

# RFID CUPONES

## Instalación, Desarrollo y Generalidades del aplicativo de lector de cupones

Javier Jesús Vela Cadena

### ABSTRACT

The purpose of this work is to develop an application applying the concepts and themes related to the Python programming language and the interaction of hardware devices such as ARDUINO, ESP32, RASPBERRY, etc.

During the process of learning concepts related to PYTHON & DEVNET they promote solidity and simplicity in their syntax and operation, in addition to their learning curve that will allow us to interact with networks, systems, hardware, etc. Highlighting the versatility that will allow us to develop direct interaction with software and / or hardware will allow its growth in the technological area of many companies at a national and international level.

### RESUMEN

El presente trabajo tiene como propósito desarrollar un aplicativo aplicando los conceptos y temáticas referidas al lenguaje de programación Python e interacción de dispositivos hardware como ARDUINO, ESP32, RASPBERRY, etc.

Durante el proceso de aprendizaje de conceptos referentes a PYTHON & DEVNET fomentan solides y simplicidad en su sintaxis y funcionamiento, además que su curva de aprendizaje nos permitirá interactuar con redes , sistemas , hardware , etc.

Destacando la versatilidad que nos permitirá a la hora de desarrollar interacción directa con el software y/o hardware permitirá su crecimiento en el ar3ea tecnológicas de muchas empresas a nivel nacional e internacional.

### PALABRAS CLAVES

RFID – CUPONES – APLICATIVO – PYTHON – ANGULAR – ARDUINO – DEVNET – DOCKER – FLASK – API – FRONTEND – SOCKETS – SERVER – CLIENT – HARDWARE – SOFTWARE – READER – CARD – COMPILAR – DOCKERFILE – MYSQL – REQUIREMENTS

### I. INTRODUCCIÓN

Los sistemas de información, programación con sensores en hardware, diseño y estructuración de redes son parte esencial en los negocios e ideas contemporáneas a día de hoy.

Cuando referenciamos el área de negocios de cupones han utilizado talonarios o bloques de papeles que justifican el uso con productos o servicios de otros negocios.

La idea principal del proyecto se justifica por medio de un aplicativo de tarjetas que nos permita consumirlos sin necesidad del consumo de más papel o talonarios sino virtualizar cada producto por medio de tarjetas asignadas a las respectivas personas y de esta manera acortar costos a largo plazo para las empresas.

El enfoque principal del proyecto es gestionar y aplicar los conceptos aprendidos referentes al lenguaje de programación Python e interacción con hardware como Arduino, esp32 , raspberry , etc , aplicando los por medio de un proyecto que nos permita establecer el dominio y control permitido de estructuración , condiciones , variables ,etc.

Además, la idea es establecer soluciones optimas para las empresas de cupones enfocadas a medios digitales como lo son tarjetas personalizadas a los usuarios por medio de lectores RFID y de esta manera llevar un control de una manera idónea y concisa para optimización de recursos ambientales y económicos para estas empresas.

### II. PREREQUISITOS

Para instalar y/o desplegar el aplicativo de tarjetas RFID es necesario tener en cuenta los siguientes programas, configuraciones y sistemas requeridos:

- Implementar algún sistema operativo como Windows, Linux, MacOS, cualquiera de los siguientes operativos funciona para desplegar el aplicativo.
- Docker Desktop: es necesario con el fin de contenerizar la aplicación y sea bastante fácil al momento de instalar o desplegar los programas y sus dependencias de manera optima y sencilla. Puedes descargar el programa por medio del siguiente

enlace: <https://www.docker.com/products/docker-desktop>

- Python lenguaje de programación esencial para la instalación y/o despliegue de nuestro aplicativo, Puedes descargar el programa por medio del siguiente enlace : (se recomienda versiones igual o superiores a la 3.6)  
<https://www.python.org/downloads/>
- Arduino IDE (Opcional) : es un programa útil que nos permite gestionar los scripts INO , dispositivo Arduino , y puentes de serial. Puedes descargar el programa por medio del siguiente enlace :  
<https://www.arduino.cc/en/software>

### III. DESPLIEGUE

Para la instalación del aplicativo de cupones RFID es requerido primero construir las dependencias y luego se despliega los contenedores por medio del siguiente comando:

```
docker-compose build
```

FIGURA 1 [BASH SISTEMA OPERATIVO]

```
docker-compose up
```

FIGURA 2 [BASH SISTEMA OPERATIVO]

una vez ejecutado el comando se nos desplegará un log en ejecución, con los detalles de los servicios que nos permitirá realizar un análisis de nuestros servicios principales

```
Starting coupon-rfid-socket-python-1 ...
Starting coupon-rfid-socket-python-1 ... done
Starting coupon-rfid-frontend-1 ... done
Starting coupon-rfid-api-python-1 ... done
Attaching to coupon-rfid-de-1, coupon-rfid-frontend-1, coupon-rfid-api-python-1, coupon-rfid-socket-python-1
```

FIGURA 3 [BASH SISTEMA OPERATIVO]

```
2021-07-21T23:00:23.0258292 @ [note] Server socket created on IP: "0.0.0.0"
2021-07-21T23:00:23.0508262 @ [warning] Insecure configuration for --pid-file: location "/var/run/mysqld" in the path is accessible to all OS users.
Consider choosing a different directory.
2021-07-21T23:00:31.688352 @ [note] Event Scheduler: loaded 0 events.
2021-07-21T23:00:31.698620 @ [note] mysqld ready for connections.
version: '5.7.32' socket: '/var/run/mysqld/mysqld.sock' port: 3306 MySQL Community Server (GPL)
[python] starting python-app.py
Server initialized for uWSGI.
* Serving flask app "app" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
Running on http://172.18.0.4:5000/ (Press CTRL-C to quit)
* Restarting with watchdog (watchdog)
* Debugger is active!
* Debugger PIN: 236-373-308
```

FIGURA 4 [BASH SISTEMA OPERATIVO]

Una vez desplegados podemos validar su funcionamiento ingresando al <http://localhost:8080> donde podremos ver el login del aplicativo:

**Ingresar**  
Ingresar con credenciales de usuario

Email

Password

INGRESAR

Bienvenido al aplicativo de cupones!

FIGURA 5 [LOGIN - <http://localhost:8080>]

Para poder configurar el correspondiente funcionamiento del arduino y conexiones adecuadas de los pines y proceder a soldarlos , podemos utilizar el siguiente modelo:

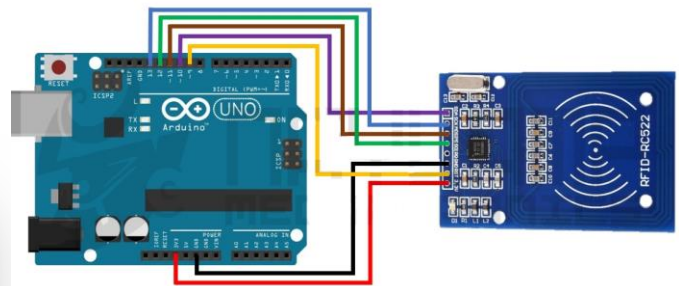


FIGURA 6 [DIAGRAMA ARDUINO -RFID]

Una vez conectado en el equipo , el sensor deberá emitir una luz que nos permite identificar que la conexión de los pines de energía están conectados de manera idónea:

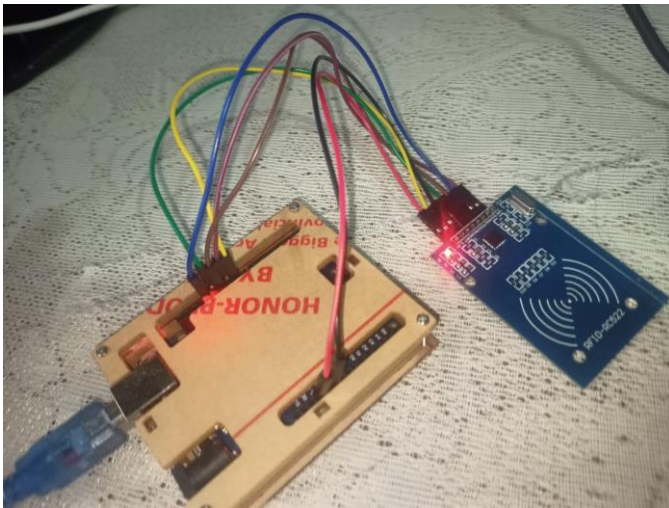


FIGURA 7 [Vela Javier - FOTO]

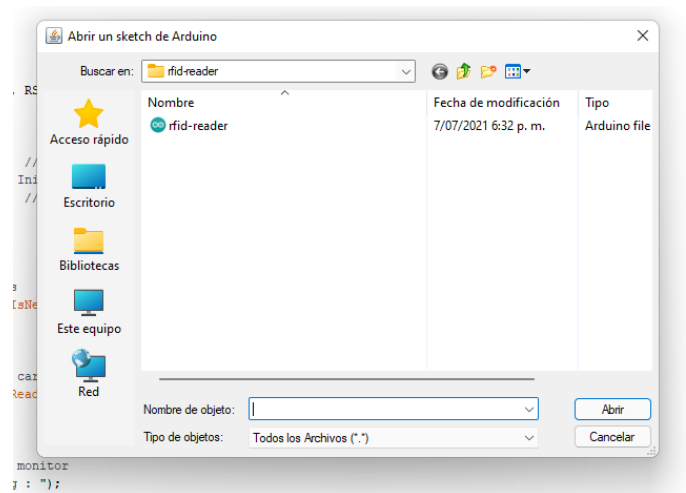


FIGURA 10 [ARDUINO IDE]

Luego validamos el puerto que utilizara

Para compilar y desplegar el programa INO de areduino podemos abrir el archivo arduino-compiler.bat si contamos con el sistema operativo WINDOWS compilando el script y descargando las dependencias

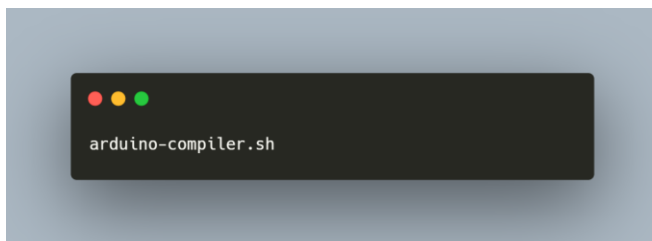


FIGURA 8 [BASH SISTEMA OPERATIVO]

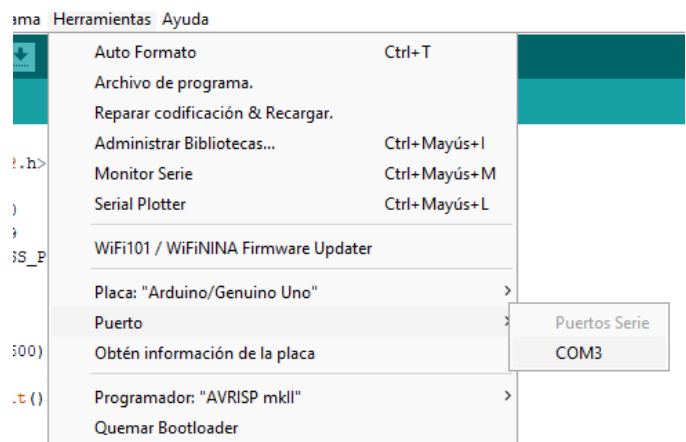


FIGURA 11 [ARDUINO IDE]

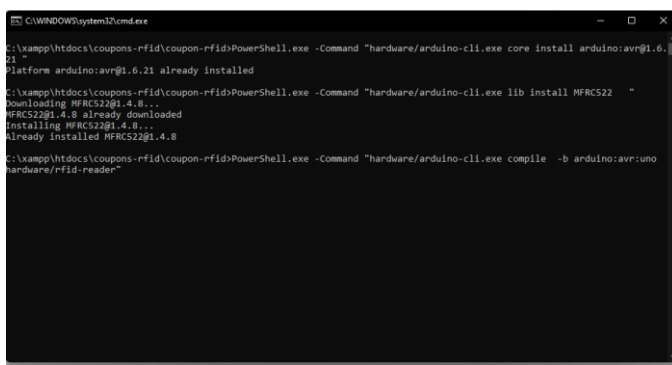


FIGURA 9 [BASH SISTEMA OPERATIVO]

En caso de no tener el sistema operativo WINDOWS , podemos compilar por medio de arduino IDE , abriendo el proyecto

Luego descargaremos la dependencia .... Corrspondiente al rfid :

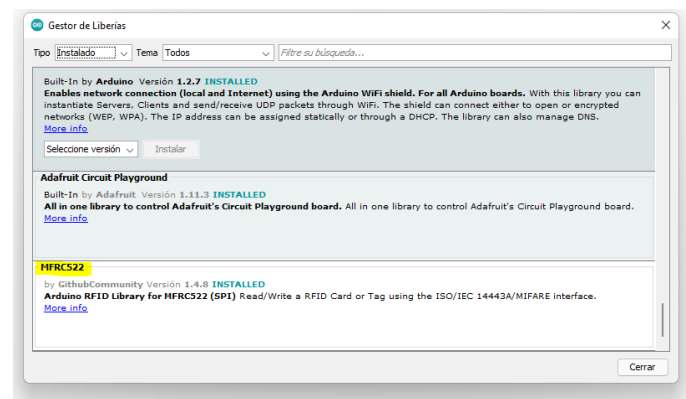


FIGURA 11 [ARDUINO IDE]

Luego procederemos a compilar el script en nuestro arduino

El Sketch usa 6624 bytes (20%) del espacio de almacenamiento de programa. El máximo es 32256 bytes.  
Las variables Globales usan 241 bytes (11%) de la memoria dinámica, dejando 1807 bytes para las variables locales. El máximo es 2048 bytes.

Y al pasar la tarjeta sobre el lector al realizar el respectivo debuggin nos debera salir lo siguiente en consola:

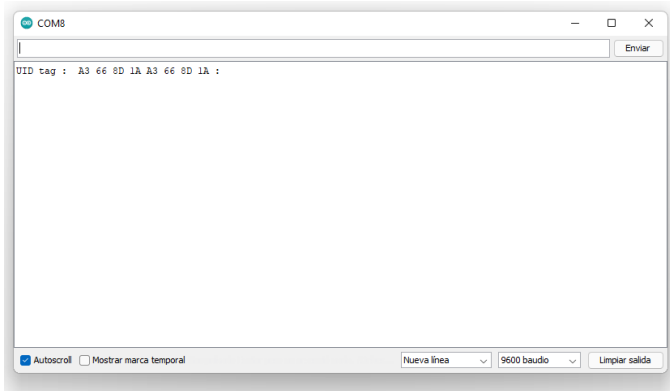


FIGURA 12 [ARDUINO IDE]

Para desplegar el lector de tarjetas RFID es necesario ejecutar el siguiente comando :

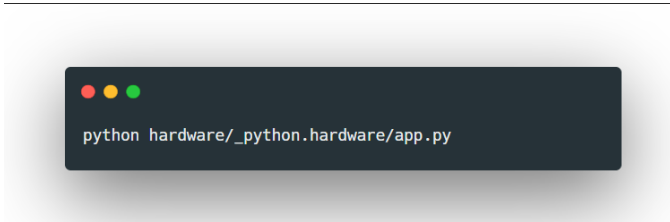


FIGURA 13 [BASH SISTEMA OPERATIVO]

Una vez ejecutado , se desplegara el siguiente mensaje advirtiendonos que se esta ejecutando :

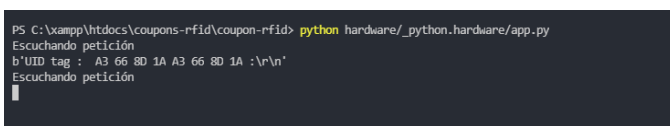


FIGURA 14 [BASH SISTEMA OPERATIVO]

En caso de que nos salga el siguiente mensaje deberemos validar el puerto serial de nuestro equipo que este activo y con el numero asignado en este caso. Ejemplo COM3, COM6,etc

#### IV. MANUAL USUARIO

Dentro del aplicativo de cupones RFID podemos encontrar una variedad de módulos o ventanas requeridas para la

realización del proceso de manera adecuada:

1. Para iniciar el proceso del aplicativo se debe ingresar al siguiente enlace:

<http://localhost:8080>

llevándonos al Login de la aplicación:

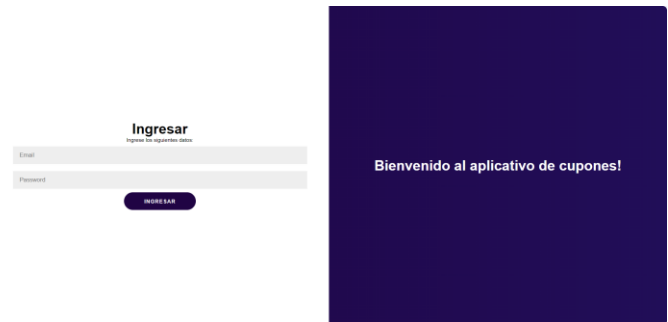


FIGURA 15 [LOGIN – <http://localhost:8080>]

Para ingresar, una vez desplegados las credenciales que cargara por defecto son las siguientes :

- **Usuario:** [admin@rfid.con](mailto:admin@rfid.con)
- **Contraseña:** password

Una vez ingresado , nos aparecerá el menú principal y una lista de iconos donde podremos ingresar a los diferentes módulos como personas, tiendas , planes ,etc.

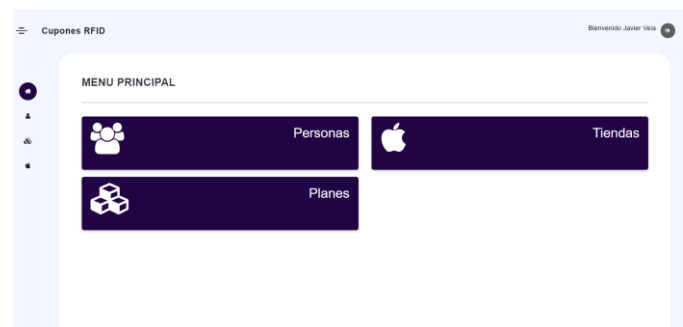


FIGURA 15 [MENU PRINCIPAL– <http://localhost:8080/admin/main>]

Dentro de modulo de personas cuando ingresamos podemos ver una lista donde aparecerán los registros agregados durante el proceso:

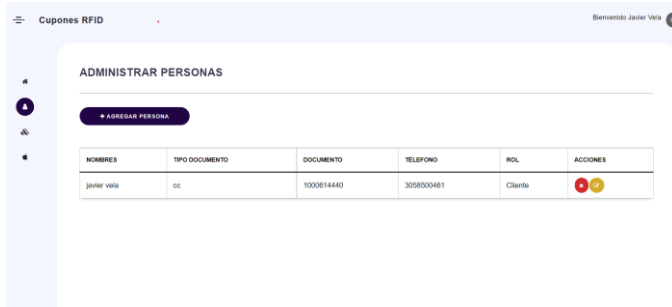


FIGURA 16 [MODULO PERSONAS – <http://localhost:8080/admin/person>]

Y al agregar o modificar nos aparecerá el siguiente modulo donde debemos tener conectado el lector de tarjetas RFID para registrar la persona con su tarjeta correspondiente:



FIGURA 17 [MODULO GESTIÓN PERSONAS – <http://localhost:8080/admin/<idPersona | add>>]

Dentro de módulo de tiendas cuando ingresamos podemos ver una lista donde aparecerán los registros agregados durante el proceso:

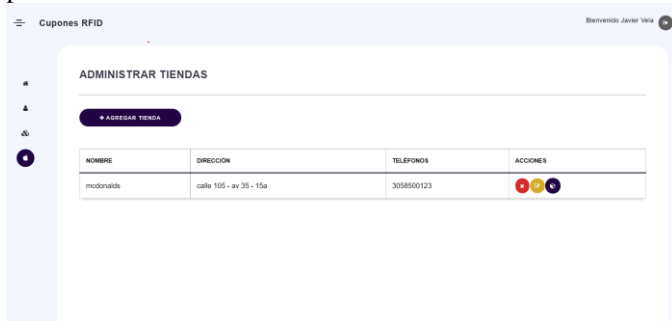


FIGURA 18 [MODULO TIENDAS – <http://localhost:8080/admin/store>]

Y podremos agregar, modificar o eliminar las tiendas en el siguiente modulo:

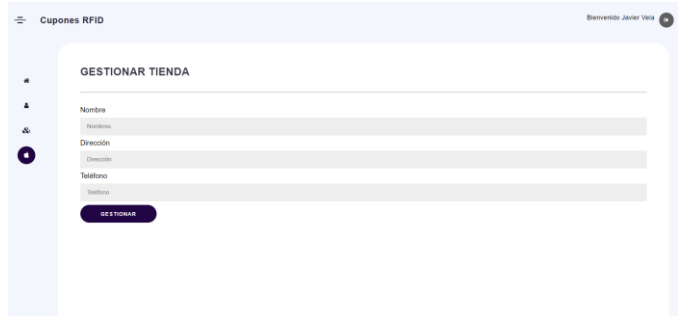


FIGURA 19 [MODULO GESTIÓN TIENDAS – <http://localhost:8080/admin/store/<idStore | add>>]

Dentro de módulo de productos cuando ingresamos podemos ver una lista donde aparecerán los registros agregados durante el proceso:

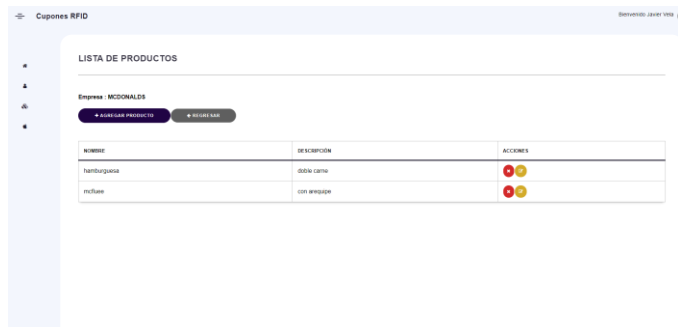


FIGURA 20 [MODULO PERSONAS – <http://localhost:8080/admin/product/<idStore | add>>]

Dentro de módulo de productos cuando ingresamos podemos ver una lista donde aparecerán los registros agregados durante el proceso:

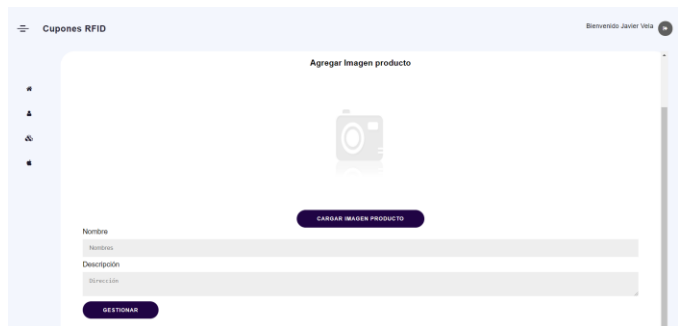


FIGURA 21 [MODULO GESTIÓN PRODUCTOS – <http://localhost:8080/admin/<idProduct | add>>]

Dentro de módulo de planes cuando ingresamos podemos ver una lista donde aparecerán los registros y podrás gestionar los planes y productos desde este módulo:

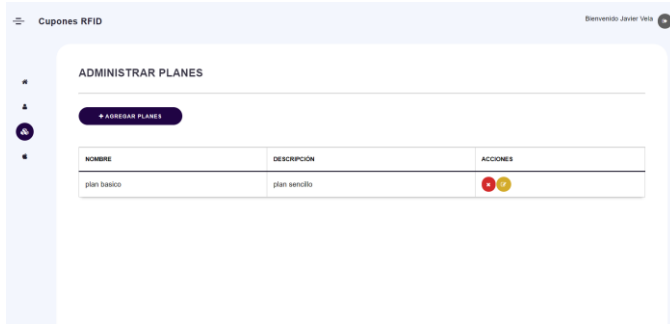


FIGURA 22 [MODULO GESTIÓN PRODUCTOS – <http://localhost:8080/admin/<idProduct | add>>]

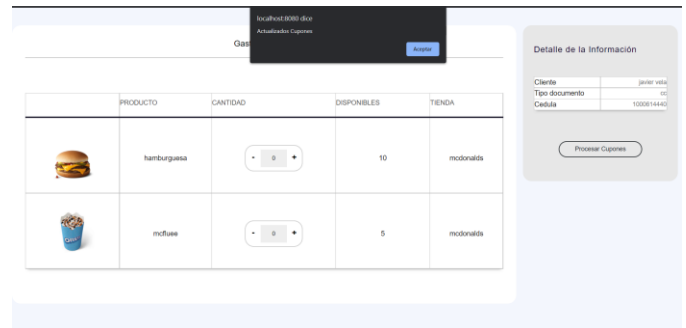


FIGURA 25 [MODULO GASTAR CUPONES – <http://localhost:8080/factory>]

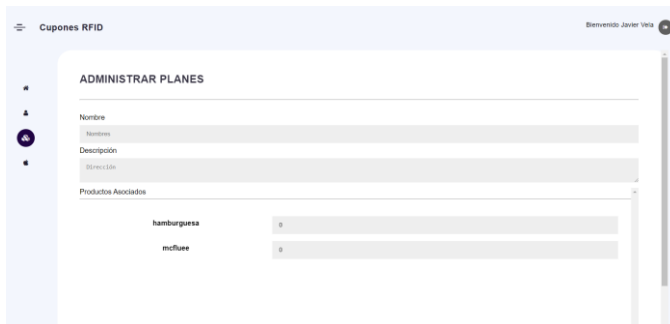


FIGURA 23 [MODULO GESTIÓN PRODUCTOS – <http://localhost:8080/admin/<idProduct | add>>]

Para gestionar los cupones podrás validarlos por medio del enlace <http://localhost:8080/factory> donde se puede interactuar con la tarjeta y el sensor RFID para identificar que cupones tiene el usuario y que productos puede consumirse :

## V. MANUAL DESARROLLADOR

### Diseño de base de datos :

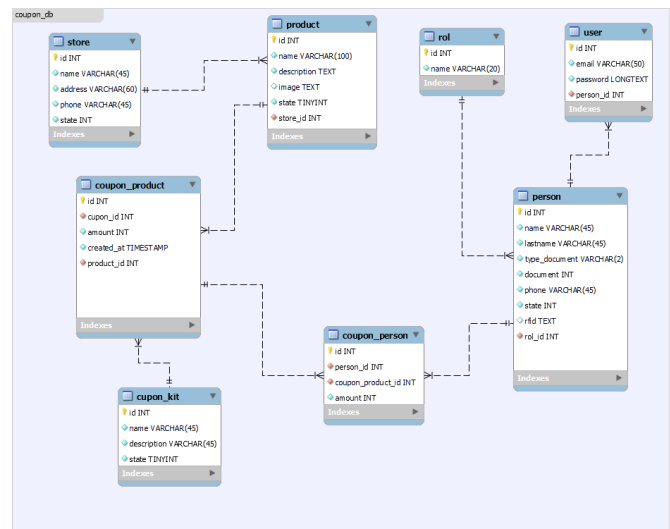


FIGURA 26 [MODELO RELACIONAL]

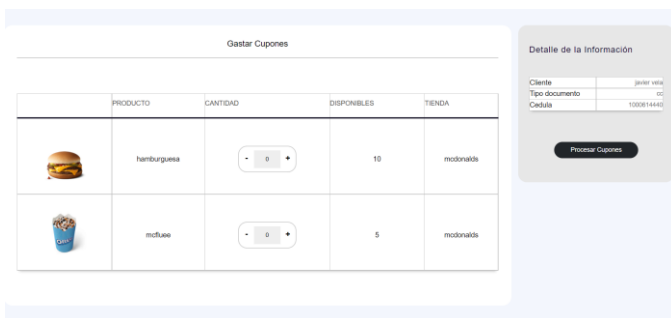


FIGURA 24 [MODULO GASTAR CUPONES – <http://localhost:8080/factory>]

### Plan de procesos

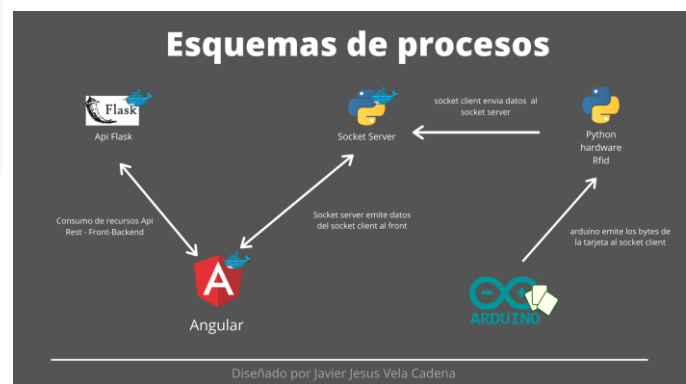


FIGURA 27 [ESQUEMA DEL PROCESO]

Una vez seleccionado los cupones a consumir le damos en el botón procesar cupones para actualizar los cupones de la respectiva persona

## Configuración de variables generales base de datos MYSQL

Ruta archivo: .env

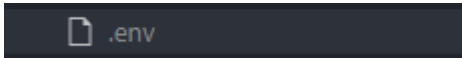


FIGURA 28 [ARCHIVO. env]

Constantes a tener en cuenta del archivo

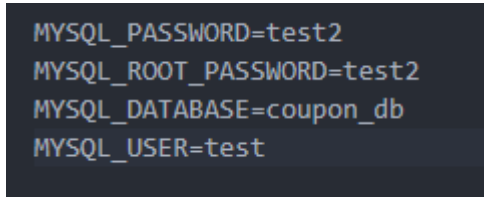


FIGURA 29 [ARCHIVO. env]

Si deseamos actualizar los datos de accesos a la base de datos de mysql se pueden actualizar desde este archivo , ademas para comprobar la conexión podemos utilizar cualquier cliente de bases de datos para validar que la conexión de los datos funciones.

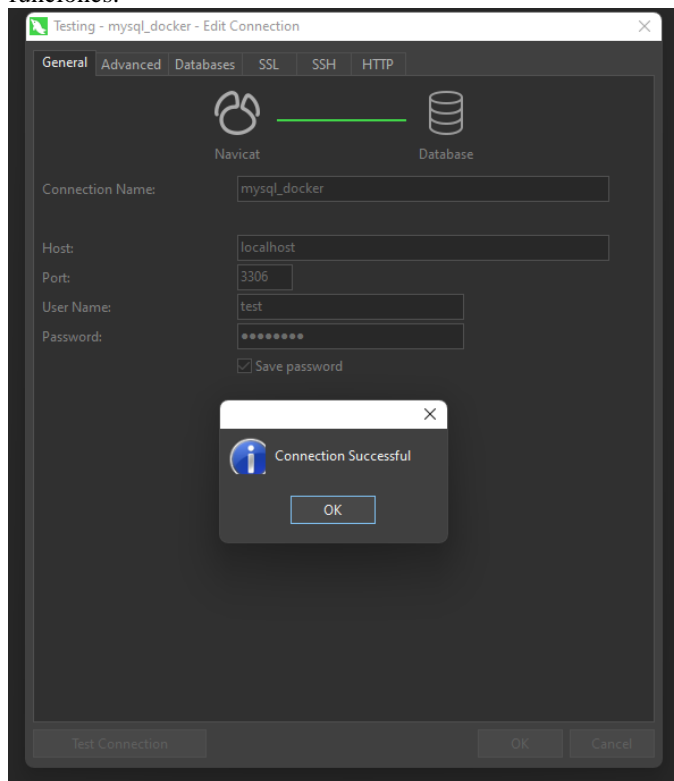


FIGURA 30 [NAVICAT – GESTOR BASE DE DATOS]

## Configurar puerto serial COM

Para configurar el puerto COM debe dirigirse al hardware/\_python.hardware/arduino/serial.py y en el parametro del inicializador o constructor se le asigna el nombre del puerto

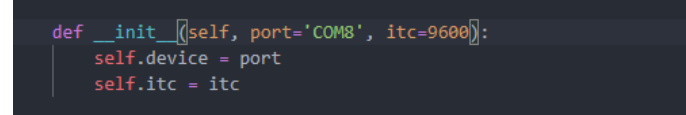


FIGURA 31 [ARCHIVO – serial.py]

## Configurar Contenedores y configuraciones de servicios

En el archivo docker-compose.yml se puede configurar datos como los puertos que escuchara los servicios , si el despliegue integrara archivos dockerfile , dependencias de servicios ,etc.

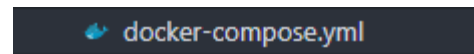


FIGURA 32 [ARCHIVO – docker.compose.yml]

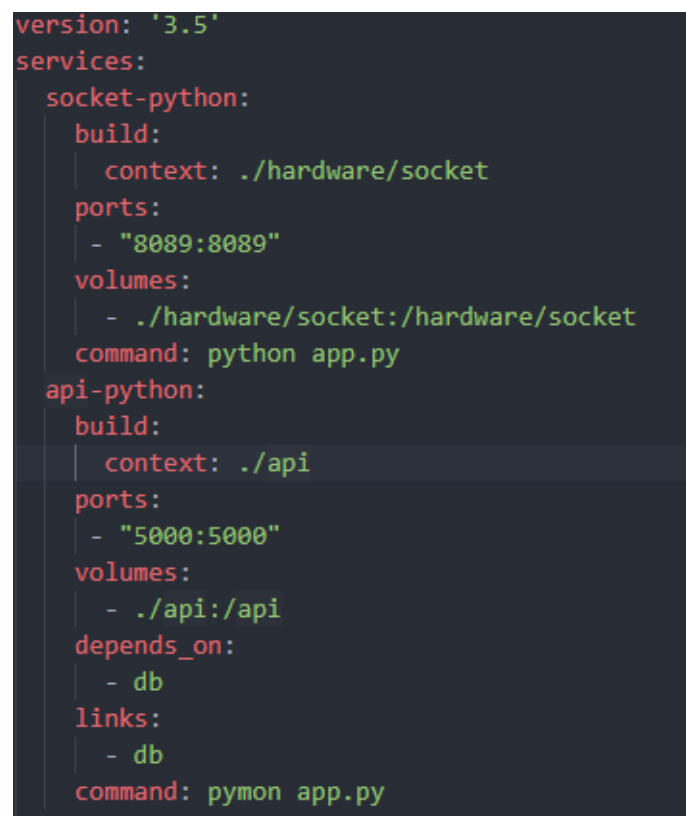


FIGURA 33 [ARCHIVO – docker.compose.yml]



## Configurar Script .SQL

Por defecto al construir y/o desplegar los servicios de Docker, por defecto nos integrará automáticamente el archivo docs/sql/coupon\_db.sql donde incluirá la creación de la base de datos, tablas y registros por defecto

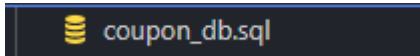


FIGURA 34 [ARCHIVO – coupon\_db.sql]

## Configuración Script Arduino

Por defecto al compilar el archivo INO del archivo hardware/rfid-reader.ino donde incluye el respectivo script para la lectura de tarjetas en el sensor RFID

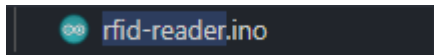


FIGURA 35 [ARCHIVO – rfid-reader.ino]

## Gestión de dependencias

Para actualizar , agregar o eliminar las dependencias de python los podemos alterar por medio de los archivos requirements.txt de la carpeta de los proyectos especificando el nombre de la dependencia y su version.

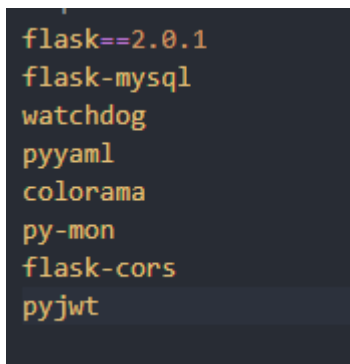


FIGURA 36 [ARCHIVO – requeriments.txt]

Para el proyecto front del aplicativo (Angular) , sus dependencias se administran por medio del archivo **package.json** en la propiedad **dependencies**

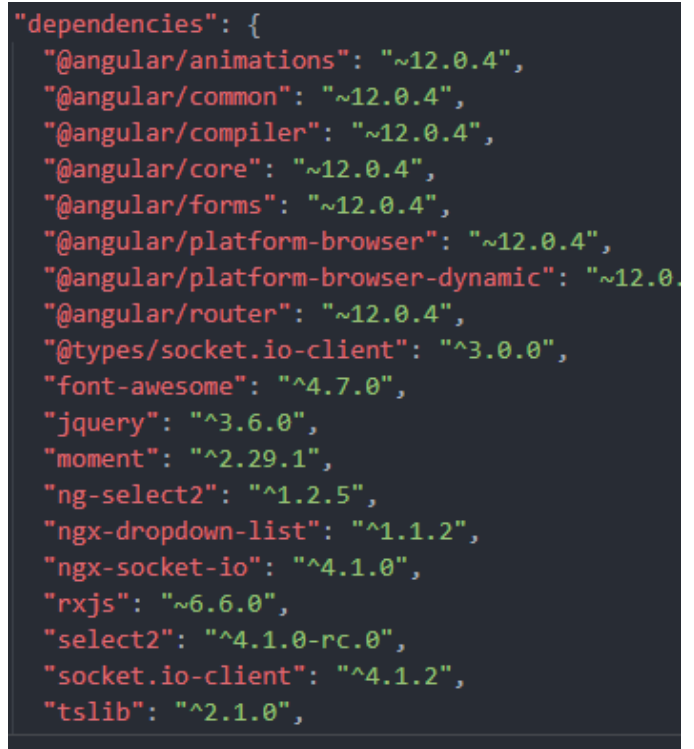


FIGURA 37 [ARCHIVO – package.json]

## Configurar Nuevas Rutas API

Podemos configurar las rutas del api de FLASK en el archivo api/controllers/loader.py donde configuraremos las rutas, controladores y tipo de método allí

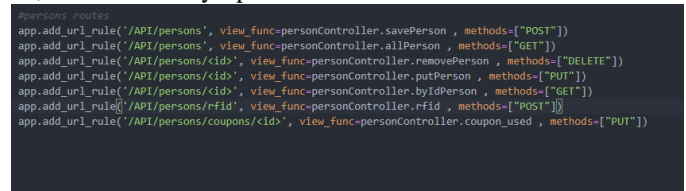


FIGURA 38 [ARCHIVO – loader.py]

## Configurar variables de la aplicación API

Dentro de la carpeta api/config podemos configurar datos de la aplicación generales y/o base de datos en estos archivos:

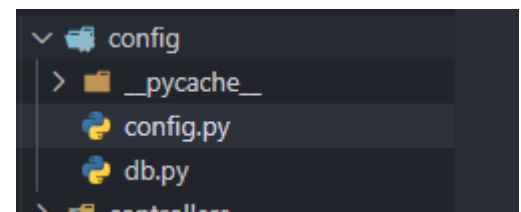


FIGURA 39 [CARPETA – config]



## Configurar modelos de la aplicación API

Dentro de la carpeta api/models podemos agregar clases con sus consultas en estos archivos:

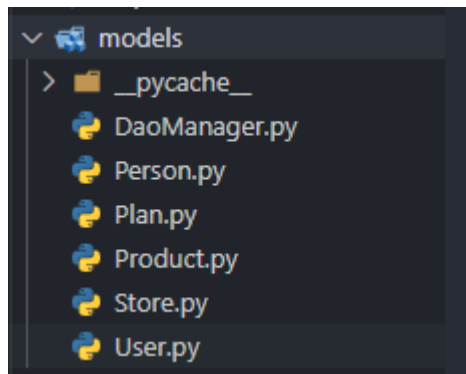


FIGURA 40 [CARPETA – models]

## Configurar rutas y componentes en la carpeta folder public

Para agregar rutas y/o componentes en angular deberá ir al archivo **public/src/app/app.routing.module.ts** donde agregaremos la ruta y el componente que deseamos renderizar

```
const routes: Routes = [
  { path: 'factory', component: ClientComponent },
  { path: 'login', component: LoginComponent },
  { path: 'admin', component: AdminComponent, data: { title: 'Menu Principal' }, canActivate: [roleGuardService], children: [
    { path: 'main', component: MainComponent, data: { title: 'Menu Principal' } },
    { path: 'person', component: PersonComponent, data: { title: 'Administrar Personas' } },
    { path: 'person/idPerson', component: ManagePersonComponent, data: { title: 'Gestionar Personas' } },
    { path: 'person/coupon/idPerson', component: ManageCouponPersonComponent, data: { title: 'Lista de cupones Personas' } },
    { path: 'store', component: StoreComponent, data: { title: 'Administrar Tiendas' } },
    { path: 'store/idStore', component: ManageStoreComponent, data: { title: 'Gestionar Tienda' } },
    { path: 'product/idStore', component: ProductComponent, data: { title: 'Lista de Productos' } },
    { path: 'product/idStore/idProduct', component: ManageProductComponent, data: { title: 'Gestionar Productos' } },
    { path: 'plan', component: PlanComponent, data: { title: 'Administrar Planes' } },
    { path: 'plan/idPlan', component: ManagePlanComponent, data: { title: 'Administrar Planes' } },
  ] },
  { path: '**', component: LoginComponent },
];
```

FIGURA 41 [ARCHIVO – app.routing-module.ts]

## Agregar Usuarios Administradores

Para agregar usuarios administradores debe agregar tanto en la tabla user y person el registro y debe tener en cuenta que el rol\_id sea 1 manualmente al archivo docs/sql/coupon\_db.sql

```
-- Data for table: coupon_db - person
START TRANSACTION;
USE `coupon_db`;
INSERT INTO `coupon_db`.`person` (`id`, `name`, `lastname`, `type_document`, `document`, `phone`, `state`, `rfid`, `rol_id`) VALUES (1, 'javier', 'hola', 'cc', 10000000, 123456789, 'C', 1);
COMMIT;

-- Data for table: coupon_db - user
START TRANSACTION;
USE `coupon_db`;
INSERT INTO `coupon_db`.`user` (`id`, `email`, `password`, `person_id`) VALUES (1, 'admin@f10.com', 'password', 1);
COMMIT;
```

## VIII. CONCLUSIÓN

En base a las respectivas clases, practicas, documentos y diferentes recursos internos y externos, además de la respectiva participación de los compañeros y seguimiento del instructor de programa puedo concluir varios puntos respecto al proyecto y los conocimientos adquiridos a lo largo del

diplomado en PYTHON + DEVNET.

Python es un lenguaje de programación bastante sencillo y solido a realizar operaciones y procesos informativos de manera robusta como la interacción de software como APIS, carga masiva de información, servidores web e interacción con hardware como ESP32, ARDUINOS, RASPBERRY y sus respectivos sensores.

Además, destaco la expansión de conocimientos como Frameworks para la gestión de APIS y librerías masivas que nos permiten llevar al cabo funciones primordiales y en un tiempo acordes a lo estipulado.

## RECONOCIMIENTO

Aludo al reconcomiendo a la academia CISCO, encargada de llevar la enseñanza, gestión y desarrollo del diplomado Python + devnet durante el periodo del diplomado, cabe aclarar que me permitió ampliar mis conocimientos en el área del desarrollo y el hardware.

La capacidad de enseñanza que tiene su plataforma y docentes altamente capacidades en énfasis correspondientes al área de desarrollo, redes y hardware, han capacitado las dudas personales y las de mis compañeros de manera satisfactoria y acorde a las estipuladas por el programa.

No queda menos que agradecer y reconocer la satisfacción del diplomado y recomendar los conocimientos adquiridos durante las sesiones y el respectivo diplomado.