MÁSTER UNIVERSITARIO EN INGENIERÍA INDUSTRIAL

TRABAJO FIN DE MÁSTER

TRACEABILITY OF BOTTLE RECYCLING USING CONNECTED CONTAINERS

Autor: Paloma Reinoso Otero

Director: Álvaro Pérez Bello

Madrid

Julio de 2021

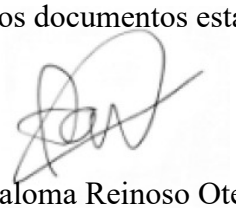
Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título
……Traceability of bottle recycling using connected containers……..……

…… en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2020-2021 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada

de otros documentos está debidamente referenciada.



Fdo.: Paloma Reinoso Otero Fecha: 15/07/2021

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Álvaro Pérez Bello Fecha: .15./ .07./ .2021

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

MÁSTER UNIVERSITARIO EN INGENIERÍA INDUSTRIAL

TRABAJO FIN DE MÁSTER

TRACEABILITY OF BOTTLE RECYCLING USING CONNECTED CONTAINERS

Autor: Paloma Reinoso Otero

Director: Álvaro Pérez Bello

Madrid

Julio de 2021

TRACEABILITY OF BOTTLE RECYCLING USING CONNECTED CONTAINERS

Author: Reinoso Otero, Paloma.

Director: Pérez Bello, Álvaro.

Collaborating Entity: Altair.

PROJECT ABSTRACT

Introduction

Over the last few years, awareness of packaging use and waste control has increased among citizens. The demand for ecological and sustainable products has grown and there is a popular trend towards recycling. Major companies are starting to support different initiatives to address climate change and making the company's products more environmentally friendly.

Despite all this progress and even though human beings are responsible for producing the largest amount of waste, it seems that sometimes people are still reluctant to recycle. The management of waste originating from any activity, as well as its correct recycling, continues to be one of the main problems against the so-called "environmental responsibility". [1]

Objective

This project arises as a solution to the low commitment of consumers regarding recycling. The main objective is to offer large beverage companies a sustainable and efficient solution for the recovery and recycling of their glass bottles. Based on the implementation of the circular economy model and making easier the recycling process to the consumer, companies will be able to re-use them directly.

Solution

To achieve this goal, a smart connected container has been developed using IoT technology.

The process that has be implemented follows the next steps:

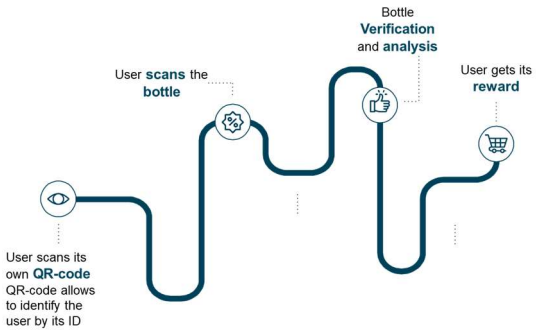


Figure 1: Glass bottle recycling chain

1. Once the user gets to a container, identification will be made by scanning a QR code. This QR code will be read using the camera placed on the surface of the container. If the user is registered in the platform, the display will show a confirmation message and the container status will change to open. only registered users can access the containers to ensure that the recycling process is done correctly.
2. After the authentication process, the image processing system will check that the item the user wants to recycle is valid (is a bottle) and that it is in good conditions. Display will show

the next steps to be followed. If the scanned item is invalid, the message rejecting the item will be displayed.

3. If the bottle is valid, the user will have to place it inside the container to scan the brand. After that, it will be stored in the corresponding box.
4. The user will receive points for each recycling action. If the bottles recycled are from one of the companies that has an agreement with the project, they will be rewarded with a discount on that brand.

This project has been developed with the help of Altair and its SmartWorks IoT platform. Altair SmartWorks IoT [2] provides all what is needed to quickly build scalable, secure web, mobile and edge applications and then iterate fast to find product-market fit. In this project it will be essential to accelerate the IoT smart application development.

SmartWorks IoT will provide real-time information about the container. This allows the transportation company to know the volume of bottles stored to decide when it is necessary to empty them. Maintenance workers will also know if there is any failure and users will have the information of where the containers are placed, and their status. Each user will have access to data about their recycling progress, accumulated money and if the container is pending to be collected there will be showed another container nearby, to avoid overflows, or when it will be emptied. The client (beverage company) will receive information about recycling trends and consumption by neighbourhoods which will allow them to launch more personalized offers. Thanks to all the information collected by the containers, more efficient collection routes can be established.

A digital model of the entities in the smart product ecosystem has been defined in Altair SmartWorks AnythingDB to store all the information. The following collections have been created to separate the information in different tables.

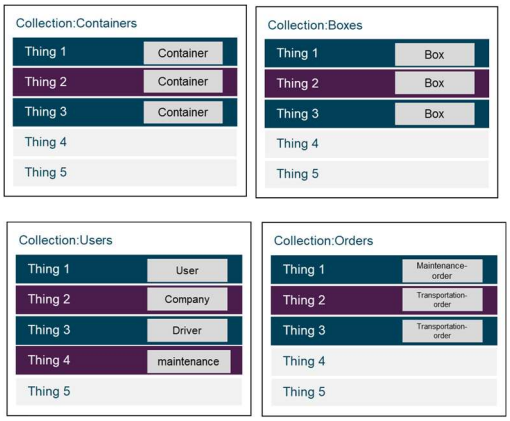


Figure 2: Collections defined at SmartWorks IoT.

Altair SmartWorks Functions have been used in this project to predict feature values and create custom business rules.

The following functions have been created:

- **Calculate bottles recycled in a container**: calculate total amount of bottles recycled on each container.
- **Calculate company bottles**: calculate total amount of bottles recycled by a company.
- **Calculate capacity**: based on the number of bottles on each box, the percentage of occupied capacity is calculated.
- **Full capacity event**: the boxes designed have a capacity of 180 bottles. Whenever the box is full, an event of full capacity will be registered, and the status of the container will change to "full". There is a data history of the events

registered.

- **Count bottles in box**: if a bottle is correctly identified by the container, it will be stored on the corresponding box and the number of bottles stored on that box will be updated.
- **Calculate number of bottles**: when the user recycles a new bottle, it will be added to the total amount of bottles recycled.
- **Calculate users' balance**: based on the number of bottles recycled, the balance of the user will be calculated and updated.
- **Set pick-up orders**: if the status of a box changes to full, a transportation order will be assigned to a driver. The status of the driver will change to unavailable and in the collection orders, the order information will be filled out. The departure time, expected arrival time and real time of arrival will also be calculated.
- **Set maintenance orders**: if the status of a container changes to failure, a maintenance order will be assigned to a maintainer. The status of the maintainer will change to unavailable and in the collection orders, the order will be filled out.

Altair SmartWorks Access Control offers the management mechanism to set authorization rules for every piece in the IoT platform. In this project, this tool will be useful to:

- Overview the users who have access to the environment - either through the SmartWorks IoT interface or through an end user application.
- Define from generalized to fine grained access control policies to carefully control which users have access to which information from the smart product ecosystem. Drivers, maintenance workers, users and companies will not be having the same role as each of them will be authorize to certain parts of the application.
- Create Apps to link SmartWorks IoT to external sources of information.

Altair® Panopticon™ technology provides the functionality and the ability to connect to a wide range of streaming data sources, time series databases and regular databases, both on-premises and in the cloud, and display data with interactive information visualization. Using Real Time Visualization (Panopticon Visualization), each user will have a different dashboard showing the relevant data for them from the smart product in real time. It helps to easily build an application for users fast and with no code.

The following schema summarizes the communication flow:

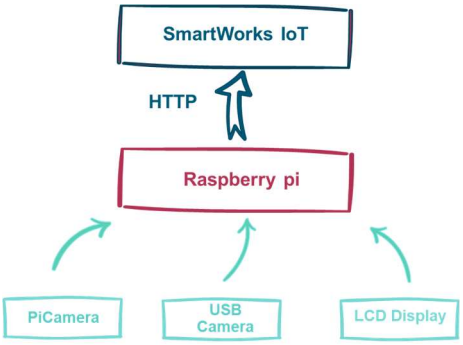


Figure 3: System communication schema

The project has been developed using a Raspberry Pi 4. It is a capable device that works as a low-cost computer. It enables to programme different applications and interact with the outside world [3]. The operating system chosen to work with has been Raspbian, the official supported OS. The following components are also being used:

Raspberry Pi Camera module to analyse the brand of the bottles.

LCD Module Display to guide the user through the recycling process.

USB Camera to scan both the QR codes and the items to identify if they are valid or not.

Heat Sink and Cooling Fan have also been installed to ensure the correct maintenance of the device.

The following Python codes have been developed:

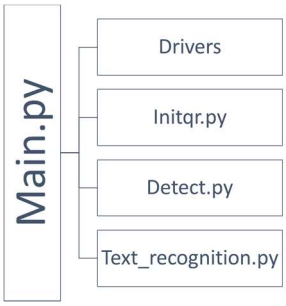


Figure 4: Python codes' schema

Once the raspberry is plugged-in, the main.py python script starts to run and makes the call for the other scripts. Each of them is in charge of one of the main process. All of them use the driver's library to show messages on the display to guide the user.

Initqr.py: reads the data on the QR code and checks if the id belongs to any of the users of the platform. If the id is valid, sends the order to open the container and displays the related message. All this data is sent to the platform to have a historical database of each time the container has been opened. It processes the QR images and identifies them by using the open-source library OpenCV.

Detect.py: identifies the type of element the user is trying to recycle and displays a message showing if the item is valid or not. If the item is a bottle, then data is sent to the platform to add one bottle to the user's account. After that, the user is asked to place the bottle inside the container to scanned its brand and store it. All this information will be sent to the corresponding box and company. This has been done using yolo object detection.

Text-recognition.py: analyses the bottle and detects its brand. If the brand is correctly detected, data is sent to the platform so the information about the boxes' capacity is updated. For this purpose, pytesseract has been used.

Results

Although the results obtained are good, new functionalities can be implemented to improve the experience of all users involved by developing new business rules and improving the Things' attributes and logic. The next step is to migrate this process to the Altair EdgeOps tool. Altair EdgeOps [4] brings web development tools and methods to the edge so that interaction with the design is easily made.

Finally, the developed prototype (Figure 5) and the design for its final implementation (Figure 6) are shown.

Figure 5: Prototype developed.

Figure 6: Design of the final model of the container.

References

[1] National Institute of Statistics (2020). *Cuentas medioambientales: Cuenta de los residuos. Año 2018*.

[2] Altair SmartWorks IoT (2021). **https://help.altair.com/2021/smartworks/topics/overview/sw_intro.htm**

[3] Raspberry Pi (s.f) **https://www.raspberrypi.org/**

[4] Altair SmartWorks IoT (s.f). **https://help.altair.com/2021/smartworks/topics/edge_ops/edge_ops.htm**

TRAZABILIDAD DEL RECICLAJE DE ENVASES UTILIZANDO CONTENEDORES CONECTADOS

Autor: Reinoso Otero, Paloma.

Director: Pérez Bello, Álvaro.

Entidad colaboradora: Altair.

RESUMEN DEL PROYECTO

Introducción

En los últimos años, la concienciación de los ciudadanos sobre el uso de los envases y el control de los residuos ha aumentado, así como la demanda de productos ecológicos y sostenibles. En la actualidad, existe una tendencia hacia el reciclaje que se extiende hasta las grandes empresas, que están empezando a apoyar diferentes iniciativas para hacer frente al cambio climático y conseguir que sus productos sean más ecológicos y sostenibles.

A pesar de todos estos avances y de que el ser humano es el principal responsable de producir la mayor cantidad de residuos, sigue habiendo gente reacia a reciclar. Debido a esto, la gestión de los residuos procedentes de cualquier actividad, así como su correcto reciclaje, sigue siendo uno de los principales problemas frente a la llamada "responsabilidad medioambiental"[|1].

Objetivos

Este proyecto plantea una alternativa para dar solución al porqué del fracaso del reciclaje entre los consumidores. El objetivo principal es ofrecer a las grandes empresas de bebidas una solución sostenible y eficiente para la recuperación y reciclaje de sus botellines de vidrio. Basándose en la implantación del modelo de economía circular y facilitando el proceso de reciclaje al consumidor, las empresas podrán reutilizarlos directamente.

Solución

Para lograr este objetivo, se desarrollará un contenedor inteligente conectado gracias a la tecnología IoT.

El proceso que se ha implementado es el siguiente:

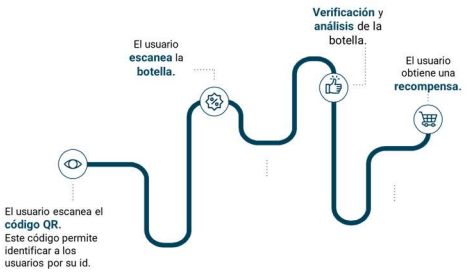


Figura 1: Cadena de reciclaje de los botellines

1. Una vez que el usuario llega a un contenedor, la identificación se hará escaneando el código QR. El código QR se leerá mediante la cámara situada en la superficie del contenedor. Si el usuario está registrado en la plataforma, la pantalla mostrará un mensaje de confirmación y el estado del contenedor cambiará a abierto.
2. Tras el proceso de autentificación, el sistema de procesamiento de imágenes comprobará que el artículo que el usuario quiere reciclar es válido (es una botella) y que está en buenas condiciones. La pantalla mostrará los siguientes pasos a

seguir. Si el artículo escaneado no es válido, se mostrará el mensaje correspondiente.
3. Si la botella es válida, el usuario deberá colocarla dentro del contenedor para que se analice la marca. Después, se almacenará en la caja correspondiente.
4. El usuario recibirá puntos por cada acción de reciclaje. Si las botellas recicladas son de una de las empresas que tienen un acuerdo con el proyecto, serán recompensadas con un descuento en esa marca.

Este proyecto ha sido desarrollado con la ayuda de Altair y su plataforma SmartWorks IoT. Altair SmartWorks IoT [2] proporciona todo lo necesario para construir rápidamente aplicaciones web y móviles escalables y seguras, y poder luego iterar rápidamente para encontrar el mercado adecuado para el producto. En este proyecto será de gran utilidad para acelerar el desarrollo de aplicaciones inteligentes de IoT.

SmartWorks IoT proporcionará información en tiempo real sobre el contenedor. Esto permitirá a la empresa de transporte conocer el volumen de botellas almacenadas para decidir cuándo es necesario vaciar las cajas. Los trabajadores de mantenimiento también sabrán si hay alguna avería y los usuarios tendrán la información de dónde están colocados los contenedores y su estado. Cada usuario tendrá acceso a los datos sobre su progreso en el reciclaje, el dinero acumulado y si el contenedor está pendiente de ser recogido se mostrará otro contenedor cercano, para evitar desbordamientos, o cuándo será vaciado. El cliente (empresa de bebidas) recibirá información sobre las tendencias de reciclaje y consumo por barrios, lo que le permitirá lanzar ofertas más personalizadas. Además, gracias a toda la información recogida por los contenedores, se podrán establecer rutas de recogida más eficientes.

Se ha definido un gemelo digital de las entidades que forman el ecosistema del producto conectado en Altair SmartWorks AnythingDB, para poder almacenar toda la información. Para separar la información en diferentes tablas y poder gestionarla cómodamente, se han creado las siguientes colecciones.

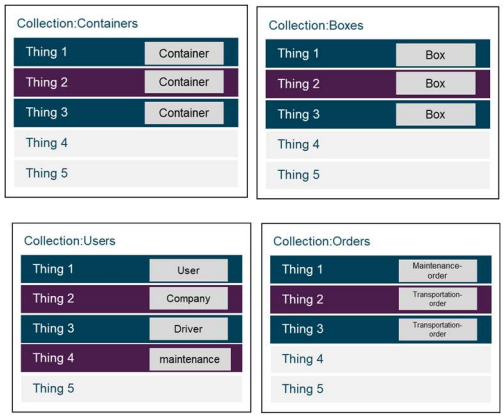


Figura 2:Collections definidas en SmartWorks IoT

Las funciones de Altair SmartWorks se han utilizado en este proyecto para predecir los valores de algunas propiedades y crear reglas de negocio personalizadas.

Se han creado las siguientes funciones

- **Calcular botellas recicladas en un contenedor**: calcula la cantidad total de botellas recicladas en cada contenedor para conocer las zonas más demandadas.
- **Calcular las botellas de la empresa**: calcula la cantidad total de botellas recicladas por una empresa.
- **Calcular capacidad**: en base al número de botellas de cada caja, se calcula la capacidad de esta.
- **Evento de capacidad completa**: las cajas diseñadas tienen una capacidad de 180 botellas. Cada vez que la caja

esté llena, se registrará un evento de capacidad completa y el estado del contenedor cambiará a "lleno". Existe un historial de datos de los eventos registrados.

- **Contar botellas en la caja**: si una botella es identificada correctamente por el contenedor, se almacenará en la caja correspondiente y se actualizará el número de botellas almacenadas en esa caja.
- **Calcular el número de botellas**: cuando el usuario recicle una nueva botella, ésta se sumará al total de botellas recicladas.
- **Calcular el saldo de los usuarios**: en función del número de botellas recicladas, se calculará y actualizará el saldo del usuario.
- **Establecer órdenes de recogida**: si el estado de una caja cambia a lleno, se asignará una orden de transporte a un conductor. El estado del conductor cambiará a no disponible y en las órdenes de recogida se rellenará la información correspondiente a esa orden. También se calculará la hora de salida y la hora prevista y real de llegada.
- **Establecer órdenes de mantenimiento**: si el estado de un contenedor cambia a fallo, se asignará una orden de mantenimiento a un trabajador. El estado del mantenedor cambiará a no disponible y en las órdenes de recogida se rellenará la orden correspondiente.

Altair SmartWorks Access Control ofrece un mecanismo de gestión que permite establecer reglas de autorización para cada pieza en la plataforma IoT. En este proyecto, esta herramienta será útil para:

- Resumir los usuarios que tienen acceso al entorno, ya sea a través de la interfaz de SmartWorks IoT o a través de una aplicación.
- Definir políticas de control de acceso para controlar cuidadosamente qué usuarios tienen acceso a qué información del entorno. Los conductores, los trabajadores, los usuarios y las empresas no tendrán el mismo papel, ya que cada uno de ellos estará autorizado a acceder a ciertas partes de la aplicación.
- Crear aplicaciones para vincular SmartWorks IoT con fuentes de información externas.

La tecnología Altair® Panopticon™ proporciona la funcionalidad y la capacidad de conectarse a una amplia gama de fuentes de datos en tiempo real, bases de datos de series temporales y bases de datos regulares, tanto en las instalaciones como en la nube, y mostrar los datos con una visualización de información interactiva. Mediante la visualización en tiempo real (Panopticon Visualization), cada usuario dispondrá de un cuadro de mando diferente que mostrará los datos relevantes para él en tiempo real.

El siguiente esquema resume cómo se comunica el sistema:

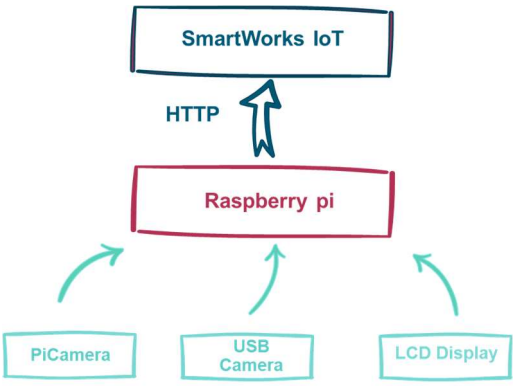


Figura 3: Esquema del sistema de comunicación

El proyecto se ha desarrollado utilizando una Raspberry Pi 4. Se trata de un dispositivo capaz de funcionar como un ordenador de bajo coste. Permite programar diferentes aplicaciones e interactuar con el mundo exterior. [3] El sistema operativo elegido para trabajar ha sido Raspbian, el SO oficial soportado. También se han utilizado los siguientes componentes:

Módulo de cámara Raspberry Pi para analizar la marca de las botellas.

Módulo LCD Display para guiar al usuario en el proceso de reciclaje.

Cámara USB para escanear tanto los códigos QR como los artículos, para identificar si son válidos o no.

También se ha instalado un **disipador de calor y un ventilador de refrigeración** para asegurar el correcto mantenimiento del dispositivo.

Se han desarrollado los siguientes códigos Python:

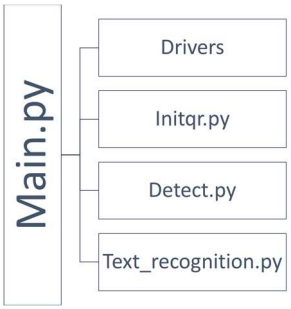


Figura 4: Esquema de los códigos de Python

Una vez que se conecta la Raspberry, el archive main.py se ejecuta e irá llamando a los demás códigos de Python. Cada uno de estos códigos se encarga de cada una de las fases del proceso de reciclaje. Todos estos códigos usan la librería "drivers" para mostrar en el display los distintos mensajes que irán guiando al usuario en el proceso.

Initqr.py: obtiene la información de los códigos QR y comprueba si el id almacenado corresponde a alguno de los usuarios registrados en la plataforma. Si el id es válido, envía la orden para abrir el contenedor y muestra el correspondiente mensaje. Toda la información es enviada a la Plataforma para tener un histórico de datos. La lectura de los códigos y el procesamiento de imágenes se realiza con la librería OpenCv.

Detect.py: identifica el tipo de objeto que el usuario trata de escanear y muestra un mensaje sobre si este es válido o no. Si el objeto es una botella, se envía la orden a la plataforma de que el Usuario ha reciclado una nueva botella. Después, el usuario debe colocar la botella dentro del contenedor para escanear su marca y guardarla en la caja correspondiente. Toda esta información se enviará a la caja y empresa correspondientes. Este proceso se realiza utilizando el detector de objetos de yolo.

Text-recognition.py: analiza la botella y detecta sum arca. Si la marca se detecta correctamente, se envía la información correspondiente para poder actualizar la capacidad de las cajas. Para ello, se emplea la librería pytesseract.

Resultados

A pesar de que los resultados obtenidos son buenos, se podrían añadir nuevas funcionalidades para mejorar la experiencia de los distintos usuarios involucrados a través del desarrollo de nuevas reglas que mejoren la lógica del proceso.

El siguiente paso será migrar este proceso a la herramienta Altair EdgeOps tool. Altair EdgeOps [4] ofrece herramientas para el Desarrollo web y la oportunidad de interaccionar con los

distintos dispositivos desplegados de manera fácil y rápida.

Finalmente se muestra el prototipo desarrollado (Figura 5) y el diseño para su implementación final (Figura 6) .

Figura 5: Prototipo desarrollado

Figura 6: Diseño del modelo final

Referencias

[1] National Institute of Statistics (2020). *Cuentas medioambientales: Cuenta de los residuos. Año 2018*.

[2] Altair SmartWorks IoT (2021). **https://help.altair.com/2021/smartworks/topics/overview/sw_intro.htm**

[3] Raspberry Pi (s.f) **https://www.raspberrypi.org/**

[4] Altair SmartWorks IoT (s.f). **https://help.altair.com/2021/smartworks/topics/edge_ops/edge_ops.htm**

Index

Figure index

Table index

Chapter 1: Introduction

Over the last few years, awareness of packaging use and waste control has increased among citizens. The demand for ecological and sustainable products has grown and there is a popular trend towards recycling. Major companies are starting to support different initiatives to address climate change and making the company's products more environmentally sound.

Despite all this progress and even though human beings are responsible for producing the largest amount of waste, it seems that sometimes people are still reluctant to recycle. The management of waste originating from any activity, as well as its correct recycling, continues to be one of the main problems against the so-called "environmental responsibility".

Households represent 16.5% of the generators of waste (Figure 1) and the main option in terms of final treatment is still waste dumping (Figure 2) [1].

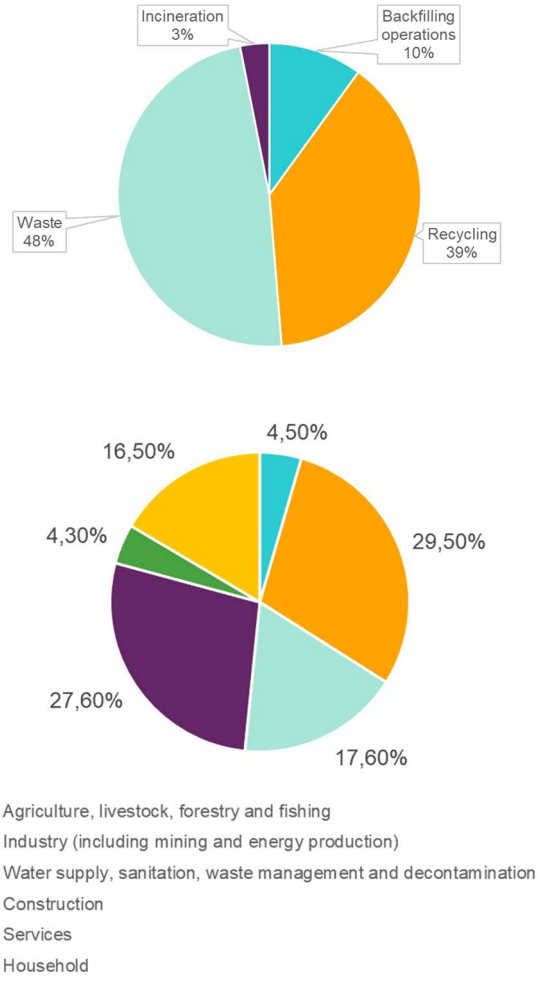


Figure 1: Generated waste, 2018 [INE]

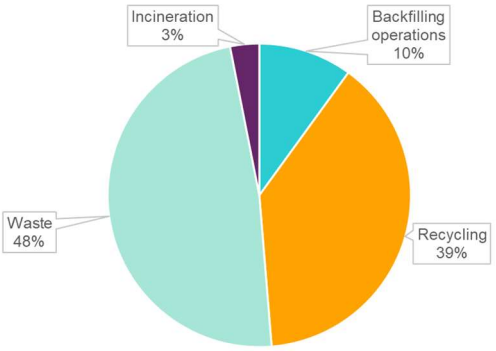


Figure 2: Waste management, 2018 [INE]

Analysing the data published [2] from last years of the final treatment of urban waste by type of treatment, there is no clear decrease in the percentage of waste landfilled.

Table 1: Percentage distribution of the final treatment of urban waste by type of treatment

	Recycling	*Waste*	*Incineration*
2018	34,79	53,61	11,60
2017	36,11	51,16	12,73
2016	33,86	54,12	12,02
2015	29,79	57,76	12,45

Besides, plastic and glass represent the lowest percentage in terms of recycling [1], as shown in Table 2, although many marketing recycling campaigns focus on the treatment of these wastes.

Table 2: Sorting of recycled waste, 2018 [INE]

Reciclado por categoría de residuos. Año 2018
Unidad: miles de toneladas

Residuos reciclados (por tipos)	Reciclado	%	Tasa anual
TOTAL	**47.245,1**	**100,0**	**5,2**
Minerales	21.155,7	44,8	3,7
Metálicos	10.747,6	22,7	1,4
Papel y Cartón	3.912,6	8,3	1,9
Residuos mezclados	3.629,0	7,7	26,3
Animales y vegetales	1.771,6	3,7	8,6
Químicos	1.274,2	2,7	14,3
Lodos comunes	1.272,0	2,7	1,2
Vidrio	1.115,6	2,4	10,6
Madera	967,0	2,0	-5,8
Plásticos	645,4	1,4	38,2
Equipos desechados	570,7	1,2	9,3
Otros	183,7	0,4	2,3

The need of reducing consumption of resources and searching new sustainable solutions to be able to face the ecological risks they cause are two of the main objectives that can be subtracted from this analysis. Both will be reflected in this project.

To understand the reasons why recycling is failing among consumers, CECU (Confederación de Consumidores y Usuarios), carried out a survey in 2019 to find out the opinions of consumers [3]. The results of this survey are summarized below:

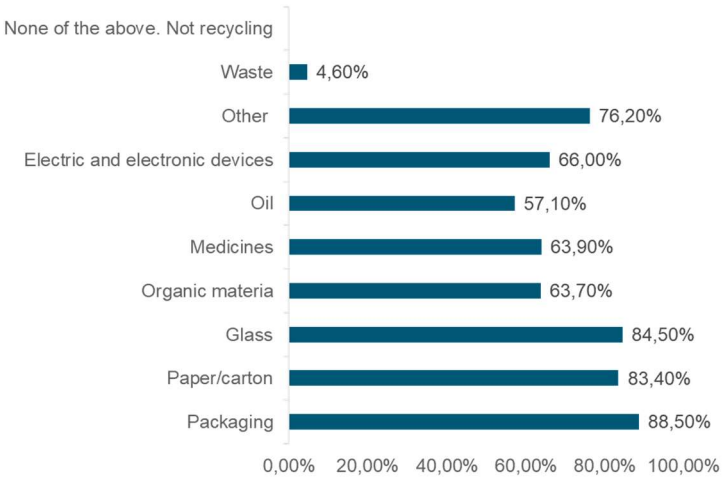


Figure 3: Type of waste recycled in the household, 2019 [CECU]

1. The majority (95.4%) of the Spanish population is in the habit of recycling at least one type of waste generated in their homes. Despite these results, it should be considered that many of these people do not recycle correctly, and many others probably do not recycle as much as they think.
2. Of the group of people who stated that they do not recycle, the reasons given were mainly:
   - Not having enough space at home to do so.
   - Not having recycling containers in their area of residence.
   - Distrust and/or disbelief in the recycling process.

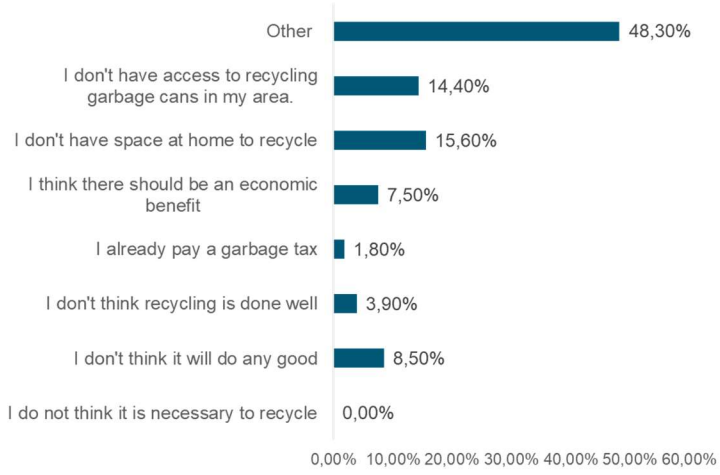


Figure 4: Reasons for not recycling,2019 [CECU]

3. Among the options to facilitate recycling at home, the most voted by the users were:
   - Artificially intelligent garbage bins focused on recycling.
   - Mobile apps on what goes in each garbage can.
   - Design of some containers so that they can be reused for other uses.
   - Designing some containers with materials that can be deformed and take up less space.

- Having an information telephone number where you can consult doubts about recycling of each product.

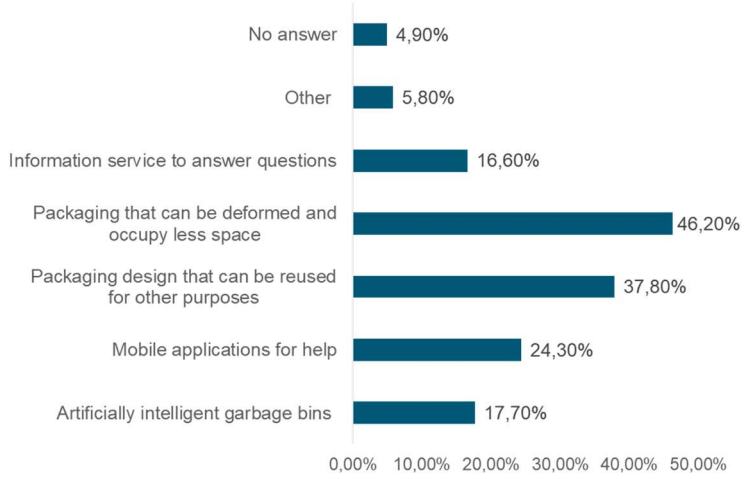


Figure 5: Options that would facilitate recycling at home, 2019[CECU]

4. From the rest of the questions asked to the respondents, it can be deduced that:
   - Most of the citizens interviewed (95.4%) consider climate change to be a major problem.
   - Almost four out of ten citizens consider that responsibility for the deterioration of the environment is shared by all stakeholders (producers, distributors, consumers, and public administrations)

As a conclusion to the study, the first step towards this new model of production and consumption is the implementation of the circular economy model, which is based on the use of waste as raw materials by reducing the consumption of non-reusable products. This model should be followed by both producers and consumers. Below is a graphic to explain the difference between linear and circular economy.

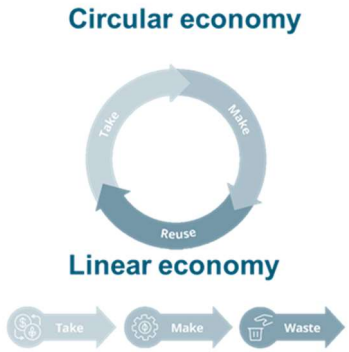


Figure 6: Circular economy vs Linear economy

Consumers represent the starting point of the project: by making easier the recycling process to the consumer, companies will be able to take advantage of the different trash materials and introduce them in the production cycle as raw materials, or even just re-use them directly. That is why this project is focus on attracting consumers to recycle.

To achieve this goal, a smart connected container will be developed to collect, send and act on data thanks to IoT technologies. In addition, many of the points raised in the survey will be answered.

Chapter 2: State of art

2.1. Latest initiatives towards recycling

Several projects based on making recycling easier for the consumer have been developed, most of them using smart IoT solutions. In Spain, the company Ecoembes stands out with the projects listed below:

SmartWaste: promoted by The Circular Lab (Ecoembes) and helped by Minsait (Indra). This innovative platform applies artificial intelligence and big data to improve the efficiency of the collection, selection, and recycling of packaging processes. The objective of Smart Waste is to move forward in the implementation of an effective model of smart city starting from a data collecting tool that can amplify the knowledge about the impact of waste management. It focuses on the use of hybrid vehicles and the intelligent management of trash containers by developing efficient routes in real time. [4]

Reciclos: mobile application for recycling that works as follows [5]:

1. The user scans the bar codes of the item that he wants to recycle.
2. The application checks the item and indicates whether it is valid or not.
3. The user scans the QR code that has been integrated to the municipal recycling container and deposits the item inside.
4. As a reward, the user gets points that can be invested in a social incentive.

The main problem with this initiative is that the use of public containers, where anyone can deposit garbage, does not ensure that all the garbage inside is correctly classified.

In other countries it is quite common the use of bottle recyclers that return money in exchange. In Spain, **Retorna** stands out as a non-profit initiative whose objective is to implement this bottle return system on the stores. This initiative is not very successful due to the following drawbacks:

- The extra cost of the products for the deposit that is "advanced" and returned when the bottle is recovered.
- Difficulty of return because it is necessary to go to the store where each bottle has been purchased. This also means higher CO_2 emissions from the vehicles that must collect them from each establishment.
- It is harmful to bars and restaurants that must manage the return of the bottles in a specific way.
- It reduces the recycling of other residues that does not carry this penalty.

There is another company, **Drago,** that stands out as they have developed an intelligent separation container that recognizes, classifies, and compresses the waste, providing an easier classification system for a better recycling process [6].

Spanish industries have mainly focused on the implementation of IoT platforms in the recycling plants, as is the case of **Urbaser.** This global company is focused on sustainable

environmental management. They have created a real-time information system to manage its fleet of vehicles, the industrial processes carried out and the different assets distributed around the city such as containers [8].

However, this model is focused on small businesses that want to make their own contribution by implementing small sustainable initiatives.

Figure 7: Overview of the initiatives explained above

Recently, a new project has also appeared in Madrid: **Tu Despensa Circular**. This online shop sells multiple beverages in glass bottles, and then offer to pick up the empty bottles in the following 60 days from the purchase [7]. Although this is a great initiative and the cost of the drinks itself is not higher than buying in other supermarkets, the total cost is higher because of the shipping costs and the penalty that exists if the user does not get the bottles back.

Samsung and its Strategy and Innovation Center have also joined this trend by developing a solution for optimizing waste management using the IoT. They have desgined a trash can system that after scanning the items, notifies the users of its recyclability. Thanks to a connected app, consumers not only will learn and recycle correctly but also shop smarter and more sustainable through suggested brands [9].

The strength of the smart connected container ,which is detailed on the following chapters, in comparison to existing alternatives is the porposal that offers which engage users but also large companies that will be able to reuse their packaging, while saving costs and promoting sustainability. In addition, this system is easy to use for all types of users and does not involve an increase in costs for them.

This will be achieved thanks to the implementation of Internet of Things to design a smart connected product. IoT devices will help to operate more efficiently, better undestand costumers and deliver enhanced customer service while increasing the value of the business.

2.2. Smart connected products and IoT systems

Michael E. Porter and James E. Heppelmann defined connected products as the sum of three main elements [10]:

- Physical component: is the product's mechanical and electrical part.
- Smart component: refers to sensor, microprocessors and data storage.
- Connectivity component: relates to the protocol that enables connections.

Smart connected products offer great capabilities to amplify product utilization and add customer value. Thanks to the monitoring, data about condition and operation can be sent, and this data collected can be used for alerts notifications. The connection of the product to the cloud provides the ability to remote control and implementation of programmed algorithms. The flow of data captured on real-time provides fast analysis and historical data allows for in-depth analysis, which will lead to a high level of optimization. Autonomy will be reached once the monitoring, control and optimization process have been fully developed.

The following image [11] on how to design an industry ecosystem, helps to understand the difference between the stages of a smart connected product ecosystem.

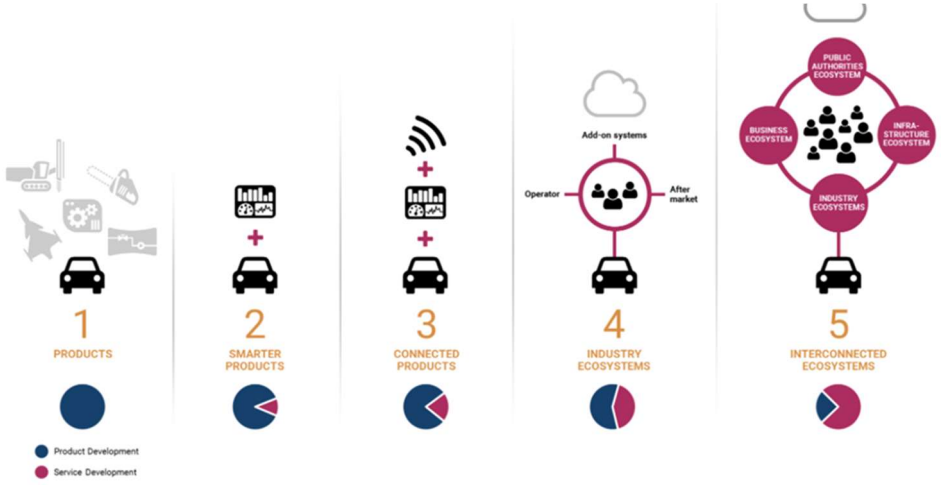


Figure 8: How to design Industry Ecosystem, 2014[Porter & Heppelmann]

This digitalization process is about value creation for customers and partners and can be reached by the implementation of IoT software ecosystems.

Internet of Things, IoT, is the inter-networking of physical devices enhanced with electronics, software, sensors, and network connectivity. One of the key aspects that has contribute to its growing is interoperability. IoT platforms collect and share data from all the devices connected and applies analytics to obtain value information with applications built to address specific needs.

As shown according to a recent study made by McKinsey, IoT will be the most impactful technology by 2025 [12].

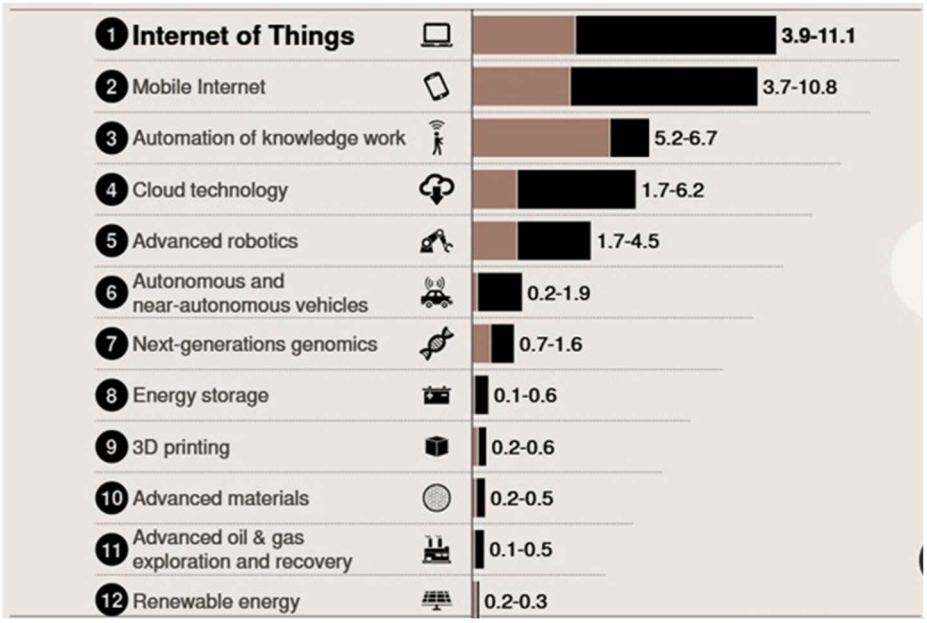


Figure 9: Disruptive technologies by 2025[McKinsey]

IoT platforms provide the tools to create, manage and deploy applications that monitor and control the connected devices, run analytics and store data while securing it. They are designed to support application program that will solve multiple business problems. Platforms include both software and hardware, which might also include above many other functions, operating environment, storage, computing power or development tools.

The main components of an IoT ecosystem are [13]:

Sensors/devices: collect data from the environment.

Connectivity: sensors are connected to a device or a part of it, so they required connectivity to send data to the cloud. IoT communication protocols ensure optimum security of the data being exchanged between the connected devices.

- IoT Networks Protocols: designed to connect devices over the network by allowing data communication within the scope of the network (HTTP, Bluetooth…)
- IoT Data Protocols: designed to connect devices without any internet connection by providing end-to-end communication with the hardware (MQTT, AMPQ…)

In this project, HTTP connection will be used to communicate between devices.

Network: connects from the device back to the cloud.

Data analytics: once data gets to the cloud, processing will be performed on it. The value of IoT is the data generated and the results obtained.

User interface: check in on the system to get information to all the end-users. Sometimes user may also be able to perform different actions that will be affecting the system and others will be performed automatically.

Security: pillar that underlines all the components of an IoT system.

Some of the advantages of using IoT systems are:

- Optimize maintenance.
- Learn more about customers and the way they are using the product to make better decisions about improving requirements.
- Extend the product market lifecycle by continuously delivering code to the edge and monitor the changes.
- Take actions to save loss if products are not being used correctly.
- Track the products from the field to the manufacturing to check the correct delivery.
- Users will have control over their environment. Real-time visualizations will help them understand the use they are doing.
- Facilitate to connect directly with the user and their circumstance when things do not go as planned.
- As software is easily extensible, additional value can be constantly delivered to the customer with apps.

Building these IoT platforms can be challenging as they required many technologies:

- Front-end development to create the product with an app.
- Back-end development to orchestrate the interactions between devices, users, and business logic.
- DevOps to control services and scale them to meet the customers' demand in a secure way.

Enabling all these capabilities required specific software, maintenance and upgrading process. All this is made by the edge development team.

In this project, SmartWorks IoT platform developed by Altair will be used to develop the IoT ecosystem required.

Chapter 3: Project definition

3.1. Objectives of the project

The main objective of this project is to offer large beverage companies a sustainable and efficient solution for the recovery and recycling of their glass bottles.

There are several reasons to implement a traceability system in a beverage factory : the legislation that controls quality and origin guarantee of the products is one of the mandatory reasons and the need of a better organized and improved work along the production chain is one of the voluntary ones. The connected containers can solve these two problems.

Companies will become interested on this project not only because of the recovery of the bottles, but also because it will be a great step forward to a more sustainable business.

Figure 10: Glass bottle recycling chain

3.2. Project planning

The project will be divided in different tasks to achieve the objectives mentioned above.

In the following table there is a task planning distributed over the time that is available (from October to June):

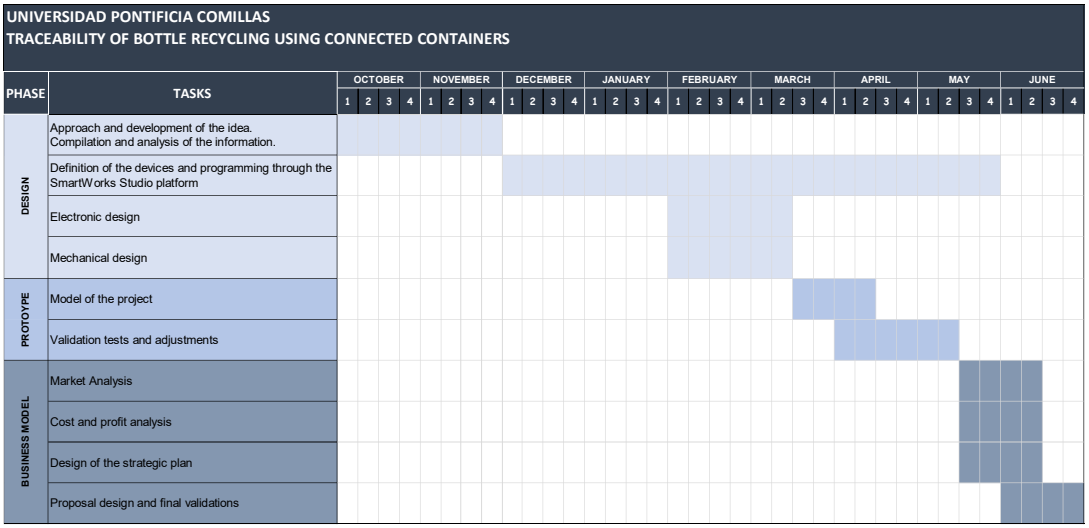


Figure 11: Schedule of the project

(*) Project has been extended one additional month.

3.3. Design overview

3.3.1. General design of the container

The process that will be implemented is summarized on Figure 12: Container processFigure 12:

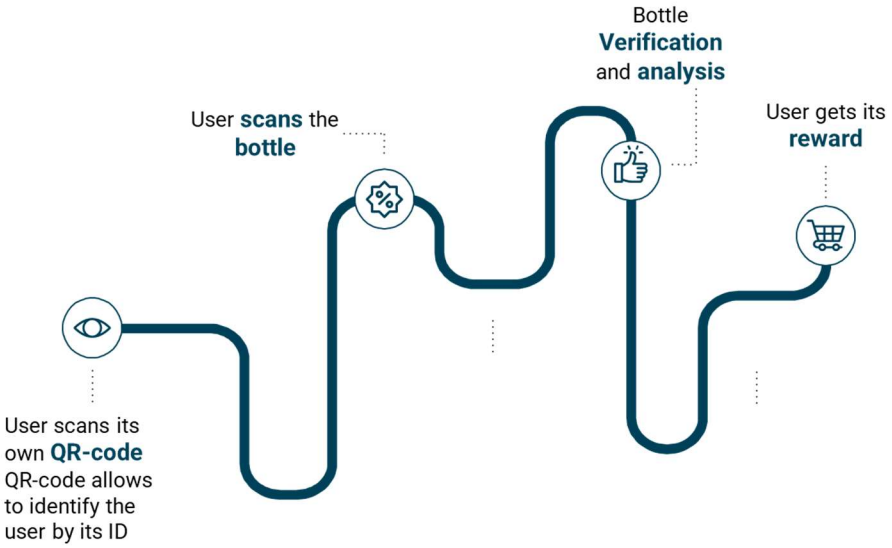


Figure 12: Container process

1ST STEP:

Each of the users will have to be registered on the platform as only registered users can access the containers to ensure that the recycling process is done correctly.

They will have a unique ID that will identify them, and the information will be stored on a QR code.

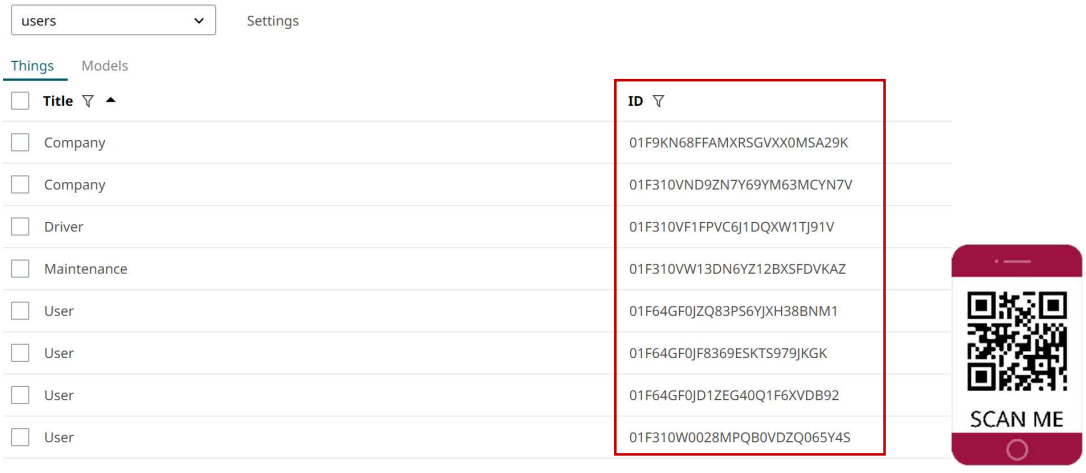


Figure 13: SmartWorks IoT User's overview

Once the user gets to a container, identification will be made by scanning the QR code. The QR code will be read using the camera placed on the surface of the container. If the user is registered in the platform, the display will confirm the user is registered and the container status will change to open.

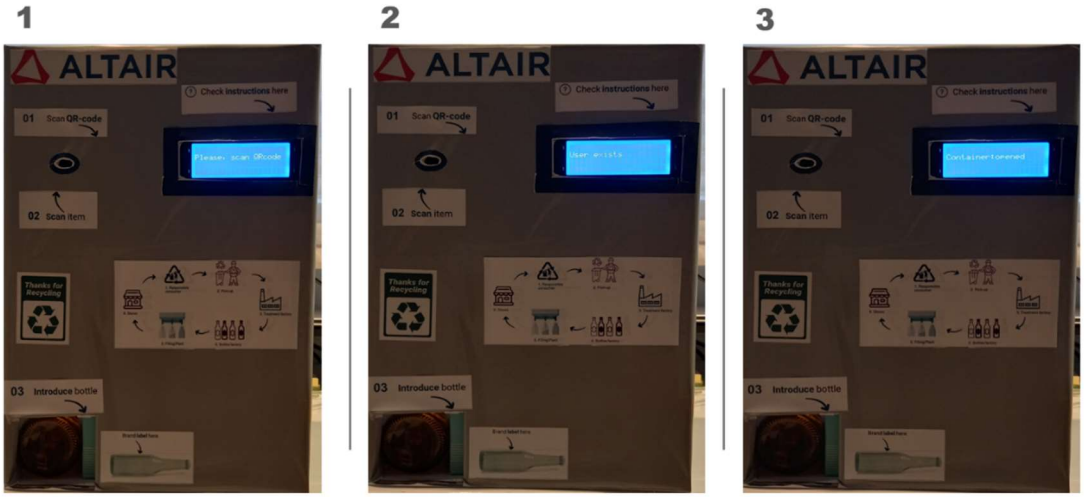


Figure 14: User identification process as shown in the container prototype.

2ND STEP:

After the authentication process, the image processing system will check that the item the user wants to recycle is valid (that it is a bottle) and that it is in good conditions. Display will show the next step to be followed. If the scanned item is invalid, the corresponding message will be displayed.

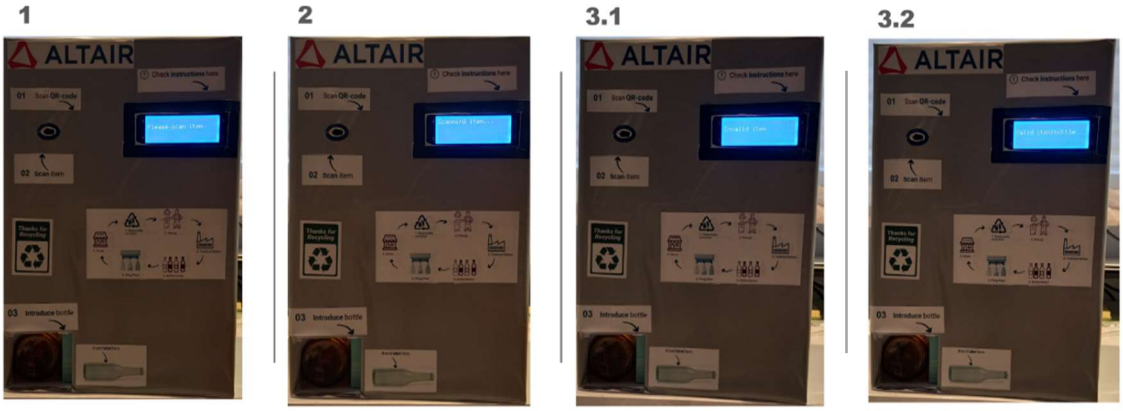


Figure 15: Item identification process as shown in the container prototype.

3RD STEP:

If the bottle is valid, the user will have to place it inside the container so the brand can be analysed. After that, it will be stored on the corresponding box.

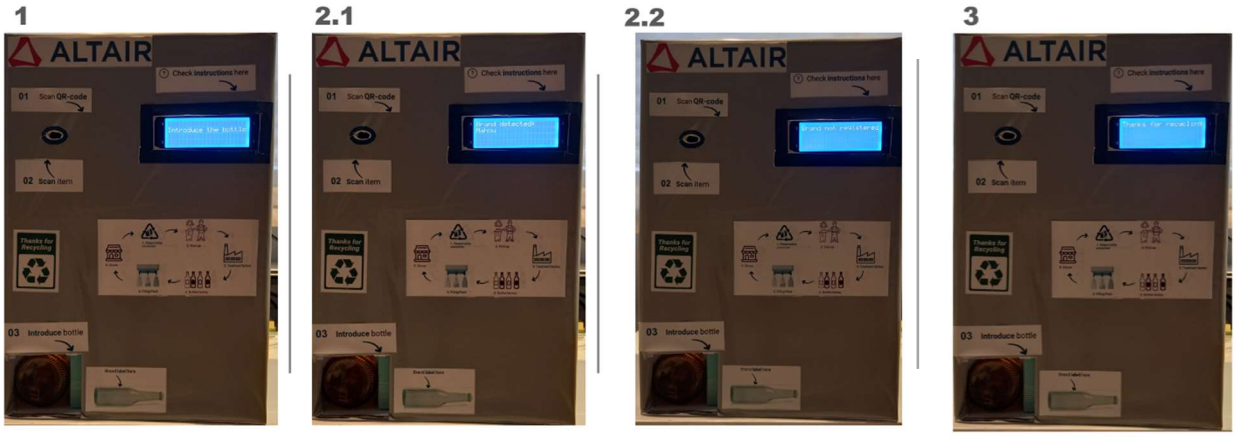


Figure 16: Brand identification process as shown in the container prototype.

4TH STEP:

The user will receive points for each action of recycling. If the bottles recycled are from one of the companies that has an agreement with the project, they will be rewarded with a discount on that brand.

In addition, information about the container will always be available. This allows the transportation company to know the volume of bottles stored to decide when it is necessary to empty them. Maintenance workers will also know if there is any failure and users will have the information of where the containers are placed, and their status.

Each user will have access to data on their recycling progress, accumulated money and if the container is pending to be collected there will be show another container nearby, to avoid overflows, or when it will be emptied.

The client (beverage company) will receive information about recycling trends and consumption by neighbourhoods which will allow them to launch more personalized offers (Example: if you recycle this week, we will double the reward).

Thanks to all the information collected by the containers, more efficient collection routes can be established, and companies will be able to analyse the type of consumer for each area so they will know what products most in demand or what areas are will need to promote projects to raise awareness about recycling.

3.3.2. SmartWorks IoT platform

This project has been developed with the help of Altair and its SmartWorks IoT platform.

Altair SmartWorks IoT [14] provides all what is needed to quickly build scalable, secure web, mobile and edge applications and then iterate fast to find product-market fit. In this project it will be essential to accelerate the IoT smart application development.

Thanks to the tools provided, developing, deploying, and monitoring automation at the edge will be simple for the container designed.

3.3.2.1. Backend development

There is set of modular services that can be customized so that the product will have the configuration needed for the backend application development process.

AnythingDB

Secure database to help smart product developers create a digital model of the entities in the world that their applications can interact with. It allows to store anything from the application and separate the information in different tables called **collections**. The individual elements of a collection are called **things.** Recurrent thing descriptions can be stored in a **model** to easily reuse them.

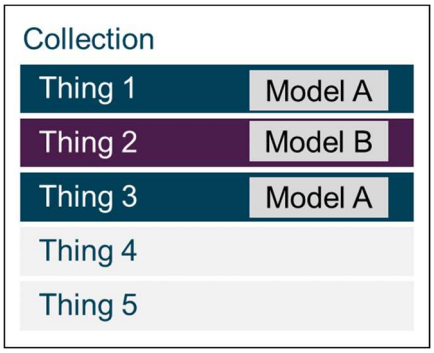


Figure 17: AnythingDB schema

For this project, the following collections have been designed:

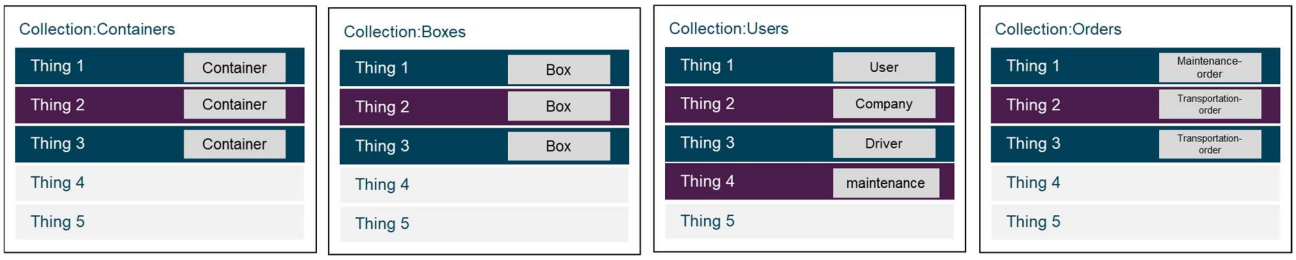


Figure 18: Collections defined at Altair SmartWorks AnythingDB

Each of the collections has multiple things. Things are defined using Web Of Things standard [15] and represent each of the entities in the smart product ecosystem. They store information about real world objects as well as information about concepts, helping create user experiences in applications that fit the data.

- In this project, each container in the collection containers represents one of the connected containers designed.
- The boxes stored at the box collection represent each of the boxes designed to collect bottles so they can be sorted by brands.
- The collection users group the different types of end-users.
- The collection orders stores both type of orders, maintenance and transportation.

Things' schemas allow to describe the devices' attributes by using properties, actions and events.

Properties: describe the devices' attributes expected for a given entity in the real world. The following properties have been defined for each of the models:

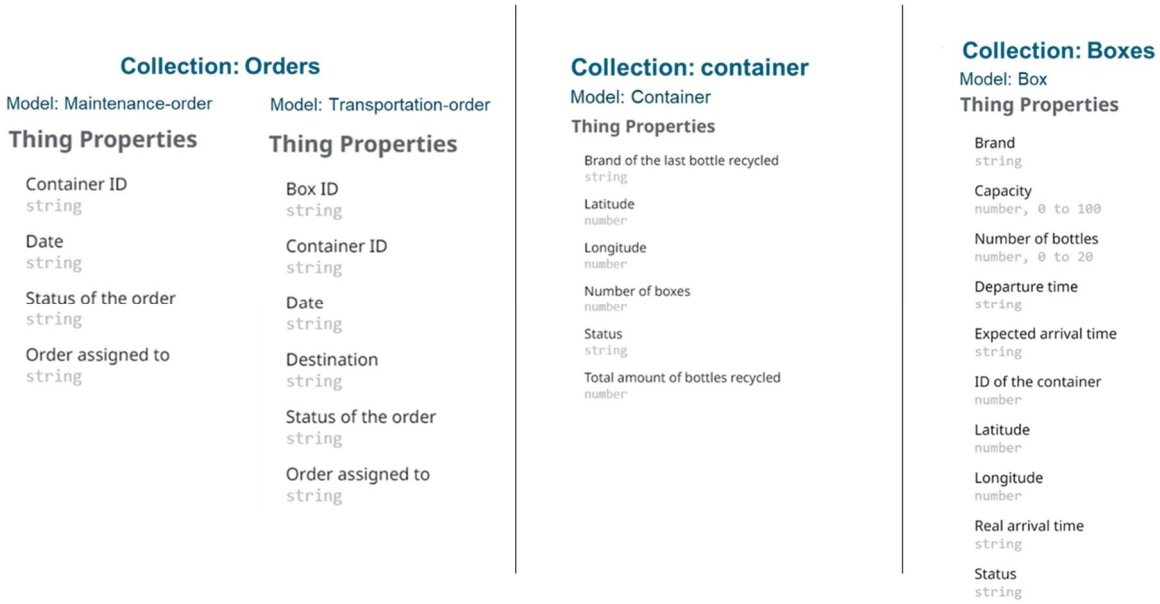


Figure 19: Properties defined for collections (I)

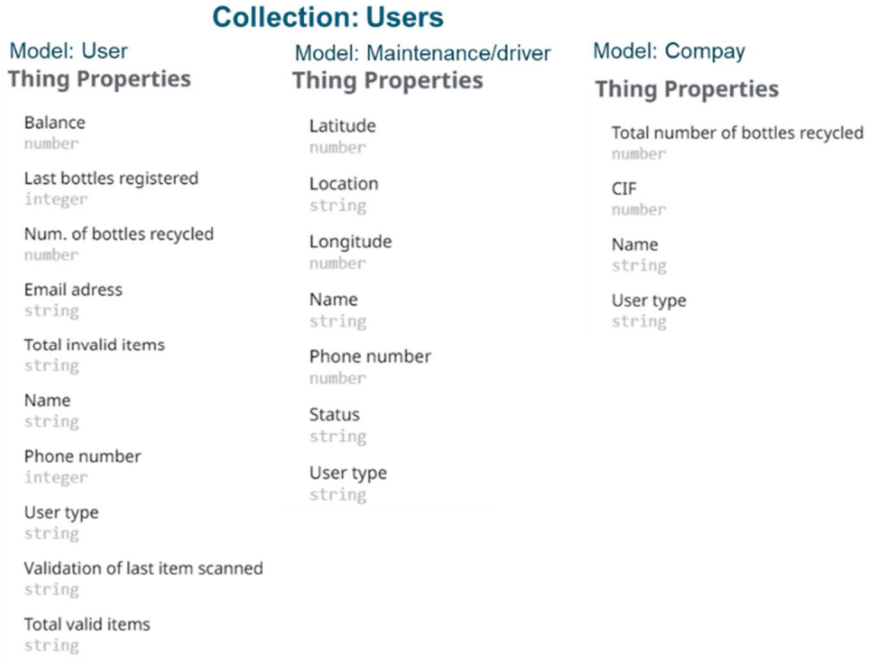


Figure 20: Properties defined for collections (II)

Actions: handle the functions that the devices can carry out. They can manipulate properties and mandate changes in the state of the device.

Events: monitor the changes that cause a modification in a thing's property by providing a log of all the changes.

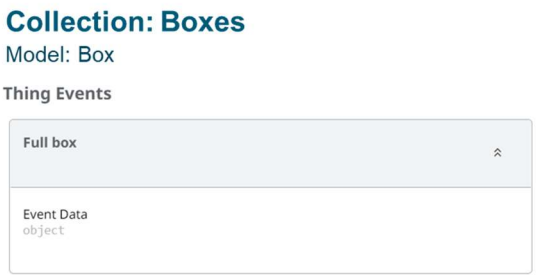


Figure 21: Events defined for boxes' collection.

Functions

Altair SmartWorks Functions provide the capability to build a smart product application by writing code and automatically deploying it without provisioning servers, building an API, or maintaining the infrastructure. It makes the process of establishing the logic for complex applications easy.

Functions have been used in this project to predict feature values and create custom business rules.

Once the functions are coded and running, they need to be invoked. Invoking a Function consists of making a request to its endpoint. Several MQTT triggers have been defined to invoke the functions created.

MQTT

MQTT is the standard for IoT Messaging and allows for communication device to cloud and cloud to device. It follows a publish/subscribe protocol that decouples the client that sends the message from the clients that received it. [16]

MQTT Topics

MQTT topics are paths inside the broker. They are used by the broker to filter the different messages to the corresponding clients. The topic name is a string hierarchically structured. There are two types of topics: set topics (to create and update a thing's properties, actions and events) and status topics (to subscribe to a certain thing 's properties, actions and events).

MQTT Triggers

MQTT triggers work establishing a connection to an MQTT broker and creating a subscription to one or more topics. When a message is received from the things defined before, the Trigger will compare the topic of that message with the topics configured in the Functions. All the matching Functions will be invoked. The payload of the MQTT message is sent in the body of the HTTP invocation request.

To configure the MQTT triggers, MQTT credentials are needed. Thanks to the labels, that enables to categorize things into defined tags, each of the components in the project ecosystem can be labelled to be triggered.

For this project, the following functions have been created:

- **Calculate bottles recycled in a container:** calculate total amount of bottles recycled on each container to know the most demanded areas.

- **Calculate company bottles:** calculate total amount of bottles recycled by a company.
- **Calculate capacity:** based on the number of bottles on each box, the capacity of the box is calculated.
- **Full capacity event:** the boxes designed have a capacity of 180 bottles. Whenever the box is full, an event of full capacity will be registered, and the status of the container will change to "full". There is a data history of the events registered.

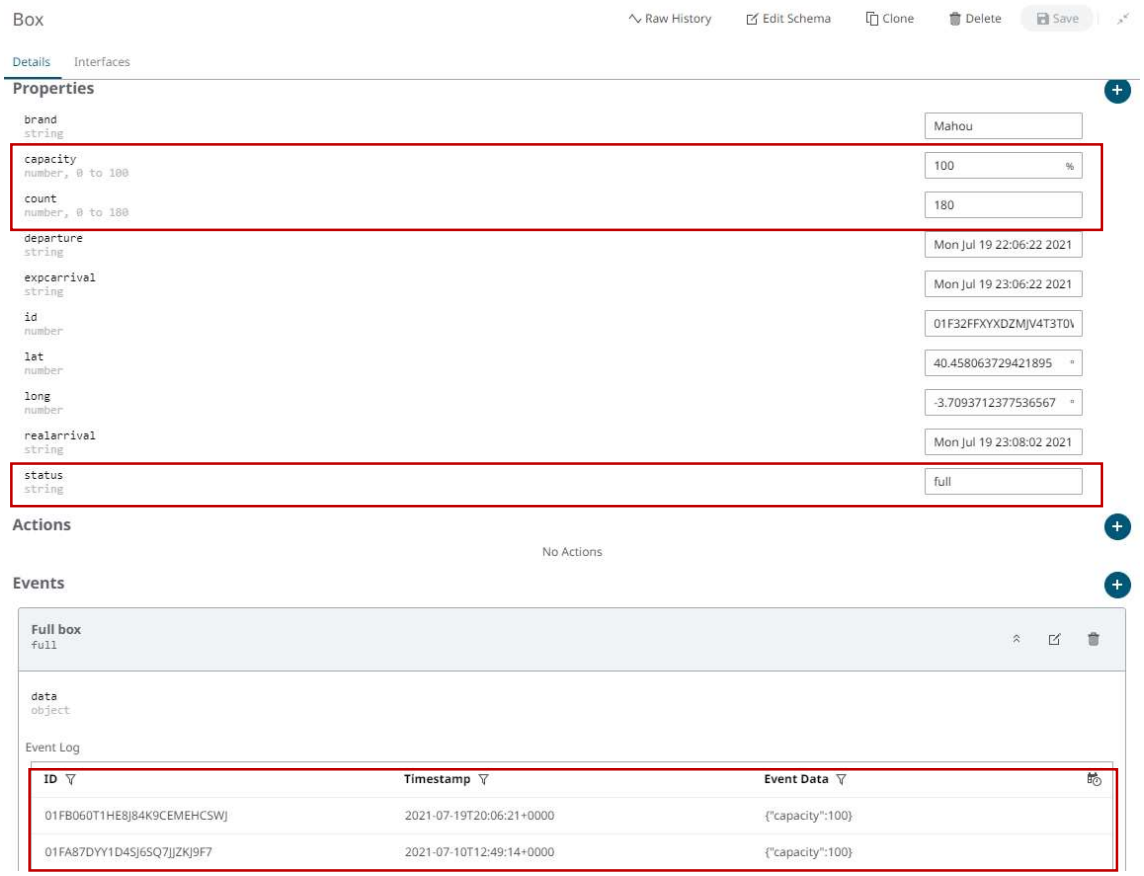


Figure 22: Box properties and events overview

- **Count bottles in box:** if a bottle is correctly identified by the container, it will be stored on the corresponding box and the number of bottles stored on that box will be updated**.**
- **Calculate number of bottles:** when the user recycles a new bottle, it will be added to the total amount of bottles recycled.
- **Set pick-up orders:** if the status of a box changes to full, a transportation order will be assigned to a driver. The status of the driver will change to unavailable and in the collection orders, the order will be filled out.
- **Set maintenance orders:** if the status of a container changes to failure, a maintenance order will be assigned to a maintainer. The status of the maintainer will change to unavailable and in the collection orders, the order will be filled out.
- **Calculate users' balance:** based on the number of bottles recycled, the balance of the user will be calculated and updated.

The codes developed to create these functions can be found at Annex B.

The triggers defined have the topics defined below. Each one of them is monitoring the changes of the properties from all the things labelled with a specific label inside a collection.

- status/containerdemo/labels/01F58GZSNVTTQKENRTPSB9TEGW/collections/boxes/things/+/properties/+
- status/containerdemo/labels/01F310WPSG6G20BYHV5KF18PPM/collections/users/things/+/properties/+
- status/containerdemo/labels/01F9HMT33RXXFJNC12PAMA1XE5/collections/containers/things/+/properties/+

A detailed example of how to correctly define a function and its trigger is explained below:

Function: Calculate bottles recycled in a container

1. All the containers available have been labelled with the same label "working":

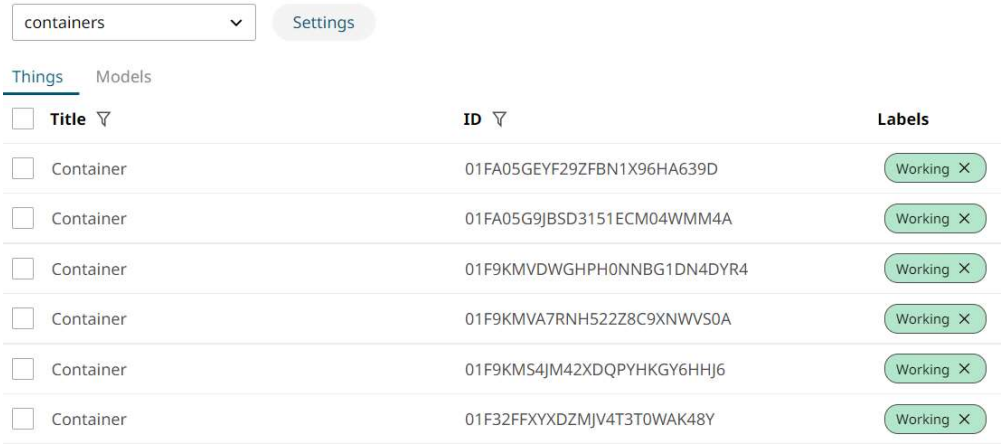


Figure 23: Overview of the containers collection

2. The trigger has been defined as follows:
   - MQTT credentials for the label "Working" (username and password)
   - The MQTT topic defined allows to subscribe to all the containers labelled with this specific label and all their properties.

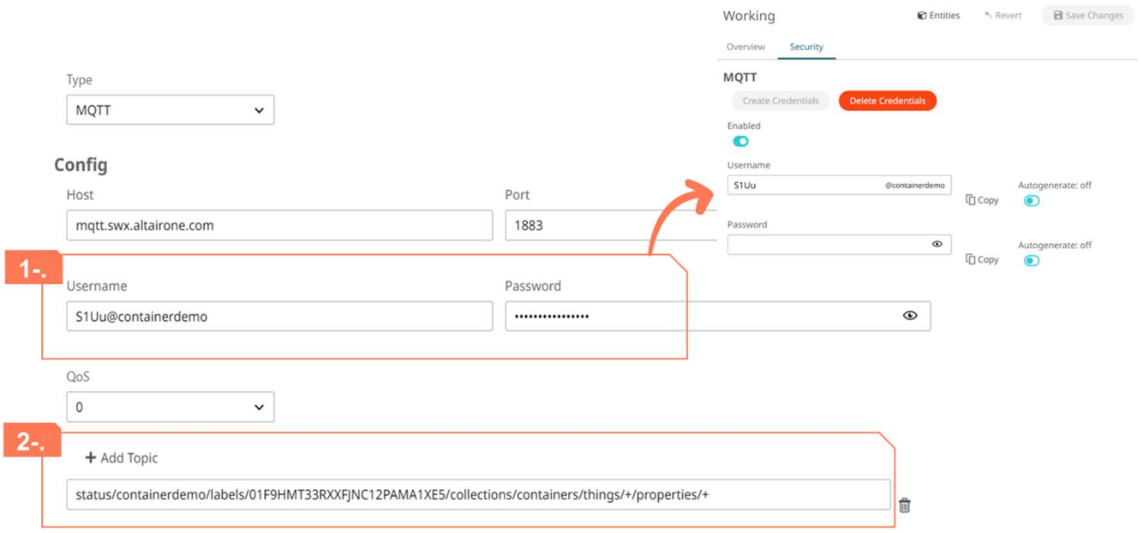


Figure 24: Trigger definition and MQTT credentials

Whenever any of the properties of one of the containers labelled changes, this trigger will be invoking the function **Calculate bottles recycled in a container** as it has been defined with the following topic:

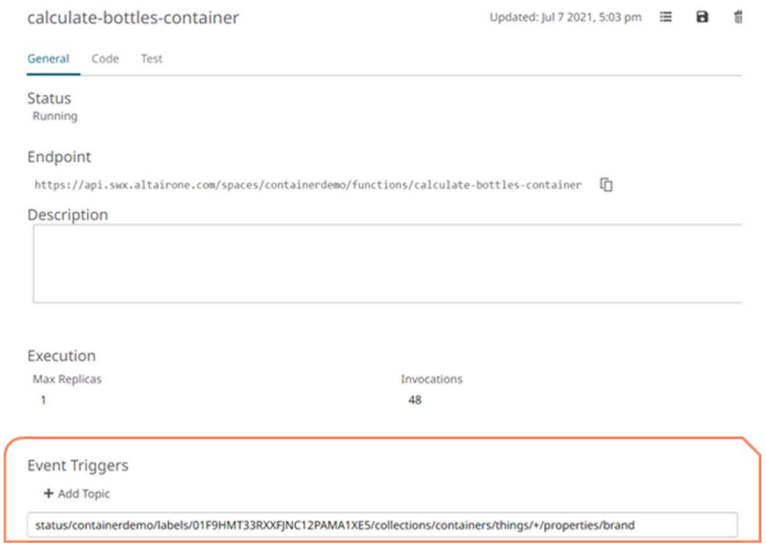


Figure 25: General overview of the function "calculate-bottles-container"

Access Control

Altair SmartWorks Access Control offers the management mechanism to set authorization rules for every piece in the IoT platform. It also includes the tools to control external access to the application, and provision security keys. To improve the security, proof of authentication and authorization is required through the SmartWorks platform. The proof the request must have is the access token. The token lifecycle in the SmartWorks platform is based in OAuth.

In this project, this tool has been useful to:

- Overview the users who have access to the environment - either through the SmartWorks Studio interface or through an end user application.

- Define from generalized to fine grained access control policies to carefully control which users have access to which information from the smart product ecosystem. Drivers, maintenance, users and companies will not be having the same role as each of them will be authorize to certain parts of the application.
- Create Apps to Link SmartWorks to external sources of information. Use scopes to limit their capacity.

3.3.2.2. Frontend development

Altair® Panopticon™ technology provides the functionality and the ability to connect to a wide range of streaming data sources, time series databases and regular databases, both on-premises and in the cloud, and display data with interactive information visualization [14].

Using Real Time Visualization (Panopticon Visualization), each user will have a different dashboard showing the relevant data for them from the smart product in real time. It helps to easily build an application for users fast, and with no code.

Dashboard graphics update continuously as sensor data is flowing in, and they allow for monitoring many potential elements simultaneously.

Each of the end-users have its own dashboard with the relevant information for them.

User

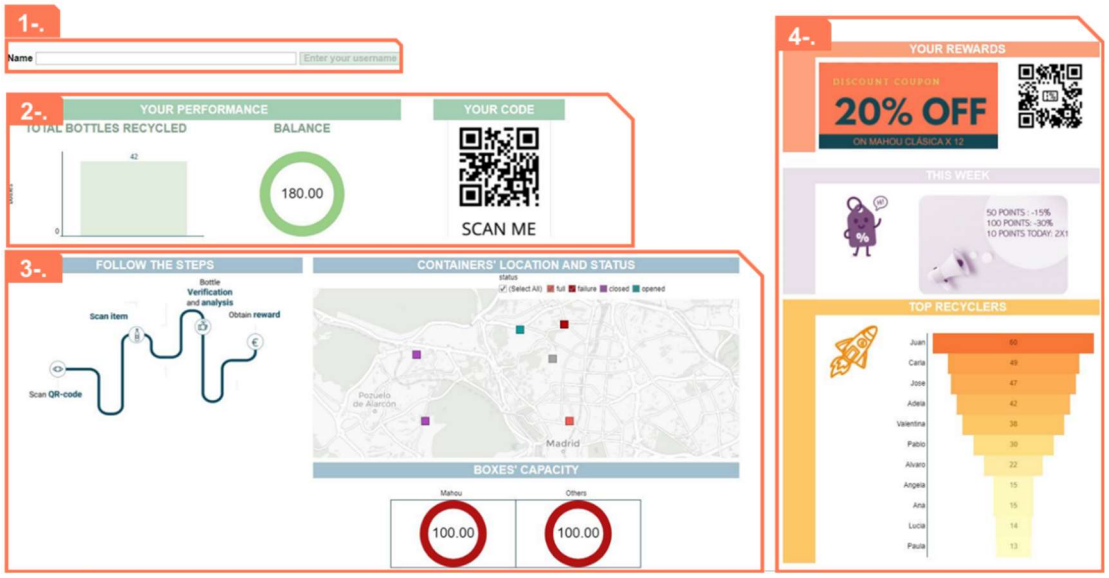


Figure 26: User's dashboard

1. Enter username to identify.
2. Overview of the user's performance (total amount of bottles recycled) and the QR code to get identified.
3. Overview of the system:
   - Instructions to remember how to use the connected container.
   - Location of the containers and their status.
   - Capacity of the boxes inside a container.
4. Rewards of the user and promotions of the week. A chart with the users listed by

total bottles recycled is also shown.

Maintenance

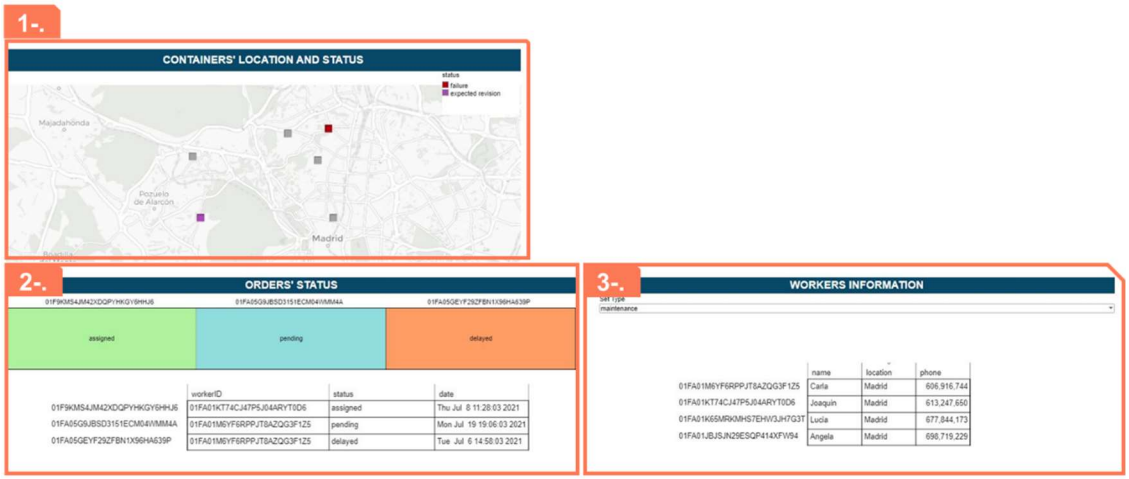


Figure 27: Maintenance dashboard

1. Containers' location and status so the maintenance workers would know where the container that needs to be fixed is.
2. Overview of the orders 'status and the worker assigned to each of them.
3. Workers' information in case they need to be contacted.

Transportation

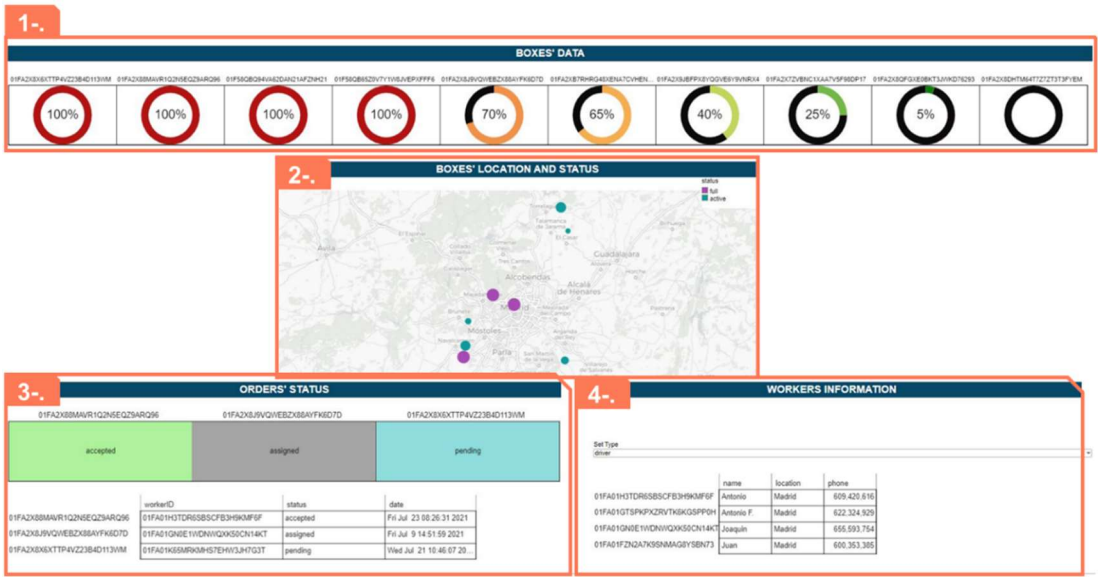


Figure 28: Transportation dashboard

1. Overview of the boxes' capacity.
2. Location and status of the boxes. Thanks to this and the boxes' capacity graph, drivers can decide to pick-up boxes that are in the same route and near to be full.
3. Overview of the orders 'status and the driver assigned to each of them.
4. Drivers' information in case they need to be contacted.

Company

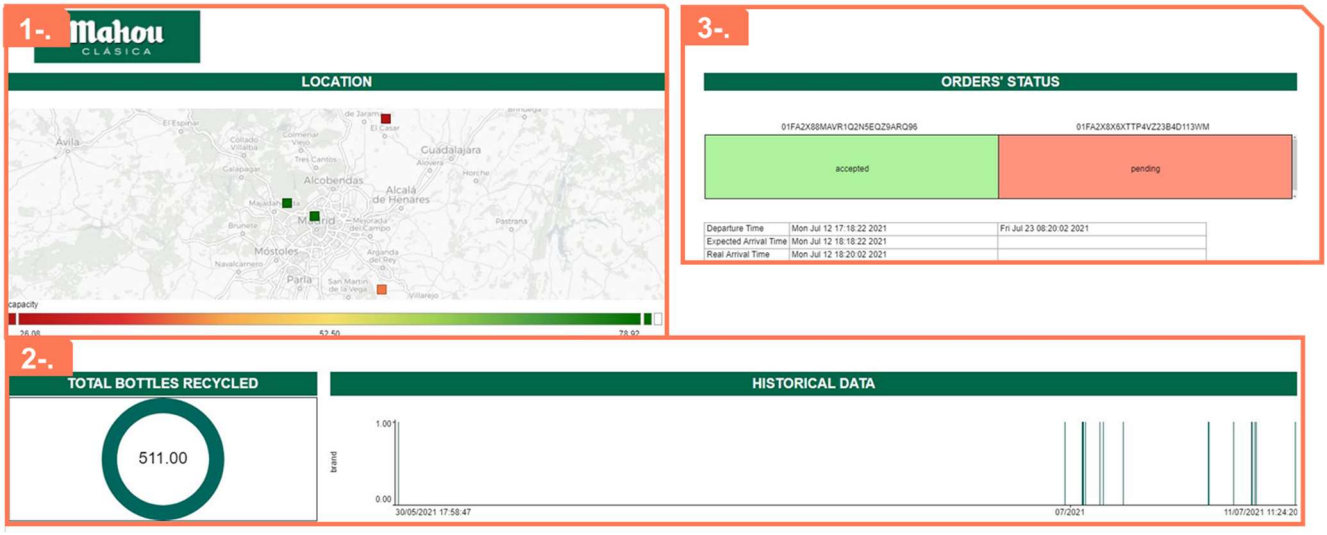


Figure 29: Company dashboard

1. Location of the Mahou's boxes and its capacity.
2. Total amount of bottles recycled and historical data.
3. Overview of the orders' status.

3.3.3. System communication

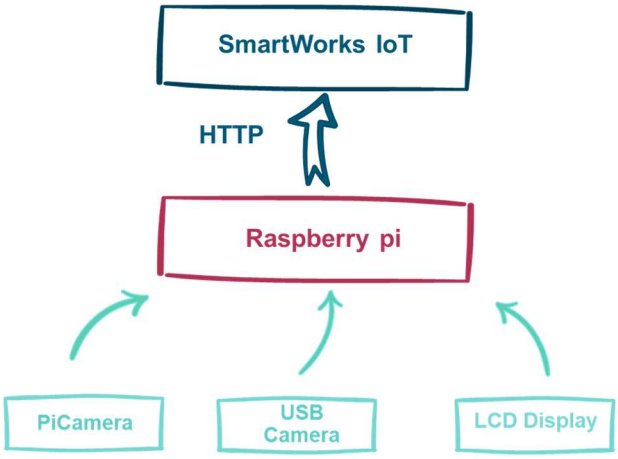


Figure 30: System communication schema

HTTP Protocol

The information store on the different things will be sent to SmartWorks IoT via HTTP (Hyper Text Transfer Protocol), which is a request/response protocol. That is done creating a connection between SmartWorks and the object from an API. To send Data through HTTP a valid client-id and secret-id are needed (they are provided by the platform). Then, using a POST request, the data will be sent.

3.4. Design of the prototype

3.4.1. Prototype design

The design to be carried out consists of a surface container where the bottle will be analysed. It will be storing different boxes to collect the bottles belonging to each company.

A small-scale prototype has been designed. All the steps to be followed are marked on the container.

Figure 31: Prototype of the connected container designed.

Detailed drawings of the connected container and the boxes to store the bottles can be found in Annex F.

3.4.2. Hardware

The project has been developed using a raspberry pi 4. It is a capable device that works as a low-cost computer that plugs into a monitor, standard keyboard, and mouse. It enables to programme different applications and interact with the outside world. [17]

The operating system chosen to work with has been Raspbian, the official supported OS.

The following components are also going to be used:

- **Raspberry Pi Camera module**: with 5 megapixels and fixed focus lens. Can be used to take high-definition video, as well as stills photographs. It is going to be used for analysing the brand of the bottles.

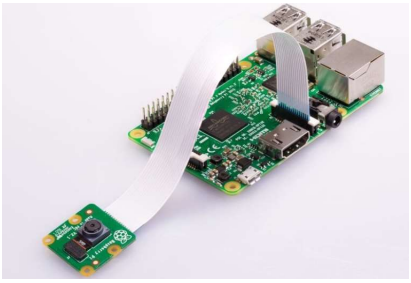


Figure 32: Pi Camera module [https://www.raspberrypi.org/]

- **LCD Module Display**: device that eases the human-computer interaction. It is going to be used for guiding the user through the recycling process.

Figure 33: LCD Display [Amazon]

- **USB Camera**: it is going to be scanning both the QR codes and the items to identify if they are valid or not.

Figure 34: USB Camera [Amazon]

- **Heat Sink and Cooling Fan** have also been installed to ensure the correct maintenance of the device.

Figure 35: Heat sink and cooling fan [Amazon]

3.4.3. Software

The code is written in Python and can be found in Annex B. It follows this structure:

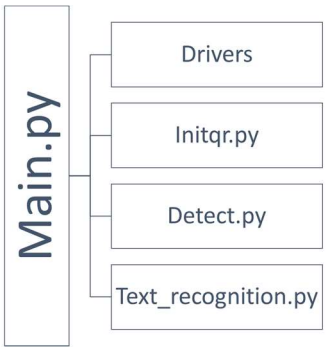


Figure 36: Python codes' structure

Once the raspberry is plugged-in, the main.py python script starts to run. This script makes the call for the other scripts, each one of them is in charge of one of the main process. All of them use the driver's library to show messages on the display to guide the user over the steps.

Initqr.py: reads the data on the QR code and checks if the uid belongs to any of the users of the platform. If the uid is valid, sends the order to open the container and displays the related message. All this data is sent to the platform to have a historic database of each time the container has been opened. It processes the QR images and identifies them by using the open-source library OpenCv.

OpenCV

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. This library has more than 2500 optimized algorithms that can be used to detect and recognize multiple objects, track camera movements and even produce 3D point clouds from stereo camera. It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. [18]

Figure 37: QR Code Python detection result

Detect.py: identifies the type of element the user is trying to recycle and displays a message showing if the item is valid or not. If the item is a bottle, then data is sent to the platform to add one bottle to the user's account. After that, the user is asked to place the bottle inside the container to scanned its brand and store it. All this information will be sent to the corresponding box and company.
This has been done using yolo object detection.

YOLO

This technique uses a type of Convolutional Neural Network called Darknet for image classification and adds the detection part. It uses the famous COCO dataset, which can classify and detect 80 different object classes. This model differs from other classifier systems because it looks at the whole image, so the predictions are informed by global context [19].

Figure 38: Detecting items tests carried out.

Text-recognition.py: analyses the bottle and detect its brand. If the brand is correctly detected, data is sent to the platform so the information about the boxes' capacity is updated. For this purpose, pytesseract has been used.

Pytesseract

Pytesseract is an optical character recognition (OCR) tool for python. It recognizes and reads the text embedded in the image captured by the PiCamera.[20]

Figure 39: Text brand detection tests carried out.

3.4.4. QR (Quick Response) code

QR codes are a type of matrix barcode to store data efficiently and carry information instantly with the scan of a mobile device. Fast readability enables product tracking, item identification , general marketing and commercial applications.

QR codes can be static or dynamic. Static codes contain uneditable information and they can´t track metrics, while dynamic codes can be updated and edited as many times as needed, which makes them the best fit for this marketing purpose.

In this project, a static QR code will be used to identify users when reciclying in one of the connected containers.

Chapter 4: Results analysis

4.1. Initial tests

The first steps of the recycling process for the connected container have been developed as initially planned, but for the item and brand detection process, two previous proposals were made, but the expected results were not obtained. Both are described below:

Check the correct condition of the bottle.

An attempt was made to verify the correct condition of the bottle by isolating parts of the bottle and comparing them with a target image. The results were inaccurate since the same light and colour conditions, same bottle and position were required. The result only worked by comparing two images with minor differences.

In Annex B, codes for detecting the bottles contour and processing the images difference are shown. Following images show some of the results obtained for detecting contours and isolating parts of the bottle and comparing two different images to show differences and detect imperfections.

Note that the tests were made with pictures and not with the images obtained from the cameras, and yet the results were not adequate.

Figure 40: Checking bottle condition tests carried out.

Detect brand by creating a classifier model.

A model was creating using Cascade Trainer GUI, a program that can be used to train, test, and improve a cascade classifier. It uses OpenCV tools for training and testing.

This Trainer works using positive and negative images samples. Positive images are the ones of the item to be detected and classified. Negative images should be anything but the object item.

After creating several versions of this model trained, none of them were as accurate as expected to be for the correct detection of the brand.

Code for this attempt can be found at Annex C.

4.2. Final testing

The results obtained with the final version of the project are satisfactory. The object type detection time could be reduced as this process is still quite latent, but it was the one with the best predictions.

However, sometimes if the user is positioned too close to the camera, the detection system may fail to classify the object properly as shown below.

This could be easily solved by telling the user on the ground where to position himself correctly so that the QR and object detection are correctly done and fast.

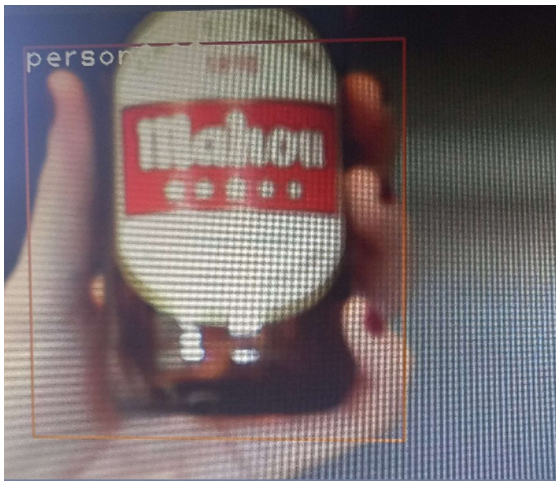


Figure 41: Wrong detection example

Chapter 5: Conclusions and future steps

Although the results obtained are good, new functionalities can be implemented to improve the experience of all users involved by developing new business rules and improving the Things' attributes and logic.

The next step is to migrate this process to the Altair EdgeOps tool.

Altair EdgeOps [21] brings web development tools and methods to the edge so that interaction with the design is easily made. It consists of two major components: Edge Compute Platform and the cloud management interface.

The Edge Compute Platform, ECP, is an open-architecture platform for the edge that will run on the Raspberry Pi. ECP leverages a slimmed down version of Kubernetes. The cloud management interface provides tools for keeping track of all the deployed code, monitoring it, and packaging up code for future deployments.

First steps have already been done and are summarized below:

1. Changed OS from Raspbian to Ubuntu 20.04, as it is the one supported.
2. Create the Docker image for the Linux/arm64 platform. Images are files to execute code in a Docker container. There are set of instructions needed to build the Docker: libraries, dependencies, application code…
3. Containerized the image built and test it on the Raspberry.

After doing the last step, some issues appeared as PiCamera is not supported in Ubuntu. Several changes are going to be made to the code to be able to correctly containerize it.

Once those problems are solved, next steps would be to create a Kubernetes manifests file to use as a resource and then build configurations on Studio to deploy using EdgeOps.

Future steps also involve the improvement of the object detection algorithm. User experience will also be improved through the development of a web application.

Chapter 6: Market study and budget

6.1. Market study

In this section the attractiveness of the product from a financial standpoint is being analysed.

Potential customers

The market for this project is formed by all the large beverage companies that want to make an impact in sustainability by reusing the bottles they used. The connected container can also be extended to other segments. For example, it can be implemented to recover wine bottles or even other materials like plastic bottles or cans.

Coca-Cola has launched a new proposition for the year 2021 called #GoalZero. The idea is to achieve within circular economy a reduction on the difficult-to-recycle materials used for their bottles, reuse those that have already been used to turn them into new resources and recover those that are discarded so that none end up as waste in vulnerable ecosystems such as marine ecosystems.[23]

Coca-Cola company is promoting the use of refillable and returnable glass containers among their Horeca customers (Hotel and Catering). The use of returnable bottles implies a lower use of raw materials, bottle production and waste generation. Once these bottles are collected from bars, cafeterias, restaurants and hotels, they are taken to the packaging plants. After that, they are inspected by means of a meticulous examination with automated sensors that show whether they are damaged, and those that can be reused in the production process are separated from those that cannot. The reusable bottles are refilled at a rate of up to 356,000 bottles per hour and shipped to distribution centres in Spain and Portugal. Each glass bottle can be reused an average of 25 times.

By using the connected container, this process can be applied not only to restaurants but to small consumers, which will allow this objective to be achieved on a larger scale and faster. All types of consumers will be encouraged to recycle even more thanks to the rewards obtained. Special rewards can be applied to restaurants to ensure they treat the bottles correctly so most of them can be refilled.

In line with Coca-Cola's commitment, Mahou's objective for 2030 is to promote the circularity of all packaging and materials, eliminating virgin plastic and ensuring that they are 100% recyclable and reusable. They estimate that 84% of packaging marketed in the hotel and catering industry are returned to their production sites for reuse [24].

As with the Coca-Cola Company, the idea of extending this recycling to small consumers and promoting it by obtaining discounts is interesting for Mahou.

Potential competitors

In Spain, Ecovidrio will be the strongest competitor as they have a special program designed to ease the recycling process to the hotel industry. They offer to locate containers near the clients' location, special containers to transport the bottles to the glass container and maintenance of the containers. The big difference is that they do not care about the conditions of the bottles recover, they just focus on the glass material itself. Another

problem is that these containers do not filter the kind of item introduced, so a filter needs to be carried after.

There are few initiatives related to recover items for its recycled but none of them are designed for glass bottles or connect users with the companies, which is the big bet of the designed container.

6.2. Business model

The strategy is based on offering these two large companies mentioned above the possibility of collaborating to promote their glass bottle recovery projects using the designed connected container. Thanks to the showcase offered by these brands, the container will be publicized, and other brands will want to join this initiative.

Madrid has been the city chosen for this first pilot project. Connected containers will be placed next to the containers that already exists. There are 7614 glass containers distributed all over Madrid [25], but the initial plan is to distribute 2000 along the following areas: Centre, Chamartín, Chamberí, Latina, Retiro, Salamanca y Tetuán (total amount of glass containers in the chosen areas is 2239 [25]). As they are going to be placed near the other public containers, the bottles that are damage will be introduce in those containers, so the bottles recover on each container will be ready for reuse.

For the second year, both companies are expected to renew their contracts and expand to other areas in Madrid. During the third year, they will expand to new cities following the same approach. New companies are expected to join the project on year 4. The expected expansion is summarized in Table 2. The expected sales growth is shown in Figure 42.

Table 2: Overview of the expected expansion

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Number of containers sold	2000	4000	6800	10200	13000	13500	14000

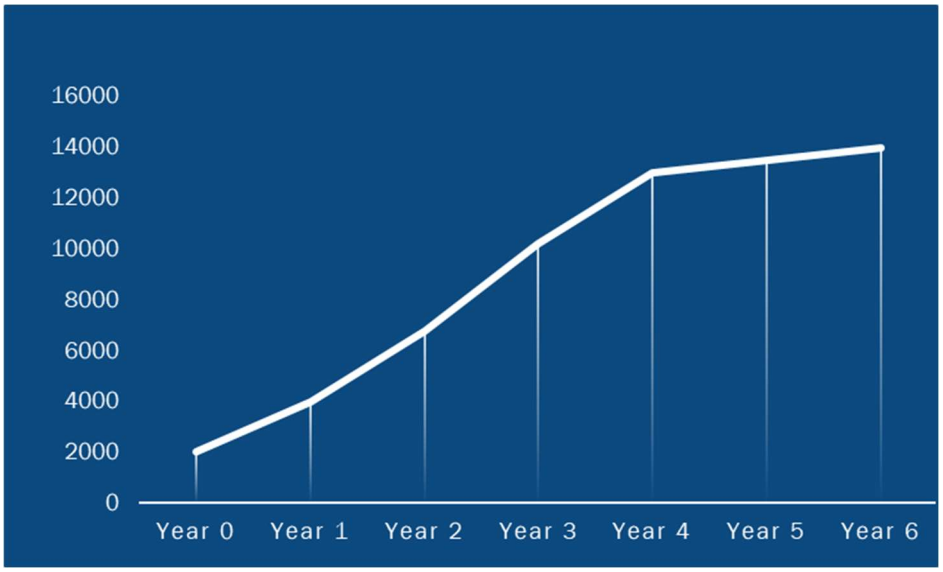


Figure 42: Expected growth of sales.

Companies can employ the same transportation company they have been using to recover the bottles from the restaurants, so that will not be a great extra cost. Moreover, with the

information that the transportation team is going to have thanks to SmartWorks IoT, they would be able to elaborate more efficient routes.

During the first year, valuable feedback to improve both the design and the logic system of the connected container will be given by the clients so new versions will be developed.

A simple application can also be implemented to ease the access to data to the different end-users of SmartWorks IoT if the companies want.

The containers that will be designed for the real development of the project will be like a vending machine, but the process to be followed by users will be the opposite: users will be placing the bottles in the container and the machine will be transporting them to each of the shelves. Each of the containers will be storing 180 bottles by brand.

Figure 43: Design of the final model of the container

The containers fabrication is going to be assigned to an external company but not the assembly of the hardware. The total cost estimated for each container is summarized in Table 3.

Table 3: Costs of the items for the prototype

ITEM	COST
Container	2.000,00 €
Raspberry Pi 4	95,00 €
Materials	15,00 €
Raspberry Pi Camera	13,00 €
LCD Module Display	11,00 €
RPI accesories: carcase, heat and heat sinks	7,00 €
USB Camera	5,00 €
Other accesories	4,00 €
Total	**2.150,00 €**

Due to scale economies, these costs will be reduced and from year 3 onwards, containers' costs will be reduced to 1600 euros.

The business plan is to rent the containers to each company for 150 euros per container. Each company should also pay for SmartWorks IoT services. The cost for this, depends on the services they will be requiring from the platform and the amount of users. For the first year, this service is not taking into account.

Estimations about the profit expected are reflected in Table 4:

Table 4: Profit estimation.

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Estimated costs	4300000	4000000	5600000	5440000	4480000	800000	800000
Estimated revenues	600000	1200000	2040000	3060000	3900000	4050000	4200000
Profit	-3700000	-2800000	-3560000	-2380000	-580000	3250000	3400000

It should be noted that the SmartWorks pricing is not included here.

References

[1] National Institute of Statistics (2020). *Cuentas medioambientales: Cuenta de los residuos. Año 2018*.

[2] National Institute of Statistics (2020). *Estadísticas sobre recogida y tratamiento de residuos. Serie 2015-2018*.

[3] CECU (2019). *Encuesta Comportamiento y percepción del consumidor en materia de medioambiente y sostenibilidad*. **https://cecu.es/publicaciones/Encuesta_reciclaje.pdf**

[4] Ecoembes (s.f). **https://www.thecircularlab.com/smart-waste/**

[5] Reciclos (s.f). **https://www.reciclos.com/es/**

[6] Drago (s.f). **https://www.dra-go.com/**

[7] Tu Despensa Circular (s.f). **https://tudespensacircular.com/**

[8] Urbaser (s.f). **https://www.urbaser.com/**

[9] The School Lab (s.f). **https://theschoolab.com/en/business-innovation-strategy/smart-cycle-how-recycling-can-be-improved-using-iot/**

[10] Harvard Business Review (2019. *How Smart Connected Products Are Transforming Competition*

[11] Combitech (2018). *How an industry could be designed (Originally from Porter and Heppelmann, 2014)*

[12] McKinsey (2019). **https://www.mckinsey.com/featured-insights/internet-of-things/our-insights**

[13] Thales Group (2020). **https://www.thalesgroup.com/en/markets/digital-identity-and-security/iot/inspired/iot-building-blocks**

[14] Altair SmartWorks IoT (2021). **https://help.altair.com/2021/smartworks/topics/overview/sw_intro.htm**

[15] Web Of Things (s.f). **https://www.w3.org/WoT/**

[16] MQTT (s.f). **https://mqtt.org/**

[17] Raspberry Pi (s.f) **https://www.raspberrypi.org/**

[18] OpenCV (s.f) **https://opencv.org/about/**

[19] Yolo (s.f) **https://pjreddie.com/darknet/yolo/**

[20] Pytesseract (s.f) **https://pypi.org/project/pytesseract/**

[21] Altair SmartWorks IoT (s.f). **https://help.altair.com/2021/smartworks/topics/edge_ops/edge_ops.htm**

[22] United Nations (s.f) **https://www.un.org/sustainabledevelopment/es/**

[23] Coca-Cola Company (s.f). **https://www.cocacolaep.com/es/al-dia/blog-rojo-y-en-botella/2021/actuando-sobre-envases-reutilizar/**

[24] Mahou Company (s.f).

[25] Datos Comunidad de Madrid (s.f).
https://datos.madrid.es/portal/site/egob/menuitem.3efdb29b813ad8241e830cc2a8a409a0/?vgnextoid=7ca99505a71d1610VgnVCM2000001f4a900aRCRD&vgnextchannel=102612b9ace9f310VgnVCM100000171f5a0aRCRD&vgnextfmt=default

Annexes

Annex A: Alignment of the project with sustainable development goals

Figure 44: Overview of the SDG involved[https://www.un.org/]

9. Industry, innovation, and infrastructure

This project will make it possible to update and restructure industries in favour of sustainability, using resources more efficiently and adopting clean and environmentally sound industrial technologies and processes [22].

11. Sustainable cities and communities

As stated in point 6 of this objective, the negative environmental impact will be reduced by taking care of the management of municipal and other types of waste.

12. Responsible production and consumption

This is perhaps the objective with the most presence, since the basis of this project includes most of the goals it reflects:

12.2 To achieve sustainable management and efficient use of natural resources.

12.4 To achieve an ecologically rational management.

12.5 To significantly reduce the generation of waste through prevention, reduction, recycling and reuse activities.

12.6 Encourage companies, especially large companies and transnational corporations, to adopt sustainable practices and incorporate sustainability information into their reporting cycle.

12.7 Promote public procurement practices that are sustainable, in accordance with national policies and priorities.

12.8 Ensure that people everywhere have information and knowledge relevant to sustainable development and lifestyles in harmony with nature.

13. Climate action

As can be deduced from the previous objective, another of the things that will be achieved is to diminish the effect of climate change derived from bad waste management.

15. Life of terrestrial ecosystems

In line with the above, by reducing pollution, the quality of drinking water will be preserved, desertification and the reduction of land quality will be avoided, the effect of global warming will be reduced, and the conditions of flora, fauna and people's health will be improved.

Annex B: Functions' codes

Calculate bottles recycled in a container

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    token=get_access_token()

PATH="/spaces/containerdemo/collections/containers/things-
status/01F32FFXYXDZMJV4T3T0WAK48Y"
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("GET", API_HOST + PATH ,
headers=headers)
    properties=response.json()['properties']
    total=properties['total']
    total =total+1

PATH="/spaces/containerdemo/collections/containers/things/0
1F32FFXYXDZMJV4T3T0WAK48Y/properties/total"
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("PUT", API_HOST + PATH,
headers=headers,
                                json={"total": total})

    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Calculate bottles by company

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"


def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    token=get_access_token()
    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''

PATH="/spaces/containerdemo/collections/containers/things-
status/01F32FFXYXDZMJV4T3T0WAK48Y"
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("GET", API_HOST + PATH ,
headers=headers)
    properties=response.json()['properties']
    brand=properties['brand']

    if brand =="Mahou":
      PATH="/spaces/containerdemo/collections/users/things-
status/01F9KN68FFAMXRSGVXX0MSA29K"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("GET", API_HOST + PATH ,
headers=headers)
      properties=response.json()['properties']
      bottles=properties['bottles']

      bottles =bottles+1
```

```
PATH="/spaces/containerdemo/collections/users/things/01F9KN
68FFAMXRSGVXX0MSA29K/properties/bottles"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST + PATH,
headers=headers,json={"bottles": bottles})

    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Calculate capacity

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''

    token=get_access_token()

    body = req.body.decode("utf-8")
    body = json.loads(body)
    count = body['count']

    capacity=count*100/180

PATH="/spaces/containerdemo/collections/boxes/things/{}/pro
perties/capacity".format(thing_id)
```

```
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("PUT", API_HOST + PATH,
headers=headers,json={"capacity": capacity})

    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Full capacity event

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F58H1HB1T06HBWN6C3Q5GBM0"
CLIENT_SECRET = "UUxs8PsI8mx6S9IDNhMGIiNF5e0CEv"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']


def handle(req):
    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''

    body = req.body.decode("utf-8")
    body = json.loads(body)
    capacity = body['capacity']

    if capacity < 100:

        return {"status_code":204}

    data={
      "full":{
        "data":{
           "capacity":capacity
```

```
        }
      }

    }

    status="full"

PATH="/spaces/containerdemo/collections/boxes/things/{}/properties/status".format(thing_id)
    headers = {"Authorization": "Bearer " + get_access_token()}
    response = requests.request("PUT", API_HOST + PATH, headers=headers,
                                json={"status": status})

PATH="/spaces/containerdemo/collections/boxes/things/{}/events/full".format(thing_id)

    headers = {"Authorization": "Bearer " + get_access_token()}
    response = requests.request("POST", API_HOST + PATH, headers=headers,json=data)

    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Count bottles in box

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-urlencoded'}
    response = requests.request("POST", API_HOST + "/oauth2/token",headers=headers, data=payload)
```

```
    return response.json()['access_token']


def handle(req):
    global count
    token=get_access_token()
    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''

    body = req.body.decode("utf-8")
    body = json.loads(body)
    brand = body['brand']

    if brand == "Mahou":
      PATH="/spaces/containerdemo/collections/boxes/things-
status/01F58QB65Z0V7Y1W8JVEPXFFF6"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("GET", API_HOST + PATH ,
headers=headers)
      properties=response.json()['properties']
      count=properties['count']

      count =count+1

PATH="/spaces/containerdemo/collections/boxes/things/01F58Q
B65Z0V7Y1W8JVEPXFFF6/properties/count"
      response = requests.request("PUT", API_HOST + PATH,
headers=headers,json={"count": count})

    else:
      PATH="/spaces/containerdemo/collections/boxes/things-
status/01F58QBQ94VA62DAN21AFZNH21"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("GET", API_HOST + PATH ,
headers=headers)
      properties=response.json()['properties']
      count=properties['count']

      count =count+1

PATH="/spaces/containerdemo/collections/boxes/things/01F58Q
BQ94VA62DAN21AFZNH21/properties/count"
      response = requests.request("PUT", API_HOST +
PATH,headers=headers,json={"count": count})

    return {
        "body": response.json(),
```

```
        "status_code": response.status_code
    }
```

Calculate number of bottles

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"


def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    token=get_access_token()
    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''

    PATH="/spaces/containerdemo/collections/users/things-
status/{}".format(thing_id)
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("GET", API_HOST + PATH ,
headers=headers)
    properties=response.json()['properties']
    bottles=properties['bottles']

    bottles =bottles+1

PATH="/spaces/containerdemo/collections/users/things/{}/pro
perties/bottles".format(thing_id)
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("PUT", API_HOST + PATH,
headers=headers,json={"bottles": bottles})

    return {
```

```
        "body": response.json(),
        "status_code": response.status_code
    }
```

Set pick-up order

```
import json
import requests
import time
import random

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''

    token=get_access_token()

    PATH="/spaces/containerdemo/collections/boxes/things-
status/{}".format(thing_id)
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("GET", API_HOST + PATH ,
headers=headers)
    properties=response.json()['properties']
    status = properties['status']

    if status=="full":

PATH_ID="/spaces/containerdemo/collections/orders/things/01
F30ZSFS8ZT73N7KJSD6E4KQ8/properties/containerID"
      headers = {"Authorization": "Bearer " + token}
```

```
      response = requests.request("PUT", API_HOST +
PATH_ID, headers=headers,json={"containerID": thing_id})

      state="assigned"

PATH_STATE="/spaces/containerdemo/collections/orders/things
/01F30ZSFS8ZT73N7KJSD6E4KQ8/properties/status"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_STATE, headers=headers,json={"status": state})

worker=random.choice(["01FA01H3TDR6SBSCFB3H9KMF6F","01FA01G
TSPKPXZRVTK6KGSPP0H","01FA01GN0E1WDNWQXK50CN14KT","01FA01FZ
N2A7K9SNMAG8YSBN73"])


PATH_WORKER="/spaces/containerdemo/collections/users/things
/{}/properties/status".format(worker)
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_WORKER, headers=headers,json={"status":
"unavailable"})


PATH_WID="/spaces/containerdemo/collections/orders/things/0
1F30ZSFS8ZT73N7KJSD6E4KQ8/properties/workerID"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_WID, headers=headers,json={"workerID": worker})

      today=time.ctime()

PATH_date="/spaces/containerdemo/collections/orders/things/
01F30ZSFS8ZT73N7KJSD6E4KQ8/properties/date"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_date, headers=headers,json={"date": today})
      time_seconds=time.time()
      dep =time_seconds + 7200
      depart=time.ctime(dep)

PATH_dep="/spaces/containerdemo/collections/boxes/things/{}
/properties/departure".format(thing_id)
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_dep, headers=headers,json={"departure": depart})
      expc =time_seconds + 10800
```

```
      expected=time.ctime(expc)

PATH_expc="/spaces/containerdemo/collections/boxes/things/{
}/properties/expcarrival".format(thing_id)
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_expc, headers=headers,json={"expcarrival": expected})

      realarrival =time_seconds + 10900
      real=time.ctime(realarrival)

PATH_real="/spaces/containerdemo/collections/boxes/things/{
}/properties/realarrival".format(thing_id)
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_real, headers=headers,json={"realarrival": real})

    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Set maintenance order

```
import json
import requests
import time
import random
API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    topic =str(req.headers.get('X-Topic'))
```

```
    thing_id = topic.split('/')[7] if topic else ''

    token=get_access_token()

PATH="/spaces/containerdemo/collections/containers/things-
status/{}".format(thing_id)
    headers = {"Authorization": "Bearer " + token}
    response = requests.request("GET", API_HOST + PATH ,
headers=headers)
    properties=response.json()['properties']
    status = properties['status']

    if status=="failure":

PATH_ID="/spaces/containerdemo/collections/orders/things/01
F30YGM3PKSKBVAHMHD2DWD84/properties/containerID"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_ID, headers=headers,json={"containerID": thing_id})

      state="assigned"

PATH_STATE="/spaces/containerdemo/collections/orders/things
/01F30YGM3PKSKBVAHMHD2DWD84/properties/status"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_STATE, headers=headers,json={"status": state})

worker=random.choice(['01FA01M6YF6RPPJT8AZQG3F1Z5','01FA01K
T74CJ47P5J04ARYT0D6','01FA01K65MRKMHS7EHW3JH7G3T','01FA01JB
JSJN29ESQP414XFW94'])

PATH_WORKER="/spaces/containerdemo/collections/users/things
/{}/properties/status".format(worker)
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_WORKER, headers=headers,
json={"status": "unavailable"})

PATH_WID="/spaces/containerdemo/collections/orders/things/0
1F30YGM3PKSKBVAHMHD2DWD84/properties/workerID"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_WID, headers=headers,json={"workerID": worker})

      today=time.ctime()
```

```
PATH_date="/spaces/containerdemo/collections/orders/things/
01F30YGM3PKSKBVAHMHD2DWD84/properties/date"
      headers = {"Authorization": "Bearer " + token}
      response = requests.request("PUT", API_HOST +
PATH_date, headers=headers,json={"date": today})


    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Calculate users 'balance

```
import json
import requests

API_HOST = "https://api.swx.altairone.com"
CLIENT_ID = "app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
CLIENT_SECRET = "V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"

def get_access_token():
    payload = f'grant_type=client_credentials&' \
              f'client_id={CLIENT_ID}&' \
              f'client_secret={CLIENT_SECRET}&' \
              f'scope=thing'

    headers = {'Content-Type': 'application/x-www-form-
urlencoded'}
    response = requests.request("POST", API_HOST +
"/oauth2/token",headers=headers, data=payload)

    return response.json()['access_token']

def handle(req):

    topic =str(req.headers.get('X-Topic'))
    thing_id = topic.split('/')[7] if topic else ''
    body = req.body.decode("utf-8")
    body = json.loads(body)
    bottles = body['bottles']

    balance = bottles*2.5

PATH="/spaces/containerdemo/collections/users/things/{}/pro
perties/balance".format(thing_id)
    headers = {"Authorization": "Bearer " +
get_access_token()}
```

```
    response = requests.request("PUT", API_HOST + PATH,
headers=headers,json={"balance":balance})

    return {
        "body": response.json(),
        "status_code": response.status_code
    }
```

Annex C: Python codes

Main.py

```
import smartworks_sdk as swx
import requests
import json
import pandas as pd
import Initqr
import drivers
from time import sleep
import detect
import text_recog as brand

collection="users"
client="app::01F64MDT8H7NX254TVA36X0KQ4"
secret="WMtCZklCK7LFBC4AjpmAeujjJHJvGA"
space="containerdemo"
host="https://api.swx.altairone.com"
namespace="{}/spaces/{}".format(host,space)

header=swx.get_smartworks_headers(client,secret,URL='https:
//api.swx.altairone.com',scope='thing')
url="{}/collections/{}/things".format(namespace,collection)
response=requests.get(url=url,headers=header)

display = drivers.Lcd()

try:

        display.lcd_display_string("Please scan QRcode ",
1)
        sleep(3)
        display.lcd_clear()
        things=json.loads(response.text)
        pd_things=pd.DataFrame(things['data'])
        uid=[]
        uid=pd_things['uid']
        Initqr.initQR(uid)
        detect.detect_items()
        brand.brand()

except KeyboardInterrupt:

    print("Cleaning up!")
    display.lcd_clear()
```

Initqr.py

```
import smartworks_sdk as swx
import cv2
import drivers
from time import sleep

space="containerdemo"
host="https://api.swx.altairone.com"
namespace="{}/spaces/{}".format(host,space)

client_id="containerdemo::01F32FFXYXDZMJV4T3T0WAK48Y"
secret_id="5jTbDz1HHsXtTHt8GlE3GMJ6HOg3aH"
thing_id="01F32FFXYXDZMJV4T3T0WAK48Y"
propcollection="containers"

def initQR(uid):
 global data
 cap = cv2.VideoCapture(0)
 detector = cv2.QRCodeDetector()

 while True:

  check, img = cap.read()
  data, bbox, _ = detector.detectAndDecode(img)

  if bbox is not None:
      for i in range(len(bbox)):
          cv2.line(img, tuple(bbox[i][0]), tuple(bbox[(i+1)
% len(bbox)][0]), color=(255, 0, 0), thickness=2)
          cv2.putText(img, data, (int(bbox[0][0][0]),
int(bbox[0][0][1])), cv2.FONT_HERSHEY_SIMPLEX,0.5, (0, 255,
0), 2)

  if data:
      print("Data:", data)
      cv2.destroyAllWindows()
      for i in range(len(uid)):
          if uid[i]==data:
              display = drivers.Lcd()
              display.lcd_display_string("User exists ", 1)
              sleep(5)
              display.lcd_clear()
              display.lcd_display_string("Container:opened
", 2)
              sleep(5)
```

```
                display.lcd_clear()
                status='open'
                headers =
swx.get_smartworks_headers(client_id,secret_id,URL =
host,scope ='thing.update%20data.create')
                urlProperties=
"{}/collections/{}/things/{}/properties".format(namespace,p
ropcollection,thing_id)
                urlStatus=urlProperties+ "/status"
                urlData=namespace+'/data'

                payload={
                   "status":status
                }

response=swx.http_post(url=urlData,payload=payload,headers=
headers,print_out=True)

                property_status={
                   "status":status
                }

response=swx.http_put(url=urlStatus,payload=property_status
,headers=headers,print_out=True)
                display.lcd_display_string("Please,scan
item", 2)
                sleep(3)
                display.lcd_display_string("It might take a
moment", 1)
                sleep(5)
                display.lcd_clear()
                return

  cv2.imshow("QR Code detector", img)
  cv2.waitKey(1)
  if cv2.waitKey(1) >-1:

      break
```

Detect.py

```
import cv2
import numpy as np
import drivers
from time import sleep
import smartworks_sdk as swx
import Initqr
```

```
space="containerdemo"
host="https://api.swx.altairone.com"
namespace="{}/spaces/{}".format(host,space)

client_id="app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
secret_id="V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"
propcollection="users"

def detect_items():
    net=cv2.dnn.readNet('yolov3.weights','yolov3.cfg')
    classes=[]
    with open('coco.names','r') as f:
        classes=f.read().splitlines()

    cap=cv2.VideoCapture(0)

    while True:
        _,img=cap.read()
        height,width,_=img.shape

blob=cv2.dnn.blobFromImage(img,1/255,(416,416),(0,0,0),swap
RB=True, crop=False)

        net.setInput(blob)

output_layers_names=net.getUnconnectedOutLayersNames()
        layerOutputs=net.forward(output_layers_names)

        boxes=[]
        confidences=[]
        class_ids=[]

        for output in layerOutputs:
            for detection in output:
                scores=detection[5:]
                class_id=np.argmax(scores)
                confidence=scores[class_id]
                if confidence>0.5:
                    center_x=int(detection[0]*width)
                    center_y=int(detection[1]*height)
                    w=int(detection[2]*width)
                    h=int(detection[3]*height)

                    x=int(center_x-w/2)
                    y=int(center_y-h/2)
                    boxes.append([x,y,w,h])
                    confidences.append((float(confidence)))
```

```
                    class_ids.append(class_id)

        indexes=cv2.dnn.NMSBoxes(boxes,confidences,0.5,0.4)

        font=cv2.FONT_HERSHEY_PLAIN
        colors=np.random.uniform(0,255,size=(len(boxes),3))

        if len(indexes)!=0:
         for i in indexes.flatten():
            x,y,w,h=boxes[i]
            label=str(classes[class_ids[i]])
            confidence=str(round(confidences[i],2))
            color=colors[i]
            cv2.rectangle(img,(x,y),(x+w,y+h),color,2)
            cv2.putText(img,label + "" +
confidence,(x,y+20),font,2,(255,255,255),2)

            cv2.imshow('Image',img)
            if label=="bottle":
                bcount=1
                display = drivers.Lcd()
                display.lcd_display_string("Valid
item:bottle", 2)
                sleep(5)
                display.lcd_clear()
                headers =
swx.get_smartworks_headers(client_id,secret_id,URL =
host,scope ='thing')
                urlProperties=
"{}/collections/{}/things/{}/properties".format(namespace,p
ropcollection,Initqr.data)
                urlBcount=urlProperties+ "/bcount"
                urlData=namespace+'/data'
                payload={
                   "bcount":bcount
                }

response=swx.http_post(url=urlData,payload=payload,headers=
headers,print_out=True)

                property_bcount={
                   "bcount":bcount
                }


response=swx.http_put(url=urlBcount,payload=property_bcount
,headers=headers,print_out=True)
```

```
                display = drivers.Lcd()
                display.lcd_display_string("Put the bottle
inside the container", 2)
                sleep(5)
                display.lcd_clear()
                return

            else:
                display = drivers.Lcd()
                display.lcd_display_string("Invalid item ",
2)
                sleep(5)
                display.lcd_clear()

            print(label)
        key= cv2.waitKey(1)
        if key>-1:
            cap.release()
            cv2.destroyAllWindows()
            break
```

text_recog.py

```
import cv2
import pytesseract
import time
from time import sleep
import picamera
import matplotlib.pyplot as plt
import numpy as np
import drivers
import smartworks_sdk as swx
from picamera.array import PiRGBArray

space="containerdemo"
host="https://api.swx.altairone.com"
namespace="{}/spaces/{}".format(host,space)

client_id="app::01F6Z2BCCDWY4ZBZCF6M14NBKA"
secret_id="V3WWAI2wQf83Mbuf4HIZ4jdqTrTh6G"
collection="containers"
brand=[]
def ocr_core(image):
    text=pytesseract.image_to_string(image)
    return text
def brand():
    with picamera.PiCamera()as camera:
        camera.start_preview()
```

```
        time.sleep(5)
        camera.capture('capture.jpg')
    image=cv2.imread('capture.jpg')
    cv2.imshow("img",image)
    brand=ocr_core(image)
    print("brand:",brand)
    b=brand.find("ORIGINAL")
    bb=brand.find("UCLASICA")

    if b!=-1 or bb!=-1:
        display = drivers.Lcd()
        display.lcd_display_string("Brand detected: Mahou",
 2)
        sleep(5)
        display.lcd_clear()
        headers = swx.get_smartworks_headers(client_id,secr
et_id,URL = host,scope ='thing')
        urlProperties= "{}/collections/{}/things/01F32FFXYX
DZMJV4T3T0WAK48Y/properties".format(namespace,collection)
        urlBrand=urlProperties+ "/brand"
        urlData=namespace+'/data'
        payload={
           "brand":"Mahou"
        }

        response=swx.http_post(url=urlData,payload=payload,
headers=headers,print_out=True)

        property_brand={
           "brand":"Mahou"
        }

        response=swx.http_put(url=urlBrand,payload=property
_brand,headers=headers,print_out=True)
        display = drivers.Lcd()
        display.lcd_display_string("Thanks for recycling",
2)
        sleep(5)
        display.lcd_clear()

    else:
        display = drivers.Lcd()
        display.lcd_display_string("Brand not registered ",
 2)
        sleep(5)
        display.lcd_clear()
```

Annex D: Other testing codes

Identify_body.py

```
import cv2

img=cv2.imread(r"C:\Users\preinoso\OneDrive\TFM\python\OK\img.jpg")
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
retval, thresh = cv2.threshold(gray_img, 127, 255, 0)
img_contours, _ = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

cv2.drawContours(img, img_contours, -1, (0, 255, 0))
cv2.imshow('Image Contours', img)
cv2.waitKey(0)

height, width = img.shape[0:2]

#Bottle body
startRow = int(height*0.55)
startCol = int(width*0.4)
endRow = int(height*1)
endCol = int(width*0.6)
croppedImage = img[startRow:endRow, startCol:endCol]
edge_img = cv2.Canny(croppedImage,100,200)
#Identify borders
cv2.imshow("Detected Edges", edge_img)
cv2.waitKey(0)
```

brand_detector.py

```
import cv2

cap = cv2.VideoCapture(1,cv2.CAP_DSHOW)

BottleClassif = cv2.CascadeClassifier('cascade.xml')

while True:

    ret,frame = cap.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    bottle = BottleClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))

    for (x,y,w,h) in bottle:

        cv2.rectangle(frame, (x,y),(x+w,y+h),(0,255,0),2)

        cv2.putText(frame,'Bottle',(x,y-10),2,0.7,(0,255,0),2,cv2.LINE_AA)
```

```
    cv2.imshow('frame',frame)


    if cv2.waitKey(1) == 27:
        break
cap.release()
cv2.destroyAllWindows()
```

Annex E: Json Things' descriptions

Container-model


```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "brand": {
      "title": "Brand of the last bottle recycled",
      "type": "string"
    },
    "lat": {
      "title": "Latitude",
      "type": "number",
      "unit": "°"
    },
    "long": {
      "title": "Longitude",
      "type": "number",
      "unit": "°"
    },
    "numboxes": {
      "title": "Number of boxes",
      "type": "number",
      "unit": ""
    },
    "status": {
      "title": "Status",
      "type": "string",
      "unit": ""
    },
    "total": {
      "title": "Total amount of bottles recycled",
      "type": "number"
    }
  },
  "title": "Container"
}
```


Transportation-user-model


```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "lat": {
```

```
      "title": "Latitude",
      "type": "number",
      "unit": "°"
    },
    "location": {
      "title": "Location",
      "type": "string",
      "unit": ""
    },
    "long": {
      "title": "Longitude",
      "type": "number",
      "unit": "°"
    },
    "name": {
      "type": "string"
    },
    "phone": {
      "title": "Phone number",
      "type": "number",
      "unit": ""
    },
    "status": {
      "title": "Status",
      "type": "string"
    },
    "type": {
      "title": "User type",
      "type": "string"
    }
  },
  "title": "Driver"
}
```


Company-user-model


```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "bottles": {
      "title": "Total number of bottles recycled",
      "type": "number"
    },
    "cif": {
      "title": "CIF",
      "type": "number"
```

```
    },
    "name": {
      "title": "Name",
      "type": "string"
    },
    "type": {
      "title": "User type",
      "type": "string"
    }
  },
  "title": "Company"
}
```


Maintenance-user-model


```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "lat": {
      "title": "Latitude",
      "type": "number",
      "unit": "°"
    },
    "location": {
      "title": "Location",
      "type": "string"
    },
    "long": {
      "title": "Longitude",
      "type": "number",
      "unit": "°"
    },
    "name": {
      "title": "Name",
      "type": "string"
    },
    "phone": {
      "title": "Phone number",
      "type": "number"
    },
    "status": {
      "title": "Status",
      "type": "string"
    },
    "type": {
      "title": "User type",
```

```
      "type": "string"
    }
  },
  "title": "Maintenance"
}
```

User-model

```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "balance": {
      "title": "Balance",
      "type": "number",
      "unit": "points"
    },
    "bcount": {
      "title": "Last bottles registered",
      "type": "integer"
    },
    "bottles": {
      "title": "Num. of bottles recycled",
      "type": "number",
      "unit": ""
    },
    "email": {
      "title": "Email adress",
      "type": "string",
      "unit": "@altair.com"
    },
    "invalid": {
      "title": "Total invalid items",
      "type": "string"
    },
    "name": {
      "title": "Name",
      "type": "string"
    },
    "phone": {
      "title": "Phone number",
      "type": "integer"
    },
    "type": {
      "title": "User type",
      "type": "string"
    },
```

```
    "validation": {
      "title": "Validation of last item scanned",
      "type": "string"
    },
    "valids": {
      "title": "Total valid items",
      "type": "string"
    }
  },
  "title": "User"
}
```


Transportation-order-model


```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "bix": {
      "title": "Box ID",
      "type": "string"
    },
    "containerID": {
      "title": "Container ID",
      "type": "string"
    },
    "date": {
      "title": "Date",
      "type": "string",
      "unit": ""
    },
    "destination": {
      "type": "string"
    },
    "status": {
      "title": "Status of the order",
      "type": "string",
      "unit": ""
    },
    "workerID": {
      "title": "Order assigned to",
      "type": "string"
    }
  },
  "title": "Transportation-order"

}
```

Maintenance-order-model

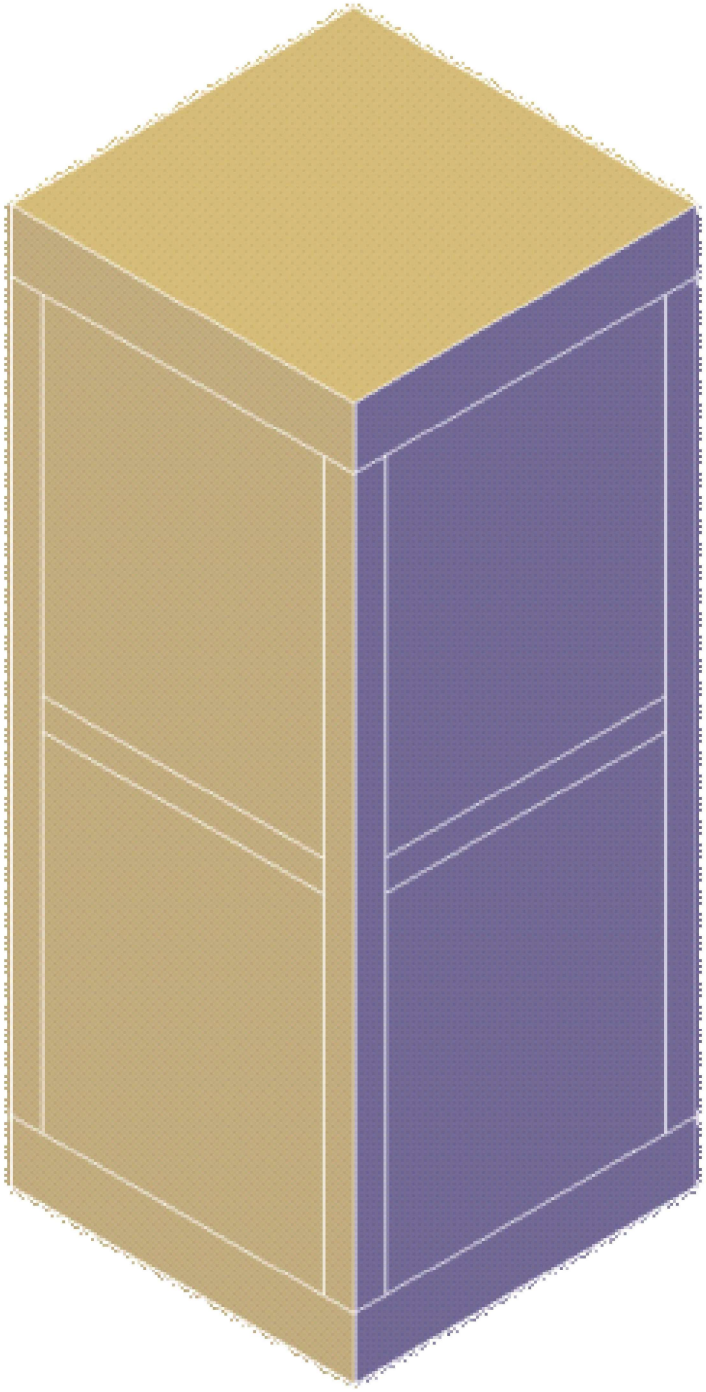
```
{
  "actions": {},
  "description": "",
  "events": {},
  "properties": {
    "containerID": {
      "title": "Container ID",
      "type": "string"
    },
    "date": {
      "title": "Date",
      "type": "string",
      "unit": ""
    },
    "status": {
      "title": "Status of the order",
      "type": "string",
      "unit": ""
    },
    "workerID": {
      "title": "Order assigned to",
      "type": "string"
    }
  },
  "title": "Maintenance-order"
}
```

Box-model

```
{
  "actions": {},
  "description": "",
  "events": {
    "full": {
      "data": {
        "type": "object"
      },
      "title": "Full box"
    }
  },
  "properties": {
    "brand": {
      "title": "Brand",
      "type": "string"
    },
    "capacity": {
      "maximum": 100,
```

```
      "minimum": 0,
      "title": "Capacity",
      "type": "number",
      "unit": "%"
    },
    "count": {
      "maximum": 20,
      "minimum": 0,
      "title": "Number of bottles",
      "type": "number"
    },
    "departure": {
      "title": "Departure time",
      "type": "string",
      "unit": ""
    },
    "expcarrival": {
      "title": "Expected arrival time",
      "type": "string",
      "unit": ""
    },
    "id": {
      "title": "ID of the container",
      "type": "number"
    },
    "lat": {
      "title": "Latitude",
      "type": "number",
      "unit": "°"
    },
    "long": {
      "title": "Longitude",
      "type": "number",
      "unit": "°"
    },
    "realarrival": {
      "title": "Real arrival time",
      "type": "string",
      "unit": ""
    },
    "status": {
      "type": "string"
    }
  },
  "title": "Box"
}
```

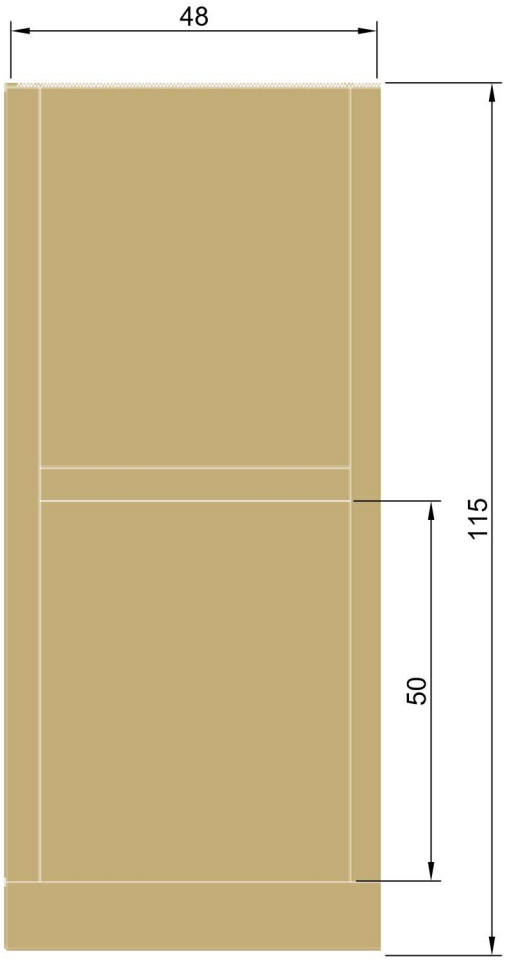
Annex F: Blueprints

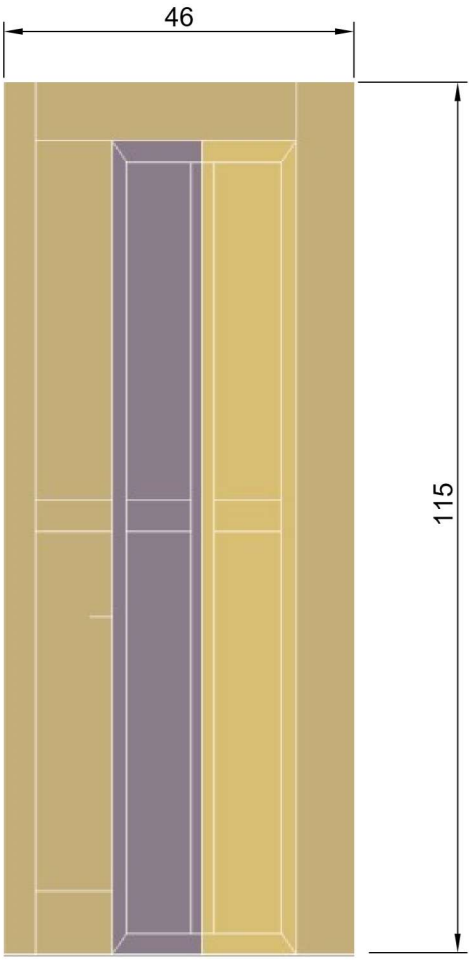


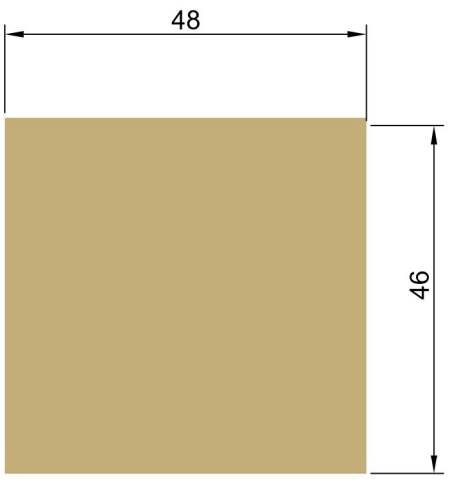
MATERIAL			
TOLERANCIA			
	NOMBRE	FECHA	Prototipo contenedor conectado
DIBUJADO	Reinoso Otero Paloma	20/07/2021	
COMPROBADO			
ESCALA: 1:1	FIRMA PalomaReinoso	I.C.A.I.	Nº DE LAMINA: 1

Traceability of glass recycling using connected containers

UNIVERSIDAD PONTIFICIA COMILLAS ICAI
Máster en Ingeniería Industrial







MATERIAL			
TOLERANCIA			
	NOMBRE	FECHA	Prototipo contenedor conectado
DIBUJADO	Reinoso Otero Paloma	20/07/2021	
COMPROBADO			
ESCALA: 1:1	FIRMA PalomaReinoso	92	I.C.A.I.
			Nº DE LAMINA: 2