



Bocatapp

Alejandro Díaz Santos, Pablo Rodríguez Roldán.

2º DAM (Salesianos de Triana, Sevilla).

DESCRIPCIÓN DETALLADA DEL SISTEMA (HISTORIAS DE USUARIO)

| ID | Asignado a | Tipo | Título / Descripción | Prueba / Criterio de aceptación |
|-------|------------------------|---------------------|--|--|
| HU-01 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de iniciar sesión, para conseguir el resultado de acceder a la aplicación | Se va a comprobar realizando el inicio de sesión mediante nick y contraseña del usuario normal. |
| HU-02 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente, quiere realizar la acción de iniciar sesión mediante Google , para conseguir el resultado de acceder a la aplicación | Se va a comprobar realizando el inicio de sesión de Firebase con Google. |
| HU-03 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente, quiere realizar la acción de iniciar sesión mediante Facebook, para conseguir el resultado de acceder a la aplicación | Se va a comprobar realizando el inicio de sesión de Firebase con Facebook. |
| HU-04 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente, quiere realizar la acción de registro, para conseguir el resultado de iniciar sesión y acceder a la aplicación. | Se va a comprobar realizando el registro rellenando los campos necesarios y cumpliendo las condiciones necesarias. |
| HU-05 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de listar los locales por cercanía al mismo, para conseguir la vista de los locales con la condición dada. | Se va a comprobar visualizando la distancia de los locales con respecto al usuario. |
| HU-06 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de filtrar la lista de locales por el nombre, para conseguir la vista de los | Se va a comprobar visualizando el nombre de los locales con respecto a lo escrito en el Searchview. |

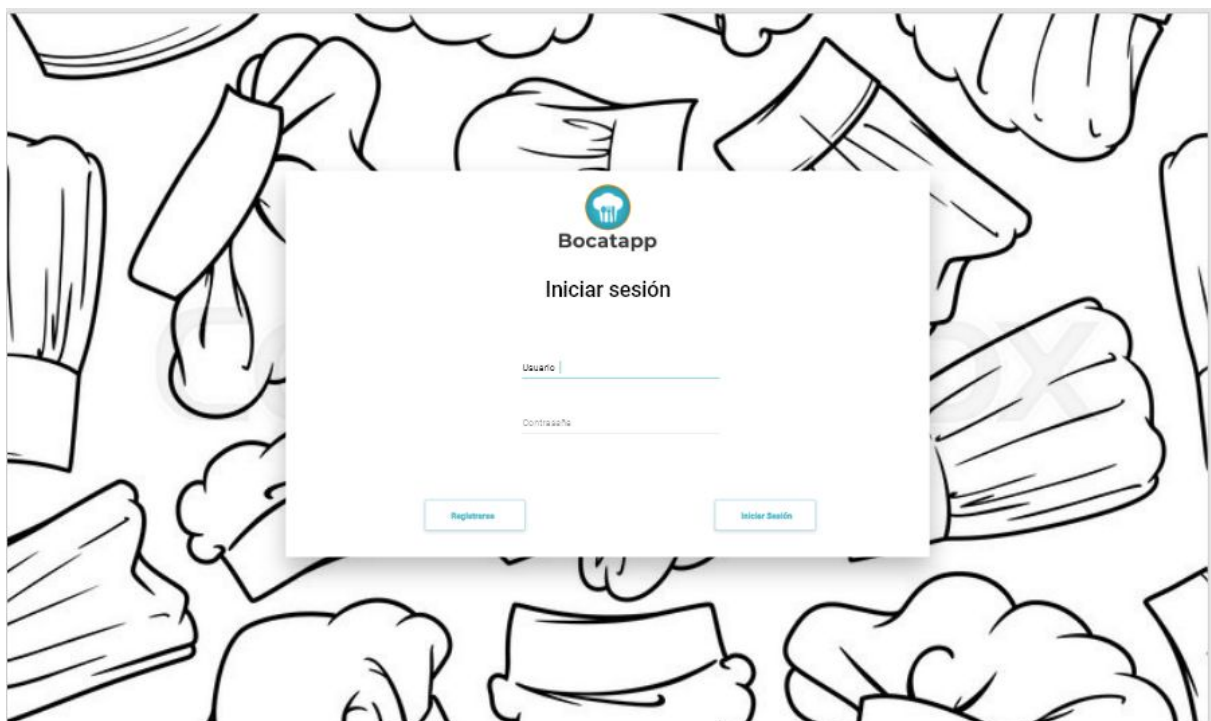
| | | | | |
|-------|------------------------|---------------------|---|---|
| | | | locales con la condición dada. | |
| HU-07 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de listar los locales seleccionados por el usuario como favoritos, para conseguir la vista de los locales con la condición dada. | Se va a comprobar con la condición de que los locales que aparezcan tengan el botón de favoritos marcados. |
| HU-08 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de listar los locales seleccionados por categoría, para conseguir la vista de los locales con la condición dada. | Se va a comprobar visualizando la distancia de los locales con respecto al usuario y que todos estos tengan la misma categoría. |
| HU-09 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de acceder al detalle de un local, para conseguir la vista del local seleccionado con sus datos. | Se va a comprobar visualizando el detalle del local que se ha seleccionado con todos sus respectivos datos. |
| HU-10 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de valorar un local, para conseguir la aportación de su opinión sobre este (si es gerente no podrá valorar el suyo). | Se va a comprobar con el resultado del aporte sobre la misma acción, acorde a las condiciones. |
| HU-11 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de aportar un comentario sobre un local, para conseguir la aportación de su opinión sobre este (si es gerente no podrá comentar el suyo, sólo responder comentarios). | Se va a comprobar con el resultado del aporte sobre la misma acción, acorde a las condiciones. |
| HU-12 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de añadir y eliminar productos al carrito, para conseguir la metodología de pedido en un local. | Se va a comprobar con el historial de pedidos una vez se proceda a pagar el pedido. |
| HU-13 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de realizar pedido, para conseguir la metodología de pedido en un local. | Se va a comprobar mediante el panel de control si se ha realizado el pedido. |
| HU-14 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de editar perfil, para conseguir la actualización del mismo. | Se va a comprobar mediante la comparación de los datos en vista y los datos recogidos en la base de datos. |
| HU-15 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de editar foto de perfil, para conseguir la actualización del mismo. | Se va a comprobar mediante la comparación de los datos en vista y los datos recogidos en la base de datos. |

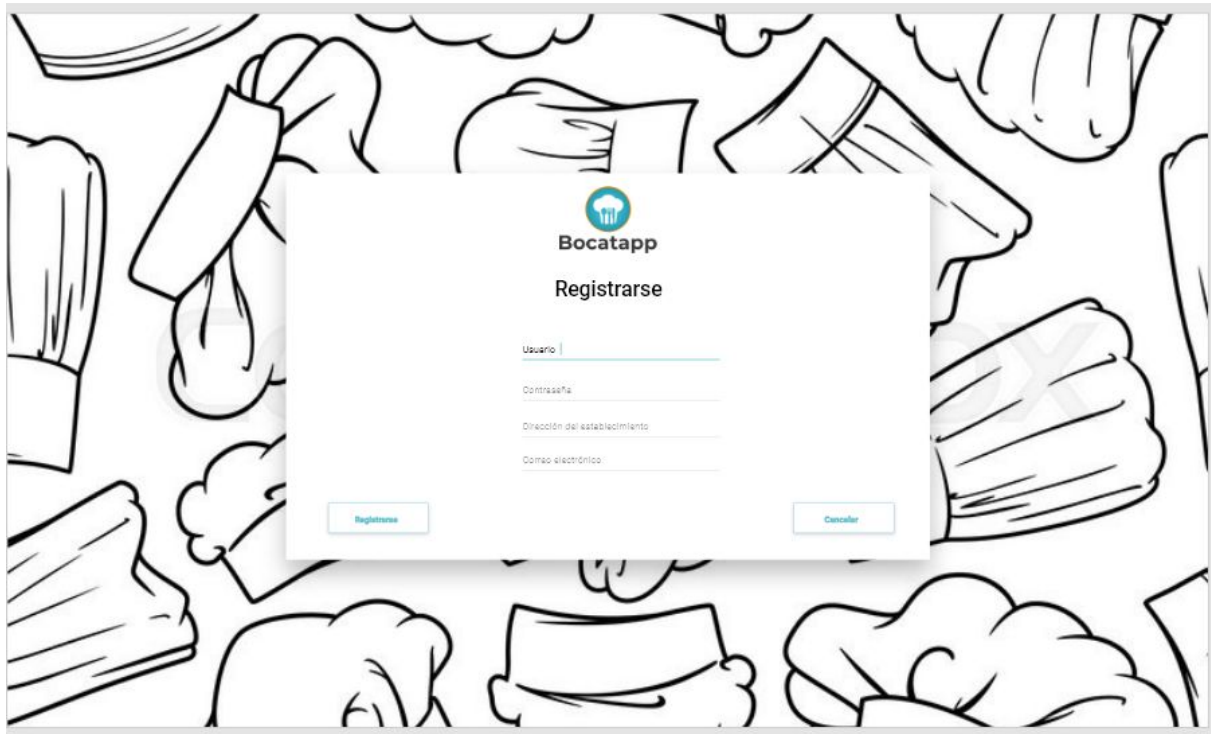
| | | | | |
|-------|------------------------|---------------------|--|--|
| HU-16 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente / administrador, quiere realizar la acción de eliminar foto de perfil, para conseguir el borrado del mismo. | Se va a comprobar mediante la comparación de los datos en vista y los datos recogidos en la base de datos. |
| HU-17 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario gerente, quiere realizar la acción de acceder al perfil del local, para conseguir la visualización de este desde la aplicación móvil. | Se va a comprobar mediante la comparación de los datos en vista y los datos recogidos en la base de datos. |
| HU-18 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario gerente, quiere realizar la acción de acceder a las estadísticas sociales del local, para conseguir la visualización. | Se va a comprobar mediante la comparación de los datos en vista y los datos recogidos en la base de datos. |
| HU-19 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario normal / gerente, quiere realizar la acción de recuperar la contraseña de su cuenta. | Se va a comprobar mediante un e-mail al correo electrónico que el usuario en cuestión utiliza. |
| HU-20 | Integrantes 1 y 2 | Historia de Usuario | Como un usuario normal / gerente, se podrá elegir la hora a la que irá a por el pedido dentro del horario del local | Se va a comprobar mediante la comparación de los horarios con la hora seleccionada a la hora de finalizar el pedido. |
| HU-21 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario gerente / administrador, quiere realizar la acción de iniciar sesión, para conseguir el resultado de acceder a la aplicación | Se va a comprobar realizando el inicio de sesión mediante nick y contraseña del usuario normal. |
| HU-22 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario gerente / administrador, quiere realizar la acción de iniciar sesión mediante Google , para conseguir el resultado de acceder a la aplicación | Se va a comprobar realizando el inicio de sesión de Firebase con Google. |
| HU-23 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario gerente / administrador, quiere realizar la acción de iniciar sesión mediante Facebook, para conseguir el resultado de acceder a la aplicación | Se va a comprobar realizando el inicio de sesión de Firebase con Facebook. |
| HU-24 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario gerente, quiere realizar la acción de registro, para conseguir el resultado de iniciar sesión y acceder a la aplicación. | Se va a comprobar realizando el registro rellenando los campos necesarios y cumpliendo las condiciones necesarias. |
| HU-25 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario administrador, quiere realizar la acción de mostrar un listado de todos los establecimientos registrados con la opción de eliminar o banear. | Se va a comprobar con un listado con todos los establecimientos registrados con la opción de banear o borrar cualquier local |
| HU-26 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario administrador, quiere realizar la acción de mostrar un listado de todos los gerentes registrados con la opción de eliminar o banear. | Se va a comprobar con un listado con todos los gerentes registrados con la opción de banear o borrar cualquier |

| | | | | |
|-------|------------------------|---------------------|---|---|
| | | | | gerente |
| HU-27 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario administrador, quiere realizar la opción de modificar o eliminar información detallada de cualquier establecimiento | Se va a comprobar en los propios detalles de cada establecimiento, con opciones de modificar o eliminar en cualquiera de los campos |
| HU-28 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario administrador, tendrá que validar todos los cambios que quiera realizar el gerente del establecimiento | Se va a comprobar con un apartado de notificaciones que recogerá todos los cambios solicitados por cada gerente |
| HU-29 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario gerente tendrá que solicitar los cambios que quiera realizar, y necesitan ser validados por el administrador | Se va a comprobar con un apartado de notificaciones para cada gerente |
| HU-30 | Pablo Rodríguez Roldán | Historia de Usuario | Como un usuario gerente podrá visualizar las estadísticas de su negocio | Se comprobará en una pestaña en la que se visualizarán diferentes estadísticas de ventas y compras |
| HU-31 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario gerente podrá crear, modificar o eliminar los productos de su establecimiento | Se va a comprobar con un formulario para registrar nuevos productos, teniendo un listado de todos ellos para poder modificarlos o eliminarlos |
| HU-32 | Alejandro Díaz Santos | Historia de Usuario | Como un usuario gerente podrá crear, modificar o eliminar las cartas de su establecimiento | |


MODELADO(SKETCHING DE LA APLICACIÓN WEB Y DE LA APLICACIÓN DE ANDROID)

SKETCHING WEB





The image shows a registration form for 'Bocatapp' centered on a background of black and white line drawings of chef hats. The form is a white rectangle with rounded corners and a subtle drop shadow. At the top center of the form is the Bocatapp logo, which consists of a green circle containing a white icon of a building with three vertical lines. Below the logo, the text 'Bocatapp' is written in a bold, black, sans-serif font. Underneath that, the word 'Registrarse' is displayed in a regular, black, sans-serif font. The form contains four input fields, each with a small label to its left: 'Usuario', 'Contraseña', 'Dirección del establecimiento', and 'Correo electrónico'. Each input field is a simple horizontal line with a small vertical cursor on the right side. At the bottom of the form, there are two buttons: 'Registrarse' on the left and 'Cancelar' on the right. Both buttons are white with a thin teal border and teal text.


Bocatapp
Registrarse


Usuario

Contraseña

Dirección del establecimiento

Correo electrónico

[Registrarse](#) [Cancelar](#)



Credil

-  Perfil
-  Carta
-  Estadística
-  Productos

Perfil



Bocatapp



PORTADA DEL ESTABLECIMIENTO:



UBICACIÓN: Calle Condes de Bustillo número 21

HORARIOS: Lunes a Viernes de 11:00 PM a 22:30 PM



Credil

-  Perfil
-  Carta
-  Estadística
-  Productos


Carta

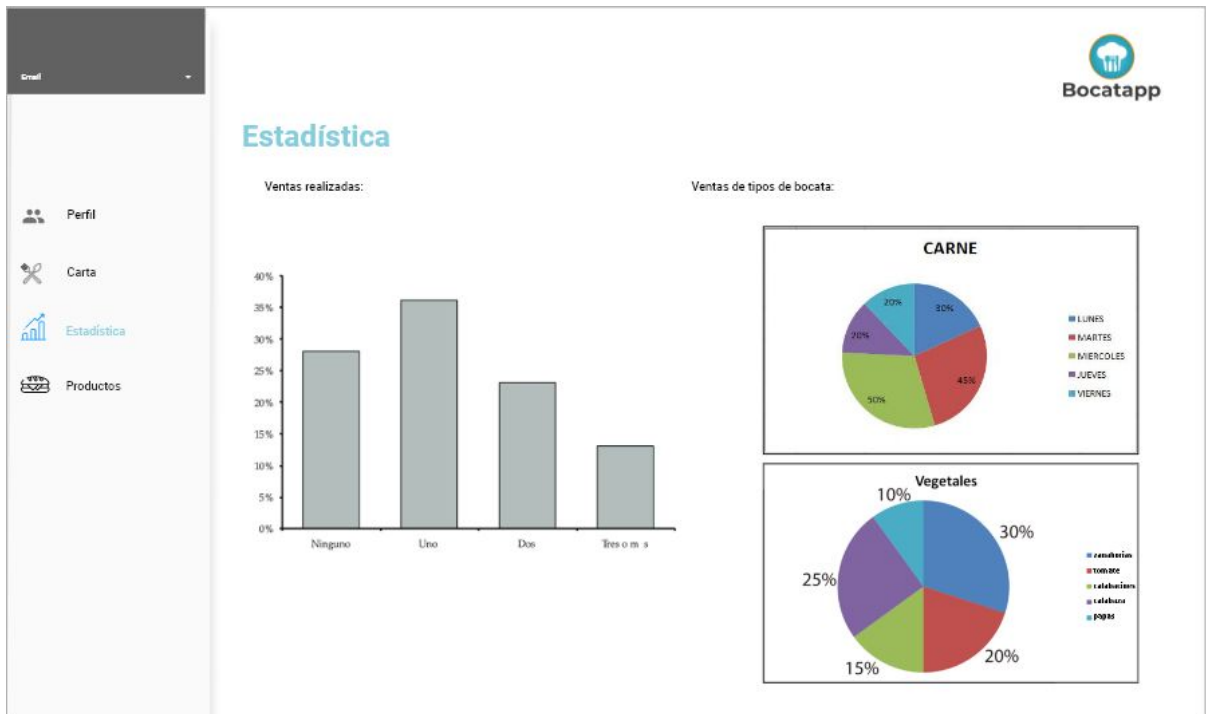


Bocatapp



| BOCATA'S BURGUER | | |
|------------------|--|--------|
| Peribují | Hamburguesa, Bacon, Queso y Salsa Rosa | 3,40 e |
| Perroñero | Hamburguesa, Pimiento, Queso y Salsa Rosa | 3,40 e |
| Tamburete | Hamburguesa, Huevo Frito, Queso y Salsa Rosa | 3,40 e |
| Tiburón | Hamburguesa, Atún, Queso y Salsa Rosa | 3,40 e |
| Titiriti | Hamburguesa, Sobrasada, Queso y Salsa Rosa | 3,40 e |
| Capricho | Hamburguesa, Lechuga, Tomate, Queso, Espárragos y Salsa Rosa | 3,40 e |
| Cucly | Hamburguesa, Jamón York, Pimiento y Salsa Rosa | 3,40 e |
| Maravilla | Doble Hamburguesa, Jamón York, Queso y Salsa Rosa | 3,90 e |
| Gutty | Hamburguesa, Bacon, Huevo Frito, Queso y Salsa Rosa | 3,90 e |

 Añadir una nueva carta



Perfil

Carta

Estadística

Productos

Bocatapp

Productos

Imagen del producto:

Información del producto: Este producto esta compuesto por ingredientes que no contienen gluten

Ingredientes:

+ Añadir productos

Imagen del producto:

Información del producto: Este producto esta compuesto por ingredientes que no contienen gluten


Ingredientes:


- Pan integral
- Jamón de York
- Lechuga
- Tomate
- Queso de Cheddar


+ Añadir productos



Small

 Perfil

 Carta


 Estadística

 Productos

 Almacén


Bocatapp

Almacén



Ingredientes:

Cantidad

- Pan integral

- 20

+

- Jamón de York

- 3kg

+

- Lechuga

- 5

+

- Tomate

- 30

+

- Queso de Cheddar

- 15

+

- Jamón Serrano

- 1kg

+

- Tortilla de patatas

- 2kg

+

- Bacon

- 4kg

+

- Lomo

- 5kg

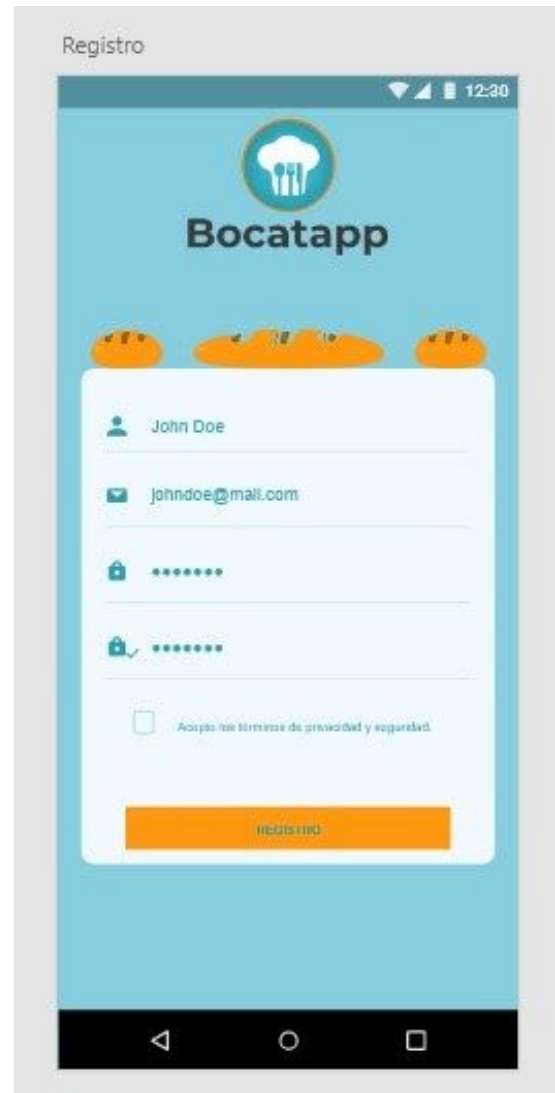
+

- Pollo

- 6kg

+

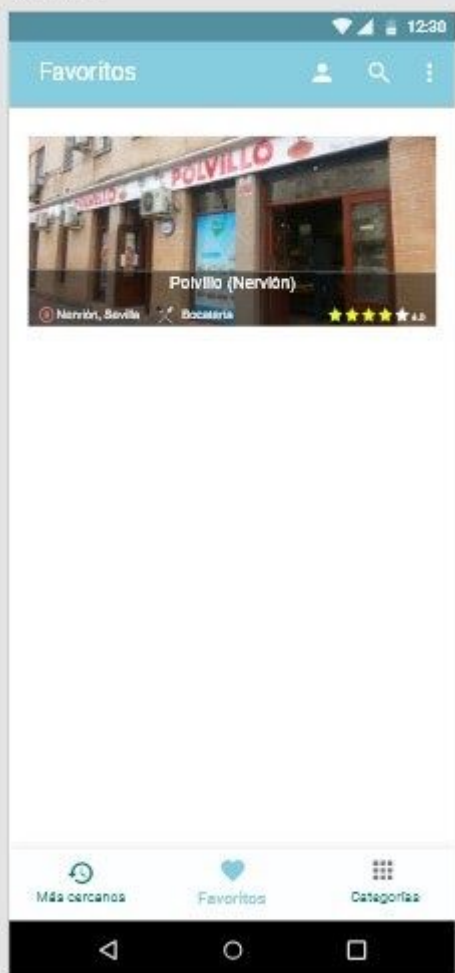
SKETCHING APLICACIÓN

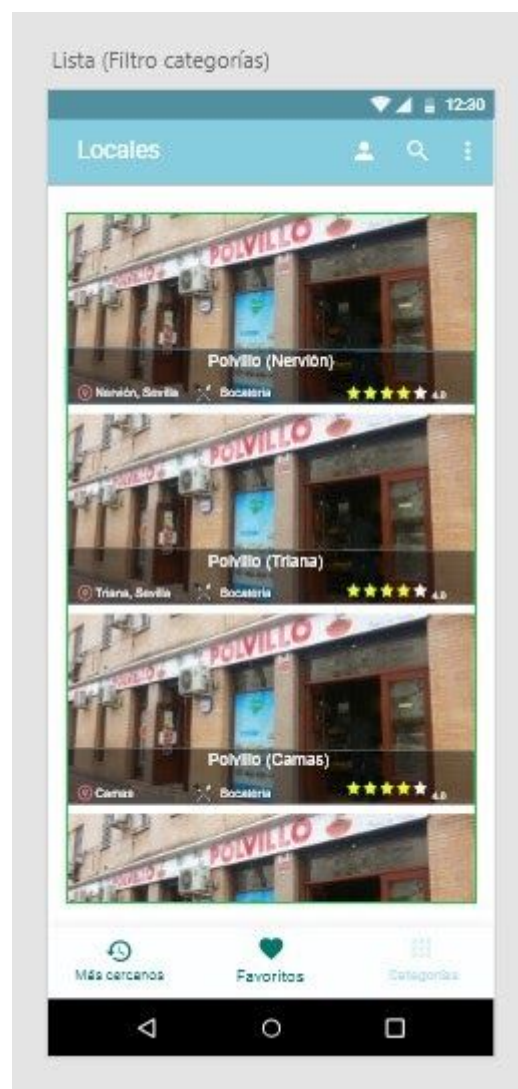
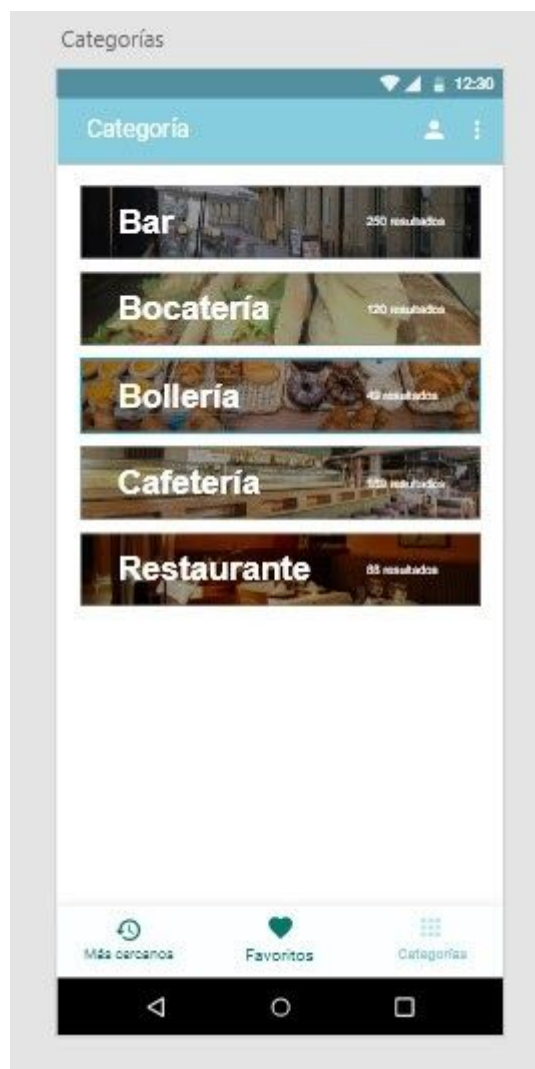


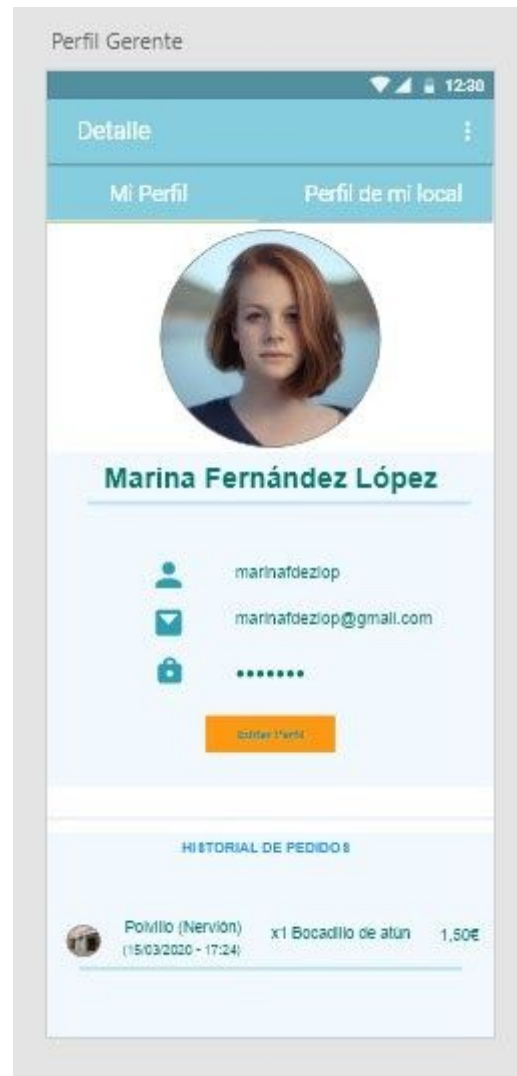
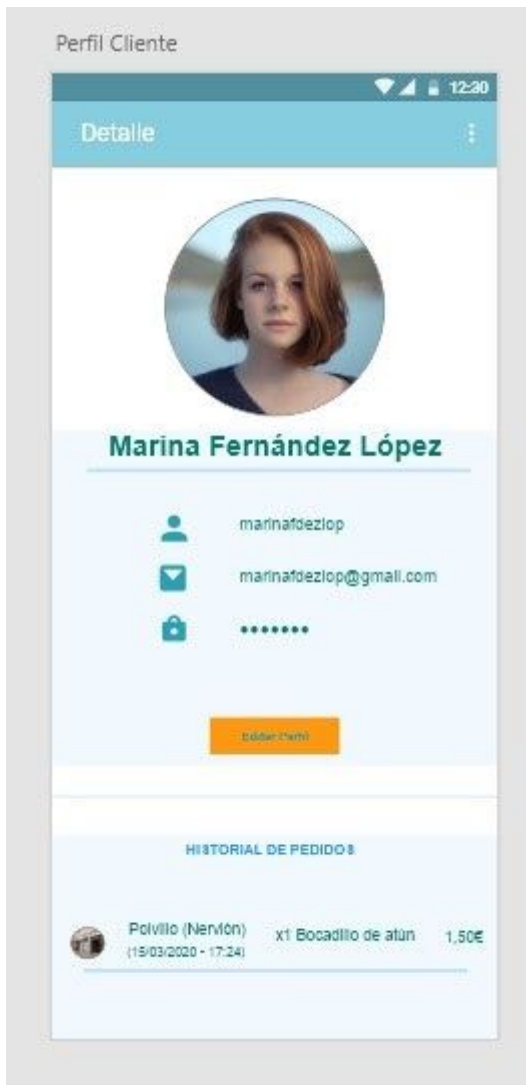
Más Cercanos



Favoritos







Detalle Local

Detalle



Polvillo (Nervión)

Nervión, Sevilla

★★★★★ 4.0

Bocateria de la franquicia Polvillo, situado en la parte central de Nervión.

✂ Bocateria

📍 Camas

→ 10/03/2020

📍 Localización

CARTA



Bocadillo de atún
(Pan integral, atún)

1,50€

−

0

+

HORARIO

OPEN

APERTURA: 09:00h

CIERRE: 15:00h

Final Pedido

Detalle

Polvillo (Nervión)

Nervión, Sevilla

→ 15/03/2020

PEDIDO



Bocadillo de atún
(Pan integral, atún)

1 ud.

1,50€

TOTAL 1,50€

PAGO

👤

💳

📶 bizum

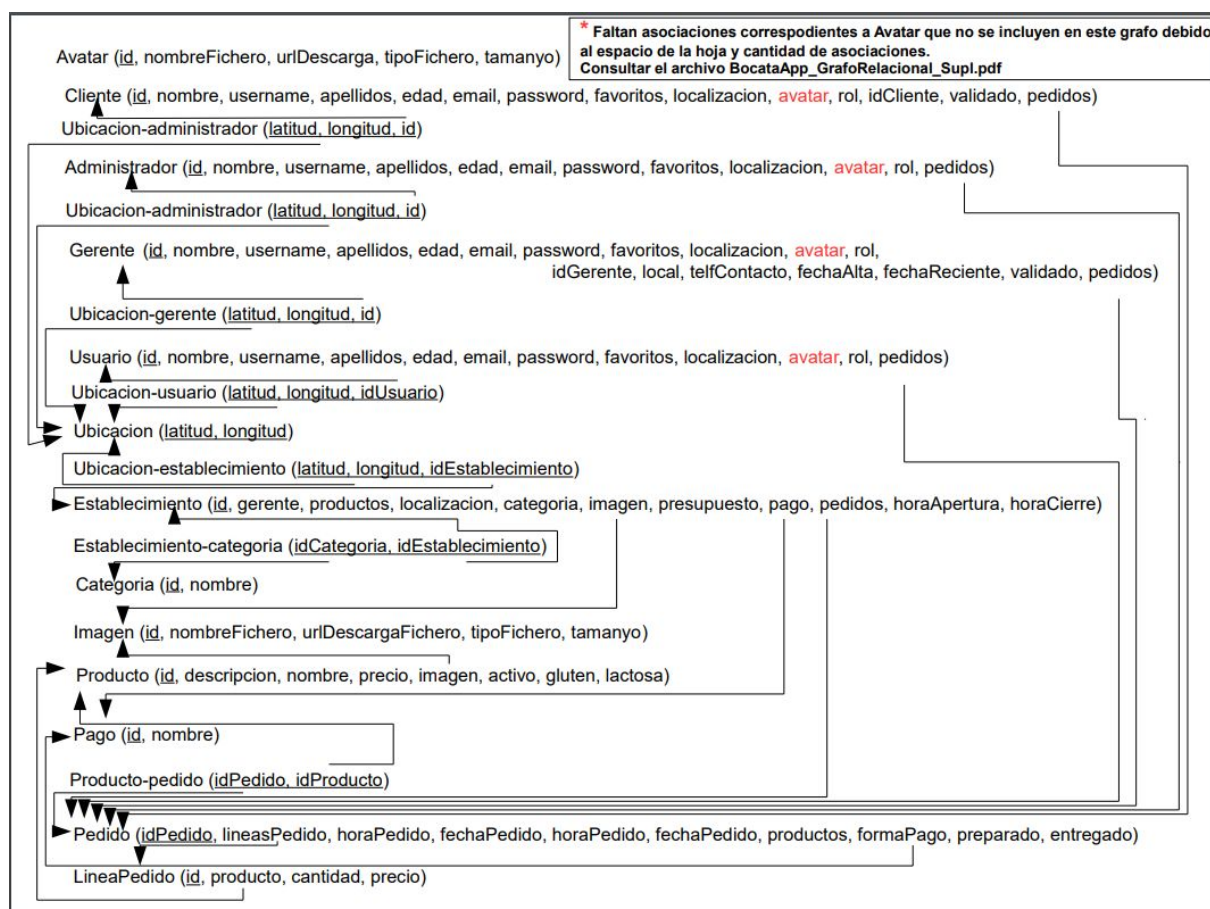
HORA DE RECOGIDA

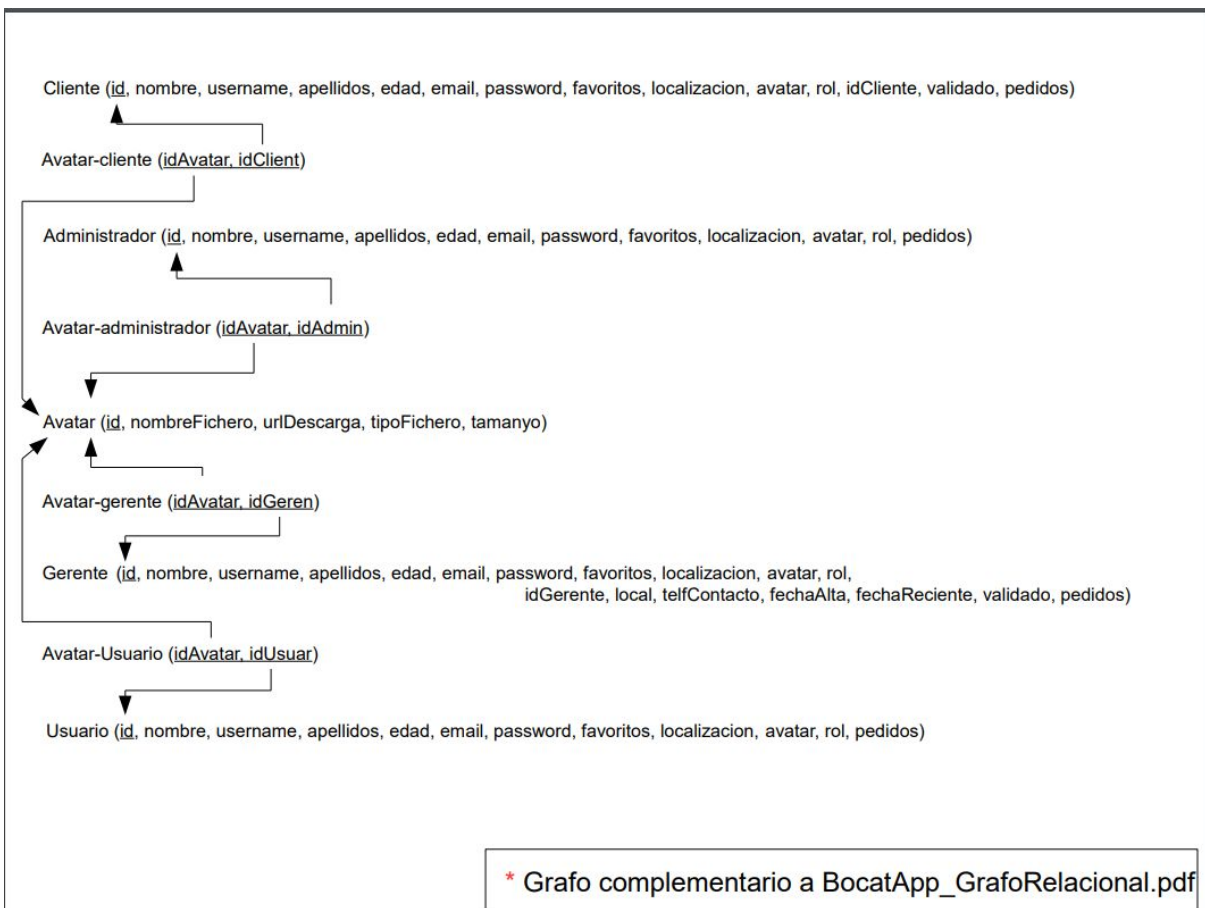
Horario: Martes

11:05 ↓

REALIZAR PEDIDO

DISEÑO WEB BOCATAPP





Administrador:

- **(POST) register: “{{base_url}}/register/”:**
 - Crea un usuario administrador
 - Se le pasará por parámetros las distintas variables, siendo una petición multiparte.
 - Devolverá el administrador creado
- **(GET) getRole: “{{base_url}}/username/”:**
 - Devuelve el rol del usuario indicado
 - Se le pasa el nombre de usuario para obtener el rol
 - Devolverá el rol del usuario indicado

Categoría:

- **(POST) nuevaCategoria: “{{base_url}}/categoria/”:**
 - Crea una nueva categoría
 - Se le pasa como cuerpo la categoría creada
 - Devolverá la categoría registrada

- **(GET) getCategoryasName: “{{base_url}}/categorias/name”:**
 - Devuelvetodos los nombres de las categorías existentes
 - Devuelve una lista con todos los nombres de las categorías
- **(GET) unaCategoria: “{{base_url}}/categoria/{id}”:**
 - Busca una categoría por su id
 - Se le pasa el id de la categoría seleccionada
 - Devuelve la categoría seleccionada
- **(PUT) editarCategoria: “{{base_url}}/categoria/{id}”:**
 - Edita una categoría
 - Se le pasa el id de la categoría que vamos a editar
 - Devuelve la categoría editada
- **(DELETE) deleteCategoria: “{{base_url}}/categoria/{id}”:**
 - Elimina una categoría
 - Se le pasa el id de la categoría que vamos a eliminar

Cliente:

- **(POST) register: “{{base_url}}/register”:**
 - Crea un usuario cliente
 - Se le pasará por parámetros las distintas variables, siendo una petición multiparte.
 - Devolverá el cliente creado
- **(GET) findByEmail: “{{base_url}}/findByEmail”:**
 - Busca un cliente por su email
 - Se le pasará el email del usuario que queremos buscar
 - Devolverá el usuario que coincida con el email

- **(GET) listarClientes: “{{base_url}}/list”:**
 - Lista todos los clientes registrados
 - Devolverá la lista con todos los clientes registrados

- **(PUT) editarFotoCliente:**
“{{base_url}}/client/{id}/editPhoto”:
 - Editará la foto del usuario cliente registrado
 - Se le pasará el id del cliente seleccionado y el cuerpo del archivo que se usará para editar.
 - Devolverá el usuario cliente con el archivo editado.

Establecimiento:

- **(GET)**
getEstablecimientoGerente: “{{base_url}}/local/me”:
 - Devuelve el establecimiento del gerente loggeado
 - Devuelve el establecimiento

- **(GET) buscarLocal: “{{base_url}}/local/”:**
 - Crea una lista de todos los establecimientos registrados

- Devuelve la lista con todos los establecimientos registrados
- **(GET) unEstablecimiento:"{{base_url}}/local/{id}":**
 - Devuelve un establecimiento buscado por id
 - Se le pasa el id del establecimiento seleccionado
 - Devuelve el establecimiento seleccionado
- **(POST) añadirLocalFavs:"{{base_url}}/local/{id}":**
 - Añade un local a la lista de favoritos
 - Se le pasa el id del establecimiento seleccionado para añadir a la lista de favoritos
- **(GET) listarLocalesFavs:"{{base_url}}/local/fav/":**
 - Lista todos los locales de la lista de favoritos
 - Devuelve la lista de todos los locales favoritos
- **(POST) nuevoEstablecimiento:"{{base_url}}/local/":**
 - Crea un nuevo establecimiento
 - Se le pasa el archivo y los parámetros que componen el objeto de establecimiento, ya que es una petición multiparte
 - Devolverá el establecimiento registrado
- **(PUT) editarEstablecimiento: " {{base_url}}/local/{id}":**
 - Edita un establecimiento seleccionado
 - Se le pasa el id del establecimiento seleccionado

- Devuelve el establecimiento seleccionado editado
- **(PUT)editarfotoEstablecimiento:"{{base_url}}/local/{id}/editPhoto":**
 - Edita una foto de un establecimiento seleccionado
 - Se le pasa el id del establecimiento seleccionado y el cuerpo del archivo que se utilizará para editar
 - Devolverá el establecimiento con la imagen editada
- **(DELETE)**
deleteEstablecimiento:"{{base_url}}/establecimiento{id}"
":
 - Elimina un establecimiento
 - Se le pasa el id del establecimiento que vamos a eliminar

File:

- **(GET) downloadFile:**
"{{base_url}}/downloadFile/{filename}":
 - Descarga un archivo
 - Se le pasa el nombre del fichero
 - Devuelve el fichero descargado
- **(GET) getImage: "{{base_url}}/image/{filename}":**
 - Trata el archivo descargado para convertirlo en una imagen
 - Se le pasa el nombre del fichero
 - Devuelve el path del fichero para ser utilizado

Gerente:

- **(POST) register: "{{base_url}}/register":**

- Crea un usuario gerente
- Se le pasará por parámetros las distintas variables, siendo una petición multiparte.
- Devolverá el gerente creado
- **(GET) listarGerentes: “{{base_url}}/list”:**
 - Lista todos los gerentes registrados
 - Devolverá la lista con todos los gerentes registrados

Pedido:

Producto:

- **(GET) getProductosGerente: “{{base_url}}/local/me/productos/”:**
 - Devuelve todos los productos de un gerente
 - Devuelve una lista con todos los productos del gerente
- **(POST) newProducto: “{{base_url}}/producto/”:**
 - Crea un nuevo producto
 - Se le pasa el archivo y los parámetros que componen el objeto de producto, ya que es una petición multiparte
 - Devuelve el producto creado
- **(DELETE) deleteProducto: “{{base_url}}/producto/{id}”:**
 - Borra un producto
 - Se le pasa el id del producto que vamos a borrar
- **(PUT) editProducto: “{{base_url}}/producto/{id}”:**
 - Edita un producto seleccionado
 - Se le pasa el id del producto que vamos a editar
 - Devuelve el producto editado

- **(PUT)editarFotoProducto**
:“{{base_url}}/local/{id}/editProducto”:
 - Edita una foto de un producto seleccionado
 - Se le pasa el id del producto seleccionado y el cuerpo del archivo que se utilizará para editar
 - Devolverá el producto con la imagen editada

Role:

- **(GET) getRole: “{{base_url}}/{username}”:**
 - Devuelve el rol del usuario indicado
 - Se le pasa el nombre de usuario para obtener el rol
 - Devolverá el rol del usuario indicado

IMPLEMENTACIÓN

API

Aquí se describen las clases y paquetes de la API

CONTROLLERS

Este paquete recoge las clases que implementa las peticiones de los modelos

- **AdminController:** Clase que implementa los servicios del modelo Administrador
- **CategoriaController:** Clase que implementa los servicios del modelo Categoria

- **ClienteController:** Clase que implementa los servicios del modelo Cliente
- **EstablecimientoController:** Clase que implementa los servicios del modelo Establecimiento
- **FileController:** Clase que implementa los servicios del modelo File
- **GerenteController:** Clase que implementa los servicios del modelo Gerente
- **LineaPedidoController:** Clase que implementa los servicios del modelo LineaPedido
- **PedidoController:** Clase que implementa los servicios del modelo Pedido
- **ProductoController:** Clase que implementa los servicios del modelo Producto
- **RoleController:** Clase que implementa los servicios del modelo Role

CONVERSORES

Este paquete recoge las clases que implementan los métodos para convertir modelos a dto y viceversa

- **ConversorCategoria** clase que implementa los métodos para convertir dtos y modelos de Categoria
- **ConversorEstablecimiento** clase que implementa los métodos para convertir dtos y modelos de Establecimiento
- **ConversorGerente** clase que implementa los métodos para convertir dtos y modelos de Gerente
- **ConversorImagen** clase que implementa los métodos para convertir dtos y modelos de Imagen
- **ConversorProducto** clase que implementa los métodos para convertir dtos y modelos de Producto
- **ConversorUbicacion** clase que implementa los métodos para convertir dtos y modelos de Ubicacion
- **ConversorUsuario** clase que implementa los métodos para convertir dtos y modelos de Usuario

DTOS

Este paquete recoge todos los dtos utilizado para la conversión de los datos de modelo

- **CategoriaDto** clase que filtra la clase modelo Categoria
- **CategoriaDto2** clase que filtra la clase modelo Categoria
- **CategoriaDtoName** clase que filtra la clase modelo Categoria
- **ClienteDto** clase que filtra la clase modelo Cliente
- **CreateAdminDto** clase que filtra la clase modelo Admin
- **CreateCategoriaDto** clase que filtra la clase modelo Categoria
- **CreateClienteDto** clase que filtra la clase modelo Cliente
- **CreateEstablecimientoDto** clase que filtra la clase modelo Establecimiento
- **CreateGerenteDto** clase que filtra la clase modelo Gerente
- **CreateLocalizacionDto** clase que filtra la clase modelo Localizacion
- **CreateProductoDto** clase que filtra la clase modelo Producto
- **CreateUserDto** clase que filtra la clase modelo User
- **EditAvatarDto** clase que filtra la clase modelo Avatar
- **EditEstablecimientoDto** clase que filtra la clase modelo Establecimiento
- **EditImagenEstablecimientoDto** clase que filtra la clase modelo Establecimiento
- **EditImagenProductoDto** clase que filtra la clase modelo Producto
- **EditProductoDto** clase que filtra la clase modelo Producto
- **EstablecimientoDto** clase que filtra la clase modelo Establecimiento
- **GerenteDto2** clase que filtra la clase modelo Gerente
- **ImagenDto** clase que filtra la clase modelo Imagen
- **ListaEstablecimientoDto** clase que filtra la clase modelo Establecimiento
- **ListaProductosDto** clase que filtra la clase modelo Producto
- **ListUsersDto** clase que filtra la clase modelo Usuario
- **ProductoDto** clase que filtra la clase modelo Producto

- **ProductoDto2** clase que filtra la clase modelo Producto
- **UbicacionDto** clase que filtra la clase modelo Ubicacion
- **UserDto** clase que filtra la clase modelo Usuario

Dto.Converter

Paquete con la clase UserDtoConverter utilizada en la seguridad

- **UserDtoConverter** clase que es utilizada en la seguridad

Error

Este paquete recoge las clases que gestionan los distintos errores

- **ApiError** clase que gestiona los errores
- **CustomAccesDeniedHandler** clase que gestiona los errores
- **CustomAuthenticationEntryPoint** clase que gestiona los errores
- **EntityNotFoundException** clase que gestiona los errores
- **RestExceptionHandler** clase que gestiona los errores

Files

Este paquete recoge las clases que gestionan subida, descarga y almacenamiento de ficheros

- **FileStorageException** clase que gestiona el almacenamiento de archivos
- **FileStorageProperties** clase que gestiona las propiedades del almacenamiento
- **FileSystemStorageService** clase que implementa el servicio de almacenamiento
- **MyFileNotFoundException** clase que implementa las excepciones del almacenamiento

Model

Este paquete recoge las clases que implementan los modelos de la api

- **Admin** clase que implementa el modelo Administrador
- **Avatar** clase que implementa el modelo Avatar
- **Categoria** clase que implementa el modelo Categoria
- **Cliente** clase que implementa el modelo Cliente
- **Establecimiento** clase que implementa el modelo Establecimiento
- **Gerente** clase que implementa el modelo Gerente
- **Imagen** clase que implementa el modelo Imagen
- **LineaPedido** clase que implementa el modelo LineaPedido
- **Pago** clase que implementa el modelo Pago
- **Pedido** clase que implementa el modelo Pedido
- **Producto** clase que implementa el modelo Producto
- **Role** clase que implementa el modelo Role
- **Ubicacion** clase que implementa el modelo Ubicacion
- **Usuario** clase que implementa el modelo Usuario

Repository

Este paquete recoge las clases que implementan los repositorios de los modelos de la api

- **AdminRepository** clase que implementa el repositorio del modelo Administrador
- **AvatarRepository** clase que implementa el repositorio del modelo Avatar
- **CategoriaRepository** clase que implementa el repositorio del modelo Categoria
- **ClienteRepository** clase que implementa el repositorio del modelo Cliente
- **EstablecimientoRepository** clase que implementa el repositorio del modelo Establecimiento
- **GerenteRepository** clase que implementa el repositorio del modelo Gerente

- **ImagenRepository** clase que implementa el repositorio del modelo Imagen
- **ProductoRepository** clase que implementa el repositorio del modelo Producto
- **UbicacionRepository** clase que implementa el repositorio del modelo Ubicacion
- **UsuarioRepository** clase que implementa el repositorio del modelo Usuario

Security

Este paquete recoge las clases que implementan la seguridad OAuth2 + Spring Security

- **Cors** clase que implementa la seguridad
- **CorsConfigServer** clase que implementa la seguridad
- **CustomUserDetailsService** clase que implementa la seguridad
- **OAuthConfig** clase que implementa la seguridad
- **ResourceServerConfig** clase que implementa la seguridad
- **ServerSecurityConfig** clase que implementa la seguridad

Services

Este paquete recoge las clases que implementan los servicios de los modelos de la api

- **AdminService** clase que implementa los métodos del modelo Administrador
- **AvatarService** clase que implementa los métodos del modelo Avatar
- **BaseService** clase que implementa los métodos básicos
- **CategoriaService** clase que implementa los métodos del modelo Categoria

- **ClienteService** clase que implementa los métodos del modelo Cliente
- **EstablecimientoService** clase que implementa los métodos del modelo Establecimiento
- **GerenteService** clase que implementa los métodos del modelo Gerente
- **ImagenService** clase que implementa los métodos del modelo Imagen
- **ProductoService** clase que implementa los métodos del modelo Producto
- **UbicacionService** clase que implementa los métodos del modelo Ubicacion

