GRADO DE INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

APLICACIÓN DE PREVENCIÓN DE RIESGOS LABORALES PARA SMARTWATCH

Autor: Fernando Celaya Oyón

Director: David Contreras Bárcena

Madrid

I declare, under my responsibility, that the Project presented with the title

Labour Risk Prevention Smartwatch Application

at the *ETS* of Engineering - ICAI of the Universidad Pontificia Comillas in the academic year 2021/22 is of my authorship, original and unpublished and has not has not been previously submitted for other purposes.

The Project is not plagiarized from another, neither totally nor partially, and the information that has been taken from other documents is duly referenced.

Sgd.: Fernando Celaya Oyón Date: …06…/ …07…/ …2022…

Thesis delivery authorized

THESIS DIRECTOR

Sgd.: David Contreras Bárcena Date: ……/ ……/ ……

Acknowledgements

To Juan Gómez Lagándara, graduate in Electrical and Computer Engineering by the University of Rochester and Software Developer at Morningstar, for providing the example Predictive Model and for his constant valuable input throughout the thesis.

To Julio Moreno, former Technical Director of Prevention Services at Zurich, for providing irreplaceable guidance on the inner workings of the Labour Risk Prevention sector in Spain.

To María Luisa Lagándara, Medical Services Chief at Madrid's *EMSF* (Municipal Funerary Services Company), for providing valuable information on the health services side of labour accidents.

To Juan Lázaro, Independent Member of the Board of Directors at several insurance companies, and to Alfredo Arán, Board Member of Mapfre Global Risks, for their time and experience, which through several conversations helped guide this thesis to its final topic.

To Ana Doncel, Insights & Analytics Expert, without whom dedicating all the time needed to write the thesis would have been impossible.

To David Contreras Bárcena, Head of the Degree in Mathematical Engineering and Artificial Intelligence at ICAI – Universidad Pontificia Comillas and Director of this thesis, for his confidence in the project and valuable advice.

APLICACIÓN DE PREVENCIÓN DE RIESGOS LABORALES PARA SMARTWATCH

Autor: Celaya Oyón, Fernando.
Supervisor: Contreras Bárcena, David.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

ABSTRACT

La prevención de riesgos laborales es una actividad clave y necesaria para cualquier empresario. Sin embargo, en España está fuertemente regulada y no han sido muchas las innovaciones tecnológicas que han podido cambiar sustancialmente la forma en la que se realiza. A lo largo de este Trabajo de Fin de Grado se propone una aplicación de smartwatch – con el backend correspondiente – capaz de recoger diferentes datos durante la jornada laboral que sería capaz de predecir y detectar accidentes laborales. Se examina el panorama normativo y los principales actores del ecosistema, se desarrolla un producto mínimo viable funcional con todos los componentes necesarios – exceptuando el modelo predictivo que ha sido parcialmente desarrollado – y se realiza un análisis empresarial y financiero. Se demuestra la viabilidad de la solución propuesta y se presentan los puntos clave de desarrollo en futuras etapas del proyecto.

Palabras claves: Prevención de Riesgos Laborales, Wearables, Aplicación de *smartwatch*

1. Introducción

La prevención de riesgos laborales es un conjunto de actividades a través de las cuales el empresario pretende planificar, reducir y minimizar el impacto de los riesgos a los que se ven sometidos los trabajadores durante su estancia en el centro de trabajo. El conjunto de obligaciones que debe asumir el empresario en este sentido está claramente señalado por la legislación española, país en el que se centra esta tesis. Esta legislación también presenta una serie de actores que intervienen en todo el proceso de prevención de riesgos laborales, entre los que se encuentran la seguridad social, las mutuas de seguros, las empresas de prevención de riesgos laborales y los servicios sanitarios.

Todo este sistema se ha creado para frenar la gran cantidad de enfermedades, accidentes o incluso muertes que sufren los trabajadores debido a su trabajo. Aunque la situación ha mejorado definitivamente en comparación con décadas atrás, el sistema es bastante rígido y no hay muchas innovaciones tecnológicas que puedan ayudar a los empleados y a los empresarios a reducir el número de accidentes.

2. Definición del proyecto

Esta tesis pretende conseguir precisamente eso: crear una solución tecnológica capaz de reducir el número y el impacto de los accidentes y patologías laborales.

Son muchas las formas en las que se ha intentado conseguir esto. Las diferentes soluciones han abordado desde la gestión de los equipos de protección individual hasta una mejor planificación de las evacuaciones de emergencia. Sin embargo, ninguna solución parece haberse centrado en el actor clave de cualquier riesgo laboral: el propio empleado. Se propone una aplicación de *smartwatch* para monitorizar constantemente los datos biométricos y de movimiento durante toda la jornada laboral del empleado.

Esos datos se envían a una aplicación remota ubicada en la nube, que los analiza y es capaz de detectar cualquier accidente repentino – por ejemplo, una caída – y predecir accidentes futuros – por ejemplo, la deshidratación. El trabajo de fin de grado se ha centrado en desarrollar la aplicación del *smartwatch* y la aplicación del servidor, en validar la viabilidad económica de la solución y en definir los futuros focos de desarrollo necesarios para implementar una versión funcional en un entorno real.

El proyecto se divide en tres partes. La primera, examina el panorama normativo en el que se desplegaría la solución. La segunda, esboza y documenta los detalles de las aplicaciones del lado del cliente y del lado del servidor. La tercera, presenta un análisis de negocio, examinando las posibles propuestas de valor para los diferentes actores, proponiendo el modelo de negocio más adecuado y realizando un análisis financiero de dicho modelo de negocio.

3. Descripción de la aplicación

La aplicación en su conjunto se compone de una aplicación del lado del cliente y otra del lado del servidor. Las diferentes partes de cada una pueden verse en el siguiente diagrama.

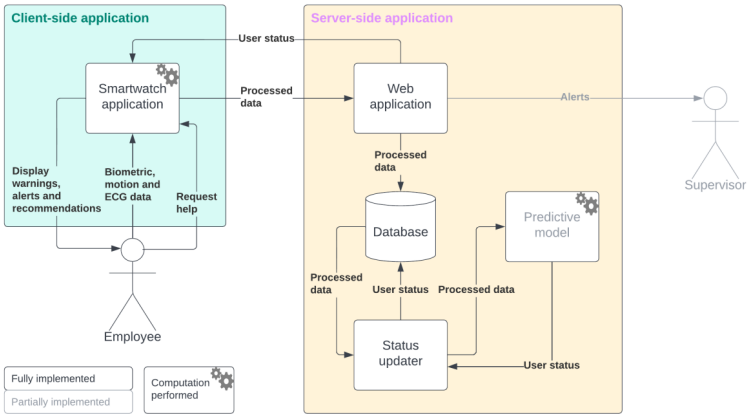

Figura 1: Representación de la arquitectura de la aplicación

La aplicación del *smartwatch* tiene tres objetivos principales. El primero es la recopilación de datos. Recoge continuamente datos del empleado a través de sus diferentes sensores. Estos datos son de tres tipos.

- Datos biométricos: Se trata de datos relacionados con la salud. Las métricas recogidas son la frecuencia cardíaca, la energía activa quemada, la energía basal quemada, el tiempo de permanencia en pie, la constancia al caminar, la exposición al audio ambiental, la variabilidad de la frecuencia cardíaca, la saturación de oxígeno, la temperatura corporal, la presión arterial sistólica, la presión arterial diastólica, la frecuencia respiratoria y la distancia caminada.

- Datos de movimiento: Son datos relativos a los movimientos del empleado. Recoge la aceleración, la rotación y la gravedad a lo largo de los tres ejes.

- Datos de ECG: Son los datos de los electrocardiogramas realizados por el *smartwatch*. No se considera parte de los datos biométricos ya que Apple no da

permisos para registrar estos datos, solo para leer los datos ya registrados, y solo se recoge un ECG al día.

Todos estos datos se recogen, se formatean y se envían a la aplicación del lado del servidor. El segundo propósito es la interacción con el empleado. Periódicamente, el *smartwatch* solicita al servidor – donde se procesan los datos mencionados anteriormente – el estado del empleado. Si se ha detectado un accidente o una futura incidencia, el *smartwatch* avisa al empleado, le da algunas instrucciones – por ejemplo, que se tome un descanso y beba agua – y se ofrece a avisar al supervisor del empleado. El objetivo final se ha implementado como prueba de concepto para validar los beneficios del *edge computing* en el *smartwatch*. Algunos cálculos básicos, como la detección de caídas, se han implementado en el *smartwatch* en lugar de en el servidor.

El lado del servidor tiene diferentes partes. La aplicación web se ha diseñado principalmente como una API REST capaz de proporcionar diferentes servicios. Los principales son recibir datos del *smartwatch* y escribirlos en la base de datos, responder a la petición del *smartwatch* con su estado – por ejemplo, Traumatismo detectado – y llamar a otros módulos como el encargado de alertar al supervisor del empleado tras la petición de éste.

La base de datos tiene diferentes tablas encargadas de almacenar todos los datos que llegan desde el *smartwatch* y el estado emitido por el modelo predictivo.

El modelo predictivo toma los datos biométricos, de ECG y de movimiento como entradas y da como resultado las probabilidades de haber sufrido algún accidente o de estar a punto de sufrir algún problema. Este modelo se ha implementado parcialmente, pero al no disponer de los datos de entrenamiento adecuados se ha sustituido por un modelo simulado que da salidas aleatorias según probabilidades preestablecidas.

El actualizador de estado recupera constantemente los datos de la base de datos, los alimenta al modelo predictivo, procesa el resultado del modelo predictivo y lo escribe en la base de datos.

4. Resultados

En primer lugar, en lo que respecta al lado tecnológico del TFG, se ha desarrollado con éxito una versión operativa de la aplicación con todas las funciones necesarias – salvo el modelo predictivo.

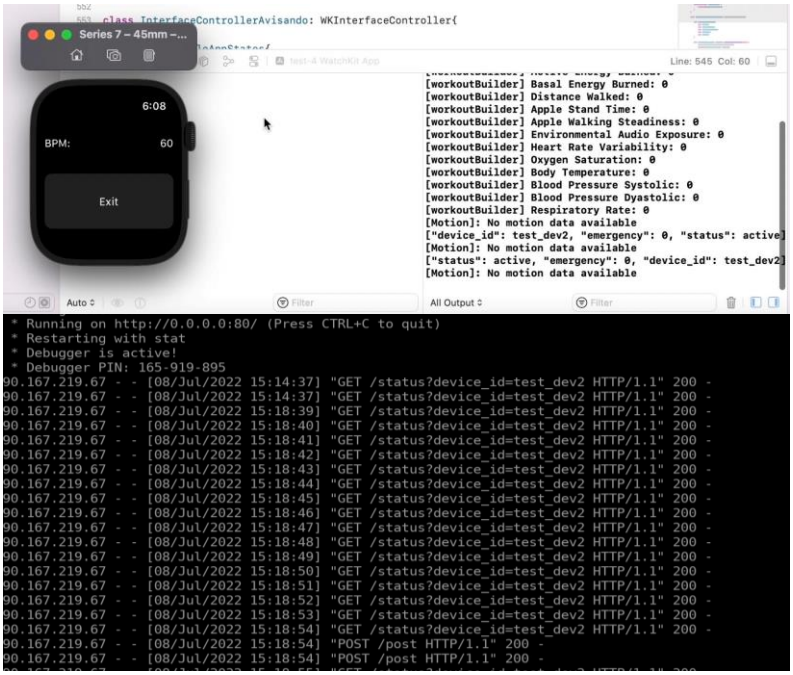

Figure 2: Capturas de pantalla mostrando el lado del cliente y del servidor funcionando

La primera mitad de la figura anterior muestra la aplicación trabajando en su estado de recolección de datos, con la consola del IDE mostrando diferentes mensajes impresos a medida que la aplicación realiza diferentes tareas – lectura de datos del sensor, lectura de estado del servidor, envío de datos al servidor, etc. La mitad inferior de la figura muestra la aplicación del lado del servidor, con las diferentes peticiones que recibe durante este periodo.

En cuanto al análisis financiero, se ha demostrado la viabilidad financiera de la solución propuesta. Se ha comprobado que el número de clientes que garantiza el umbral de rentabilidad es de 101 empresas clientes utilizando la solución, que según las hipótesis utilizadas se alcanzaría en el año 3. La inversión inicial mínima requerida es de 408.725 €, con una inversión recomendada de 500.000 €. La figura siguiente muestra la evolución de los ingresos, los costes y las reservas de efectivo (sin inversión inicial).

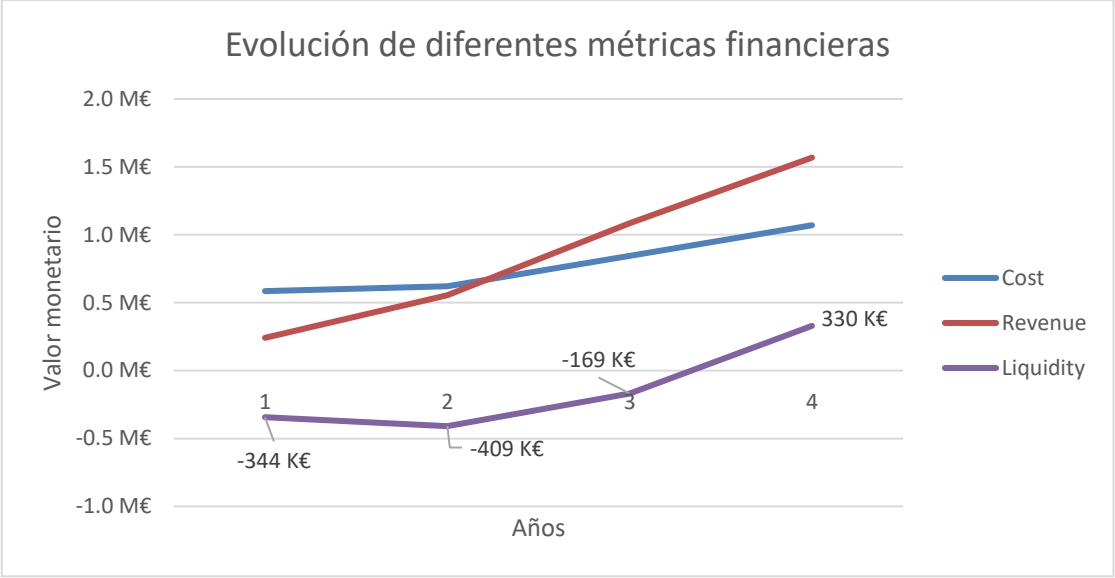

Figure 3: Evolución de métricas financieras clave obtenidas del análisis financiero de la aplicación (sin inversión inicial)

5. Conclusions

La principal conclusión es que la viabilidad de la solución propuesta ha quedado demostrada desde el punto de vista regulatorio, tecnológico y económico, siendo el ajuste y el entrenamiento del modelo predictivo la única pieza que falta por validar.

Aparte de esta conclusión general, se han determinado varios puntos de interés para la siguiente fase de desarrollo, en la que la solución se aplicará en un entorno real:

- Desplazar una mayor parte del procesamiento de datos y los cálculos necesarios fuera del servidor y hacia el smartwatch.

- Mejorar el envío y recepción de datos de la aplicación cliente, reduciendo la frecuencia y el tamaño de los mensajes enviados y recibidos. Este es el mayor factor del coste en el modelo de negocio propuesto.

- Validar que el modelo de negocio es atractivo para las mutuas. Ellos son los actores que más se benefician de la implantación de la aplicación y los que más pueden ayudar a mejorarla y hacerla crecer. Por tanto, son el cliente objetivo del modelo de negocio.

- Probar la capacidad real de reducción de costes de la solución para las mutuas, que está directamente relacionada con la precisión del modelo predictivo. Este es el motor clave de los ingresos de la aplicación y es, por tanto, la hipótesis financiera que necesita una validación más urgente.

LABOUR RISK PREVENTION SMARTWATCH APPLICATION

Author: Celaya Oyón, Fernando.
Supervisor: Contreras Bárcena, David.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

Labour risk prevention is a key and necessary activity for any employer. It has however been heavily regulated and not many technological innovations have been able to make an impact on how it is performed. Throughout this thesis, a smartwatch application – with the corresponding backend component – capable of gathering different datapoints and which would be able to predict and detect labour accidents is proposed. The regulatory landscape and key players in the ecosystem are examined, a functioning minimum viable product of all required components is developed – except for the predictive model which was partially developed – and a business and financial analysis is conducted. The viability of the proposed solution is proven and key development points for future stages of the project are presented.

Keywords: Labour risk prevention, Wearables, Smartwatch application

1. Introduction

Labour risk prevention is a set of activities through which an employer aims to plan for, reduce and minimize the impact of any risks the employees must undergo while at the workplace. The set of obligations the employer must undertake in this regard is clearly pointed out by the Spanish legislation, country where this thesis is focused. This legislation also presents a series of stakeholders which take part in the whole labour risk prevention process, including the social welfare system, mutual insurance companies, risk prevention companies and health services.

This whole system has been set up in order to curb the vast amount of diseases, accidents or even deaths that workers fall into due to their jobs. Even though the situation has definitely improved when compared to decades ago, the system is quite rigid and not many technological innovations have been able to further help employees and employers reduce the number of accidents.

2. Project definition

This thesis sets out to achieve exactly that: creating a technological solution capable of reducing the number and impact of any labour related accidents and pathologies.

There are many ways in which this has been tried. Different solutions have tackled anything from managing individual protection equipment to better planning emergency evacuations. However, no solutions seem to have focused on the key actor of any labour risk: the employee itself. A smartwatch application is proposed to constantly monitor biometric and motion data throughout the employee's workday. That data is sent to a remote cloud application, which analyses it and is able of detecting any sudden accidents – e.g. a sudden fall – and predicting any upcoming problems – e.g. dehydration. The focus of the thesis has been the development of the smartwatch application and the server application, the validation of the economic feasibility of the solution and the definition

of future development points of focus needed to implement a functional version in a real environment.

The project is split in three parts. The first, examining the regulatory landscape in which the solution would be deployed. The second, outlining and documenting the specifics of the client-side and server-side applications. The third, presenting a business focus analysis, examining the possible value propositions for the different stakeholders, proposing the most suitable business model and conducting a financial analysis of said business model.

3. Application description

The application as a whole consists of a client-side application and a server-side application. The different parts of each can be seen in the diagram below.

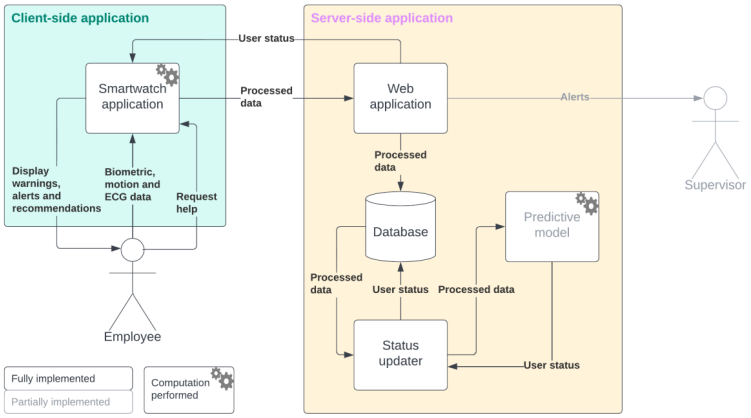

Figure 1: Representation of the application's architecture

The smartwatch application has three main purposes. The first one is data gathering. It continuously gathers data from the employee through its different sensors. This data is of three types.

- Biometric data: This is health related data. The collected metrics are heart rate, active energy burned, basal energy burned, stand time, walking steadiness, environmental audio exposure, heart rate variability, oxygen saturation, body temperature, systolic blood pressure, diastolic blood pressure, respiratory rate and distance walked.

- Motion data: This is data relative to the employee's movements. It gathers acceleration, rotation and gravity along the three axes.

- ECG data: This is data from electrocardiograms conducted by the smartwatch. It is not considered part of the biometric data since Apple does not give permissions for recording this data, just to read already recorded data, and only one ECG is gathered each day.

All of this data is gathered, formatted and sent to the server-side application. The second purpose is interaction with the employee. Periodically the smartwatch requests from the server – where the data mentioned above is processed – the status of the employee. If an

accident has been detected or a future incidence is detected, the smartwatch alerts the employee, gives him/her some instructions – e.g. take a break and drink some water – and offers to alert the employee's supervisor. The final purpose has been implemented as a proof of concept to validate the benefits of edge computing in the smartwatch. Some basic calculations, like fall detection, have been implemented in the smartwatch instead of in the server.

The server-side has different parts. The web application has been mainly been designed as a REST API capable of providing different services. The main ones are receiving data from the smartwatch and writing it to the database, responding to the smartwatch's request with its state – e.g. Traumatism detected – and calling other modules like the one in charge of alerting the employee's supervisor after the employee's request.

The database has different tables tasked with storing all data incoming from the smartwatch and the status outputted by the predictive model.

The predictive model takes the biometric, ECG and motion data as inputs and outputs the probabilities of having suffered some accident or of being about to suffer some problem. This model has been partially implemented, but due to not having the appropriate training data it has been substituted by a simulated model which gives random outputs according to preset probabilities.

The status updater constantly retrieves data from the database, feeds it to the predictive model, processes the predictive model's output and writes it to the database.

4. Results

First, regarding the technical side of the thesis, a working version of the application capable of all required features – except for the predictive model – has been successfully developed.

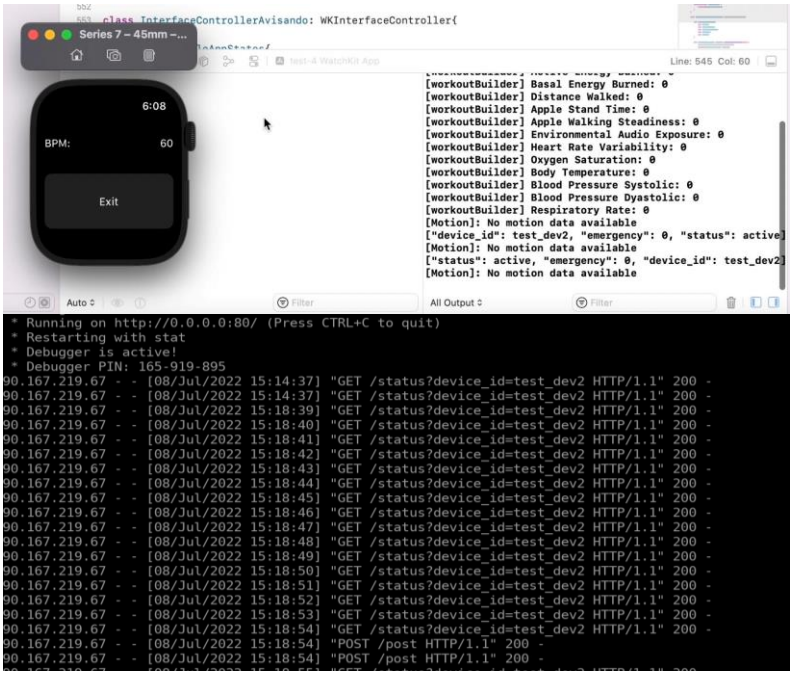

Figure 2: Screenshots showcasing the application-side and server-side applications working

The first half of the Figure above showcases the application working in its Data gathering state, with the IDE console showcasing different messages printed as the application performs different tasks – sensor data reading, status reading from the server, data sending to the server, etc. The bottom half of the figure shows the server-side application, with the different requests it receives during this period.

As for the financial analysis, it has demonstrated the financial viability of the proposed solution. The break-even number of end clients has been found to be 101 client companies using the solution, which according to the used hypotheses would be achieved in year 3, and the minimum required initial investment is 408,725 € with a recommended investment of 500,000 €. The Figure below showcases the evolution in revenues, costs and cash reserves (without initial investment).

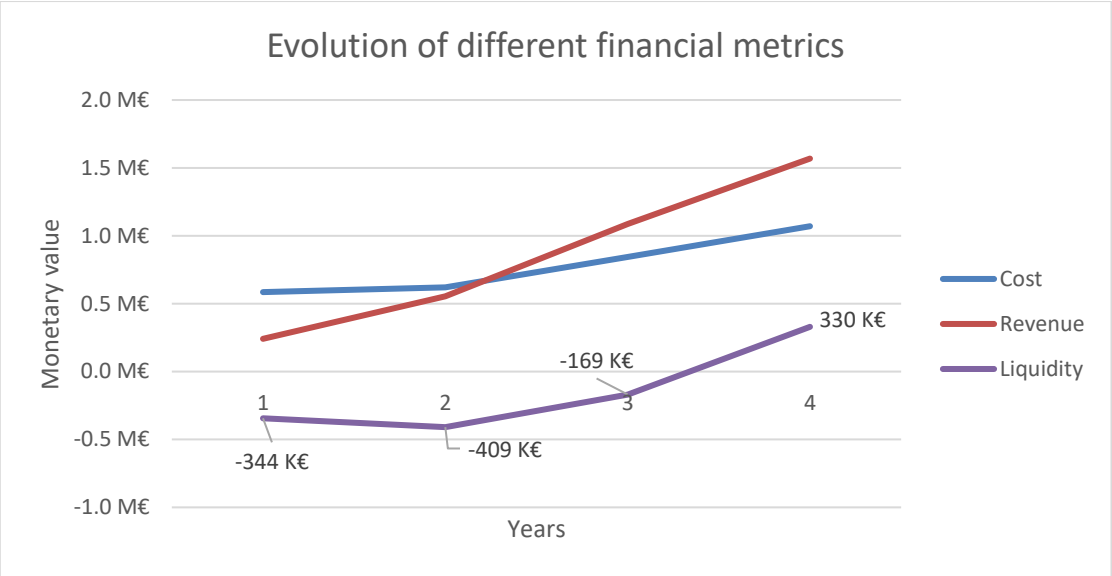

Figure 3: Financial metrics evolution obtained through the application's financial analysis (without initial investment)

5. Conclusions

The main conclusion is that the viability of the proposed solution has been proved from the regulatory, technological and financial perspectives, with the adjustment and training of the predictive model being the only piece that still needs validation.

Apart from this general conclusion, several points of focus for the next development stage – where the solution is to be implemented in a real setting – have been determined:

- Moving a bigger part of the required data processing and computations away from the server and to the smartwatch.
- Improving the networking aspect of the client application, reducing the frequency and size of messages sent and received. This is the biggest cost factor.
- Validating that the business model is attractive to Mutual insurance companies. Them being the stakeholders that stand to benefit the most from the

implementation of the application and being the ones able to help improve and grow the application the most. They are thus the client targeted in the business model.

- Testing the real cost reduction capabilities of the solution for the mutual insurance companies – which is directly related to the accuracy of the predictive model. This is the key driver of the application's revenue and is thus the financial hypothesis that needs the most urgent validation.

Table of contents

Chapter 1. LABOUR RISKS

1.1 REGULATORY LANDSCAPE

We live in a modern democracy governed by the rule of law. A such, the law is a clear delimiter of what we can and cannot do in many cases, even more so in a matter as regulated as labour risks. Thus, before trying to understand the current situation, its main problems and possible solutions it is of the uttermost importance to understand how this matter is legislated, to be able to understand the incentives of all the agents involved and the range of possible solutions.

This legislation is different in every country, so focusing in one is required at least as a starting point. The focus will be put in Spain, given the greater familiarity with this country of everyone involved in this thesis.

The current regulation on the matter in Spain is outlined mainly by the *Ley 31/1995, de 8 de noviembre, de prevención de Riesgos Laborales*. This 40-page long document outlines all the details relevant to the topic, however a review of the most fundamental points will be sufficient for all the intended purposes of this thesis.

The aforementioned legal document is composed of seven main chapters.

1. Object, application scope and definitions
2. Policy in labour risk prevention in order to safeguard the security and safety of the employees
3. Rights and obligations
4. Prevention services
5. Employee enquiries and participation
6. Obligations of producers, importers and suppliers
7. Sanctions and responsibilities

1.1.1 RIGHTS AND OBLIGATIONS

Of all these chapters, the one that is of the most interest to this thesis is the third, in which the rights and obligations of employers and employees are outlined, where the rights of the employees are the counterpart to the employer's obligations. The obligations of the employer

can be summed up in the following points, paraphrased from Articles 15 to 29, Chapter III *(Ley 31/1995 de prevención de Riesgos Laborales*, 1995):

- Article 15. Protecting employees from labour risks: the employer must guarantee the safety and security of the employees in all activities concerned with their work. The employer needs to integrate the preventive tasks in the company and will adopt whatever necessary measures to protect the employers, paying special attention to the specifities outlined in the following points. The main principles of these preventive tasks will be
    - Avoiding risks
    - Evaluating the risks that cannot be avoided
    - Fighting these risks at their source
    - Adapting the work to the given worker, paying special attention during the work post design and team and team dynamic design to reducing monotonous and repetitive work and reducing its effects on the employees' health
    - Considering the evolution of technology
    - Substituting dangerous for less dangerous tasks
    - Planning work safety, searching for a coherent whole that integrates technologies, work organization, work conditions, social interactions and the effect of ambient factors
    - Prioritizing measures that concern collective protection against those concerning individual protection
    - Give the appropriate instructions to employees.
- Article 16. Developing a labour risk prevention plan, risk evaluation and preventive activities planification: Labour risk prevention must be integrated in the company's general management system, in all activities and hierarchical levels. The labour risk prevention plan must include organizational structure, responsibilities, functions, practices, procedures, processes and the necessary resources to carry out the risk prevention activities in the company. The essential instruments for the application of said plan are the labour risk evaluation and the planification of the preventive activities.

- Article 17. Work equipment and protection means: The employer must make sure that the work equipment provided is adequate to the professional activities undertaken, guaranteeing the health and safety of the employees that use them. The employer must also provide employees with individual protective equipment and oversee their effective use.
- Article 18. Information, enquiry and employee participation: The employer must make sure that all employees receive all the relevant information pertaining to health and safety risk of collective and individual activities and protective and preventive measures taken. Employees must be asked and allowed participation in the risk prevention process.
- Article 19. Employee training: The employer must guarantee that all employees receive the necessary theoretical and practical training needed for the safe development of their tasks, both at the time of onboarding and whenever new technologies are introduced. Time spent in this training will count as working hours and its cost may never fall upon the employees.
- Article 20. Emergency measures: The employer must analyse the possible emergency situations and take the needed measures in terms of first aid, fire risks and employee evacuation. Some employees within the company must be designated to carry out the necessary measures when needed. The employer must also start relationships with the pertinent external services, particularly concerning first aid, urgent medical attention and firefighting and rescue.
- Article 21. Imminent and grave risk: When employees are currently or could potentially be exposed to grave risk, the employee must inform as soon as possible all affected employees of the risks and the measures to be taken, adopt these measures and give the needed instructions to employees, allow employees to leave their work until the risk has disappeared and make sure that employees not able to get in contact with their immediate superior have all necessary knowledge and technical measures to adopt the necessary measures.
- Article 22. Health checks: The employee must guarantee the periodic checking of the employees' health, according to their activity's level of inherent risk. The employees

have the right to object to these checks as long as doing so does not put them or others in a dangerous situation. These must be carried out by medical staff with the appropriate training and they must be carried out respecting the worker's dignity and right to privacy, always keeping the confidentiality of the outcome of these checks. The results stemming from these checks must be communicated to the employees. As for the employer, it is forbidden for him to use the data coming from these checks with discriminatory ends or seeking harm for the employee. The medical information of a personal nature will only be accessed by the medical staff and the health authority in charge of watching over the health of the employees, being shared with the employer only after the employee's explicit consent. The employer will only be informed of the conclusions derived from these checks, in order to help him better carry out his labour risk prevention functions. In some cases, the employees' right to health checks can be prolonged over his duration in the company.

- Article 23. Documentation: The employer must construct and conserve the following documentation, always having it available for the labour authorities:
    - Labour risk prevention plan
    - Evaluation of security and health risks
    - The planification of the preventive activities, including all preventive and protective measures to be carried out
    - Verification of all medical checks and the conclusions derived from them
    - Collection of all labour accidents and sickness that have derived in a sick leave of more than one day.
- The employer is also required to notify in written form to the labour authorities the health damages of its employees derived from their work.
- Article 24. Coordination of business activities: When in a given space two or more different businesses are carrying out their activities they must cooperate to apply the current labour legislation. The owner of the workspace is responsible for informing all workers – including those of the other companies located in his workspace – of the required instructions in relationship to existing risks and labour risk prevention and protection measures. When a third-party company is hired to carry out construction or

other services in the workspace, the hiring company must oversee the compliance of the hired company with the labour risk prevention measures. The same that has been said about companies applies as well to autonomous workers.

- Article 25. Protection of employees sensitive to certain risks: The employer must specifically guarantee the safety of the workers that due to personal characteristics or a recognized biological state – like employees with physical, mental or sensorial disabilities – are especially sensitive to certain labour risks. In this regard, people that can put themselves or others in danger, transitorily or permanently, due to these vulnerabilities cannot be employed.
- Article 26. Maternity protection: If the risk evaluation determines a health risk for the mother or the foetus in the mother's current labour activities, the employer must take the necessary measures to avoid the mother's exposition to the given risk. These measures will usually be an alteration of the working conditions or the working time, including when necessary the avoidance of working night shifts and shift work. When adapting conditions and working time is not possible and the Medical Services of the *Instituto Nacional de la Seguridad Social* or of the Mutual insurance company certify the potential negative impacts of the mother's current position, she must be allowed a change in work position or function, always conserving all her original work retributions. If not even this change is possible, the mother can have her contract suspended. All these changes will be in effect until the mother's health allows her to return to her original position. All of the above also applies for mother during their lactation period. Pregnant employees will also have the right to skip work, without it affecting their remuneration, for the realization of pre-natal exams and partum preparation techniques, having notified the employer beforehand and justifying the need for carrying them out during the work shift hours.
- Article 27. Minors protection: Before onboarding any minor – under 18 years old – and before any substantial change to his or hers working conditions, the employer must assess the working environment in order to determine the nature, severity and length of exposure to any agents, processes or working conditions that could harm the health or security of the minor. The employer must consider the specific risks related to the

employee's lack of experience, maturity for evaluating existing and potential risks and development. The employer must also inform the minor and his parents or tutors all the possible risks and adopted measures for the protection of his health and safety.

- Article 28. Temporary working relationships, of determined duration and in temporary employment agencies: Employees with temporary working relationships or of determined duration as well as those hired by temporary employment agencies must enjoy the same level of protection as the employees in the company where they are carrying out their activity. Every right outlined in the previous articles will also be safeguarded for the beforementioned workers, including the right to being informed about the risks to which they are being exposed and the pertinent protection and prevention measures and the right to receiving medical checks. It will be the hiring company – not the temporary employment agency – the one responsible for guaranteeing the safety of the working conditions. The temporary employment agency will have the responsibility of guaranteeing that all necessary training is performed and of checking the health of the workers.

This final article is the only one that outlines the obligations not of the employer, but of the employees. Given how the proposed solution will be primarily used by the employees, it is also necessary to review these obligations as they will help understand what kind of compliance can be expected from the employees when an employer proposes a new labour risk prevention tool.

- Article 29. Employees' obligations concerning labour risk prevention: Every employee must watch out for his own safety and for the safety of those affected by his work. In particular, according to their training and following the employer's instructions, all employees must:
    - Use adequately all machines, apparatus, tools, dangerous substances, transportation equipment and any other mean by which they carry out their work
    - Use adequately the protective means and equipment provided by the employer, according to the instructions given out by him
    - Use adequately all safety devices and not disable them

- Immediately inform of any situation that may lead to health and safety risks of other employees to the immediate supervisor, employees designated to carry out protection and prevention activities or the prevention service
- Contribute to the compliance of the established obligations
- Cooperate with the employer so he may guarantee safe working conditions

As can be seen, the labour risk prevention law outlines a wide array of obligations and rights, ranging from required training to special attention to given collectives. Not all of these obligations are going to be addressed by the proposed solution. However, this section is of a purely descriptive nature of the situation *as is*, and thus no further analysis of the provided articles from the framework of the proposed solution will be carried out. What will be said, in order to better frame and understand future sections, is that the obligations which the proposed solution aims to tackle are the ones outlined in Articles 20, 21, 22, 23, 25 and 29.

1.1.2 PREVENTION SERVICES

Even though the Chapter relating to the rights and obligations of both employer and employees is the one that gives the best general picture about labour risk regulation, another fundamental chapter that will have to be considered is Chapter IV. This chapter outlines the regulation relative to prevention services, and will be key mostly to understand the business implications of the proposed solution and how it will fit in the current landscape of risk prevention companies.

Since this chapter is of shorter length and easier to understand, it won't be necessary to go over all articles one by one. They all can be summarized in the following paragraphs.

The law states that in order for the employer to fulfil its prevention duties, he must do one of three things:

- Designate one or more employees from within the company to be in charge of the prevention activities.
- Constitute a prevention service within the company
- Outsource that service to a third-party entity specialized in the matter

No matter what choice the employer makes, he must collaborate with the chosen team by providing them with all the information already outlined in Articles 18 and 23. If the choice is to designate internal human resources to carry out these tasks, the employer must also guarantee that no harm will come to them stemming from their risk prevention activities and he must also make sure that the internal team and resources is audited externally.

Which of the three outlined choices the employer is able to make will depend on three main factors, being these:

- Size of the company
- Type of risks which the employees can be exposed to
- Risk distribution within the company

An exception can occur if the company is smaller than 10 employees, or if it is smaller than 25 employees and the company only has a single work centre. In these cases, all prevention duties can be carried out by the employer himself.

No matter who carries out these duties, it will be considered a prevention service, which is defined as "all the human and material resources necessary to carry out preventive activities in order to guarantee the adequate protection of workers' health and safety, advising and assisting adequate protection of the safety and health of workers, advising and assisting the employer, the workers, their representatives the employer, the workers and their representatives and the specialised representative bodies" (*Ley 31/1995 de prevención de Riesgos Laborales*, 1995). The explicit duties of the prevention services are to support and advice the company in anything relative to, as outlined in Article 31:

- The design, implementation and application of an occupational risk prevention plan that allows for the integration of prevention in the company.
- The evaluation of risk factors that may affect the health and safety of workers
- The planning of preventive activity and the determination of priorities in the adoption of preventive measures and the monitoring of their effectiveness.

- The information and training of workers, under the terms already provided in Articles 18 and 19.
- The provision of first aid and emergency plans.
- The monitoring of the health of workers in relation to the risks derived from work.

The solution proposed in this thesis will be key in aiding prevention services in these fifth and sixth points.

If there is further interest in the matter, the general guidelines outlined above are further developed in the (*Real Decreto 39/1997 por el que se aprueba el Reglamento de los Servicios de Prevención*, 1997)

1.1.3 SOCIAL WELFARE SYSTEM (*SEGURIDAD SOCIAL*)

Another key part of the Spanish legislation that needs to be considered is legislation relative to the Spanish social welfare system. Its main legislative text is the *Real Decreto Legislativo 8/2015, de 30 de octubre, por el que se aprueba el texto refundido de la Ley General de la Seguridad Social*.

There is however no need to analyse its 229 pages as long as an understanding of the general concepts and those specific to labour risks is attained. So, what is the Spanish social welfare system? It can be understood as a public organism and set of policies designed to protect citizens who find themselves in a situation of vulnerability. Every employee and company operating in Spain is obligated by law to monetarily contribute to the organism, which manages the funds and distributes them to citizens in times of need. The benefits provided by the welfare system are those, according to the official welfare's system magazine, relative to retirement, widowhood, orphanhood, permanent disability and leaves of absence due to illness, maternity, paternity and risk during pregnancy and lactation (*Qué es realmente la Seguridad Social*, 2021). Apart from this it also covers pharmaceutical and medical services benefits.

As stated above, all contributions to the welfare system are regulated by law, and one of such contributions is that relative to labour risks. Every employer must pay a given amount to the welfare system for each of its employees, so that if the employee has a sick leave, is left

permanently disabled or dies in a work place accident the welfare system can pay the employee or its family all the benefits associated with the situation.

The amount each employer pays for each employee is a set amount that is also regulated by law. It depends on the activity developed by the employee, so more risky activities with a higher risk of resulting in accidents have a higher contribution, as the welfare system's expected cost of such employees' benefits is higher. The set amount for each type of activity is presented in a table, the latest of which is presented in the fourth additional provision of the *Ley 42/2006 de Presupuestos Generales del Estado para el año 2007*, the 2007 General State Budgets. A sample part of this table has been included below.

Cuadro I		Tipos de cotización		
Códigos CNAE-2009 y título de la actividad económica		IT	IMS	Total
01	Agricultura, ganadería, caza y servicios relacionados con las mismas Excepto:	1,50	1,10	2,60
0113	Cultivo de hortalizas, raíces y tubérculos.	1,00	1,00	2,00
0119	Otros cultivos no perennes.	1,00	1,00	2,00
0129	Otros cultivos perennes.	2,25	2,90	5,15
0130	Propagación de plantas.	1,15	1,10	2,25
014	Producción ganadera (Excepto el 0147).	1,80	1,50	3,30
0147	Avicultura.	1,25	1,15	2,40
015	Producción agrícola combinada con la producción ganadera.	1,60	1,20	2,80
016	Actividades de apoyo a la agricultura, a la ganadería y de preparación posterior a la cosecha (Excepto 0164).	1,60	1,20	2,80

Table 1: Contribution rate for accidents at work and occupational diseases

As can be seen, the different types of regulated activities are presented together with two types of contribution rates:

- IT (*Incapacidad Temporal*): The contribution rate attributed to temporary incapacity.
- IMS (*Invalidez, Muerte y Supervivencia*): The contribution rate attributed to permanent incapacity and death.

The employer must contribute the sum of both rates, which are expressed as percentage points for the employee's base welfare system contribution. This rate ranges from 1.5% for multiple jobs, such as "Manufacture of knitted and crocheted garments" to 7.5% for jobs such as "Regular work inside mines".

There is however one exception to these set rates that is of real interest to this thesis. That exception can be found in the *Real Decreto 231/2017 por el que se regula el establecimiento de un sistema de reducción de las cotizaciones por contingencias profesionales a las empresas que hayan disminuido de manera considerable la siniestralidad laboral*. As the name suggests, this law establishes a reduction in the labour accidents and diseases rates for companies that have been able to significantly reduce labour accidents. This is another incentive – apart from the ones that will be explained in more detail in the economic viability section of the thesis – for all parties involve to reduce the frequency and severity of accidents and illnesses in the workplace, which will make companies more prone to adopting the proposed solution

1.1.4 MUTUAL INSURANCE COMPANIES (*MUTUAS*)

The social welfare system is however only one side of the coin, being mutual insurance companies the other. Their main legislation is the same text analysed in the previous segment (*Real Decreto Legislativo 8/2015 por el que se aprueba el texto refundido de la Ley General de la Seguridad Social*, 2015).

Starting in a similar fashion, the first question is what are mutual insurance companies. Obtained from the same official magazine, mutual insurance companies are non-profit associations of business owners that collaborate with the welfare system in the execution of many of its services (*¿Qué son las mutuas?*, 2018). They can be seen as the independent, execution branch of the welfare system. Thus, they are the ones in charge of receiving the payment done by the employer and managing everything related to labour accident claims.

Another important point regarding MICs (mutual insurance companies), as explained in Article 82 (*Real Decreto Legislativo 8/2015 por el que se aprueba el texto refundido de la Ley General de la Seguridad Social*, 2015), is that they will be the ones in charge of providing the health services related to labour accidents. These "shall be provided through the means and facilities managed by the mutual insurance companies, by means of mutual insurance companies, through agreements with other mutual insurance companies or with the public health administrations, as well as through agreements with private means".

1.1.5 DATA PRIVACY

All the legislation relative to the structure of labour risk prevention in Spain has already been covered, allowing for an understanding of the different players and their obligations. There is however a key part of legislation that still needs to be addressed.

In recent years, the world in general and the European Union in particular have started to realize the importance of data privacy. After the appearance of many business models – such as the whole internet advertising complex which work by extracting and selling personal data –, people have begun valuing their privacy and have started defending their right to have ownership over their own data. There is still much to be done in this aspect, but some steps have already been taken in the form of legislation trying to regulate what can and cannot be done with user's data. The main piece of legislation with this concern is the General Data Protection Regulation (GDPR). Its implementation in 2018 was pivotal for the industry, introducing rights and obligations such as users being able to demand that all of their data be erased from some company's database, also being able to require to see all the information a company has on them or forcing websites to allow users to reject all non-necessary cookies.

The solution proposed in this thesis will have to handle a great amount of user's data of the most sensitive type, health data. It is thus fundamental to look at the GDPR, not only to make sure that the solution is developed within the bounds of the law, but also as an ethical and moral compass, to understand how far the data retrieval and processing should go. Below are presented the main GDPR articles with concern to the proposed solution. Only the relevant sections of each article are presented (Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016, 2016):

- Article 5. Principles relating to processing of personal data: This article is of interest not so much for the legal implications it may have but more for the ethical and moral considerations it introduces, as it is quite vague. The main principles which any data handling operation should abide by are:
    - Lawfulness, fairness and transparency: The data should be processed in accordance to the law and in a fair and transparent manner to the user.

- Purpose limitation: All data should be collected for a "specified, explicit and legitimate purpose and not further processed in a manner that is incompatible with those purposes".
- Data minimisation: Only the minimal amount of data required for the purpose at hand should be processed.
- Accuracy: The gathered data should be accurate, and an explicit effort should be made to keep that data up to date and to fix any errors.
- Storage limitation: Data should be kept in a form in which it allows for personal identification for as little time as possible, and longer storage after that point should be done only as long as it is for the public's interest or for scientific, historical or statistical research. In this cases, specific measures – outlined in another article – should be taken to ensure the proper long term storage of the data.
- Integrity and confidentiality: The proper technical and organizational measures should be taken to prevent unauthorised or unlawful processing of data and the accidental loss, destruction or damage of said data.
- Accountability: The entity in control of that data must be able to demonstrate compliance with all previous principles

- Article 6. Lawfulness of processing: For the processing of data to be lawful, at least one of the following circumstances must apply:
  - The user has given consent for the processing of his/her data for one or more specific purposes.
  - The processing of said data is required for the fulfilment of a contract of which the user is part or to fulfil the request of the user even if no contract exists.
  - The processing of said data is necessary for the company to fulfil its legal obligations.
  - The processing of said data is necessary to "protect the vital interests" of the user.
  - The processing of said data must be done in the public interest or to comply with an official authority's demands.

  - The processing of said data must be done to comply with the legitimate interests of the user, always when these interests do not conflict any fundamental rights and freedoms.

- Article 9. Processing of special categories of personal data: Processing of some categories of data, such as racial or ethnic origin or political or religious beliefs shall be prohibited. Processing of genetic or biometric data for the purpose of identifying a person shall be prohibited. Processing of data concerning health or a person's sex life or sexual orientation shall be prohibited. The previous shall not apply if one of the following applies:
  - The user has given explicit consent for the processing of said data.
  - Processing of said data is necessary for the carrying out of obligations and exercising of rights of the entity or of the user in the field of employment and social security, always providing safeguards for the fundamental rights and interests of the user.
  - Processing of said data is necessary in order to protect the vital interests of the user.
  - Processing of said data is carried out by foundations, associations or other non-profits when the processing relates to members or former members of the body, and that data is not disclosed outside that body without explicit consent.
  - The processing relates to data that has been explicitly made public by the user.
  - Processing of said data is necessary for the fulfilment of judicial requirements.
  - Processing of said data is necessary for the pursuit of substantial public interest, while always respecting the fundamentals of data protection.
  - "Processing is necessary for the purposes of preventive or occupational medicine, for the assessment of the working capacity of the employee, medical diagnosis, the provision of health or social care or treatment or the management of health or social care systems and services on the basis of Union or Member State law or pursuant to contract with a health professional".

- Processing of said data is necessary for the pursuit of public interest on the topic of public health, such as cross-border transmission of pathologies or ensuring the quality and safety of health products.
- Processing of said data is necessary for the pursuit of public interest or scientific, historical or statistical research purposes, while always safeguarding the fundamental rights and interests of the user.

An important caveat to the eighth point – completely in between quotes – is that the processing to which that point refers must be done "by or under the responsibility of a professional subject to the obligation of professional secrecy under the Union or Member State law".

These Articles cover just the first eleven pages out of the total of seventy-eight the document covers. The next articles talk about further specifities, such as specific data categories like criminal data or introduce specific technical and non-technical measures to ensure the proper implementation of the principles already stated. These first articles are however enough to understand the general gist of the whole document and the most important considerations when determining the legal viability of the proposed solution.

Apart from the special measures that will have to be taken, the main takeaways relative to its legality is that consent from the employees will be needed, and some medical staff will have to oversee and be responsible for the data processing of the proposed solution.

As a footnote on data privacy, it is important to note that the GDPR is not the only legislative document which must be followed for this topic. The Labour risk prevention law itself *(Ley 31/1995 de prevención de Riesgos Laborales*, 1995) also addresses the health data privacy issue. It will not be further developed here as it has already been explained in the Rights and obligations subsection, in Article 22.

1.2 MONETARY FLOWS

Now that the legislative landscape is covered, the set of rights and responsibilities of each stakeholder in the labour risk prevention landscape has been thoroughly examined. A part of

the current landscape that is however not as well understood just by looking at the regulation is the monetary flows between the stakeholders. These flows can help understand the underlying relationships that lie behind each one's responsibilities, and will be key for understanding the business model and financial analysis performed in the Economic analysis section.

Here is presented the diagram explaining how money flows throughout the system.

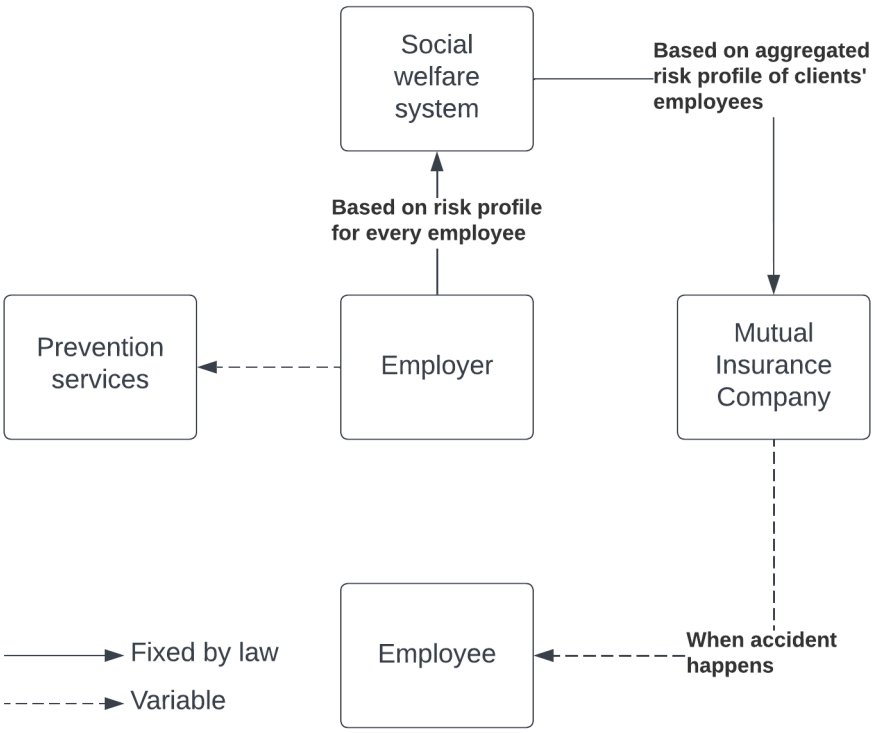

Figure 1: Representation of monetary flows relevant to labour risk prevention

As it can be seen in the diagram above, the Employer must pay two different things relevant to labour risk prevention. One is a payment made to the social welfare system, which is one part of the social security contribution the employer makes to the social welfare system on behalf of the employee. This contribution has different reasons, like potential unemployment benefits, future retirement benefits, etc. and one of them is the potential labour accidents payment. This payment's amount is set by law and depends on the risk level of the employee's activities. The

other payment is the one made to the prevention services providers, which can either by a third-party company, be part of the employer's company or be offered by the mutual insurance company. The social welfare system then distributes the payment made by all employers to the different mutual insurance companies, based on the risk profiles of their client's employees. When an employee suffers an accident, it makes a claim to the insurance company, which pays for the death, permanent incapacity or temporal leave that stems from the accident. Even though the amount the mutual insurance must pay to each employee is set by law, the total amount it pays to all employees throughout a given year depends on the total amount of accidents that happen.

1.3 OPPORTUNITIES AND CURRENT SOLUTIONS

As it has been seen in the previous sections, the labour risk prevention market is heavily regulated and particularly mature. This leads to there being a very wide range of activities and obligations that need fulfilment – where each of them is an opportunity for a solution to facilitate the fulfilment of said obligation – but some of these obligations can only be fulfilled by government entities and the ones that do not are fulfilled by settled players with great economies of scale and experience in the sector. That means that breaking into the main market can be difficult, so most of the new solutions do not attempt to fulfil main obligations, but to aid settled players in better carrying out these tasks or attempt to fulfil niche tasks. Here are some examples of companies offering these kinds of labour risk prevention services:

- EPC Tracker: This startup from Andalusia has released an app to manage and reduce accidents in construction sites. More precisely, this app allows for "checking safety equipment instantly and digitally on the move, keeping track of all risk prevention actions, accessing instructions and safety methods for each task via mobile terminals, declaring incidents by providing graphic documentation and even controlling IPE (individual protection equipment, *EPI* in Spanish), among other things." (Startups andaluzas desarrollan soluciones para mejorar la seguridad en el trabajo, 2021)

- Safeguru: As stated in Article 17 *(Ley 31/1995 de prevención de Riesgos Laborales*, 1995), part of the employer's obligations is providing employees with IPE. The market for this type of equipment used to be really traditional, with the employer having to go through a long, arduous and many times analogue process to acquire the proper, approved equipment that suited his needs. That is the problem Safeguru, a startup from Málaga born in 2021, tried to tackle. They did it by creating an e-commerce platform focused on speeding up the buying and selling of IPE. (La nueva "Start Up" del EPI y PRL, que simplifica el proceso de compra, 2021)
- WavyDrive: This Andalusian startup aims to prevent labour risk for two types of scenarios. The first is for employees that must drive a vehicle as their main activity – truck drivers, bus drivers, delivery van drivers… The second, is for every employee during a part of the work shift often overlooked by other labour accident prevention companies: the commute to and from work. This commute is legally established to be part of the working activities, so any accident that happens during it is considered a labour accident. Their application allows employees to analyse their driving habits, get alerts when the vehicle's maintenance is due and offers them indications on better driving – covering things such as braking, speed limit surpassing and curve taking. (Startups andaluzas desarrollan soluciones para mejorar la seguridad en el trabajo, 2021)
- Millas Digitales: The covid pandemic meant the appearance of a serious and widespread new labour risk: that of being infected by covid in the workplace. In order to tackle this new risk, the Madrid startup called Millas Digitales released a new product to enforce social distancing and capacity limits. Using Bluetooth and the employees' phones, they alerted two people when they got too close together and monitored continuously the capacity of every building, allowing employers to monitor these two metrics – infractions of social distancing and capacity – in real time and to derive insight from the generated reports. (Delgado A. , 2020)

- Archangelus System: This startup, also from Andalusia, aims to tackle a completely different part of labour risk prevention. They have invented a set of smart evacuation lights that are able to improve security during emergency evacuations and can dynamically and automatically manage evacuation routes, adapting to the specific incidence. (Startups andaluzas desarrollan soluciones para mejorar la seguridad en el trabajo, 2021)

Looking at all these solutions, one is able to see some common patterns. All of them leverage some form of IT innovation, some of them to help transform a traditional task – like buying IPE, managing evacuations or managing risk in construction sites – making them easier and more efficient and some of them to add some predictive or analytic capabilities to things that up to that point were not a problem or were just assumed to be part of normal working conditions – like traffic accidents.

All of them offer some value in their own way, the surprising part however is that none of them seem to go to the root of labour risk prevention. Labour risk prevention is at the root just about one thing: preventing labour accidents or injuries from happening. That is the main goal, and every task like risk analysis, prevention planning, EPI distribution, etc. is just a controlled way of tackling the issue of labour accidents, making them less probable or confining their consequences as much as possible.

The intuitive market gap thus becomes a solution able to skip all these middle steps and tackle the problem at its root. That is, a solution able to directly predict and prevent accidents, or able to truncate their negative health effects when they do happen. That will be the goal of the proposed solution. However, to better understand it, labour accidents must be examined first.

1.4 LABOUR ACCIDENTS

To create a product capable of predicting and preventing labour accidents, one must first understand what labour accidents are and how they happen. Note that this solution is specially tailored to manual workers, which in general are some of the most accident-prone workers

there are, with some special niches, such as construction workers or miners being even more prone. It is important to point out that accident-prone relates to two different types of accidents:

- Chance accidents: These are immediate accidents, in which the transition from complete health to injury happens in a matter of seconds. Some examples are having a heavy object fall in your head or having a limb amputated by industrial machinery. In order to lessen the harm, to the employee and the company, stemming from this type of accidents two types of things can be done:
    - *A prioir*: The more effort that goes into risk analysis, employee training, labour risk prevention plan, etc. The more of these accidents that can be prevented. If these activities are performed properly, the employee will not really notice their impact in his day to day, but the overall effect will be a reduction in the number of accidents.
    - *A posteriori*: With this kind of accidents, the speed and agility of the moments after the accident are of the utmost importance. The faster the accident is noticed, the faster the employee is taken to the appropriate health staff and the faster the diagnosis is made, the better the outlook of the accident will be

  These accidents can lead to both injury and leave of absence or death.
- Extended accidents: Usually, these aren't even called accidents, but they will still be referred to as such in this thesis for simplicity's sake. These are the kind of accidents that don't occur from one moment to the next, but that happen due to an extended behaviour. This behaviour can happen in a matter of minutes, like intense physical activity leading to a heart attack; in a matter of hours, like extended heat without drinking leading to dehydration; or in a matter of days or even months, like extended stress and anxiety leading to depression. This type of accidents are prevented again in two different ways:
    - Planning: By setting the right guidelines and policies, it can become more difficult for employees to find themselves in one of these situations.

- Monitoring: By spotting the unhealthy behaviour in advance, it can be stopped before it leads to a given pathology or injury, and an opposite positive behaviour can be recommended to balance the effects of the negative one.

Someone could argue that these categories are not completely mutually exclusive. If someone has bad sleeping habits for example, being more tired can lead to being less focused at work and thus to a higher probability of suffering a chance accident. This is completely true, even though tracing the occurrence of a given chance accident to a given behaviour pattern is usually a lot harder than tracing an extended accident, as in chance accidents – as the name suggests – there are many more chance factors and many different behaviour patterns involved.

Now that the two main categories are described, the main labour accidents (Contreras & Bullon, 2021), (Delgado S. , 2020), (Los tipos de accidentes de trabajo más comunes, n.d.) will be examined and classified:

- Falls: Falls from first or second heights are one of the most common types of workplace accidents, with around fifty people dying in Spain from falls from stairs alone (Los 7 accidentes de trabajo más frecuentes #DíaMundialDeLaSeguridadYSaludEnElTrabajo, 2017).
- Getting hit by falling objects: Mostly in factories or construction sites, unsecured objects can fall on people causing them grave injuries. This is another type of chance injury.
- Back injuries: Back injuries, such as herniated discs, are also quite common for manual workers of medium age. They happen most usually due to the lifting of excessively heavy objects, or due to the repetitive lifting of objects. These injuries can be both chance and extended accidents.
- Muscle tears: They are a subset of possible back injuries, but are also quite frequents in other parts of the body like shoulders or legs. Same as back injuries, they can happen due to extreme effort as a chance accident or as an extended accident due to fatigue, overuse or improper use of a muscle (Muscle Strains, 2018).

- Repetitive motion wear: These are the type of injuries that happen due to making the same movement over and over again. They are generally not as intense as a muscle tear but can also leave someone incapacitated to work. Some examples of precise pathologies are tendinitis or bursitis. As its name suggests it is an extended accident.
- Lacerations and amputations: These injuries caused by blunt trauma are most common in construction sites or industrial settings, and are an extreme form of chance accident.
- Intoxications: When working with chemical compounds, in the production of chemical products or in the use of these products, repeated exposure can end up causing multitude of respiratory problems, from asthma to lung cancer. This is another type of extended accident.
- Electrocutions: Electrical accidents can also happen in many different settings and range in severity, from a little scare to death. This is also a chance accident.
- Loud noises: Repeated exposure to loud noises can lead to what is called industrial deafness, or partial or complete loss of hearing due to loud noises at work. These are most usually extended accidents but in extreme cases can also occur due to a single event.
- Vibrations: The extended exposition to vibrations, mainly due to the handling of vibrating machinery, can lead to a vast range of musculoskeletal diseases. These are also Extended accidents.
- Vehicle accidents: As its name suggests, these are accidents that happen while driving or being in a vehicle. Everyone is familiar with this type of accidents as they are quite common outside of the workplace. They are chance accidents.
- Mental disorders: This encapsulates the whole range of mental disorders, like anxiety disorder, stress or depression, that is caused due to conditions found at the workplace. Even if this is not the type of labour accidents people think of when thinking about labour risks they should not be overlooked. Depression alone is the second cause of work leave – only behind musculoskeletal disorders – in Spain, where its economic effect between work leave, permanent incapacity, diminished productivity or accelerated retirement is estimated to be between 150 and 370 million euros annually (La depresión es la segunda causa de baja laboral, incapacidad permanente o jubilación

anticipada en España, 2020). This is actually a bad thing, since this type of accidents will be the hardest to predict and detect with the proposed solution, if it is at all possible. These are almost always extended accidents.

This list is by no means comprehensive. It is a diverse selection of some of the most common workplace accidents. For an exhaustive list, one can examine the 2010 revised list of occupational diseases, by the International Labour Organization (ILO List of Occupational Diseases, 2010). This is a list of 106 diseases classified by organ and causing agent. Apart from this list, they also present some very interesting studies related to trends in Labour accidents and diseases, presenting for example the proportion between the deaths caused by occupational accidents (chance accidents) and caused by work-related diseases (almost always extended accidents). This figure can help understand the stark difference in proportion between both.

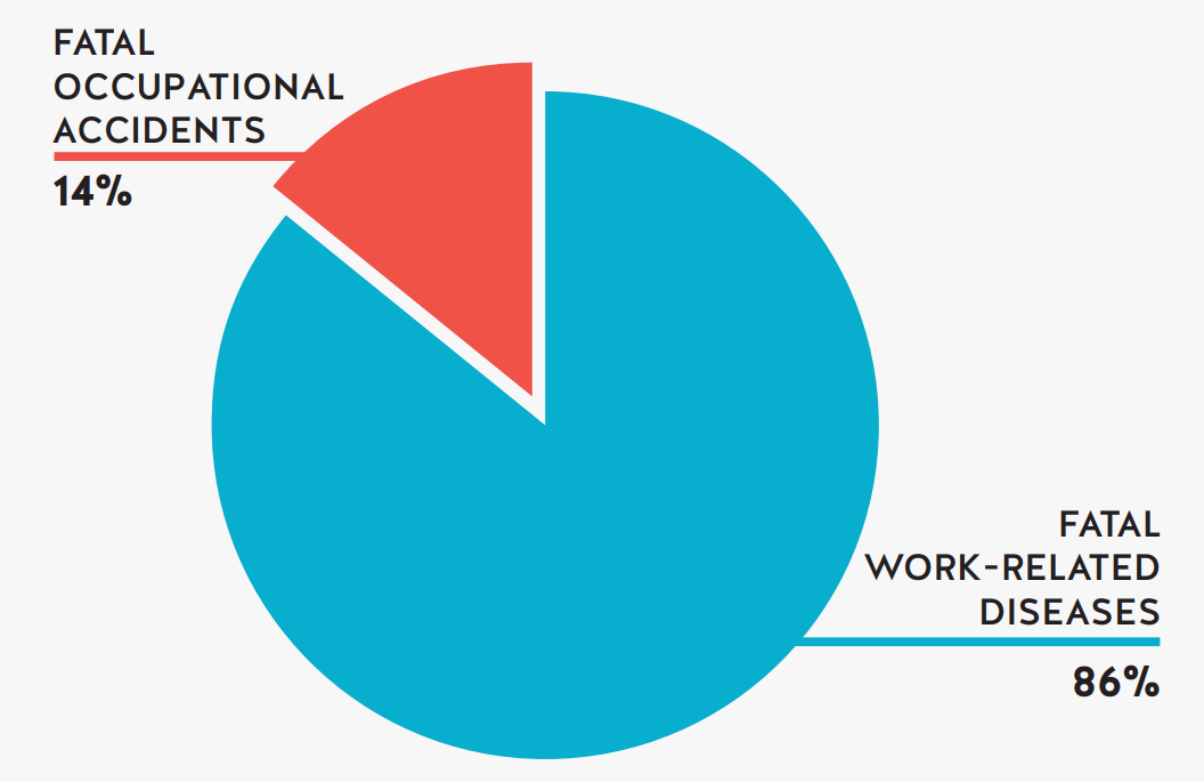

Figure 2: Occupational fatal accidents and diseases. (Global trends on occupational accidents and diseases, 2015)

This different in proportion showcases the importance of focusing on the prediction and prevention of diseases, which almost always occur not immediately but as an extended accident. This is quite fortunate, since it will also be easier for the proposed solution to address, as chance accidents are by definition almost impossible to predict.

That does not mean that the solution will completely leave aside chance accidents, as it will try to address both. As different as the diseases and accidents are in this presented list, practically all of them exhibit some sort of effect on biometric and motion data. This data will be the one leveraged by the proposed solution to predict, prevent and detect them in time. That is precisely what will be explained in the following section.

Chapter 2. PROPOSED SOLUTION

After understanding the regulatory landscape in which the Spanish labour risk prevention ecosystem is immersed and understanding the different obligations and opportunities involved, it is now time to discuss the proposed solution.

This solution is a labour risk prevention smartwatch app for manual workers. From here on, the overall solution will be referred to as "the solution", "the proposed solution" or "the application". Before going into more detail about how it is laid out and the technologies used, it is necessary to understand the big picture about what it attempts to do.

The solution aims to predict and prevent labour accidents, and to mitigate their effects when they do happen. Regarding the two main types of accidents discussed in the previous section, here follows a brief description about how it plans to tackle each type of accident:

- Chance accidents: The solution aims to lessen the severity of such accidents through an *a posteriori* approach. The smartwatch will be continuously monitoring different parameters about the employee's behaviour and health – parameters which we will specify in detail in the following sections. Thus, if a chance accident happens the application will be able to detect its immediate effects: spike in heart rate, decrease in blood oxygen levels, etc. or even the accident itself, like detecting the fall of an employee thanks to the accelerometer. This allows for a practically immediate detection of the accident in practically all environments. The application can also potentially allow for indications to the employee to lessen the effects of the accident in case he can carry them out and for an immediate notification to the employee's immediate supervisor or labour risk prevention representative. The diagnosis can also be sped up, by measuring all these parameters after the accident, which could potentially be sent to the appropriate health services for a faster evaluation of risk and diagnosis, which allows for a faster acceptance into treatment.
- Extended accidents: The solution aims to prevent this kind of accidents through a continuous monitoring of the parameters already presented in the previous paragraphs.

These parameters can be fed into a predictive algorithm that is able to understand the overall picture given by them and give a probability that the employee will end up suffering some kind of extended accident. When that probability reaches a certain threshold, the application is then able to notify the employee and offer some advice as to how to prevent the accident from happening. It is important to note how the longer the behaviour needs to be for the accident to happen – like with the depression example previously mentioned – the more difficult it could be for the predictive model to accurately predict the probability of the accident happening.

This is the general idea for the proposed solution. However, it is of critical importance to point out that the technical development carried out in this thesis is that of a minimum viable product (MVP). The basic technical foundation and the basic functionalities have been built, however building a complete version of the solution with all features is outside of the scope of this thesis. That is why in some discussions, mostly the business-oriented ones, some features that have not been built in the presented MVP will be discussed. In the technical specification the MVP features will be discussed in depth, touching on how to build some of the not-yet-included features in the future, integrating them into the current architecture.

2.1 GENERAL ARCHITECTURE

After the big picture introduction to the proposed solution, it is time to dive deeper into the technical intricacies of the MVP development. First, the general architecture will be discussed, followed by an in-depth discussion of the different components. The main components have been outlined in the following diagram.

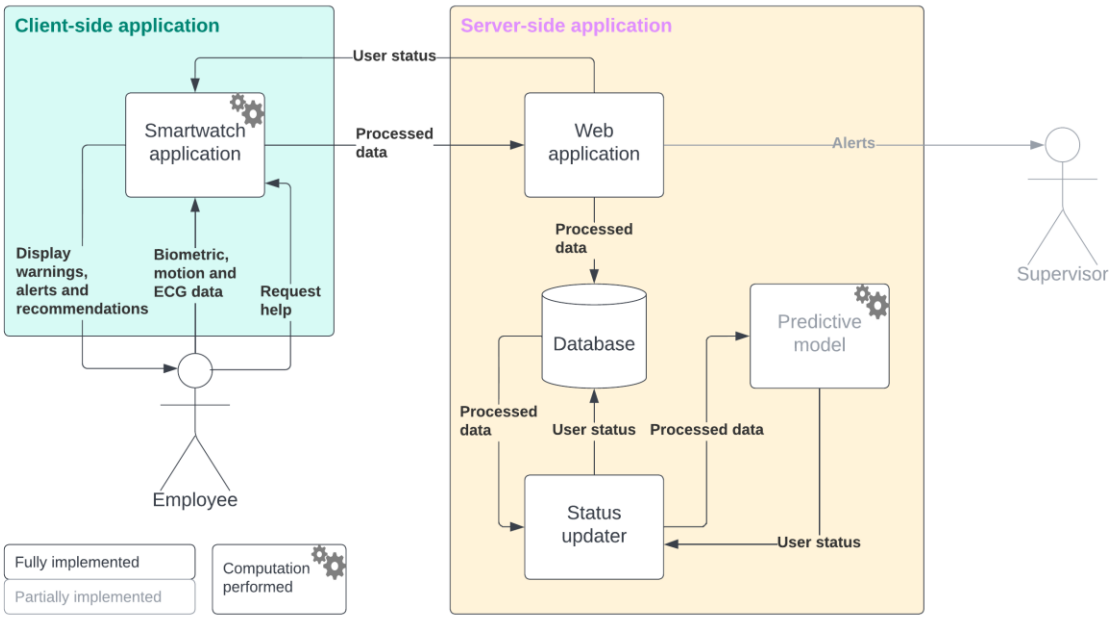

Figure 3: Representation of the application's architecture

As it can be seen in the diagram above, there are 2 main sides to the application:

- Client-side application: This is the part of the application developed for the smartwatch. It has 3 main purposes:
    - Data: Here is where all of the dynamic datapoints are gathered. Dynamic are any parameters that change (e.g. Blood Oxygen Saturation) v.s. those parameters that, even if they are used to make the extended accident's prediction, don't change or take much longer to change (e.g. sex, height, weight, etc.). The client-side application then formats these datapoints and sends them to the server-side application.
    - User interaction: The smartwatch is also fundamental to manage interactions to and from the user. It is the only open channel communication with him/her, and is thus the one used for the two types of needed communication.
        - Application-user communication: When the application suspects something may be wrong, or wants to warn the user or something, this is presented in the smartwatch's user interface (UI) for the user to see.

        - User-application communication: When the application suspects something is happening, it asks the user for confirmation that he/she is okay, or whether he/she wants to take a break or continue working. This user input is also handled within the client-side application.
    - Basic calculations: The smartwatch application also performs some basic calculations and data formatting.

  The different aspects of the smartwatch application will be explained in depth in the Client-side application section.
- Server-side application: This part of the application is hosted on the cloud and has two main purposes
    - Gathering data: All datapoints sent by the smartwatch are gathered here, together with some other information like the current state of a given device.
    - Analysing data: This is where the predictive model, which takes in the datapoints gathered by the smartwatch and calculates the probability of the user suffering an extended accident in the near future or having suffered a chance accident. Developing this predictive model is outside of the scope of this thesis, so this model has been substituted by a function that provides a random output, following certain pre-specified probabilities.

  The different components of the server-side application will be explained in depth in the Server-side application section.

2.2 *TOOL DESCRIPTION*

In this chapter is presented an outlook of the different tools used for the development of the application.

2.2.1 CLOUD INFRASTRUCTURE

- Amazon Web Services (AWS): After exploring the main options in the market (Azure, Google Cloud and AWS), AWS was chosen to host the server-side application. AWS is the cloud hosting and computing business of Amazon, the pioneer large scale player

in the industry. Today it remains the industry leader with a 33 % market share (Panettieri, 2022). Initially, AWS was chosen due to its great range of custom solutions and free-tier pricing for many of them. Some of the services that were considered for using in this thesis where:

- Managed Relational Database Service (RDS)
- Managed NoSQL Database (DynamoDB)
- Lambda: Platform for serverless computing, where all computations are performed without any infrastructure design or management.
- Lightsail: Service bundling compute power and memory, targeted towards new or inexperienced users, allowing users to launch Virtual Private Servers connecting the different independent components.
- IoT Core: Used for connecting IoT devices to the cloud.
- AWS Amplify: Platform with tools, frameworks and app services for the development of mobile and web apps
- Amazon Elastic Beanstalk: An end-to-end web application management platform, for deploying web applications and services

- However, none of these services were chosen for two main reasons. The first one is flexibility. Most of these solutions are custom made for a given purpose, so they don't allow as much creativity and flexibility in creating a custom solution. However, the main reason is academic. Given how this thesis has an educational purpose, building the full solution from the ground up was thought to be more instructive than configuring and connecting pre-built solutions. That is why a single resource was chosen in the end.
  - Virtual Cloud Servers (EC2): This service provides empty virtual servers with certain computing power and storage, so any solution can be built within them. It is a virtual server – and not a dedicated one – as the users are not reserved a certain physical server for them, but rather Amazon manages the given computing power and storage across different physical servers according to the user's needs. This is a much cheaper and scalable solution than physical servers.
  - The EC2 server was configured with the following parameters:

- OS image and prebuilt applications: Different operating systems are offered by Amazon, such as Amazon Linux, Ubuntu, Windows, Red Hat Linux, etc. Ubuntu was chosen, due to it having the most widespread adoption and thus the best support and documentation, and also because of the WSL. The Windows Subsystem for Linux (WSL) is a Windows tool that allows you to have a Linux subsystem without partitioning your disk. It is very helpful for testing before deploying in the cloud, and its flavour is Ubuntu, so going from test to deployment is easier if the cloud OS is also Ubuntu.
- Instance type: The chosen instance was t2.micro. It consists of 1 virtual CPU with 1 GB in memory – more than enough for the application. It was chosen due to having a free tier of 750 hours per month for a year. After that its pricing is 0.0116 USD/hour for the all Linux OS flavours (Precios de Amazon EC2, n.d.).
- Key pair: A key pair file was generated in order to add an extra layer of security when connecting remotely to the instance.
- Network settings: A new security group was created and the firewall was configured to allow SSH, HTTP and HTTPs traffic from any IP. The SSH was allowed in order to connect remotely when configuring the server and the HTTP and HTTPs for communicating with the web application. With this configuration, the following ports were enabled:
    - Port 22: For SSH connection over TCP.
    - Port 80: For HTTP traffic.
    - Port 443: For HTTPs traffic.
    - Port 3306: This port was later manually enabled for direct connection with the database over TCP.
- Storage settings: As root volume a 30 GB general purpose SSD was chosen, as this was the maximum storage allowed by the free tier
- Advanced settings: None of the advanced settings – allowing for configurations such as having an elastic GPU for intense computation or

CloudWatch monitoring for tracking and metrics gathering – were used, as they weren't deemed necessary for the current thesis.

2.2.2 REMOTE CONNECTION

- PuTTY: PuTTY is, among other things, an open source SSH (Secure Shell) client. It was used to connect remotely to the EC2 instance. By specifying the IP address, the port, the user and the keypair one is able to create a remote terminal and work in the EC2 via de command line interface.
- WinSCP: WinSCP is, among other things, an open source FTP (File Transfer Protocol). It is used to transfer files to other computers over the internet. In this case, it was used to transfer the potential predictive model to the EC2 instance. Later, it was substituted by GitHub for the transfer of files and scripts between the local testing environment and the remote deployment environment.

2.2.3 DATABASE

- MariaDB: MariaDB is an open source relational database developed by one of MySQL founders, which guaranteed it would remain open source after MySQL's acquisition by Oracle.
- A relational database was chosen as the stored data is well structured, allowing for a simple database design ahead of time. Although it is not clear that relational databases have a higher performance than non relational databases even with this kind of structured data (Wang & Yang, 2017), the simplicity of working with relational databases vast outweighs any possible performance benefit, especially for an MVP.
- Within relational databases, MariaDB was chosen over MySQL – the traditional relational database alternative – for a general preference for free open source software.

2.2.4 PROGRAMMING LANGUAGES

- Swift: Swift is the compiled programming language used to develop applications for macOS, iOS, watchOS and tvOS (the different Apple operating systems). The only other alternative for writing these applications is Objective – C, a 30 year old extension

of C, and Swift was chosen due to it being easier to learn, more readable, and the official recommendation by Apple, meaning it has a much better and newer documentation.

- Python: Python is an interpreted programming language that can be used for pretty much anything. It is one of the easiest languages to learn and understand and has a vast number of available libraries that allow it to perform many different tasks. In this case, it was used for the development of all the components of the server-side application.
- SQL: Although not a programming language in the same sense as the previous two, SQL is the language generally used to interact with relational databases. It has been used for all interactions with the MariaDB database, from creating the tables to writing incoming data to reading data from the database.

2.2.5 IDE (INTEGRATED DEVELOPMENT ENVIRONMENTS)

- XCode: XCode is the official Apple IDE for applications development. Although it is technically possible to write these applications without the use of XCode, it can be a lot more complicated as no other development environment has the support and all the tools provided by Apple, which allows you to easily configure, develop and test any Application. There have been 4 main features of XCode used to develop the application.
    - Code editor: This is where the code for the logic of the application is written. XCode also incorporates code completion, debugging and compiling features among others.
    - StoryBoards: This is a graphical interface for the design of the UI of the application. It allows for the creation and modification of the XML file that defines the UI in a simple and visual way.
    - Application configuration: XCode also allows for an easy modification of info.plist, the XML file which holds the configuration to the application. The several modifications that have been made, such as requesting access to read health data, are made a lot easier with XCode.
    - Apple watch emulator: One of the most useful features, XCode has incorporated emulators for different Apple watches and watchOS versions, allowing for easy testing and showcasing of the application within the IDE itself.

- Finally, XCode provides an easy building and exporting of the application.
- VSCode: Visual Studio Code is a semi open source – source code is open under MIT license but the binary code isn't – developed by Microsoft. It has been chosen due to different reasons, the main being that it is really easy to work with, previous experience working with it, the great number of plugins that allow working on pretty much anything with it and its great integration with the WSL – explained within the Testing subsection in this section. It has been used for the development of the server-side application.

2.2.6 FRAMEWORKS

- Flask: Flask is a Python Web Application Framework. It allows for an easy development of web applications that can give responses to the client application's requests. It is contained within a Python library and is thus really easy to incorporate into any code and implement. It has been used in part for this reason, but also because it allows for really fast development. This is mainly because it has incorporated an integrated web server, so just with Flask the whole server-side can be handled.

  It must be admitted that using Flask's integrated is not recommended for production environments. The official recommendation is to use a 3 components combination as described below:

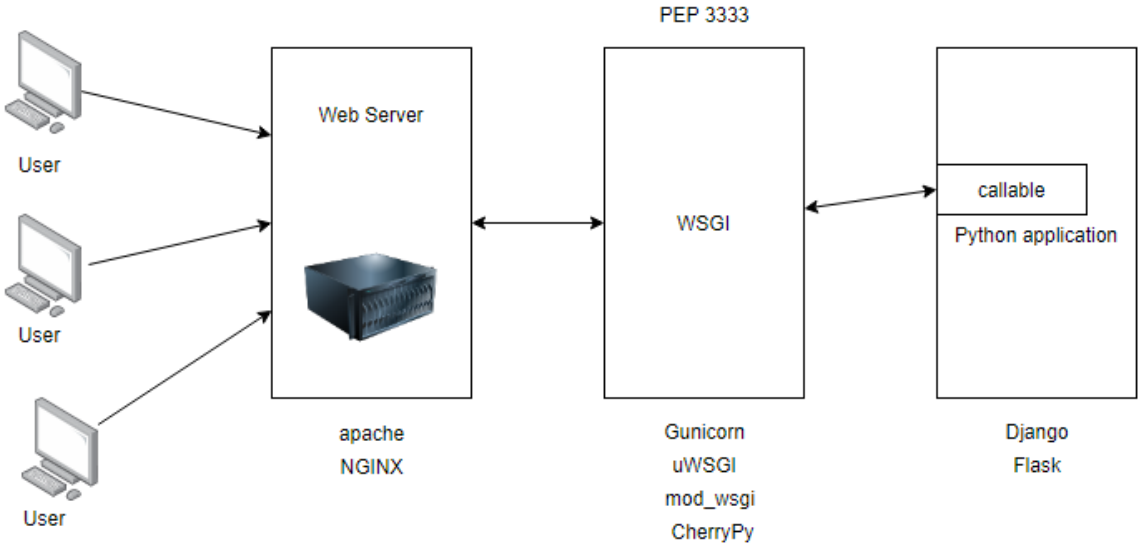

Figure 4: Recommended setup for Python web applications (Stud, 2020)

The three different parts are:

- Web Server: In an oversimplified way, the web server is responsible for receiving user's requests and serving the appropriate file, be it a web page a document or whatever it may be.
- WSGI: The Web Server Gateway Interface is the intermediary that facilitates the communication between the web server and the web application. It is a concept that belongs exclusively to Python web applications, and every WSGI server is designed following the PEP 3333 specifications for a correct integration with Python web applications.
- Web application: This last step holds the logic needed to serve any kind of dynamic content and handles any interactions with other parts of the backend such as the database.

Flask comes with all three parts integrated. And why is the Flask incorporated web server suitable only for a development environment and not for a production environment? There are two main reasons, being one that it does not scale well, as it is only able to handle one request at a time. The second one is that with the debug mode enabled, if an error occurs during the handling of a request anyone can open a shell and execute any code on the server (Blender, 2012). However, for the purpose of building

an MVP this is not really an issue, as the application will not be receiving multiple requests from multiple devices at the same time and it won't be used by anyone that could exploit the security vulnerability.

2.2.7 TESTING

- WSL: Already mentioned above, the Windows Subsystem for Linux is a Windows application that allows any Windows computer to have a Linux subsystem without partitioning the disk. Any development for the web application was first done in WSL, as it was easier to work with than EC2 through putty. All requests were tested in the localhost and only once everything seemed to work the EC2 application was updated. This update was carried out with GitHub, explained below.
- Beekeeper Studio: Beekeeper Studio is a free and open source SQL editor and database manager. It was used to connect to the local database and test all the different queries, being able to see in a lot more detail the different tables, the data contained within them and what the different errors were.

2.2.8 VERSION CONTROL

- Git+GitHub: Git is an open source version control system that allows you to make changes and add features to your code in an organized, being able to track these changes and compare different versions. GitHub is probably the most widely used code hosting and version control platform, built on top of Git. Github makes interacting with Git a lot easier, and makes it easier to synchronised work between different people or different computers. Github was used both for the smartwatch application and the web application. It was vital for both. For the smartwatch application because it was developed using different computers, and thus allowed for it to be stored remotely and to work from the different computers in an organized way. For the web application it was used to streamline the testing-deployment pipeline. Changes were introduced in a local version of the repository. When the version was stable, it was uploaded to Github through Github Desktop. Then, on the EC2 side, updating to the latest version could be

done through a single line command – git pull – which fetched the latest version from Github and merged it with the EC2 repository.

2.3 CLIENT-SIDE APPLICATION

After having seen an overview of the solution as a whole, in this section an in-depth analysis of the client-side application will be explored. First, a comparison of the available hardware options and the criteria for choosing will be presented. It will be followed by an analysis of the general flow of the application through an examination of the UI. Then, the logic and intricacies of the data gathering and sending will be examined, finishing with some special remarks on some topics which are worth mentioning in their own section.

2.3.1 HARDWARE COMPARISON

Nowadays there are multitude of smartwatches and activity trackers in the market, so the first choice must be to decide which one to use, as this choice will determine the direction of all of the application development. For choosing the appropriate hardware, there are two sets of criteria.

- Deal breakers: These are criteria that cover the bare minimum the smartwatch needs to be able to do in order for the application to work as intended
    - Activity sensors: The smartwatch needs to be able to record the required datapoints, of which the fundamental ones are, gathered from conversations with a doctor working in the labour accidents branch of a public hospital:
        - Heart rate
        - Blood pressure
        - Glucose – no smartwatch is able to measure it as of now, but other datapoints can be used to estimate it
        - Blood oxygen saturation
        - ECG (ElectroCardioGram)
        - Solar exposition
        - Number of hours standing/sitting

  - Programmable: Access to these sensor's information needs to be accessible through some kind of API, and an application that can sit in the smartwatch needs to be able to be developed.

- Preferences: There are many more criteria, that although don't make it impossible to choose one smartwatch that doesn't have them, help tip the scale in favour of one or another choice.
  - Official SDK (Software Development Kit): This point is mentioned in relation to the Programmable requirement mentioned before. Some smartwatches don't have an official SDK or API access to sensors, but third parties have developed unofficial ones. Although these could be used, having an official one allows for easier development and more guarantees that it will work.
  - Other sensors: Although the sensors mentioned above provide the baseline of what any smartwatch should have, the more data that can be gathered the merrier. This is so specially with machine learning predictive models, that benefit greatly from having a great number of features from which to extract patterns and predictions.
  - Other properties: There are many more criteria, that although not necessary, would help choose one smartwatch over another. These criteria are:
    - Battery life
    - Price
    - Durability of the materials it is built from
    - Connectivity options
    - Waterproofness
    - Storage
    - Haptic feedback and Audio capabilities

The comparison started with about ten different brands of smartwatches and activity trackers. After an initial comparison of the hardware capabilities of all of them, this list was brought down to 3 main brands: Apple, Samsung and Fitbit. The three of them made it to the shortlist, as they are the only brands that allow for an easy development of native apps with access to

the smartwatch's sensors. An in-depth comparison of the hardware capabilities of some models of these brands can be found in *Appendix I: Hardware comparison of main options*.

The determining factor ended up being the sensors to which developers have access through the official API, because even though the 3 brands have models with more than enough sensors, only one of them provides access to almost all of the information, while the other two only use the data coming from those sensors for official applications. That is why the final choice ended up being Apple.

It is worth noting that aside from the above criteria, Apple would also probably have been the chosen choice, due to the great ecosystem it has built for developers and all the help it provides in the form of great documentation. The main drawback of Apple over the other options is the price of its smartwatches. However, for an MVP development this is not really an issue, and once the proof of concept has been proven to work with the most favourable smartwatch, other less optimal alternatives may be explored. Furthermore, in the Economic analysis section it will be shown how this price is not really an important drawback.

2.3.2 CLIENT APPLICATION FLOW & UI

In this section, the flow of the application will be explored through the different screens designed for the UI.

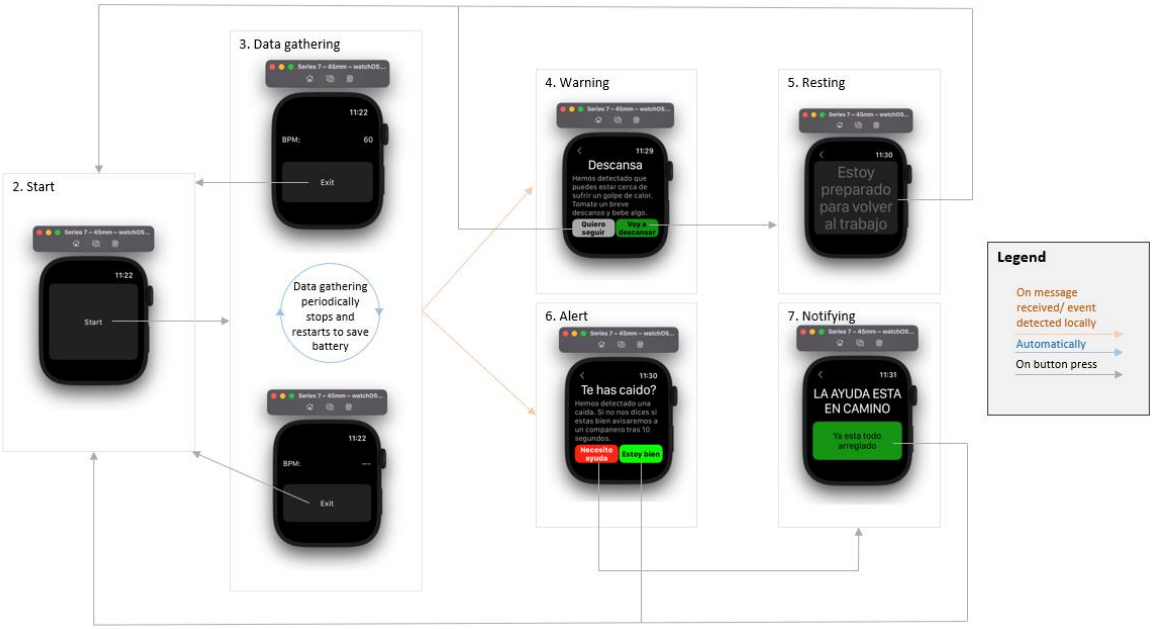

Figure 5: Application UI flow (Permissions screen not shown)

These are the main screens, their purpose and how they interact among themselves:

1. Permissions: This screen is only shown the first time a user goes into the application. It is really two screens: one that requests access to reading ECG data and one that requests access to reading and creating all the health data used by the application. Once the user grants these permissions, they are able the access the application and don't get shown this screen again.
2. Start: This is the first screen of the application itself. It is a very simple one, with just one button. When the user presses the button, the application goes to the next screen and starts recording and sending data
3. Data gathering: This is the screen where the user will spend most of his/her time. In the foreground it presents quite a simple UI, with one Exit button that stops the data gathering and takes the user to the Start screen and a label showing the heart rate in beats per minute. This data is not necessarily useful to the user, but it shows whether the data gathering is in active or inactive mode – more on this in the Data gathering section. In the background however, it is recording all the relevant data and sending it to the server and is constantly reading the emergency status of the device.

4. Warning: When the predictive model calculates a probability over a certain threshold that a given user will suffer an extended accident in the near future, it changes the emergency status of the device to that accident (e.g. insolation). When the application reads this emergency status in the Data gathering screen, the application is taken to this screen with haptic feedback – vibration – for the user to notice the warning. In this screen, a quick recommendation is shown as the title, with more details provided in the text below. The user is given two choices in the form of two buttons. In one of them the user signals willingness to follow the advice, in which case he/she is taken to the Resting screen. However, the user is also given the choice to ignore the warning, in which case the user is taken to the Start screen so he/she can restart the data gathering again.
5. Resting: This is a very simple screen with just a button. It signals that the user is resting due to a received warning. When the user feels ready again, he/she can press the button and is taken back to the Start screen so he/she can restart the data gathering again.
6. Alert: This screen appears when the applications reads that the emergency status of the device has changed to a given emergency – i.e. chance accident, like a heart attack. The appearance of this screen is also preceded by haptic feedback. Similar to the warning screen, this screen shows a main question – e.g. Are you feeling okay? – and a more detailed text explaining why the alert came up. The user is given again two choices. One is to notify a supervisor, and the other is to say he/she is okay, which takes the user back to the Start screen. If there is no recorded interaction for 15 seconds, the supervisor is notified, as in a really bad emergency the user could be not responsive.
7. Notifying: This screen appears when the user requests to notify a supervisor, mainly when an emergency has occurred. In the background, the application sends a message to the server so the notifying routine can be started. In the UI itself, the screen shows two main things. One is a reassuring message, telling the user that help is on the way, and the other is a button to signal to the application that every thing is okay, which takes the application back to the Start screen.

2.3.3 CLIENT APPLICATION LOGIC

In this section, the different parts of the application code itself will be discussed.

2.3.3.1 Code structure

Usually Apple watch applications are built with an associated iPhone application, however the one created here is a standalone watch application. Any Apple watch standalone application is structured in two main folders: the WatchKit App and the WatchKit Extension. The main file in the first one is the Interface.storyboard, the XML file that holds the UI. The second folder usually has two main files. One is the Info.plist, another XML file which holds the application's configuration, permissions and properties. The other one is InterfaceController.swift. This is the swift file that holds the logic for the application itself. In this case, two extra files called UtilHTTP.swift and UtilHealth.swift have been created. These are just for code cleanliness purposes, as they only extend the classes created in InterfaceController.swift. Thus, when throughout this section InterfaceController.swift is mentioned, in reality the function or portion of code could be located in one of the other two files, but InterfaceController.swift will be mentioned for simplicity's sake.

The main code of the application is composed primarily of classes. Each screen in the application is governed by one class, which extends the WKInterfaceController standard class. These are the following classes present in the code:

- InterfaceController: This is the main class and the one that governs both the Start screen and the Data gathering screen, as they are in reality the same screen but with a dynamically changing interface.
  This main class, apart from extending the WKInterafaceController class also extends the HKWorkoutSessionDelegate and HKLiveWorkoutBuilderDelegate standard classes, as they are the ones needed to gather the health Data. The code is split in the following sections.
  - Basic configuration: In here, some general variables and configurations are set, such as the device ID, the server's IP address, the set of possible application states.

- Motion-specific configuration: The object of the class CMMotionManager used for collecting motion data is declared.
- Health-specific configuration: The HKWorkoutSession object and HKLiveWorkoutBuilder objects are declared. The first, represents the workout session in which the health data is collected and configures the sensors for the activity at hand, managing and holding all the workout data. The second is a protocol that monitors live workout builders, which are objects that incrementally construct a workout session based on live data from the sensors. Both of these standard classes and protocol are needed for the gathering of health data. The length of the interval in which the application collects data and in which it rests is set and the dictionary which holds the motion data before it is sent is declared. The dictionary that holds the health data before it is sent and the one holding the ECG data are also declared here.
- Function awake: Method of the class WKInterfaceController which is executed once the interface comes on screen. Here the motion collection starts and a timer is created, which introduces the motion data into the motion dictionary as it comes with a pre-specified frequency. The change from actively recording to not recording health data is also stored here.
- Function willActivate: Method of the same class, it is executed when the interface is about to come on screen. It checks whether the user has given permission to the application to record the required data, and if not, it requests that permission. The device ID is added to the different data dictionaries here.
- Function didDeactivate: Once again, this function is also a method of the same class and it is executed once the view controller – the screen – is no longer visible. In this application's case it is completely empty.
- Function startMotionCollection: In here a queue is created. Queues are data structures that hold a given message from the moment it is sent until the moment the receiver acts on it. The motion data collector is put on this queue for the management of the incoming motion datapoints and it is started.

- Function workoutSession: Required by the HKWorkoutSession class, executed when the state of the workout changes from one state to the other. In the application it is only used to log a notification to the console.
- Function workoutBuilder: This function builds the workout, gathering the different specified health datapoints as they are gathered from the sensors. All metrics and their respective units are specified, and whenever a new measurement comes in – e.g. the temperature sensor records a new temperature – this measurement is stored in the health dictionary. If the gathered datapoint is from the heart rate sensor, it is displayed on the screen.
- Function initWorkout: Required by the HKWorkoutSession class, this function is in charge of setting up the workout configuration – e.g. indoors or outdoors activity –, starting the actual workout session and creating the workout builder delegate.
- Function startWorkout: Function built around initWorkout, which after calling it starts the actual session and data gathering by the workout builder. It also changes the state of the application and the workout session.
- Function stopWorkout: The opposite to the previous function, it stops the data gathering, finishes the session and erases both the builder and the session objects. It also changes again the application's and the workout's state.
- Function requestECG: It queries the watch for the latest ECG data collected in the Health Kit.
- Function getSendECG: This function calls the previous function, and after it receives the queried ECG – if it is available – it introduces the data in the ECG dictionary and sets the asynchronous task to send the ECG dictionary to the server by calling the following function.
- Function postHTTP2: This function is in charge of delivering the post requests to upload data to the server. It starts by creating an URL session – object in charge of coordinating al data transfer tasks – and creating the URL request. This URL request's method is set to POST, and the dictionary that the function takes as input is converted into JSON and added to the request's body. The

JSON is added to both the content-type and accept header fields. The appropriate task is created with the specified URL request, which sends the request to the server and handles the response. The response's JSON is converted into a swift dictionary and it is printed into the console together with the response code.

- Function getHTTP2: Very similar to the above function, but instead of creating a POST request it creates a GET request as the name suggests, and it is not directed towards the /post endpoint used to upload data but rather towards the /state endpoint used to retrieve the device's emergency state. In the response handling, instead of printing the JSON-turned-dictionary, the emergency state is extracted, and if the state is that of an emergency the application changes state and is taken to the relevant screen. This function is executed continuously with a pre-specified frequency.
- Function sendAndSave: This function is in charge of sending the motion and health data with the already explained postHTTP2 function and emptying both dictionaries.
- Function buttonPressed: This function is connected to the screen's only button and it is executed when the button is pressed. Depending on the application's state, whether it is in the welcome, activeWorkout or activeNotWorkout state, the button works in one way or in another. In welcome state, the function goes into activeWorkout state, sends the ECG data, starts the health data recollection process and modifies the UI into the Data gathering screen. In the activeWorkout, the UI is modified into the welcome screen, the application's state also changes to the welcome state and the recorded data is sent to the server and the data recollection is stopped – by finishing the workout. In the activeNotWorkout, the same steps as in the activeWorkout variation are taken, but without sending the collected data as no data is collected during the activeNotWorkout period.

The rest of the classes are much simpler and quite similar to each other, as most of the heavy lifting is done in the InterfaceController class.

- InterfaceControllerWarning: This class governs the Warning screen and only conforms to the WKInterfaceController standard class. Apart from the functions required by this class – awake, willActivate, didDeactivate – it only has two additional functions, each one executed with the pressing of one of its two buttons. The resting button takes the user to the resting view controller while the one indicating the application that the user is going to keep working takes the user back to the Start home screen.
- InterfaceControllerAlert: This class governs the Alert screen and also only conforms to the WKInterfaceController standard class. It also has the same required methods for this class, also empty, and also has two functions for governing the two buttons present in the screen. The resting button is completely equivalent to the one in the previous class. The help button also takes the user to the corresponding Notifying screen, although it also while the one indicating the application that the user is going to keep working takes the user back to the Start home screen.
- InterfaceControllerAvisando: It controls the Notifying screen. Apart from the three standard functions it has just one other function that controls the button that takes the user back to the home screen.
- InterfaceControllerDescansando: Equivalent to the previous class.

2.3.3.2 Data collection

Although the topic of data collection has already been touched upon in previous subsections, this one will dive in depth in the type of data collected, explaining its purpose and collecting mechanics.

To understand what data should be collected, it is important to first remember what the purpose of the application is. As it has already been explained, its purpose is to predict and prevent extended accidents before they happen and to detect and speed up the response to chance accidents. Thus, the application will collect any available data that can help in any of these two purposes.

By further inspecting the most frequent occupational accidents and diseases, practically all of them are caused or can be identified by some motion and/or have some kind of effect in the

health metrics of the employee. Thus, both motion and biometric data will be collected and sent to the server-side application. These will help predict and identify many different kinds of accidents. These should therefore be enough to fulfil that first goal of the application. There is however another goal, which has already been explained, which is that of facilitating and accelerating the medical diagnosis of the employee. After speaking with María Luisa Lagándara, who works in the labour accidents unit in a public hospital in Madrid, she was able to provide a list of routine medical checks that are needed for most diagnoses. The list, already presented in the Hardware comparison subsection, is the following:

- Heart rate
- Blood pressure
- Glucose
- Blood oxygen saturation
- ECG (ElectroCardioGram)
- Solar exposition
- Number of hours standing/sitting

Some of these can be collected directly as biometric data from the watch. Others will have to be inferred from biometric data. There is however, another type of data which doesn't entirely fit with the rest. That is the ECG. For a correct diagnosis an ECG from the current day should be presented, but there is to continuously read ECG data – which isn't allowed by Apple anyways. This will thus be the third type of data, ECG. What follows is an in depth examination of the three types of data retrieved by the smartwatch application:

- ECG: The ElectroCardioGram cannot be written to the Apple Watch Health kit from the application. The application can however read the latest ECG performed by the Apple watch. The application will thus, at the beginning of each working day, read and send the latest ECG to the server.
- Motion data: This data will be used both to detect accidents like falls or getting hit by objects, and to identify repetitive movements which can lead to different kinds of musculoskeletal diseases. All of this data is retrieved with a pre-specified frequency –

currently 5 Hz – and sent in bulk to the application also at a pre-specified frequency – currently every 30 seconds. The motion data retrieved by the application is:

  - Acceleration: The linear acceleration in all three axes. It is read by the built-in accelerometer and automatically cleaned by Apple, so there is no need to retrieve the raw acceleration data.
  - Rotation: The rotation in all three axes. It is read by the built-in gyroscope and automatically cleaned by Apple, so there is no need to retrieve the raw acceleration data.
  - Gravity: From the previous two, Apple automatically calculates the acceleration provided by gravity in all three axes:

- Health data: This is biometric data, either read by dedicated built-in sensors in the smartwatch or calculated by Apple's proprietary algorithms. These metrics are either the ones obtained in the already mentioned conversation, proxies to those that cannot be explicitly retrieved or extra data, that even if it is not necessary for a medical diagnosis, could be really helpful for the predictive model to extract additional patterns and improve its predictions. The different metrics are:
  - Heart rate: Calculated in beats per minute. Read by the optical and electrical heart rate sensor in the watch.
  - Active energy burned: The calories the employee has burned due to physical activity. Although not as good a measure as the blood glucose level, it could serve as a proxy, with higher energy burned relating to lower glucose levels.
  - Basal energy burned: The calories the employee has burned just to maintain the body's proper normal functioning, in a resting state. It depends on measures such as sex, weight, height, etc. And together with the active energy burned makes up the total calories burned. It is also part of the approximation to glucose blood level.
  - Apple stand time: As its name suggests, it is a metric calculated by Apple that shows the total standing time of the employee. This was one of the metrics that were useful for a proper diagnosis.

- Apple walking steadiness: Metric calculated by apple which measures walking coordination and steadiness on a scale from 0 to 1. It can be useful to prevent falls.
- Environmental audio exposure: This metric measures the intensity of the surrounding noises in decibels. This will help with preventing all kinds of hearing related accidents. But it will not be used just to present recommendations. The maximum audio exposure levels are regulated by law *(Real Decreto 286/2006 sobre la protección de la salud y la seguridad de los trabajadores contra los riesgos relacionados con la exposición al ruido*, 2006), being the maximum allowed levels a daily 87 dB – calculated as an integration of the day's decibels – or a peak level of 140 dB – calculated as the instantaneous maximum noise level. If these levels permitted by law are exceeded both the employer and the supervisor could be warned. If the levels are closed to being exceeded the employer could be warned to prevent any future infractions.
- Heart rate variability: This is calculated as the standard deviation of beat-to-beat measurements – what is commonly identified as HRV (SDNN). This measure is increasingly being related to general health, with researches uncovering relations between HRV and anything from a healthy diet (Young & Benton, 2018) to Health-Related Quality of Life (HRQOF) in the physical domain (Lu, et al., 2016). This metric will therefore probably be very useful for the prediction of a range of occupational diseases.
- Oxygen saturation: This metric records the blood oxygen level, also a key datapoint needed for a good medical diagnosis.
- Body temperature: As its name suggests, it records the body's temperature, with a mix between a thermometer at skin temperature and a proprietary algorithm to infer the body's core temperature. It can be used as a proxy to solar exposition or to other infections which have fever as a symptom.
- Blood pressure systolic: This is the maximum blood pressure recorded during a heartbeat, recorded in Pascals.

- Blood pressure diastolic: This is the blood pressure reading at its low point, in between heartbeats. By combining it with the systolic blood pressure the overall blood pressure can be obtained.
- Respiratory Rate: This metric measures the frequency of the employee's breaths, which can be affected by things such as intense physical activity, which in turn can lead to a range of occupational accidents.
- Distance Walked: As its name suggests, it measures the employee's walked distance in meters, which when taken in conjunction with energy burned can give an idea of physical exertion.

2.3.3.3 Edge computing

Any project remotely related to IoT or distributed devices must reflect on edge computing. Edge computing is a trend, driven by the exponential increase in devices storage and computational capacity, in which data is processed in the devices located at the periphery of the network.

This new architecture paradigm has a number of advantages, among which the main are (Bhardwaj, 2021):

- Response latency: By performing all relevant data computations at the source and destination of the data, one is able to skip the trip the data must take from the client to the server and back to the client. This greatly improves the speed of the response.
- Data security: In a normal cloud architecture there are three main points where security could be breached and data could be captured. At the client, where data is being created; in the transmission from the client to the server or the other way around, where data packets could be captured by malicious actors; or in the cloud itself, where hackers could gain access to. In contrast, for edge computing architectures there is a single vulnerable point for most data, the client itself.
- Transmission costs: Sending data to and from the server very often has some costs, like the data plan costs of the device. By performing most computations in the device itself

the amount of data that needs to be sent to the server is reduced, and so are therefore the transmission costs.

Of these three advantages, to one that is really relevant to the proposed solution is the reduction in transmission costs. The response latency is not really as critical, as the latency improvement generally is an improvement from milliseconds to microseconds. It is true that in this case the improvement could be substantially more, as the data from the client is sent in batches, and if the computations were performed in the smartwatch it could be practically real time, so the improvement is from the few seconds that are elapsed in between data batch sending to the microseconds it would take to make the computation in the device. Even in this worst-case scenario, the difference it makes to detect a chance accident or to warn the employee of an extended accident a few seconds early is very rarely significant. And regarding data security, this hasn't been a focus given the MVP nature of the project, so many improvements could be made to improve the security of the whole application before every computation is moved to the edge. Once these improvements are made, it is also important to consider the great advances that have been made in cloud and transmission security, with most technological companies relying on a cloud architecture without significant security issues. Transmission costs are however a significant problem where edge computing could help, as it will be seen in the Financial conclusions subsection.

So with the benefit of reduced networking costs the question remains. Why not move practically all computation to the smartwatch, with its more than excellent specifications by any measurable standard? There are several reasons:

- Data synergies: In an edge architecture each device usually has just its own data – if not, most of the edge computing benefits go away. In the proposed application, the predictive model's accuracy, as most ML models, greatly improves as the amount of training data increases, so having every device's data stored in a central location is needed for training purposes. Even so, one could argue that the cloud could just act as a long-term data storage, receiving data every day, week or month. This however closes the door to many features. Some of them are more vital, like providing the supervisor

with a monitoring platform that can see statistical data of its employees – as long as the employees agree to their data being used for statistical analyses. It also closes the door to some minor features but which could become more important as the predictive model improves, like leveraging data from employees of the same company working on the same activity to make predictions about each of those individual employees.

- Streamlined improvements: If the predictive model is located in each smartwatch, then one must rely on each end-client company to carry out a responsible maintenance of the watches and periodically update the application. If the model is however stored in the cloud, improvements could be made to the model day to day, with every employee benefiting from those improvements without having to make any additional effort.
- Predictive model testing: Along the same lines as the previous point, the predictive model is still very early stage and a lot of questions remain around which metrics will be useful, which won't, and what additional computations and data processing will need to be performed to obtain good predictions. The decision has been thus to have all available data in the server, where it can be aggregated and experimented with, in order to better answer all the current predictive model unknowns.
- Lean development: As it has already been explained in the Tool description and the Code structure subsections, all coding for the client application must be done in either Swift ore Objective C. These are programming languages mainly designed for mobile applications, which are much less versatile than other languages like Python. If the predictive model were to be developed necessarily in these languages, none of the libraries designed for other languages would be available. This would make development of the model itself much harder. But that is not the only development which would be more challenging, as any other task such as long-term storage would have to be done within the watchOS constraints, which is an OS much less general, more difficult to develop on and with less tools than the one used in the server-side application (Linux).

These arguments provide more than sufficient reason to design the architecture as mostly cloud based. There is no need however to completely forego all benefits of edge computing, as a

mildly hybrid architecture is still possible. Specifically, two types of computations have been implemented in the client-side application:

- Fall detection: The detection of falls has also been included as an edge computation, mainly due to it being the most time sensitive metric. To understand its computation one must first understand what a fall is. A fall by definition is a movement whose acceleration is equal to that of gravity in all axes. The detection is thus triggered when

$$\frac{|a_{x,i} - g_{x,i}|}{g_{x,i}} \leq \sigma$$

$$\frac{|a_{y,i} - g_{y,i}|}{g_{y,i}} \leq \sigma$$

$$\frac{|a_{z,i} - g_{z,i}|}{g_{z,i}} \leq \sigma$$

$$\forall i \in F$$

Where $a_{k,i}$ is i[th] measurement for the acceleration in the k axis and $g_{k,i}$ is i[th] measurement for gravity in the k axis. $\frac{|a_{x,i}-g_{x,i}|}{g_{x,i}}$ thus becomes a normalized measure for the difference between acceleration and gravity for a given axis and measurement. This difference would ideally be zero. However, there will probably be small measurement errors and it is very difficult for falls to be a perfect fall. This normalized measure has thus been required to be smaller than σ, a possible error measure initially set to 0.05, although it still needs to be adjusted upon further experimentation. The only item left is F, which is the set of all measurements for which the above inequalities must hold true for the movement to be classified as a fall. The duration of this interval has been calculated through the formulae for free falling objects starting from a resting position:

$$y = \frac{1}{2} \cdot a \cdot t^2$$

Where the t is the length of the F interval. a is the acceleration of the movement, which in the case of a vertical fall is equal to 9.81 m/s^2, and y is the vertical distance traversed by the falling object. It has been set to 1.5 m, which is an estimation of the average fall

distance of the hands of a working standing adult to the floor. Solving for t, the length of the interval turns out to be 0.553 s.

This calculation has been specified through several approximations, and many effects like the involuntary movement of hands when a person is falling are not considered. It nonetheless provides a starting point for the fall detection with which experiment and from which improve.

- Apple local computations: Although not explicitly specified in the application code, the Apple watch is performing many different computations behind the scenes, the output of which is then transmitted to the server. These computations are of two kinds:
    - First order computations: These are the computations that take in the raw signal from the different sensors and output a human-readable metric. This is done for example for all motion sensors or for the heart rate sensor.
    - Second order computations: These computations take in the previous metrics and other static datapoints as inputs and output more abstract metrics, like active energy burned or Apple walking steadiness.

  Although people now take those computations as granted in their wearable devices, they are only possible thanks to the great advances in computing achieved in the last decades. Before that, many of these computations – specially the more abstract ones – would have been impossible to perform at the edge and would have to be taken to the server to be processed.

Although the main computational power is still being used in the server-side application, these first, already implemented use cases provide an insight into the advantages edge computing can provide for the given solution and how to implement these computations. In further versions of the application, the trend will probably be to send to the server less raw data and more summary data, sending the minimum amount of data required for the predictive model to output accurate results.

2.3.3.4 Networking

This section will focus on the external communication aspect of the application. Since the application is limited in its processing power, memory and battery life, data will have to be

stored and processed in a remote server capable of storing a higher quantity of this data and of processing it in a more complex way. Moreover, having all aggregated data in one place allows for better analysis of general tendencies and patterns. However, smartwatch-server communication isn't the only required communication. Since much of the processing is performed in the server, the results of this processing will need to be retrieved by the smartwatch to give the pertinent message to the user. Thus, server-smartwatch communication is also possible.

Different types of communication protocols could be implemented:

- Watch Connectivity: This framework has been designed for two-way connectivity between an Apple watch application and a paired iPhone application. Many apple watch applications are designed as an extension of an iPhone application, where all the computing is performed in the iPhone and the Apple watch is little more than a more convenient UI for the user to interact with the application. This framework is very heavily recommended by Apple and has many advantages such as not having to rely on any WiFi or cellular connection. However, in this case this option has not been chosen as the Apple watch application should be able to work on its own, without having to rely on the user's iPhone. Many of the manual workers towards whom the solution is targeted will not have an iPhone, and providing an iPhone on top of the Apple watch significantly increases the unit costs of implementing the project. Furthermore, adding an iPhone just for connectivity reasons seems like an absolute waste of resources which contributes little more than complicating the whole user experience. (Apple, n.d.)
- MQTT: Message Queuing Telemetry Transport is a communication protocol thought for IoT – Internet of Things – devices, with constrained computational resources and bandwidth. Its communication system is a publish and subscribe system, where any device can post a message on a certain topic and any device that has previously subscribed to this topic will be sent the message. The whole system relies on a broker, which receives all messages, filters them by topic and sends the appropriate messages to the subscribed devices (Santos, 2017). In this application's case, it could be used mainly for the watch application to receive messages relative to changes in its state in

a very resource efficient manner. However, there are two main reasons against the implementation of an MQTT protocol for this use case. The first is that the range of possible messages is very limited, and these messages are always one-to-many and not many-to-many like in many IoT use cases. Thus, the implementation of MQTT does not provide such a simplification as it could have provided in other cases. The second reason, and the one that proved to be decisive in deciding against this protocol, is that the Apple watch currently provides no native support for MQTT. There seem to be some cocoa pods – swift libraries – designed for the implementation of this protocol, however when reading several discussions in different development forums no definitive answer as to whether they would work on a standalone Apple watch application were obtained.

- WebSocket: WebSocket is a communication protocol that provides a persistent fully bidirectional and real time connection between two devices. This seems like the ideal connectivity protocol, as it would allow the server to receive real time data from the watch, accelerating the detection of any accident, while at the same time also providing real time feedback to the watch application. However, this protocol could not be implemented. There is a native class used for the implementation of this protocol called URLSessionWebSocketTask. As it can be inferred from the name, this class is an extension of the URLSessionTask class which was initially designed for HTTP connections. Thus, by not having an independent implementation, this class is quite limited in its implementation of the WebSocket protocol. The main of such drawbacks is the implementation of the message receival process. In other WebSocket APIs in other languages the receive function is implemented in a callback, thus it is called once and it automatically receives all future messages. In this implementation however, probably because of it being an adaptation from the URLSessionTask, the messages will not be received unless the receive function is called, which thus needs to be constantly called (Zenechenko, n.d.). This practically eliminates the persistence of the WebSocket protocol, thus taking away one of the main arguments in favour of its use. However, if this argument was not enough, there is another one that proves impossible the use of this protocol. The URLSessionWebSocketTask is only implemented for iOS

13.0 or higher. Thus, it is impossible for an independent Apple watch application to use it.

- Lower level protocols: These are protocols such as TCP, TLS, QUIC or UDP. Apple has developed a series of classes for the implementation of these protocols at a lower level than with other connectivity APIs (Apple, n.d.). Some of these protocols could have possibly been used, but no significant advantages were found over using the protocol outlined below.
- HTTP: Hyper Text Transfer Protocol is an application layer communication protocol that sits on top of TCP (Mozilla Foundation, n.d.). It is generally used in client-server architectures, where a client application sends a request to the server for some data and the server sends the client application the requested resources, be it an HTML, JSON, JPEG or whatever file has been requested. This protocol was invented in the 90s and is probably one of the most used by individuals in their day to day. It allows for the sending and requesting of practically all types of data, which is the core networking need of the Apple watch's application. But more importantly, it has native support in watchOS, thus making it very easy to use for any watchOS application. Those are the main reasons for why HTTP was chosen as the communication protocol.

Elaborating further on the chosen solution, HTTP uses several request methods that the client can send to the server. All HTTP requests generally have two or three parts (IBM, n.d.).

- Request line: The request line is composed itself by three components
    - Method: Instruction for what operation the client is requesting the server to perform. It could be to fetch some resource (GET), get rid of some resource (DELETE), modify an existing resource (PUT), etc.
    - Path: The URL path of the indicated resource on the server.
    - HTTP version number: As its name suggests, this is just the HTTP version number.
    - [Optional] Query string: In some cases, a string of additional parameters can be specified, which the requested resource can use for some purpose.

    - [Optional] Scheme and host: These components of the URL are sometimes specified in addition to the path, usually when the request is going to go through a proxy server.
- A simple example request line could be: `GET /data/device/6234fe46a65b/status.json HTTP/1.1`
- HTTP headers: These are key-value pairs of information about the message that are given to the server to help it better process the request. Some possible headers are the language in which the client expects the response, what kind of resource is expected, the minimum last modified date of the resource…
- [Optional] Message body: The body contains additional data that may be necessary to process the request. If the client performs for example a PUT request to write some new data to a database, that data needs to be sent in the request. The data would be sent in the message body. The body's data can take different formats, but JSON is a very widely used one.

Of all possible request methods, the two below are the ones used by the application:

- GET: Used to request the server for the current device's emergency state. The request line, apart from the request method – GET – directs the request to the path /status/, as this is the endpoint where the server is designed to show the device's status. However, for the server to know what device's status it must fetch the request needs to tell the server what the requesting device is in some way. That is where the optional query string comes in, attaching a key-value pair that transmits the device ID. The full URL to which the request is sent would thus be `http://{{IPaddress}}/status?device_id={{deviceID}}`
  The server answers by presenting a JSON response, which the client application reads and transforms into a swift dictionary, proceeding differently depending on what the response was.
- POST: Used to post the data collected by the watch to the server. The structure is quite similar to the GET request, but the method instead of being GET is POST and the path is /post/. In this case, two key-value pairs are added to the header, which are "Content-

Type: application/json" and "Accept: application/json". The Content-Type header, which sometimes is provided by the server's response, in the case of a POST request indicates to the server how the data is being sent. The Accept header indicates to the server what data type the client is willing to accept as a response. Another difference is that in this case the request contains a message body, which contains the data that needs to be uploaded in JSON format. The last difference is that in this case, the device ID is not added as a query parameter, but is rather included within the sent data.

2.4 SERVER-SIDE APPLICATION

After addressing the client-side application, half of the complete technical solution has already been explained. However, the server-side application is also fundamental for the correct functioning of the whole solution, thus its explanation is fundamental for the understanding of the complete solution.

2.4.1 GENERAL SERVER-SIDE DESIGN

Throughout previous sections, many characteristics of the server's architecture have already been explained. In order to avoid being too repetitive, below is presented a very brief recapitulation of the points that have already been explained.

- Physical server: As explained in the Cloud Infrastructure subsection, the chosen infrastructure for the server has been AWS. The specific choice has been a Virtual Private Server (EC2), which does not live in a specific machine but rather has allocated a certain computing power and storage that is distributed by Amazon among its available physical machines. This virtual server, even though it is not a real machine, has its own IP address which can be used to access it, by the client-side application or by anyone trying to connect remotely to work on the server.
- Server architecture: As explained in the Frameworks section, the chosen server architecture is a WSGI architecture, specific to Python. This application has three parts, which are the Web server, the WSGI interpreter and the Web application. In this case the three parts are encapsulated in the Flask module for python, which allows for an

easy setup of HTTP endpoints within a Python application – which serves as the Web application. These endpoints are different URL paths which have different services associated to each one.

Now that the first points already pointed out in previous sections are clear, they can be developed and the server can be examined in further detail.

2.4.1.1 REST API

The whole server-side application has been designed as a REST API. REpresentational State Transfer API architectures are standard architectures that allow applications to access resources of a different application over HTTP. Note that "resource" is an abstraction of the information contained in the server. It could be an HTML document, a picture, a dynamic service or a set of other resources for example. This architecture was chosen due to its flexibility, allowing for a very diverse set of communications and operations between both applications while being relatively easy to implement. To understand a little bit better what differentiates a REST API from other architectures, here below are the main design principles (IBM Cloud Education, 2021) (Gupta, 2022):

- Uniform interface: All requests for the same resource should look the same, independent of the source. When a given resource is requested, that resource should be located in only one place, so one resource must be connected to a unique URI – Unique Resource Identifier – and this is the only piece of information that should be known by the client application, driving the server application the rest of the resource fetching.
- Client-server decoupling: Both applications must be completely independent of each other. The client application should only interact with the server through the HTTP request, and the server must not modify the client application in any way, it should limit itself to delivering the requested data.
- Statelessness: A single request should contain in itself all the required data to deliver the resource. The whole processing is performed during the request, and after it no data is stored of the performed request in the server.

- Cacheability: When possible, resources should be cacheable – stored for later use without having to perform the request again. This cache could be done in the client application, saving the need for the request entirely, or in the server application, allowing for faster access to frequently used resources and better performance.
- Layered system architecture: It must be assumed that the client and server do not connect directly to each other, as the requests can go through many layers such as proxies before reaching the final destination. This possibility should be addressed such that the client and server application do not know and do not care if they are communicating with the final application or an intermediary.
- Code on demand (Optional): In some cases, apart from sending static resources, REST APIs can answer a request by sending a piece of code. In these cases, this code should be ready to be executed by the client application with prior configuration or the need for other resources.

Of all the aforementioned principles, only the first three have been key in the designing of the REST API. Due to the simplicity of the server-side application, the three later ones have been ignored. Regarding cacheability, the information that can be requested by the client application is really simple and always the same, which is already stored in a very accessible location, so caching information in the server side is not really necessary. And the client application needs to act on the received information in real time, discarding it later, so no caching is needed in the client side either. Regarding the layered system architecture, the communication is directly between the two application without any intermediary layer, so this point hasn't been considered either. And finally, regarding the code on demand, all information served by the server is static, so this point hasn't had to be implemented either. However, the first three design principles have been followed, resulting in a uniform, decoupled and stateless API.

2.4.2 WEB APPLICATION

Now that the general design principles of the server-side application have been presented, it is time to dive deeper into the actual implementation of the Web application. This is the part of the server implementation responsible for, after receiving a request, performing the relevant

computations and if needed serving the requested information. The whole application has been written in python, as it is a very versatile language that can perform all needed tasks within the same script.

The Web server functions have already been explained. The library Flask can be imported into the Web application script and handle all these functions, together with the WSGI interpretation. Flask makes the whole process really easy in mainly two ways.

First, the creation of the whole web server/application is as simple as creating an object of type Flask. Starting it then is done with a single line of code, by running the run method on the application object. This method takes in the following parameters (Pallets, n.d.):

- Host: The hostname to listen to. In this case set as "0.0.0.0" to allow for external communication.
- Port: The port of the server. In this case it is set as "80", as this is the usual port used for HTTP communication and the one that has been opened in the server's firewall for HTTP traffic.
- Debug: Sets the debug mode. When the debug mode is activated, all errors are logged to the screen, as are all requests done to the server, and the server automatically restarts when the Python file is changed and saved.

The second Flask feature that makes it ideal for the quick setup of a web application/server is how easy it is to configure new endpoints. This is done through function decorators. Python decorators – recognised because they start with a @ sign – are functions that take in other functions as one of their parameters, modify it, and output the modified functions. Flask's application object has a *route* decorator, which apart from the function takes in other parameters. The two that have been used are:

- Rule: A string representing the path in which the resource is to be located.
- Methods: The HTTP request methods the resource expects.

The decorator is then located above certain function, and when a request comes in to the path specified in the rule parameter, that function is automatically called and the request is passed

on to it. After the function performs the necessary computations, the object returned by the function is the response the resource outputs. Just to showcase how simple everything is here is a minimal example of how this is actually implemented:

```
@app.route("/some-path",methods=['GET','POST'])
def foo():
    //handle the request, stored in the "request" object
    return //the response the resource should output
```

2.4.2.1 Available resources

Now that the code architecture of the Web application is understood, it is time to showcase the actual resources that have been implemented in the application. These resources will be presented by their URL path:

- /: The base URL, it just outputs a line of text with a simple welcome and indicating what the available endpoints are.
- /db: It outputs an array containing all the tables present in the database. This was mainly used during development to test the connection to the database.
- /post: This is the endpoint to which the smartwatches must periodically upload all their data. The data is uploaded in JSON format in the message body. In that same message, apart from the data itself, there is some other parameters like the type of data that is being sent or de device ID of the smartwatch that sent the data. The function reads the data type that is being sent, formats the JSON to transform it into a table like structure that can fit the database structure and writes the data onto the table corresponding to the data type, together with the device ID. The function the reads the status of the device and responds with its current status, together with a message in which it points out the additions done to the database.
- /alert: This is the endpoint used by the client application to request that a given device's user's supervisor be notified of an emergency. This notification works through a POST request in which a JSON containing the key-value pair of "device_id" and the device's ID is sent. The relevant function then reads this ID and calls the function responsible for notifying the supervisor.

- /update-emergency: The function located at this endpoint is responsible for changing the emergency status in the server's database when an emergency is detected in the watch. It accepts POST requests, in which the emergency code and the device ID are uploaded, changing then the emergency status in the devices table in the database
- /status: This resource is responsible for providing the client application with the emergency status of the requested device. Unlike many of the other endpoints, this one is prepared to accept only GET requests. The GET request must have attached a query with the device ID as a parameter. The resource's function then fetches the appropriate device's emergency status and responds with the requested emergency status in JSON format.

2.4.3 DATABASE

The database is the collection of stored information the application uses to function. It is located in the server. As it has been already stated, the selected database model has been MariaDB. It is a widely used relational database very similar to MySQL. It has been installed in the EC2 and worked on directly from the PuTTY remote terminal, after testing the queries in the local Beekeeper Studio. Note that for any configuration to be performed, or to make any query to the database, the database server must be started any time the AWS instance is started.

As for how it is operated on from the Web application, MariaDB has a Python library that allows for easy manipulation of the database. To work with it in this library, one must first create a connection to the database server. In order to create this connection, one must specify the following parameters:

- User: The user who will perform the operations on the database. A user called fcelaya was created, as a security measure in order to use the root user as little as possible. This user was given all the relevant permissions to read and write to the database.
- Password: The user's password, needed to authenticate it.
- Host: The host where the MariaDB server is located. In this case, since both the script requesting the connection and the database server are located in the same machine, the host is always "localhost" or "127.0.0.1".

- Port: The port where the connection will be attempted. The MariaDB default port, which has remained unchanged, is 3306.
- Database: The database to which the connection must be made. The database created for the application is called “prl”.

After the connection has been tried and successfully accomplished, a cursor belonging to said connection is created. The cursor is used both to execute queries and to manage the output from these queries. Queries in SQL are passed as strings and executed, and the results, obtained in a tuple, are then processed as needed. After retrieving all the desired data, it is important to close both the cursor and the connection, because if all connections are left open this will damage the database performance over time.

2.4.3.1 Tables

In any relational database data is stored in tables. Tables have columns, each of which represents a data dimension – e.g. in a table storing product information tables would be price, colour, volume, etc. – and rows, each of which represents an entry. The tables that have been implemented in the prl database are:

- Devices: Used to store the state and emergency state of all devices. The different columns are:
    - ID: Used to identify the device, specially useful if it needs to be referenced from other databases. It goes from 1 onwards, it must be unique and is created automatically when a record is created.
    - Device_id: A string that identifies the device, must also be unique.
    - Status: A string representing the status of the device – active, inactive, emergency, etc. Currently it is not in use
    - Emergency: What has been called emergency status throughout the thesis. It is the code of the emergency state of the device, being 0 the code for no emergency.
    - Last_update: The moment when the last update on the record was performed. It is updated automatically anytime a change is made to a record.

  - Created_at: The moment the record was created. It is added automatically when a record is created

- ECG: Used to store the ECG records of all devices. The different columns are:
  - ID: Used to identify the ECG record. It goes from 1 onwards, it must be unique and is created automatically when a record is created.
  - Device_id: A string that identifies the device which uploaded the ECG, must also be unique.
  - Ecg: A BLOB – Binary Large OBject – containing the ECG
  - Device_time: The client application records the time when the ECG is retrieved and sends it in its POST request. That date and time is stored here.
  - Created_at: The moment the record was created. It is added automatically when a record is created
- Health: Used to store all the health data retrieved by the smartwatch's sensors. The different columns are:
  - ID: Used to identify the health data record. It goes from 1 onwards, it must be unique and is created automatically when a record is created.
  - Device_id: A string that identifies the device which uploaded the health data, must also be unique.
  - (Multiple) Health metrics: Each health metric has its own column – e.g. heart_rate, active_energy_burned, body_temperature, etc.
  - Device_time: Currently the client application does not send the time at which a health metric is recorded, as every metric is recorded at a different time but metrics of different types recorded at different times are stored in the same row. However, it was created in case this implementation is changed in the future and the record time starts being received from the device.
  - Created_at: The moment the record was created. It is added automatically when a record is created.
- Motion: Used to store all the motion data retrieved by the smartwatch's sensors. The different columns are:

- ID: Used to identify the motion data record. It goes from 1 onwards, it must be unique and is created automatically when a record is created.
- Device_id: A string that identifies the device which uploaded the health data, must also be unique.
- (Multiple) Motion metrics: Each motion metric has its own column – e.g. accx, accy, accz, gyrx, etc.
- Device_time: In this case, as all motion metrics are recorded at the same time, the client application does record the time a record is read and sends that datetime in the request. This column then, the same as with the ECG, records that time.
- Created_at: The moment the record was created. It is added automatically when a record is created.

2.4.4 PREDICTIVE MODEL

The next key part of the server-side application is the predictive model. It is important to note that implementing this part of the application was never in the scope of this project, as the creation of an accident prediction model worth mentioning could easily be a whole thesis in and in of itself. A general overview of what a production level predictive model will however be presented, in order to provide a complete view of the application.

The predictive model is the algorithm tasked with interpreting the data received from the client-side application and calculating the probability of something happening or having happened. In order to understand better the predictive model its requirements will be examined, followed by lifecycle, actual implementation and experimental implementation.

2.4.4.1 Requirements

The first step before developing all the model's details is to understand what it must accomplish and what form it may take. Before going into further detail into the actual implementation of the model, a black-box interpretation can be made, where the inner workings aren't discussed, just its relationship with the outside world. These are the general requirements of this black box.

- Output requirements: Starting with the end, what exactly should the model provide us with. Given what the solution aims to achieve, the output must be a vector containing the probability of different events having happened – for chance accidents – or being about to happen – for extended accidents.
- Input requirements: To reach those outputs, the model must take in some data from the user. This input data will take different forms:
    - Real-time time series data: This will be the latest health and motion data
    - One-off time series data: This will be the last ECG obtained
    - Static data: This will be personal data such as sex, height or weight
    - Processed data: Probably, feeding in some of this raw data alone won't be enough. There probably will have to be some feature extraction and pre-processing for some of the time series data.
- Other requirements: Apart from these requirements relative to the way the model interacts with data, there are other general design and execution requirements:
    - Modularity: As time progresses, new pathologies and accidents will probably become more relevant and will need to be included in the set of the model's predictions. This increase in the outcome set shouldn't require a whole redesign of the model. The same can be said about adding new inputs.
    - Quick processing: The model must perform its predictions in a close-to-real-time way, as speed in reacting to its predictions can sometimes increase the chances of successfully avoiding or mitigating the effects of an accident. Furthermore, the server-side application will have to process the information from thousands of devices and provide real time feedback. If the computation needed to output the prediction is too-time consuming it will either really limit the number of clients the application can have or really drive up the infrastructure cost.
    - Upgradeability: As more data is gathered, the model is expected to improve over time, providing more value to the users.

2.4.4.2 Model lifecycle

Now that the abstraction of the model has taken shape, a walk through what the general lifecycle of the model will be presented. Some of the general steps presented when talking about Machine Learning models (Visengeriyeva, Kammer, Bär, Kniesz, & Plöd, n.d.) such as Business problem framing are skipped, since they have already been explained throughout the thesis.

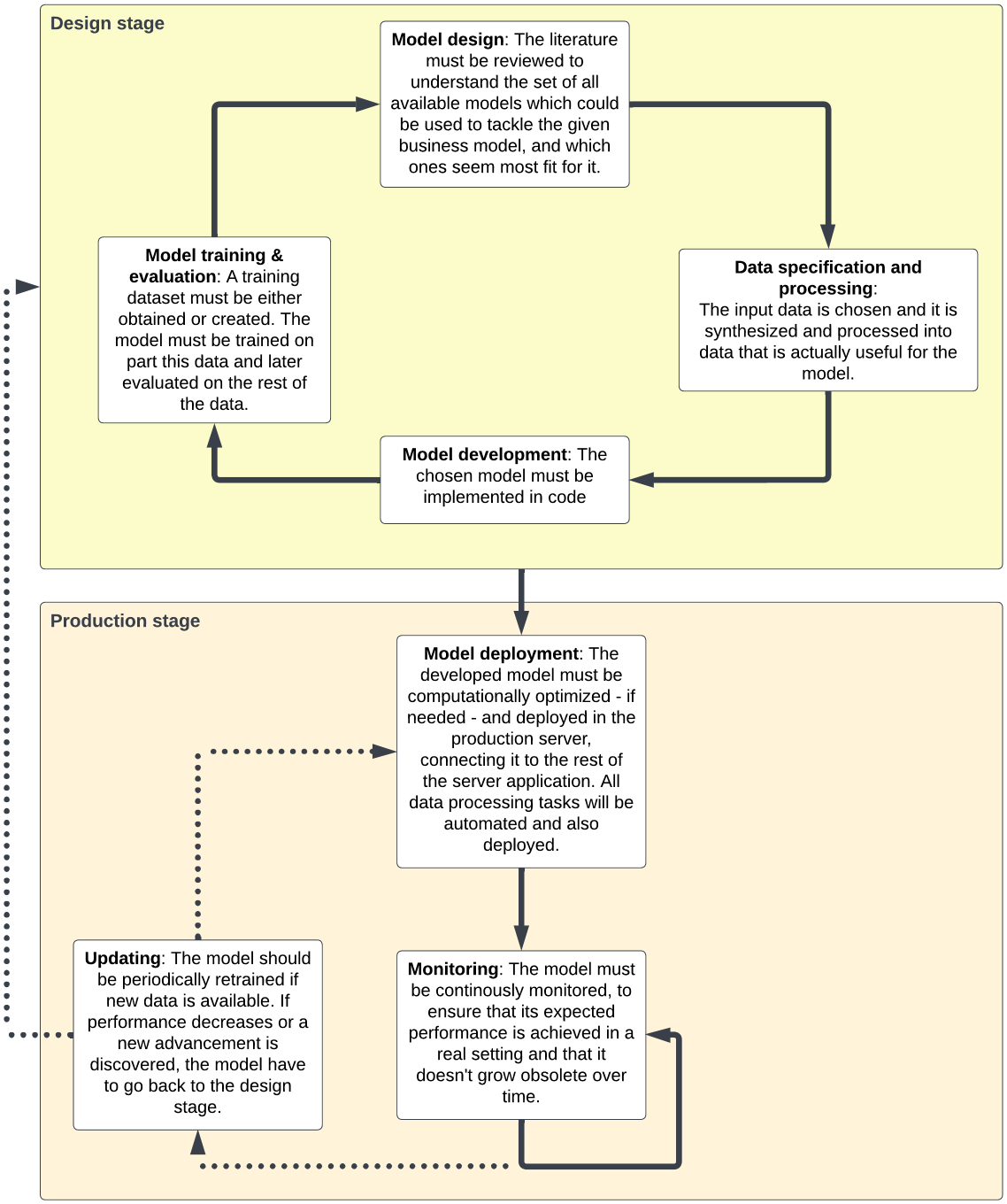

Figure 6: Predictive model lifecycle

There are two important remarks to add to the diagram above when talking about the specific implementation for the server-side application:

- Location: Different parts of this process are to be performed in different locations. The whole design stage flow is to be performed in a local environment, probably with the use of prototyping tools such as python notebooks. The only part of this flow that could be performed remotely is the model training step. If the local CPU is not powerful enough and hardware acceleration is not available, performing this step in online platforms such as Google Colab or a Kaggle notebook could really speed it up. In this case, using the AWS EC2 instance probably would not be of much use as its computing power is less than most computers. This means that for model retraining in the Updating step, this probably would have to be done also in one of the previously stated options – local machine or online platform – and later uploading the updated model to the production environment, which is the EC2 instance.
- Model tuning: The explanation provided in the figure above is completely superficial since the focus of the thesis is not ML models. There is however a specific part of the ML process which is not showcased in the more general diagram above, but which should be mentioned. That is the process of model tuning. In every model, there are many parameters which can be, and many times there is no right solution to how to determine them. These parameters can be anything from the number and size of layers in a Neural Network to the probability thresholds which will be explained in the Status updater subsection. There are many techniques that can be used for optimizing these parameters. And even though it is taken for granted that these techniques will be used for the parameters inherent to the model design – layer size in the example above – it was worth noting that these techniques would also be leveraged for parameters outside of the model "black box".

2.4.4.3 Implemented model

After the general overview given above, it is time to dive into what actual implementation has been done, and it is nothing like the explanation before. The predictive model part has only been implemented to allow for all other parts to work, so the simplest possible version of a model has been implemented. This model takes in a user's heart rate and outputs a random output between 3 possibilities: Everything is fine, Risk of heatstroke and Fall detected. Even

though fall detection has already been implemented in the watch itself – see the Edge computing subsection – it was included here just as an example of a chance accident. The probabilities of each of these three random outputs are specified by the user.

2.4.4.4 Experimental model

Even though creating a viable predictive model was never part of the scope of the thesis, an attempt was made to deploy something as similar to a real model as possible. This model was kindly provided by Juan Gómez Lagándara, who agreed to participate in the thesis as an external collaborator.

Of all possible models, the one chosen was a collaborative filtering label consistent Restricted Boltzmann Machine (RBM). The model was a direct implementation of the one presented in a recent paper (Verma, Patel, & Majumdar, 2019). Here are some basic definitions.

- RBM: The RBM is a specific type of Neural Network configuration. It is generally composed by just two layers: an input layer and a hidden layer. These layers are connected by a matrix of weights – each with a vector of biases – and the connection is, unlike in other Neural Network configurations, made in both directions. Also, unlike other NN architectures, this model isn't trained through backpropagation. The training generally uses a gradient-based maximization method to maximize the probabilities that the set of inputs are generated by the RBM (Deb, 2018). This is done in two steps. In the first one – called Gibs sampling –the values of the hidden layer are predicted with the visible layer values and the probability that a given visible layer neuron is one given the hidden layer. Later, the values of the visible layer are predicted with the calculated hidden layer vector and the probability that a given hidden layer neuron is one given the visible layer. The next steps – called Contrastive divergence – updates the weights matrix with the difference in probabilities of the old and new visible layer vector. This architecture allows for many supervised and unsupervised tasks such as regression, classification, collaborative filtering, feature learning, dimensionality reduction and topic modelling.

- Collaborative filtering: Collaborative filtering, in the ML sense of the expression, is the process of making automatic predictions about the interests of a user by collecting preferences information from many users. (Collaborative Filtering, n.d.)

The model from the paper aimed to solve the movie recommendations problem, where one must find out what movies to recommend to users. This is a classic collaborative filtering example, where the rating of a user for a movie he/she has not seen must be predicted from the ratings of other users. In this case several visible layers are added, encoding user occupation, user gender, user age and/or movie genre, depending on the model being tested.

Although the prediction problem presented in this model is not a collaborative filtering problem as such, it can be approached as one. In this case the different accidents would be equivalent to the movies, and the user's static metadata – sex, weight, height – as well as dynamic metadata – motion, health and ECG – could be added as visible layers to the model. The dataset of available ratings would be equivalent to a dataset of available diagnosis, where a given user with its relevant metadata has a probability of 1 of suffering from the diagnosed pathology or having suffered the diagnosed accident and a probability of 0 for the rest.

There are of course many differences between both cases. In this case, the set of available pathologies/movies is much smaller than the one in the paper's dataset. On the other hand, the set of inputs is much bigger and much more complex, featuring different types of more complex data such as time series data. Furthermore, if the equivalence is implemented as explained in the paragraph above, there would only be 1 sample per user. In the paper, each user had a set of metadata related to it which did not change, and that same user made many ratings. In this case, there would only be one user per accident in the dataset. First because it is unlikely that a given person is diagnosed a great number of times, and second because even if that were the case, in each of the diagnosis the metadata would change, so the samples would belong to different users for all effects and purposes.

It is unclear how these changes would affect the accuracy of the model, or even if it would at all make sense to implement it. However, as it has already been reiterated throughout the

section, this was only an example of what a final model may look like, in order to understand how it would behave inside the server-side application infrastructure.

As for that integration in the EC2 instance, the model was provided by Juan Gómez Lagándara as a Python class. This class had all relevant methods to create the RBM with the specified parameters, train it, output predictions, etc. The mathematical RBM itself was created with Numpy and Theano, two mathematical Python libraries used for the manipulation of vectors and matrices, evaluation of mathematical expressions and optimization.

The train-test dataset from the movie recommendation paper was also provided, with a set of experiment configurations – with different RBM parameters and datasets – provided in JSON format. Everything was uploaded to the EC2 instance, and one of the experiments was run to guarantee that the instance specifications were enough memory-wise to handle the training of the model. The model was created and trained successfully.

The initial aim was to adapt this model to the labour risk prevention application and to train it with relevant data, at least in a small subset of possible accidents and using a subset of the available data. In order to carry out this training, Juan was in talks with a Spanish hospital to start a collaboration so they could share the medical data for research purposes, so the modified RBM could be trained. However, due to the bureaucracy of the whole process, the hospital was not able to provide the data in time for the delivery of the thesis. That is why the random predictive model was finally implemented.

2.4.5 STATUS UPDATER

The final missing part for the whole server-side application is the status updater. Thus far three different components have been examined. The database, responsible for storing all the data; the Web application, responsible for interacting with the client application by writing the data it receives to the database, responding to the web application's requests with the pertinent data and calling the necessary functions – e.g. the one responsible for alerting the supervisor –; and the predictive model, capable of calculating the likelihood that a given device's user is going

to suffer an extended accident or has suffered a chance accident. There is however one missing part.

The predictive model is a black box that takes as input the data received by a device and outputs the likelihood of different events happening or having happened. This black box however needs to be fed the relevant data, and its output also needs to be managed. That is where the status updater comes in.

The status updater is an independent Python script which is continuously running and works as follows.

1. It creates a connection to the database
2. It retrieves a list of all devices from the devices table. Ideally, it would retrieve a list of all active devices. However, since that information is currently not being synced between the client-side application and server-side application, there is no use in retrieving all the devices marked as active in the devices table.
3. It iterates through all devices carrying out the following tasks:
    a. It retrieves all the store data from that device. Probably the oldest data is not necessary, but since the predictive model has not been completely developed, there is no way to know from how long ago it needs data, so all of it is retrieved. In the current implementation only the heart rate is retrieved, since again this does not make a difference for the current predictive model, however changing it to retrieving all data is very straightforward.
    b. This data is fed to the predictive model, and the response is received.
    c. Currently the response is a single state, so that response is written to the database. In the case that the response was a vector of probabilities, this vector would be contrasted with a given set of rules and the conclusion would be written to the database. The set of rules would be mainly of three types:
        i. Thresholds: A minimum probability should be set for each pathology/accident, so that outputs below that probability are not considered. For example, a probability of 60 % of suffering dehydration

is probably not enough to alert the employee but one of 60 % of suffering a stroke is. These thresholds would be completely dependent on the specifics of the predictive model and would ideally come from some type of trial, with people with medical expertise involved.

ii. Priorities: In case that two pathologies/accidents are above their threshold, of which one should the employee be alerted first?

iii. A mixture of both: Blindly applying priorities is probably not the best idea. For example, a probability of dehydration of 98 %, with respect to a 55 % threshold, is probably more urgent than a probability of suffering a stroke of 53 %, with respect to a 50 % threshold, even though as a general rule of thumb the stroke has higher priority over the dehydration. Thus, a mix of multiple thresholds – i.e. mildly probable, probable, very probably, urgent, etc. – with different priority relationships to other pathologies' thresholds would have to be set.

4. The staus updater closes the connection and cursor and sits idle for a pre-specified time interval, before starting again.

As it can be seen, the status updater works independently of the Web application and the predictive model, helping tie both of them together and keep the whole application running smoothly.

Since it is a script independent of the Web application, but which always needs to run alongside it, a shell script has been written that runs both files, starting the status updater in the background so the different requests and debug messages of the Web application can be seen in the console.

Chapter 3. ECONOMIC ANALYSIS[1]

Since the beginning, one of the key aims of this thesis has been to not only develop a technically interesting solution from which to learn different topics, but also to create an MVP of an economically viable product. This section is dedicated to analysing the economic viability of said solution, considering the regulatory environment and the technical specifications already covered.

This analysis will follow several steps, studying qualitatively first the value proposition of the solution for the different involved stakeholders, proceeding to a basic quantitative financial analysis and finishing with a summarized business development plan.

3.1 STAKEHOLDER'S VALUE PROPOSITIONS

This subsection is very tightly related to the whole regulatory analysis performed in the first section of the thesis. In that section, the main stakeholders were presented together with their regulatory obligations. Here, these obligations and some other considerations will be translated into tangible value propositions, which will help understand why anyone would be interested in this solution. The different stakeholders are

- Employer: There are two main direct interests the employer would have in implementing the solution
    - Fluid business activities: As terrible as suffering an illness or accident is for the employer, it also harms the employer. When an employee leaves for a period due to a labour accident, the business activity usually carried out by him/her is necessarily affected. Even if not a single euro comes out of the employer's pocket, the unattended work position will have to be filled by someone else with

[1] During this whole section, "PRLApp" will be used to refer to the company that would be built around the application and would commercialize it

more tasks making them less productive, or will be left unattended thus harming the proper functioning of the company. By reducing the frequency of labour accidents, the employer will better preserve the correct functioning of its business.

  - Employee morale: In risky activities where accidents are not too uncommon, the presence of these accidents can have a great impact on the morale of employees. When several companions have recently been injured due to the same activity that you are supposed to perform, it is normal for you to be less eager to perform said activity and thus become less productive. Taken to the extreme, this morale problem can also affect turnover rate and hiring challenges. By reducing the number of accidents this problem is also ameliorated.

- These two are the most important consequences of implementing the solution for the employer, but there are many other lesser secondary value propositions, some of which are:
  - Easier capital raising: By demonstrating that he/she cares about employees, it will probably be easier for the employer to raise capital from investor, particularly from those concerned with ESG metrics, trend which has recently sped up a lot in the investing world (Visram, 2021).
  - Accidents analysis: By having more detailed data about accidents and how they happen the employer could better understand why they are happening and elaborate a better labour risk prevention plan.
- Employee: For employees the main value proposition is also very obvious. By using the solution, their probability of suffering an accident will decrease, and the effect of accidents will be less severe when they do happen. It is however important to acknowledge that this value proposition will be very probably be faced by scepticism of equal or bigger magnitude. Employees could perceive the option of using the solution as a fake privacy vs. safety dilemma, where they must give up the first to receive the second. If this is the case they may refuse to participate, due to concerns of being monitored by their employer or third parties. It is very important to communicate how the data is only shared with prior consent, to health professionals and for their benefit,

and that their employers or any other third party will under no circumstances have access to their individual data.

- Mutual insurance companies: Recapping, these are the companies that receive a set amount from the social welfare system according to the activities of the employees of the companies they are insuring, and if any of these employees suffer an accident they are in charge of paying them the appropriate compensation. Economically, these companies are the ones that stand to win the most because of two main reasons:
    - Fixed money supply: The money the social welfare system gives them is set by law and does not depend on their individual number of incidences. It is set in accordance to each industry's standards. This means that if the mutual company accomplishes to reduce the number of accidents of its insured companies, it will be getting the same money from the social welfare system but paying out less money in claims. Revenue will remain the same while costs will go down. It is true that mutual insurance companies are forbidden by law to distribute benefits, so that reduction in costs will not result in a direct increase in dividends for shareholders. It will however give the company a lot more money for multitude of other purposes, like increasing its employees' wages, investing in R&D, acquiring new customers, improving other business lines…
    - Scale: For a medium sized individual company it can be quite hard to see the benefits of implementing the solution. If the average number of accidents per year is 4, that number going to 2 or to 6 could be due to pure mere luck independently of the lowering in accident probability provided by the solution. Mutual companies however insure thousands of employees of multitude of different companies. That means that small changes in the probability of suffering an accident are carried a much higher number of employees, making its effects a lot more evident and predictable.
- Prevention services: Paradoxically, prevention services probably have the least attractive value proposition from this labour risk prevention application. That is because, going back to the legislation section, almost all of the activities performed by these services – be it third party services or internal services – are *a priori*. They are

more concerned with the planning of risk prevention, training of employees, etc. thank with the actual execution of these plans. And they also have very little ownership over the actual results in labour risk prevention. The most interesting value proposition that could be presented is one that has already been explained when talking about the employer's value proposition. By having a much better compilation of accident data, these companies would probably be able to better understand the particular risks of an industry or of a type of company and thus improve their prevention planning and other activities in which they help companies. And similarly, to mutual insurance companies, by working with multiple companies they would be able to better leverage these small advantages and create synergies from the data compiled by their different clients.

- Health services: The advantage this solution brings to health services is twofold. On the one hand, they are better to take care of the injured employee. By receiving detailed information about his/her health state and about how the accident happened they would be able to speed up the treatment of the patient. And by improving the detection and response speed, it is much easier for them to reach a beneficial outcome. The implementation of this solution in some of its potential patients will however not only improve the prospects of these patients but also of all the rest. By receiving detailed information even before the patient reaches the hospital, they accelerate the triage process – assessment and prioritizing of some patients over others according to the severity of their problem – and are able to better assess its risk and manage the whole set of patients.

3.2 BUSINESS MODEL

After understanding what the different stakeholders can win from adopting the proposed solution, it is time to formulate a specific business model which can later be analysed more in depth.

The chosen client for the product is the mutual companies. They have been chosen for several reasons

- Available capital: Unlike the final employer, which has a primary economic activity and the labour risk prevention is for them one of a thousand administrative tasks, labour risk prevention is part of the key activities of the mutual insurance companies. That means they will be much readier to invest in it than the final company, which has a whole range of more urgent potential investments to choose from. Moreover, in the Spanish landscape, these companies are generally really large players with high liquidity which can be directed to R&D projects like the implementation of the solution.
- Industry knowledge: These companies have spent many years working with many different companies on their labour risks. This has helped them retrieve a vast amount of specific knowledge, which can help them determine what specific sectors or company types could benefit most from using the solution. By making them responsible of deciding to which of their clients to propose this solution, their knowledge can be leveraged to improve the chances of a successful implementation.
- Industry connections: These companies already have a huge network of existing clients that rely on them for anything labour risk related. By leveraging their sales distribution channels, the proposed product can reach a much wider audience of companies, and in a much more persuasive way.
- Product synergies: This product can be easily integrated into other of their existing products or can work alongside them. For example, they can make an offer to their clients in which if they accept implementing this solution, they can receive a host of other benefits. Reducing the insurance cost, which would probably be the favourite of the employer, is not an option since this insurance cost is regulated by law. However, there are other things the company can offer, from better health services to reductions in the price of other insurances like industrial insurance.

Probably the biggest problem of choosing the insurance companies as the clients is a concentration of clients which would have a huge bargaining power. Since the industry is very concentrated with very few players controlling most of the market, very probably there would end up being just a handful of clients in the best case. This would make PRLApp overly reliant on all of them, giving them a lot of room to bargain for more favourable conditions. This

problem has been considered, and it has been determined to not be a deal breaker for three main reasons:

- Symbiotic relationship: If the insurance companies understand PRLApp's value proposition, it is in their interest to keep working with it, and more importantly to help develop and improve it. This would mean that, at least in the beginning, they would probably act in the best interest of the product.
- Possible diversification: Even though the most important potential clients are few and concentrated, this does not mean that there are no more options. There are mainly two options for diversification
    - Individual companies or groups: When companies are big enough, some of them decide to become self-insured or insured in a group of industry partners. In these cases, all the economic benefits that were outlined for mutual insurance companies in the previous subsections also apply for these companies or groups of companies.
    - International expansion: Even though labour risk laws are different in each country, many western countries have the concept of a compulsory labour risk insurance for employees, and for more liberalized countries many private companies offer this kind of insurance even if it is not compulsory. In all these cases, there is a party – be it the mutual insurance company, the client company or any other player – that stands to benefit from a reduction in labour accidents. And the set of most common accidents are pretty standard across multiple countries, meaning that the application could be launched in different countries with minimal marginal effort.

Having described the client and the benefits of choosing it, it is now possible to understand the business model in itself. That is, how would PRLApp make money, and what would it offer in return.

The chosen model has been decided to be as flexible and scalable as possible. The whole client-side application and server-side application are licensed to mutual insurance companies,

becoming the white label infrastructure, while mutual insurance companies can offer its services as PaaS: Prevention as a Service, to their clients. In this model the end client needs to make zero effort to implement the solution. PRLApp would provide the necessary trainings to employees, the maintenance of the application and the management of accidents. All the while, being an invisible entitiy to the end client which would be conducting business with its usual mutual insurance company. To sum up, from the end client's perspective PRLApp could be seen as the Product and Operations department while the mutual insurance company would carry out the tasks of the Marketing and Sales departments and would be the visible brand.

The relationship between the mutual insurance company and PRLApp would be a little bit different. Under the licensing model, the mutual insurance company would pay a fixed onboarding price for implementing the solution in a client, followed by an ongoing management cost calculated based on the number of employees of the company and the riskiness of their activity. PRLApp would not be responsible for providing the Apple watches. Who provides them would have to be agreed between the insurance company and the client company, but very probably the insurance company would lease these smartwatches to the client company as part of their solution offering.

3.3 FINANCIAL ANALYSIS

As promising as any business may sound when discussed superficially, some kind of numerical financial assessment is needed to determine its viability. In this case, two different kinds of assessments could be performed.

- PRLApp focused analysis: This analysis is the typical financial analysis. It consists in estimating costs and profits and performing a forecast, which would show the main cost drivers, the main profitability leverages and whether the business seems profitable.
- Mutual insurance company analysis: This is a client-focused analysis, in which the solution is studied from the financial viability for the potential client. It can help determine whether this solution would be viable for them to implement, and therefore whether there would be a market for it.

A combination of both analyses will be performed, with the client-focused analysis being used to determine the possible pricing of the solution. Since this pricing is required to analyse and forecast the potential revenue of PRLApp, this analysis will be performed first.

3.3.1 CLIENT-FOCUSED ANALYSIS

This simple analysis will be a revenue-cost examination from the PRLApp's clients' perspective. The main revenue source for the mutual insurance companies is the reduction in claim payments, while their costs would be the buying of the apple watches and the payment for PRLApp's services. The different hypotheses are the following:

- Claim payment reduction: To determine how much mutual insurance companies pay in claims each year, a simple approximation has been made. Since the money they get and the money they must pay in claims is regulated by law, and they are not allowed to have profits, the money they pay in claims is roughly the same money their clients pay to the social welfare system in the concept of potential labour accidents. This money depends on several factors:
    - Gross salary: Each worker's gross salary is the base from which the employer's contribution to the social welfare system on its behalf is calculated. It has been estimated to be 25,000 € per year.
    - Contribution for occupational accidents and diseases: This is the percentage of the gross salary paid to the social welfare system with the concept of potential labour accidents. This is regulated by law (*Ley 42/2006 de Presupuestos Generales del Estado para el año 2007*, 2006) and depends on how risky the activity is. The average contribution has been estimated to be 5.2 %, which is the rate charged for crude oil extraction, forestry and logging or others of the sort.
    - Reduction in labour accident costs: This is the reduction in the claims the mutual insurance company will have to pay. It is determined by the accidents the application is able to completely predict and prevent and by the diminishment

in the effects of those that do happen. The base case has been determined to be a reduction in costs of 45 %.

- Apple watch costs: Since the mutual insurance company will be the one in charge of paying for the apple watches this is a cost that should be considered. Two main hypothesis have been made here
    - Price: The price of a single Apple watch 6 GPS+cellular – which has all the necessary capabilities – with cellular connection, needed to transmit the data from wherever the employees work –, has been found to be from $ 310.11 for a second-hand device. This price being the lowest, probably means that the watch's conditions is not the best, so a higher price could be expected. However, the smartwatch's aesthetics' or external state should not be of much concern as long as all sensors work, since it is to be used only during the working day as one work tool more. Furthermore, PRLApp could probably find cheaper-than-retail distributors or reach some agreement with Apple thanks to the volume of Apple watches it would have to buy, acting then as a distributor to the mutual insurance companies. Considering all of this, a final reasonable price of $ 319.99 has been decided as hypothesis.
    - Apple watch lifetime: The Apple watch is expected to get broken down, stolen or incapacitated in some other way. The mutual insurance companies will thus have to buy these Apple watches periodically. The average lifetime of a single Apple watch has been estimated to be 3 years.
- Mutual insurance company profit: Probably, especially in the beginning, mutual insurance companies could be convinced to collaborate at 0 % revenue. They would be interested in supporting the project as an R&D effort, as if it works they could benefit greatly from it. However, just in case as an incentive part of this "revenue" has been reserved for the mutual insurance company, as an incentive to collaborate. This percentage has been estimated to be 20 %

By combining all of these hypotheses, a price for the management of the service per employee can be calculated. This final price is € 386.34 per year.

3.3.2 PRLAPP FOCUSED ANALYSIS

Now that the client-focused analysis has been performed, a price for the services is determined and a proper revenue-cost analysis can be performed. These are the different hypotheses used to estimate the revenues and costs:

- General hypotheses: These are hypotheses that apply to both costs and revenues. They are the following three:
    - Number of end clients: An estimation of end client companies. These are not mutual insurance companies, as this number is irrelevant for the financial projections, but their clients which choose to adopt the solution. The total end clients schedule has been estimated to be 20, 50, 100, 150 companies, for each of the four first years.
    - Employees per end client: Each of these companies will have a different number of employees which choose to use opt in to use PRLApp, but the average number has been estimated to be 25.
    - Dollar – Euro conversion: Since some of the costs are in dollars but the general finances of PRLApp are in Euros, a conversion rate needs to be specified. Even though this rate is particularly unfavourable for the Euro right now, probably due to the events happening in Ukraine, the energy crisis suffered by Europe and the rise in interest rates by the Federal Reserve among other things, the current rate of 0.96 EUR/USD has been chosen as a conservative reference.
- Cost hypotheses: Below are explained the different hypotheses, split by cost type.
    - Personnel costs: The main cost component is the one dedicated to personnel. These are the different job positions needed to carry out implement the solution.
        - 1 frontend software developer: Developer in charge of the smartwatch application. Will be responsible of optimizing the application to reduce networking and battery usage, implement new features and general maintenance. Estimated salary is 50.000 € year

- 1 backend software developer: In charge of managing the server-side application and the AWS infrastructure. Estimated salary is 50.000 € year.
- 1 designer and web developer: In charge of the graphic design for the company and the design and development of company's website. This will be a very simple website with general information about the application. Estimated salary is 40.000 € year.
- 1 data & machine learning engineer: In charge of optimizing the data infrastructure in the backend, testing the predictive model and introducing improvements to the model. Ideally will be someone with specific knowledge in medical machine learning. Estimated salary is 60.000 €.
- 2 operations managers: In charge of onboarding new companies, training their employees, managing incidences and carrying out the day to day operations. Estimated salary is 35.000 € each for a total of 70.000 €.
- 1 business developer: In charge of managing the relationship with mutual insurance companies and acquiring new customers. Estimated salary is 40.000 € year.
- 1 CEO: The CEO will encompass different roles, among which the most important will be Chief Product Officer, coordinating the efforts of the different technical developers and understanding the client's needs, Chief Financial Officer, in charge of raising capital and managing RPLApp's finances and Human Resources manager, making decisions on talent acquisition and retention. Apart from these two roles, he/she will also hold the typical tasks of the CEO, like addressing investors and guiding the long-term vision for the company. Estimated salary is 60.000 € year.

- Administrative costs: Apart from the already mentioned personnel, there are some other tasks that are necessary for any company, but which do not require having a dedicated employee for a startup of this size. These are mainly two:
  - Administrative agency: Usually called *Gestoría* in Spanish, it is an external company in charge of many administrative tasks, like accounting or anything related with the governmental administration. Estimated cost is 16.000 € annually.
  - Legal agency: An external company in charge of providing legal advice. Tasked with anything from revising employment contracts to writing the collaboration contracts to be signed with the mutual insurance companies. Estimated costs are 8.000 € annually.
- Outsourcing costs: Not every technical development will be done in-house, as this would require to either create a really big technical team or to wait very long for every development. The development that is planned to be outsourced is that of the management platform for the client companies' supervisors. This will be a web platform from which supervisors will be able to see information such as how many active employees are using the smartwatch, any alerts that the employee wants to send to the supervisor, statistics about incidences, etc. The estimated cost for the development of this platform is 100.000 €. This will be a one time cost in the first year.
- Infrastructure costs: There are several costs associated with running the necessary technical infrastructure to make everything work. They can be divided in mainly three categories:
  - Storage costs: A lot of health and motion information will need to be stored in the server's database. The space required for all this data, together with PRLApp's website and the different parts of the server-side application has been estimated to be for the first years around 1TB GB. As the solution is adopted by more companies, more data will have to be stored, but this has been deemed to be a conservative estimate during at least the first years. As with the developed MVP, the best

option is to store this in a an EBS (Elastic Block Store), used as part of the EC2. The cost of storage is $ 0.08 per GB per month, for a total of $ 960 annually. (Amazon Web Services, n.d.)

- Computing costs: Apart from the storage, computing power is also necessary to perform the different data processing tasks. The required instance type has been determined to be the t3.medium, which has 2 vCPU for a total of 4 GiB – 1024^3 bytes instead of a GB, which is 1000^3 bytes. The cost of said EC2 instance is $ 0.0416 per hour, for a total of $ 360.67 in annual costs. (Amazon Web Services, n.d.)
- Networking costs: As many employees will be performing their job-related activities in locations without wifi access, it is vital for the Apple watches to be enables with cellular connectivity. The data rate needed to send and receive the needed data to and from the server-side application also has a cost. The message payload to be sent from the smartwatch has been estimated to be 300 bytes, which are sent every 30 seconds. This accounts for a total of around 5.7 GB per month, without considering received data. This is by all means a conservative estimate, and the data networking designed in the MVP can be greatly optimized, both in payload size and frequency. However, also to retain a conservative estimate, an unlimited rate targeted for IoT applications has been found. The data plan, which belongs to Things Mobile, offers Unlimited data for $ 30 per month. This will be thus the networking cost per employee which requires cellular connection. However, in many factory or enclosed settings WiFi will be accessible. The second networking hypothesis is that the split between cellular and WiFi connectivity is around 50 % - 50 %.

- Revenue hypotheses: Now that all the cost hypotheses have been addressed, the same must be done with the revenue side of the equation.
  - Onboarding cost: When a new end client adopts PRLApp, an onboarding cost is charged to the mutual insurance company. This is partly due to the initial cost

of having to implement the solution and having to train the employees and partly as a good measure to ensure a certain degree of commitment. This onboarding cost has been set to be equal to the ongoing cost of three months.

- Ongoing cost: This is the cost charged for the ongoing management of each end client's employee. The pricing comes from the client-focused analysis already performed in the previous subsection, which came out to be 386.34 € per year per employee.

3.3.3 FINANCIAL CONCLUSIONS

By combining all the previous hypotheses, several conclusions can be extracted:

- Required number of clients: The more end clients that are obtained, the more the fixed costs are spread among them. This results in a break-even number of end clients, a number of end-clients for which costs are equal to revenues. This number comes out to be 101 clients.
- Required initial investment: In the beginning, the number of end-clients lies below the break-even point calculated before. Plus, there are some initial investments which bring the costs at the beginning even below what they would be. By observing the evolution in liquidity, one can see what the initial investment needs to be. Below is portrayed the evolution of costs, revenues and liquidity without initial investment.

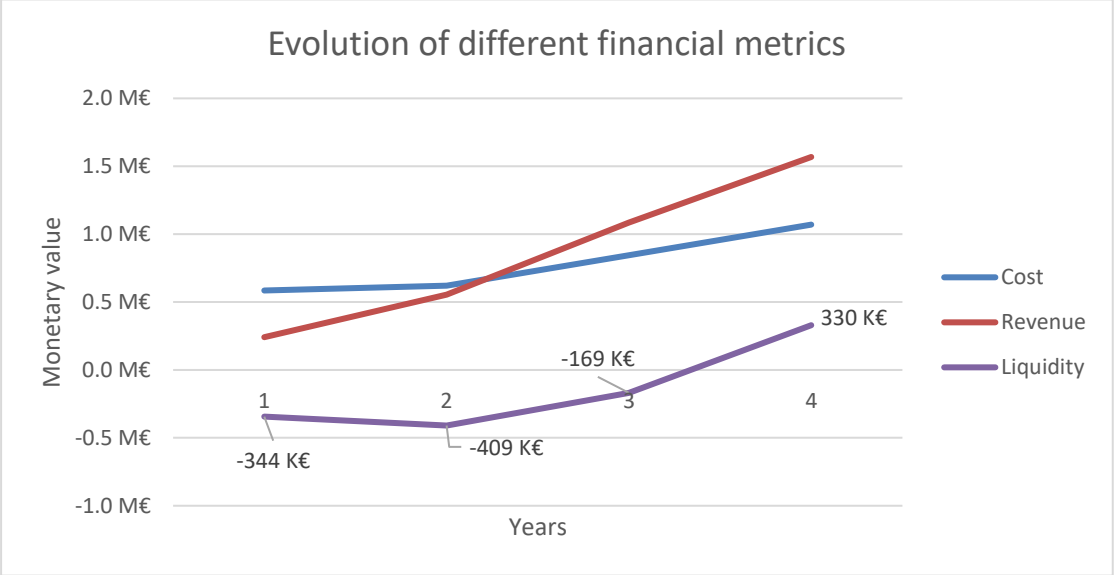

Figure 7: Financial metrics evolution obtained through the application's financial analysis (without initial investment)

As can be seen in the above graph, costs start higher than revenues mainly due to the initial investment needed to develop the monitoring platform and the low number of end clients at the beginning. With time, as the number of end clients increases, break-even is achieved shortly after the end of year two. However, some initial cash reserves are needed, as the initial investment is not recouped until after the end of year three. The initial investment technically needs to be the point of lowest liquidity, which is 408,725 €. This however would leave the liquidity reserves at 0 € at some point, and if anything goes wrong PRLApp would have to ask for a loan or raise capital again. Thus, a safer initial investment would probably be 500,000 €.

- Fundamental hypotheses: Of all the hypotheses used to come up with the above conclusions, some are more important than others. If required storage ends up being 0.5 TB or 2 TB instead of 1 TB this would have little impact in the bottom line. There are however some hypotheses which have been found to be the main cost or revenue drivers, and which are thus very important to get right as the whole financial viability of PRLApp depends on that. These are:
    - Accident cost prevention: This is the most fundamental revenue driver. If this prevention rate went from 45 % to 35 %, the break-even number of end-clients

would go up to 153 companies, and the lowest liquidity point would go up all the way to 674,426 €, with PRLApp taking way longer than 4 years to recover the initial investment.

- Networking cost: Networking cost are by far the biggest of the infrastructure costs, becoming bigger than all personnel costs after just 86 end clients. This is both good news and bad news. It is bad news because if networking hypotheses have been too optimistic, with a higher than expected data fee or proportion of cellular over WiFi devices, the whole endeavour could quickly become economically unfeasible altogether. The good part is that any small improvement in networking costs, be it by improving the frequency or size of the payload or by improving the networking price, could have incredible effects on the bottom line.

The financial analysis is both one of the most important and less reliable parts of any business analysis. It is fundamental for assessing the viability of any endeavour, but at the same time it is built on many assumptions which are at times very hard to check. It shouldn't be taken as holy text, believing any number it outputs up to the last cent. It can however give general ideas about in which financial scales one is moving. And more importantly, it is really useful for finding out what cost and revenue drivers one needs to be most concerned about, and it can be used to continuously reassess all financial planning once the business actually starts functioning and hypotheses start being substituted by realities.

Chapter 4. CONCLUSIONS

4.1 GENERAL FINDINGS

This thesis' focus has been to evaluate the viability of a technical-business idea, by building a proof of concept version of the solution and performing the necessary regulatory and business analyses. More specifically, this thesis has:

- Analysed the regulation and key actors in the Spanish labour risk prevention landscape.
    - The labour risk prevention regulation imposes several restrictions and guidelines on how the proposed solution should be implemented. Partly as a consequence of over regulation, it creates a kind of legal certainty in which the playing field rules are guaranteed to stay extremely stable. This can help guarantee that any business model that fits in the regulatory framework will be sustainable for a long time.
    - The GDPR analysis highlights the importance of maintaining employee intimacy protected and giving the employee control of their data. Any solution stemming from this thesis needs to guarantee that data is being collected and analysed for the benefit of the employee, that only summary metrics and conclusions are shared with the employer and only after explicit consent from the employee and that no final diagnoses are made without proper medical supervision. Abiding by those rules, the proposed solution is viable from a data privacy standpoint.
    - Mutual insurance companies were determined to be the customers in the proposed business model after analysing all relevant stakeholders and the potential benefits they would obtain from the solution's deployment.
    - Finally, a potential competitors analysis ensured that there are no relevant players in the market right now addressing labour risk prevention by directly monitoring the employee's biometric and motion data.

- Proposed a functional solution, that allows to prevent most of the most common typologies of labour accidents – falls, getting hit, back injuries, muscle tears, repetitive motions wear, lacerations and amputations, intoxications, electrocutions, loud noises, high vibrations, vehicle accidents and mental disorders – through the analysis of personal data – age, sex, height, weight –, biometric data – heart rate, active energy burned, basal energy burned, stand time, walking steadiness, environmental audio exposure, heart rate variability, oxygen saturation, body temperature, systolic blood pressure, diastolic blood pressure, respiratory rate and distance walked –, motion data – acceleration, rotation and gravity in all three axes – and ECG data collected through the smart watch. Through the constant analysis of this data through a predictive model, the system will alert the employee, provide advice on how to prevent said accident or mitigate the effects of an already happened accident and alert a supervisor if requested by the employee. The initial predictive model should focus on those accidents that are more directly detectable – those caused by noise, movement and some kinds of traumatism and with consequences focusing mainly on heart rate, blood pressure and blood oxygen saturation –, expanding later that list to accidents requiring more complex interrelations between metrics, as more data is gathered and the solution is perfected.
- Analysed the economic feasibility of the initiative. After defining the business model, some pricing hypotheses have been set. Based on those hypotheses, the initiative would require a ~500 K€ initial investment and reach breakeven point in ~100 end clients. The key cost factor is data costs for the smartwatch cellular connectivity. Any improvement on the networking efficiency of the application or the pricing deal obtained for this connectivity is therefore key for the profitability of the company. The key revenue factor is the reduction in claim payment achieved by the solution, which is a direct consequence of the accuracy of the predictive model.
- Built a functional proof of concept solution ready to be expanded into a fully implementable solution. After an initial analysis of technical options, the solution has been built on:

- Smartwatch: Apple Watch 6.0 or further. Apple was found to be the most advanced and developer friendly, with model 6.0 being the first one to include all required sensors.
- Infrastructure: AWS' EC2 instance
- Database: MariaDB
- Programming languages: Swift and Phyton
- Web: REST API implemented in Flask

This proposed architecture has several advantages:

- It leverages existing modules: Both in the client-side application and server-side application, the advantages of leveraging modules and classes already created by other people have been incredibly beneficial. They allow for anything from faster prototyping to accessing capabilities in the hardware that would have been impossible to benefit from otherwise.
- Leverages asynchronous tasks: A key to the proper functioning of the application has been the importance of queues and threads and asynchronous processes. Many of the smartwatch application tasks, like packaging the incoming sensor data into the proper data structure, sending this data, or reading information from the server, has been designed to be performed asynchronously. This allows for a correct functioning of the application and a more fluid change in UI, making the application a lot more usable.
- Leverages edge computing: With the growing computational and storage capabilities of edge devices – smartwatch in this case – it is clear that a growing part of the data processing and analysis will be performed at these edge locations. The proposed application would benefit from more edge computation mainly due to less networking load and faster alert times. This shift is however not possible at the current stage, since it is not yet known what specific datapoints and data processing routines will be needed for the final predictive model, so sending all possible data to the server-side application allows for better data centralization and discovery of the needed processing tasks.

- Leverages REST API's versatility: Even though other networking protocols where considered, HTTP requests through a REST API architecture was found to be a completely appropriate option. This architecture allowed for the performing of all necessary tasks between the client and the server in a straightforward way.
- Uses a Python web application: By developing the web application in Python, with the proper tools to handle the server's job, this application can be very easily and quickly integrated with all other parts of the server-side application, from database to other processes like the predictive model.
- Leverages established scientific knowledge: When developing a predictive model, the importance of looking at the relevant literature and examining models developed by researchers for similar problems cannot be understated. The amount of research going into ML model development is huge and should be leveraged. The second finding was not really surprising but had a great influence on the thesis. It is the importance of finding a good dataset on which to train the model.

The main takeaway from the points above is the overall viability of the project. However, before massively implementing this solution in a real-world environment, some further tasks need to be carried out.

4.2 NEXT STEPS

After the general findings have been pointed out, it is time point out the pending developments that should be carried out in next iterations to improve the quality of the project and leave it at a fully implementable stage.

- Working predictive model: The most fundamental pending development is that of a working predictive model. Even though a tentative model has been proposed, it hasn't been appropriately modified or trained. The training dataset needs to be acquired and

the suitability of the model to the problem at hand needs to be assessed, developing a new model if this one doesn't prove to be adequate.

- Networking optimization: Being networking costs the biggest cost driver, the frequency and size of messages sent to and from the server needs to be improved. This can be done by eliminating all data which is not needed – after the model has been developed and the required data specified –, changing the frequency of messages to the server to the minimum possible in order to have a reasonably real-time prediction and detection of accidents, reduce the frequency of the status requests to the server also to the minimum possible and improve the message format to optimize its size as much as possible. Not that by reducing the frequency the total data consumed is reduced even if the same data is sent, since the message overhead – things like message header, keys in the JSON, etc. – are spread out among more data per message.
- Migrate more computations to the edge: Along the same lines, once the model is specified and the required data is known, as many data processing tasks as possible should be migrated to the apple watch. Also, as many as possible of the chance accident detection algorithms should be taken to the smartwatch in order to improve the speed of the response. Also, some of the more straightforward extended accident prediction capabilities could be taken to the edge, like measuring and adding the exposure to sound, detecting when spot or accumulated maximum levels are reached.
- Additional features: The client-side application could be improved with many changes or additional features. Some of them are:
    - Add the option to notify a supervisor in every screen, not just when a chance accident has been detected.
    - Improve synchronization between server and client-side applications so the server can keep track of every change in the client-side application state. Right now, it only keeps track of the emergency status of the device.
    - Improve the UI to make it more useful to employees and easier to navigate.
    - Redirect to the data gathering screen instead of to the start screen when an employee signals he/she is ready to start working again.

  - Continue with the data gathering once the application goes to the warning, alert, resting or notifying screens. Increase the data gathering frequency in the notifying screen, since this is where the employee is expected to be suffering some kind of illness so it is when that data is most useful to ensure the safety of the employee.

- Integration with the client's and health services' systems: For the application to be fully implemented in an end client, two key connections need to be made. One is to the company's incident management systems, so all notifications of any incident can get to the supervisor. If the company doesn't have such a system, other options can be explored for how to notify the supervisor like SMS, Telegram, independent application, etc. The second key connection is to the company's health services provider, be it private or public, so any relevant medical data can be sent in real time to them when an emergency happens.
- Test solution in a real-world environment: Even though the solution seems to work in a laboratory setting, it would have to be tested in a real company with real employees. This would allow for the optimization of many parameters, like the frequency of the application state change between data gathering and not data gathering, and to uncover any unforeseen errors before launching the solution in multiple end clients.
- Validate business model: Even though the business model seems reasonable, meetings with different mutual insurance providers should be scheduled to ensure that the proposed solution addresses a real problem that they have, that they would benefit as expected from using it and that they would be interested in trying the proposed solution.

Chapter 5. BIBLIOGRAPHY

¿Qué son las mutuas? (2018, August 17). Retrieved from La revista de la Seguridad Social.

Amazon Web Services. (n.d.). *Precios de Amazon EBS*. Retrieved from AWS: https://aws.amazon.com/es/ebs/pricing/

Amazon Web Services. (n.d.). *Precios de las instancias bajo demanda de Amazon EC2*. Retrieved from AWS: https://aws.amazon.com/es/ec2/pricing/on-demand/

Apple. (2021). *Apple Watch Series 7 - Technical Specifications*. Retrieved from Apple Support: https://support.apple.com/kb/SP860

Apple. (n.d.). *Implementing Two-Way Communication Using Watch Connectivity*. Retrieved from Apple Developer: https://developer.apple.com/documentation/watchconnectivity/implementing_two-way_communication_using_watch_connectivity

Apple. (n.d.). *Network*. Retrieved from Apple Developer: https://developer.apple.com/documentation/network

Apple Watch SE (40 mm) (GPS) Specifications/Features/Review and Price . (n.d.). Retrieved from Smartwatch Charts: https://smartwatchchart.com/smartwatches/apple-watch-se-40mm-gps-specifications-features-review-and-price/

Bhardwaj, P. (2021, September 07). *Advantage and Disadvantage of Edge Computing*. Retrieved from Tutorialspoint: https://www.tutorialspoint.com/advantage-and-disadvantage-of-edge-computing

Blender. (2012, September 4). *Is the server bundled with Flask safe to use in production?* Retrieved from StackOverflow: https://stackoverflow.com/questions/12269537/is-the-server-bundled-with-flask-safe-to-use-in-production

Collaborative Filtering. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Collaborative_filtering

Contreras, G., & Bullon, J. (2021, August 25). *Accidentes de trabajo más comunes en las empresas*. Retrieved from Blog QHSE: https://www.blog-qhse.com/es/accidentes-de-trabajo-m%C3%A1s-comunes-en-las-empresas

Deb, S. (2018, November 20). *Restricted Boltzmann Machine Tutorial - A Beginner's Guide To RBM*. Retrieved from Edureka: https://medium.com/edureka/restricted-boltzmann-machine-tutorial-991ae688c154

Delgado, A. (2020, July 04). *La solución completa a la prevención de riesgos laborales frente al coronavirus*. Retrieved from Emprendedores: https://www.emprendedores.es/ideas-de-negocio/millas-digitales-prl-coronavirus/

Delgado, S. (2020, January 12). *Los 10 accidentes más comunes en entornos de trabajo*. Retrieved from Preven-ir: https://preven-ir.com/los-10-accidentes-mas-comunes-en-entornos-de-trabajo/

Fitbit Sense Full Specifications, Features and Price. (n.d.). Retrieved from Smartwatch Charts: https://smartwatchchart.com/smartwatches/fitbit-sense-full-specifications-features-and-price/

(2015). *Global trends on occupational accidents and diseases*. International Labour Organization.

Gupta, L. (2022, April 07). *What is REST*. Retrieved from REST API Tutorial: https://restfulapi.net/

IBM Cloud Education. (2021, April 06). *REST APIs*. Retrieved from IBM Cloud: https://www.ibm.com/cloud/learn/rest-apis

IBM. (n.d.). *HTTP requests*. Retrieved from IBM Documentation: https://www.ibm.com/docs/en/cics-ts/5.3?topic=protocol-http-requests

(2010). *ILO List of Occupational Diseases.* International Labour Organization.

La depresión es la segunda causa de baja laboral, incapacidad permanente o jubilación anticipada en España. (2020, December 14). Retrieved from Observatorio de RRHH: https://www.observatoriorh.com/orh-posts/la-depresion-es-la-segunda-causa-de-baja-laboral-incapacidad-permanente-o-jubilacion-anticipada-en-espana.html

La nueva "Start Up" del EPI y PRL, que simplifica el proceso de compra. (2021, June 03). Retrieved from Comunicae.es: https://www.comunicae.es/nota/la-nueva-start-up-del-epi-y-prl-que-simplifica-1225643/

Ley 31/1995 de prevención de Riesgos Laborales. (1995, November 8). Boletín Oficial del Estado.

Ley 42/2006 de Presupuestos Generales del Estado para el año 2007. (2006, December 28). Boletín Oficial del Estado.

Los 7 accidentes de trabajo más frecuentes #DíaMundialDeLaSeguridadYSaludEnElTrabajo. (2017, April 28). Retrieved from Peris: https://www.peris.es/seguros-de-salud/los-7-accidentes-de-trabajo-mas-frecuentes-diamundialdelaseguridadysaludeneltrabajo/

Los tipos de accidentes de trabajo más comunes. (n.d.). Retrieved from Cruz & Asociados abogados: https://cruzfirm.com/es/areas-de-practica/accidente-de-trabajo/tipos-de-accidentes-de-trabajo-comunes/

Lu, W.-C., Tzeng, N.-S., Kao, Y.-C., Yeh, C.-B., Kuo, T. B., Chang, C.-C., & Chang, H.-A. (2016). Correlation between health-related quality of life in the physical domain and heart rate variability in asymptomatic adults. *Health and Quality of Life Outcomes*, 149.

Manage fall detection on Apple Watch (Apple Watch SE and Apple Watch Series 4 and later only). (n.d.). Retrieved from Apple Watch User Guide: https://support.apple.com/en-gb/guide/watch/apd34c409704/watchos

Mozilla Foundation. (n.d.). *Generalidades del protocolo HTTP*. Retrieved from MDN web docs: https://developer.mozilla.org/es/docs/Web/HTTP/Overview

Muscle Strains. (2018, January 11). Retrieved from Healthline: https://www.healthline.com/health/strains

Pallets. (n.d.). *API Documentation*. Retrieved from Flask: https://flask.palletsprojects.com/en/2.1.x/api/#application-object

Panettieri, J. (2022, April 29). *Cloud Market Share 2022: Amazon AWS, Microsoft Azure, Google Cloud*. Retrieved from Channel E2E: https://www.channele2e.com/news/cloud-market-share-amazon-aws-microsoft-azure-google/

Precios de Amazon EC2. (n.d.). Retrieved from AWS: https://aws.amazon.com/es/ec2/pricing/

Qué es realmente la Seguridad Social. (2021, September 7). Retrieved from La revista de la Seguridad Social: https://revista.seg-social.es/-/qu%C3%A9-es-realmente-la-seguridad-social

Real Decreto 231/2017 por el que se regula el establecimiento de un sistema de reducción de las cotizaciones por contingencias profesionales a las empresas que hayan disminuido de manera considerable la siniestralidad laboral. (2017, March 24). Boletín Oficial del Estado.

Real Decreto 286/2006 sobre la protección de la salud y la seguridad de los trabajadores contra los riesgos relacionados con la exposición al ruido. (2006, March 10). Boletín Oficial del Estado.

Real Decreto 39/1997 por el que se aprueba el Reglamento de los Servicios de Prevención. (1997, Enero 17). Boletín Oficial del Estado.

Real Decreto Legislativo 8/2015 por el que se aprueba el texto refundido de la Ley General de la Seguridad Social. (2015, October 30). Boletín Oficial del Estado.

Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016. (2016, April 2016). *On the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation).*

Samsung. (n.d.). *Samsung Galaxy Watch 4 Specifications*. Retrieved from The Official Samsung Galaxy Site: https://www.samsung.com/global/galaxy/galaxy-watch4/specs/

Santos, R. (2017). *What is MQTT and How It Works*. Retrieved from Random Nerd Tutorials: https://randomnerdtutorials.com/what-is-mqtt-and-how-it-works/

Simon, M. (2020, August 25). *Fitbit Sense vs Versa 3: Small differences make a huge difference*. Retrieved from Macworld: https://www.macworld.com/article/234535/fitbit-sense-vs-versa-3-design-sensors-fitness-health-price.html

Stables, J. (2022, June 21). *67 Apple Watch tips and features: Hidden features revealed*. Retrieved from Wearable: https://www.wareable.com/apple/apple-watch-tips-and-tricks-iwatch-7383

Startups andaluzas desarrollan soluciones para mejorar la seguridad en el trabajo. (2021, April 28). Retrieved from Europapress: https://www.europapress.es/andalucia/noticia-startups-andaluzas-desarrollan-soluciones-mejorar-seguridad-trabajo-20210428140355.html

Stud, @. (2020, September 30). *What is WSGI (Web Server Gateway Interface)?* Retrieved from Medium: https://medium.com/analytics-vidhya/what-is-wsgi-web-server-gateway-interface-ed2d290449e

Verma, S., Patel, P., & Majumdar, A. (2019). Collaborative Filtering with Label Consistent Restricted Boltzmann Machine.

Visengeriyeva, L., Kammer, A., Bär, I., Kniesz, A., & Plöd, M. (n.d.). *CRISP-ML(Q). The ML Lifecycle Process*. Retrieved from Machine Learning Operations: https://ml-ops.org/content/crisp-ml

Visram, T. (2021, December 28). *ESG investing continued to soar in 2021. The government could boost it even more*. Retrieved from Fast Company: https://www.fastcompany.com/90706552/esg-investing-continued-to-soar-in-2021-the-government-could-boost-it-even-more

Wang, R., & Yang, Z. (2017). SQL vs NoSQL: A Performance Comparison. *University of Rochester*.

What are the Sustainable Development Goals? (n.d.). Retrieved from United Nations Development Programme: https://www.undp.org/sustainable-development-goals

Young, H. A., & Benton, D. (2018). Heart-rate variability: a biomarker to study the influence of nutrition on physiological and psychological health? *Behavioural Pharmacology*, 140-151.

Zenechenko, S. (n.d.). *How to use the URLSessionWebSocketTask in Swift. Post WWDC deep-dive review.* Retrieved from AppSpector: https://appspector.com/blog/websockets-in-ios-using-urlsessionwebsockettask

APPENDIX I: HARDWARE COMPARISON OF MAIN OPTIONS

DEVICE	PRICE	NATIVE APPS	CONNECTION	GPS	ALTIMETER	BLOOD O2	ELECTRICAL HEART SENSOR	OPTICAL HEART SENSOR	ECG	COMPASS	ACCELEROMETER	GYROSCOPE	AMB LIGHT SENSOR	MICROPHONE	SPEAKER	BIA	EDA	SKIN TEMP	ATRIAL FIBRILATION	HIGH/ LOW HR	FALL DETECTION	CALORIES	EXERCISE TIME	STAND ALERT	CASING	STORAGE (GB)	BAT LIFE (h)	WATER RESISTANT (m)
Apple Watch 7	$399	Y	LTE+3G, WiFi, Bluetooth 5.0, NFC	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	Y	Y	Y	Y	Y	Y	ALUMINUM	32	18	50
Apple Watch SE	$279	Y	WiFi, Bluetooth 5.0, NFC	Y	Y	N			Y*	Y	Y	Y	Y	Y	Y	N	N	N	Y	Y	Y	Y	Y	Y	ALUMINUM	32	18	50
Galaxy Watch 4	$250	Y	LTE+3G,WiFi, Bluetooth 5.0, NFC	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	Y	Y	Y	Y	Y	ALUMINUM	16	+24	50
Fitbit Sense	$200	Y	WiFi, Bluetooth 5.0, NFC	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	Y	Y	Y	ALUMINUM	4	+6*24	50
Fitbit Charge 5	$120	N	Bluetooth 4.0, NFC	Y	N	Y	N	Y	Y	N	Y	N	Y	N	N	N	Y	Y**	Y	Y	N	Y	Y	Y	ALUMINUM		7*24	50
Fitbit Versa 3	$180	Y	WiFi, Bluetooth 4.0, NFC	Y	Y	Y		Y	N		Y	Y	Y	Y	Y	N	N											

*(has sensor but no tracking capabilities)

** (No dedicated sensor but capable at night)

Sources: (Apple, 2021) (Stables, 2022) (Samsung, n.d.) (Simon, 2020) (Fitbit Sense Full Specifications, Features and Price, n.d.) (Manage fall detection on Apple Watch (Apple Watch SE and Apple Watch Series 4 and later only), n.d.) (Apple Watch SE (40 mm) (GPS) Specifications/Features/Review and Price , n.d.)

APPENDIX II: CLIENT-SIDE APPLICATION CODE

The client-side application code can be found in this folder of the following GitHub repository:

https://github.com/fcelya/prl-app/tree/main/watch-app/test-4%20WatchKit%20Extension

APPENDIX III: SERVER-SIDE APPLICATION CODE

The code can be found in the following GitHub repository:

https://github.com/fcelya/flask-server

APPENDIX IV: PREDICTIVE MODEL CODE

The code provided by Juan can be found in the following GitHub repository:

https://github.com/fcelya/rbm-prl-example

APPENDIX V: ALIGNMENT WITH SDG

Nowadays, the world is more globalized than ever. And thanks to the advancements in communication technologies, practically everyone is able to see what is happening practically everywhere around the globe. This makes it easier to realize than ever that our local decisions have global consequences.

That is why, when coming up with a project, it is necessary to think about the implications of every decision taken. A good framework to think about the moral implications of these decisions are the UN Social Development Goals. They are a set of goals agreed upon by the UN which aim to set different aspirational targets for countries and companies to aim towards (What are the Sustainable Development Goals?, n.d.). Of this set of 17 targets, the proposed solution has been designed to tackle the following:

- 3. Health and wellbeing: Everyone's health should be a priority. During the last few years many different advances have been achieved, in everything from the fight against HIV to that against cancer. However, this advance has been rooted in inequality, favouring more those already better off. The implementation of the presented solutions aims to be a direct improvement of everyone's labour health. This improvement is however even more targeted towards manual workers, which are generally those of lower socioeconomic status. This solution will therefore directly aim to improve worker's health and to do so prioritizing those worse off.
- 8. Decent work and economic growth: Over the last 25 years, the amount of people living in extreme poverty has been reduced dramatically while the percentage of people that make up the middle class in developing nations has tripled. However, this growth seems to be slowing down and there seems to be a widening inequality. The creation of this jobs needs to be done in a sustainable manner, without forgetting of the adjective "decent", as this is the only way for this job creation to lead to human fulfilment. The presented solution aims to improve the working conditions of those

which have generally been stuck with the worst working conditions. Thus, it aims to guarantee that this necessary job creation is done while also improving job quality, and not just quantity.

- 17. Partnerships for the goals: Having a world more interconnected than ever, fostering cooperation is the only way to achieve the Sustainable Development Goals. This cooperation needs to be cooperation between countries – emphasizing specially cooperation between North-South and South-South hemispheres – but also between private companies among themselves and with public entities. In this last idea about private-private and private-public partnerships is where the proposed solution comes it. Its aim is to act as an enabler between different parties, improving the connection between mutual insurance companies and their clients and between companies and the public health infrastructure.