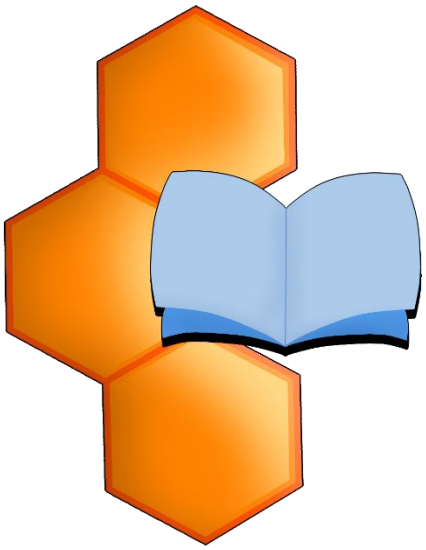


COLMENIAL:

PROYECTO FINAL

Memoria técnica y migración de tecnologías realizada sobre el proyecto realizado durante el ciclo de Desarrollo de Aplicaciones Web

Autor: García Fernández, Álvaro

Tutor: Del Toro Gómez, Salvador

Promoción: 2016-2018

Agradecimientos:

En primero lugar agradecer al centro CIPFP Mislata los conocimientos y dinámicas aprendidos. Así como a los amigos y compañeros que he encontrado a lo largo de estos dos últimos años, por su compañía y su apoyo.

Por último, agradecer a Ramón Martínez Tarín, desarrollador de Adding Technology, su apoyo y ayuda en mi aprendizaje de las tecnologías JSP.

ÍNDICE

1. Introducción

Colmenial es un proyecto nacido durante el vigente curso de Desarrollo de Aplicaciones Web en la asignatura de Empresa e iniciativa emprendedora, el objetivo de la actividad era la de la creación teórica de una start-up. Durante el proceso de creación fue sufriendo diferentes cambios derivados del propio proceso y otras circunstancias.

El objetivo del presente documento es informar sobre el proceso de migración de la tecnología back-end, así como tratar de presentar una idea general del diseño y funcionamiento de la aplicación en su versión práctica, también presentar las características técnicas y organizativas que hemos empleado en él.

1.1 Idea de negocio

Como ya he citado anteriormente la idea original del proyecto fue sufriendo cambios. La idea final del proyecto teórico presentado en la asignatura de Empresa e iniciativa emprendedora era la de la creación de una plataforma que aunase todas las necesidades de los estudiantes universitarios, especialmente dirigida a aquellos que tienen que movilizarse para llevar a cabo sus estudios, ya que son los que más necesidades de este tipo atesoran.

Desde Colmenial sopesamos las necesidades de adaptación a una nueva ciudad por parte de los estudiantes universitarios y valoramos soluciones para algunas de ellas:

- ***Selección de centro**: Presentaríamos diferentes valoraciones y críticas de estos, estas valoraciones serian realizadas por usuarios de la aplicación, de esta manera los estudiantes podrían conocer las opiniones de otros estudiantes acerca de su centro o grado.*

- ***Alojamiento**: Nos convertiríamos en un canal para facilitar la comunicación entre demandantes y ofertantes de alojamiento universitario.*

- ***Apuntes**: Ofreceríamos una plataforma donde los estudiantes pudieran acceder a apuntes redactados por sus compañeros.*

- ***Ocio**: Presentaríamos una agenda lúdica y cultural dirigida especialmente a un público universitario.*

Finalmente, y dado el gran volumen que una aplicación de estas características llevaría consigo, decidimos centrar el proyecto práctico exclusivamente de una de estas funcionalidades, y, dado que era la que más se ajustaba a los requerimientos demandados por nuestros profesores, decidimos centrarnos en la funcionalidad de compra-venta de apuntes universitarios.

1.2 Tecnologías empleadas

Esta explicación de las tecnologías empleadas continua en el apartado Herramientas y tecnologías empleadas, donde se desarrolla con más detalle la funcionalidad de cada una de ellas.

Front-end:

Esta parte es la relacionada con el aspecto más visual del proyecto y en su momento empleamos tecnologías como la librería de Javascript Jquery, HTML5, el framework de CSS3 Bootstrap o el metalenguaje Sass.

Todo esto conectado a la parte back-end gracias a las peticiones AJAX de forma asíncrona.

Back-end:

Es la parte del proyecto que realiza las funciones de acceso a datos, el CRUD de modificación y visionado de estos datos y el tratamiento de los mismo, en esta parte empleamos el sistema gestor de bases de datos MySQL.

Esta es la parte del proyecto que ha sido modificada, anteriormente estaba realizada en el lenguaje PHP, no obstante, a lo largo de este proyecto he decidido realizar una migración de este lenguaje a las tecnologías JSP Spring Boot y JPA.

Los motivos por los cuales he realizado esta migración son los siguientes:

- *PHP es un lenguaje de programación de una envergadura más ligera, excelente cuando deseamos crear aplicaciones web sencillas, no obstante considero que en una aplicación como Colmenial, una vez terminada y con todas las funciones explicadas en el apartado de Idea de negocio, tendría una envergadura que podría dar problemas en el lenguaje PHP .*

- *La normalización empresarial en la utilización de tecnologías JSP hace que, en el mercado laboral, sean tecnologías muy demandadas y por lo tanto considero conveniente con mi carrera profesional el aprendizaje de estas.*

2. IMPLEMENTACIÓN

2.1 Análisis

2.1.1 Requerimientos iniciales

Diferenciamos dos tipos de requerimientos iniciales: Interfaz, Seguridad

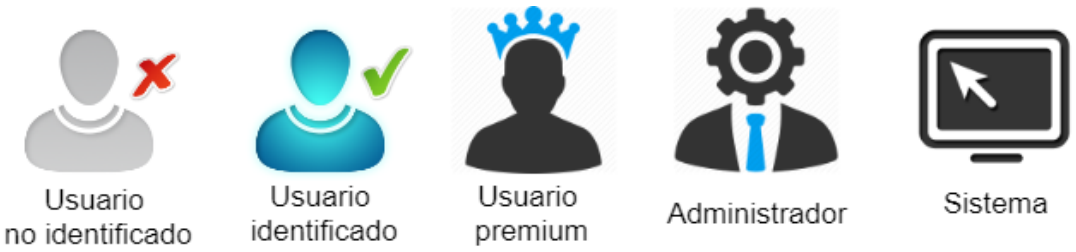
- ***Interfaz:*** *El interfaz visual de esta aplicación está diseñado de forma intuitiva, con el objetivo de facilitar la familiarización y habituamiento de uso de la aplicación con respecto al usuario, ya que consideramos que el usuario debe perder el mínimo tiempo posible en adaptarse a la aplicación o, de lo contrario, terminará por no utilizarla.*

- ***Seguridad:*** *En Colmenial diferenciamos cuatro tipos diferentes de usuarios diferenciados entre sí a la hora de realizar su autentificación*.

Nivel de seguridad	Tipo usuario	Funcionalidad
0	No identificado	Observar los servicios ofrecidos por la aplicación.
1	Usuario identificado	• Observar los servicios ofrecidos en la aplicación • Adquirir los servicios ofrecidos en la aplicación.
2	Premium	• Observar los servicios ofrecidos en la aplicación • Adquirir los servicios ofrecidos en la aplicación. • Publicar un nuevo servicio.
3	Administrador	Gestión de la aplicación.

2.1.2 Actores del sistema

Diferenciamos a los actores del sistema en función de su nivel de seguridad es por ello que podemos establecer una relación entre los niveles de seguridad previamente explicados los actores del sistema, de este modo diferenciamos 5 tipos de actores del sistema si añadimos el propio sistema.



- ***Usuario no identificado**: Es el usuario que no ha realizado la labor de iniciar sesión y probablemente no esté registrado en el sistema, su funcionalidad en la aplicación será principalmente la de visualizar los servicios ofrecidos y si alguno de ellos satisface sus necesidades iniciar sesión o registrarse y finalizar el proceso de compra.*

- ***Usuario identificado**: Es el usuario que está registrado dentro del sistema, puede, además de realizar las mismas funciones del usuario no registrado, realizar compras y hacer valoraciones de los productos adquiridos, esta valoración repercutirá en la valoración media del usuario premium que ofrece el servicio.*

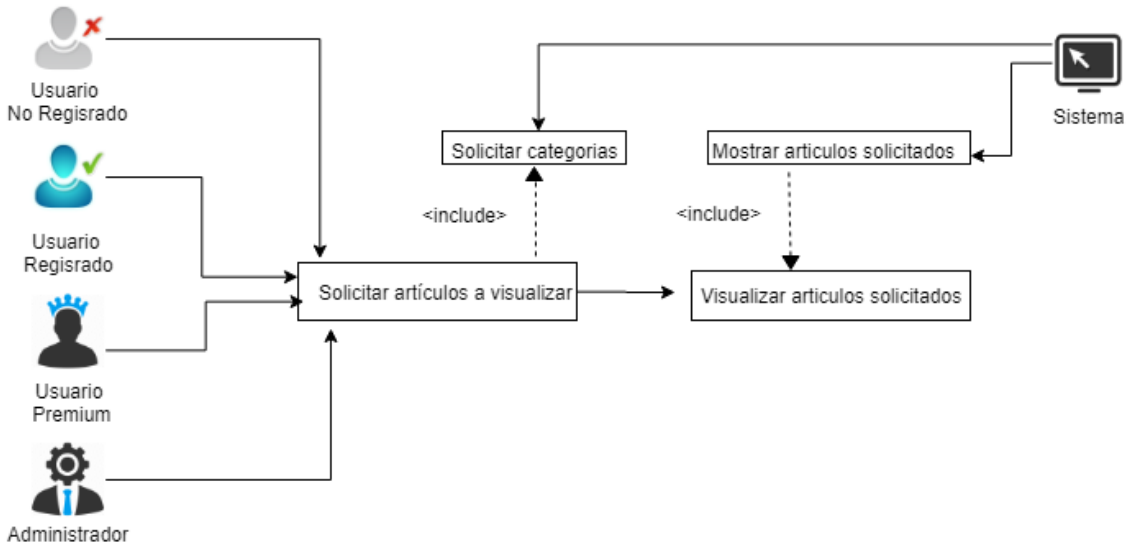
- ***Usuario premium**: Este es el usuario que proporciona los servicios, para llegar a ese rango debe de pagar una cuota mensual/semanal, que indicará el tiempo en el cual sus anuncios estarán publicados en la aplicación. Además, también puede llevar a cabo las funciones del usuario identificado.*

- ***Administrador**: Es el encargado de regular la lógica de la aplicación, tiene poderes administrativos sobre usuarios, apuntes, módulos, grados y centros.*

- ***Sistema**: Es el agente que dada la información ofrecida por los usuarios llevará a cabo las acciones que formarán el sistema.*

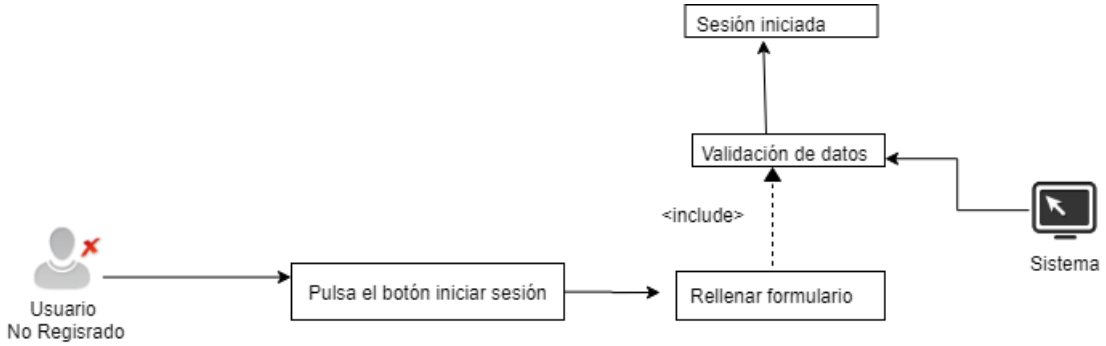
2.1.3 Diagramas de casos de uso

- Visionado de servicios:



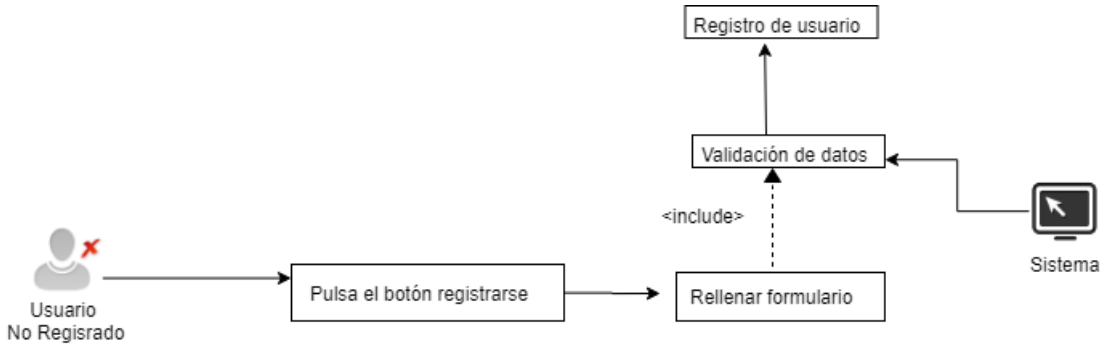
Nombre	Visionado de servicios anunciados
Descripción	Visualización de los apuntes anunciados
Actores	Usuario no registrado, usuario registrado, usuario premium, administrador, sistema.
Precondición	
Caso normal de uso	1. Acceder a la dirección de la aplicación. 2. Seleccionar la universidad de la cual queremos ver los apuntes disponibles. 3. Seleccionar el grado del cual queremos ver los apuntes disponibles. 4. Seleccionar el modulo del cual queremos ver los apuntes disponibles. 5. El sistema mostrará los artículos de la categoría seleccionada 6. Ver los apuntes.
Flujo alternativo	

- **Inicio de sesión:**



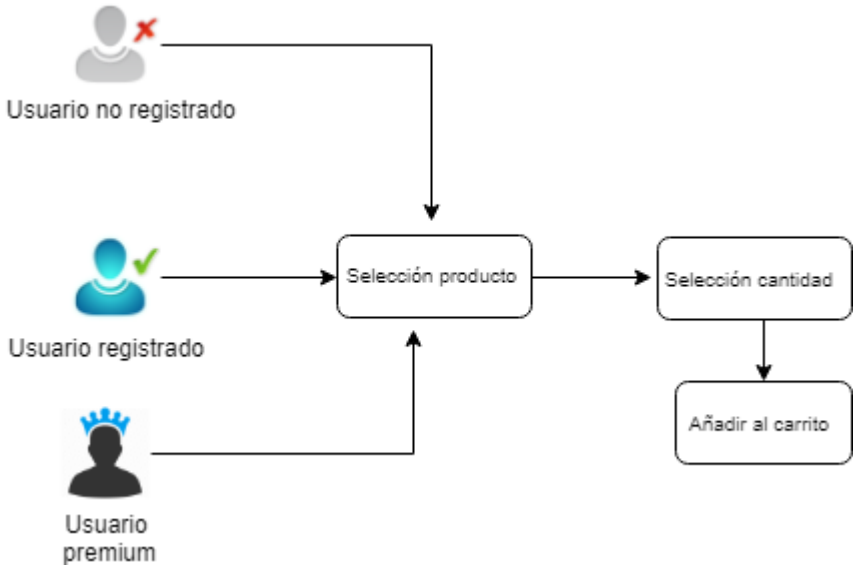
Nombre	Inicio de sesión
Descripción	Usuario registrado quiere iniciar sesión
Actores	Usuario no registrado, sistema
Precondición	Usuario registrado
Caso normal de uso	1. Pulsar el botón de inicio de sesión. 2. Rellenar formulario. 3. Enviar solicitud de inicio de sesión a sistema. 4. Validación de sesión por parte del sistema. 5. Inicio de sesión.
Flujo alternativo	4. Sesión errónea 5. Volver al paso 2

- **Registrar usuario**



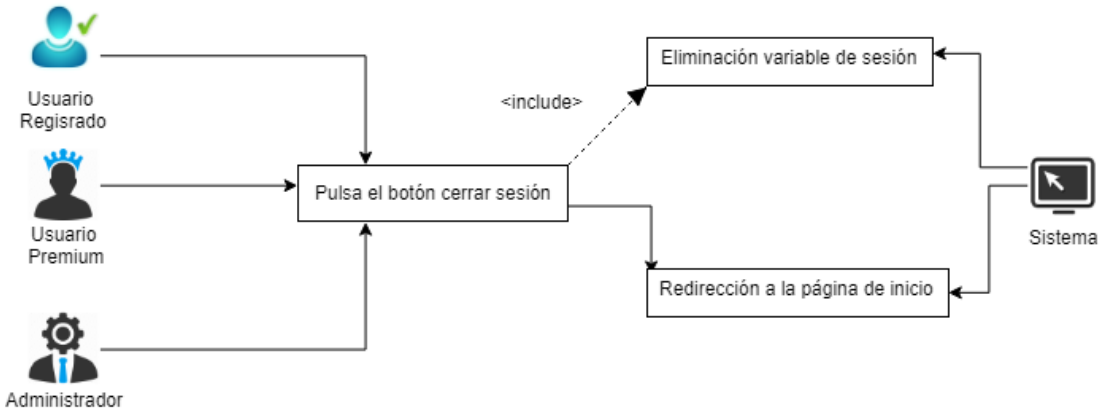
Nombre	Registrar usuario
Descripción	Usuario se registra en el sistema
Actores	Usuario no registrado, sistema
Precondición	
Caso normal de uso	1. Pulsar el botón de registrarse. 2. Rellenar formulario. 3. Enviar solicitud de registro al sistema. 4. Validación de datos. 5. Registro del usuario por parte del sistema 6. Inicio de sesión.
Flujo alternativo	4. Los datos introducidos ya existen en el sistema. 5. Volver al paso 2

- **Añadir al carrito**



Nombre	Añadir al carrito
Descripción	Producto añadido al carrito para posterior compra
Actores	Usuario no registrado, usuario registrado, usuario premium
Precondición	Visionado de servicios
Caso normal de uso	1. Seleccionar el apune. 2. Seleccionar la cantidad de apunes deseada. 3. Pulsar añadir al carrito.
Flujo alternativo	

- **Cierre de sesión:**

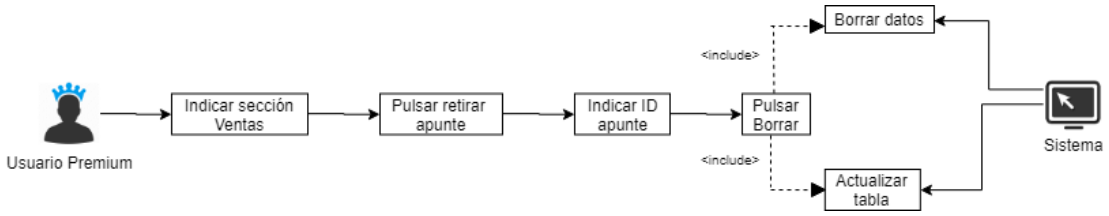


Nombre	Cierre de sesión
Descripción	Cierre de sesión de un usuario registrado
Actores	Usuario registrado, usuario premium, administrador
Precondición	Iniciar sesión
Caso normal de uso	1. Pulsar botón de cerrar sesión. 2. El sistema eliminará la variable de sesión y redireccionará al usuario a la página de inicio.

Flujo alternativo	

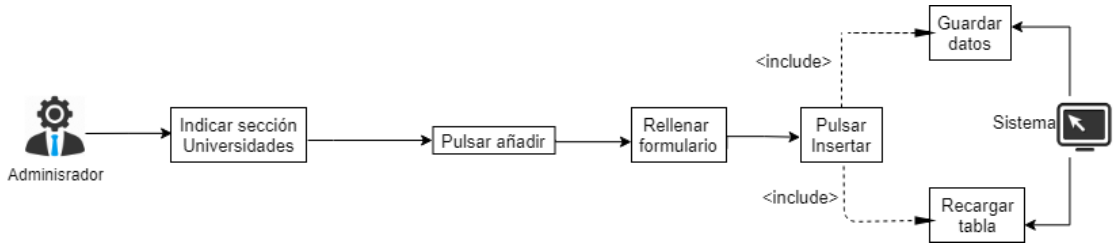
Los siguientes diagramas son diagramas representativos de todas las funciones de los paneles de administración (inserción, visionado, edición y borrado), estos diagramas son idénticos a los de la misma función en otra sección.

- **Retirar apunte:**



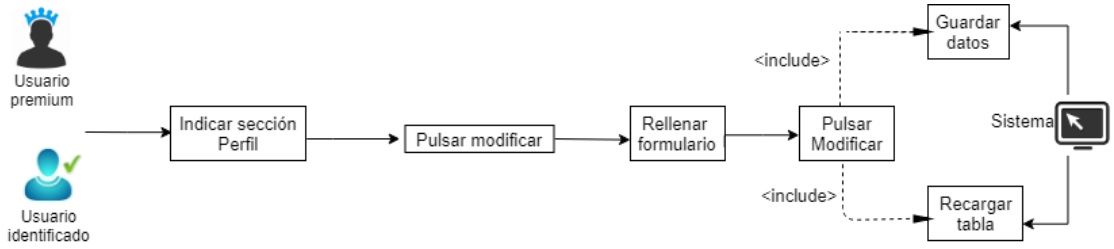
Nombre	Retirar apunte
Descripción	Retiro de venta de un apunte
Actores	usuario premium
Precondición	Iniciar sesión
Caso normal de uso	1. Entrar en el panel de usuario. 2. Indicar sección Ventas 3. Pulsar botón retirar apunte. 4. Indicar la id del apunte. 5. Pulsar Borrar. 6. El sistema borrará el apunte y actualizará la tabla de apuntes a la venta.
Flujo alternativo	

- **Insertar universidad:**



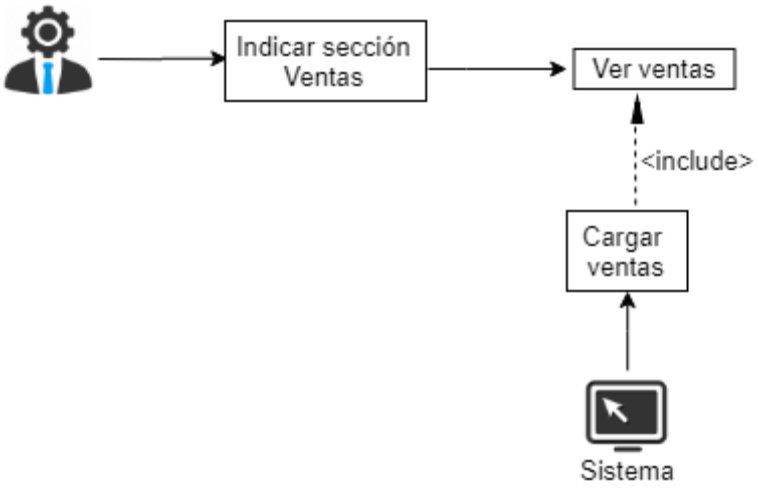
Nombre	Insertar universidad
Descripción	Inserción de una nueva universidad
Actores	Administrador
Precondición	Iniciar sesión
Caso normal de uso	1. Entrar en el panel de administrador. 2. Indicar sección Universidades 3. Pulsar botón añadir universidad. 4. Rellenar formulario con los datos de la universidad. 5. Pulsar insertar. 6. El sistema insertará la universidad y actualizará la tabla de universidades.
Flujo alternativo	

- **Modificar perfil:**



Nombre	Modificar perfil
Descripción	Modificación del perfil de un usuario
Actores	Usuario premium, usuario identificado
Precondición	Iniciar sesión
Caso normal de uso	1. Entrar en el panel de usuario. 2. Indicar sección Perfil 3. Pulsar botón modificar perfil. 4. Rellenar formulario con los datos del perfil. 5. Pulsar modificar. 6. El sistema modificará los datos introducidos.
Flujo alternativo	

-Ver ventas totales:



Nombre	Ver ventas totales
Descripción	Visionado del total de las ventas realizadas
Actores	Administrador
Precondición	Iniciar sesión
Caso normal de uso	1. Entrar en el panel de administrador. 2. Indicar sección Ventas. 3. El sistema mostrará las ventas realizadas almacenadas en este.

	4. Ver ventas.
Flujo alternativo	

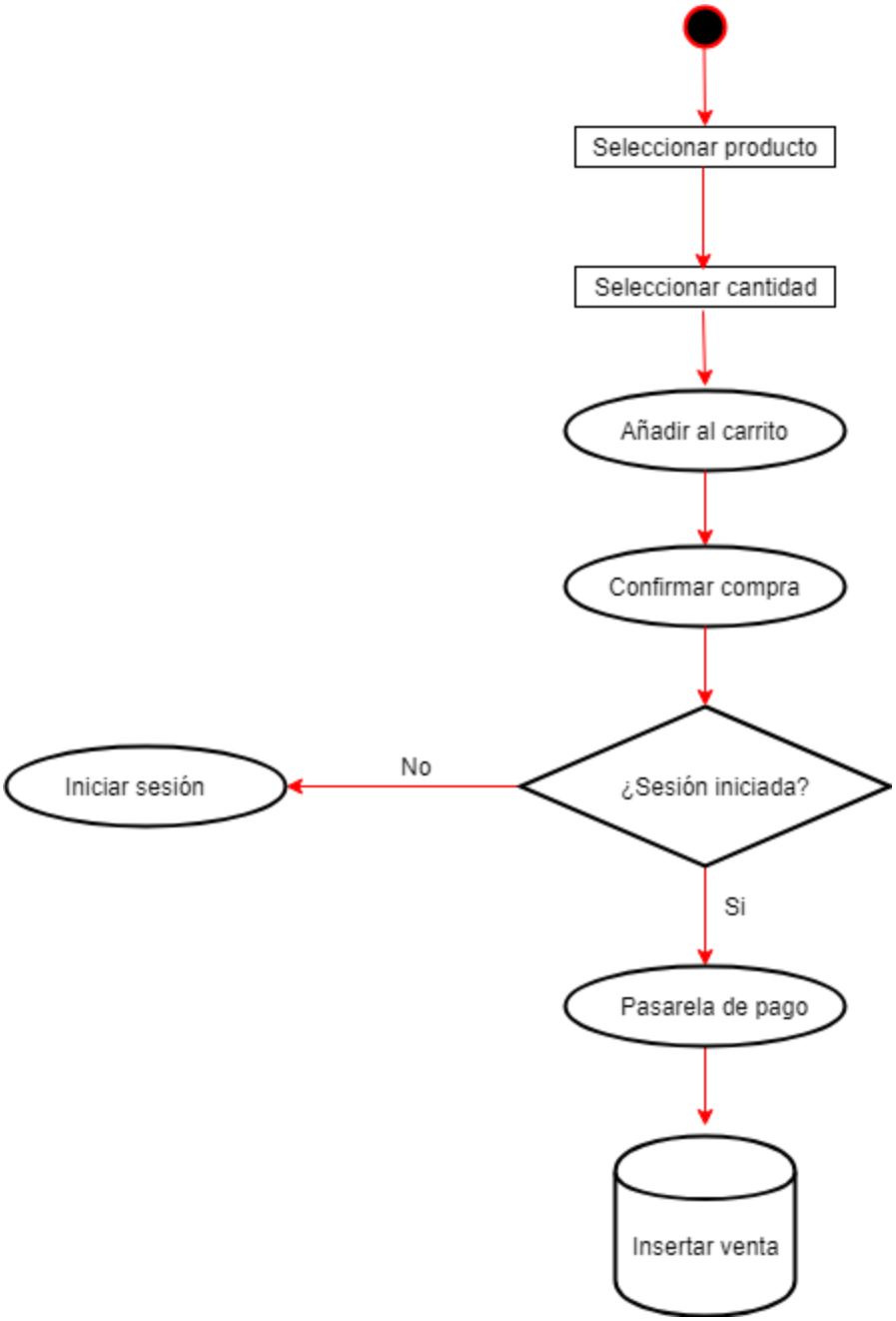
2.1.4 Diagramas de actividad

- **Visionado de servicios:**

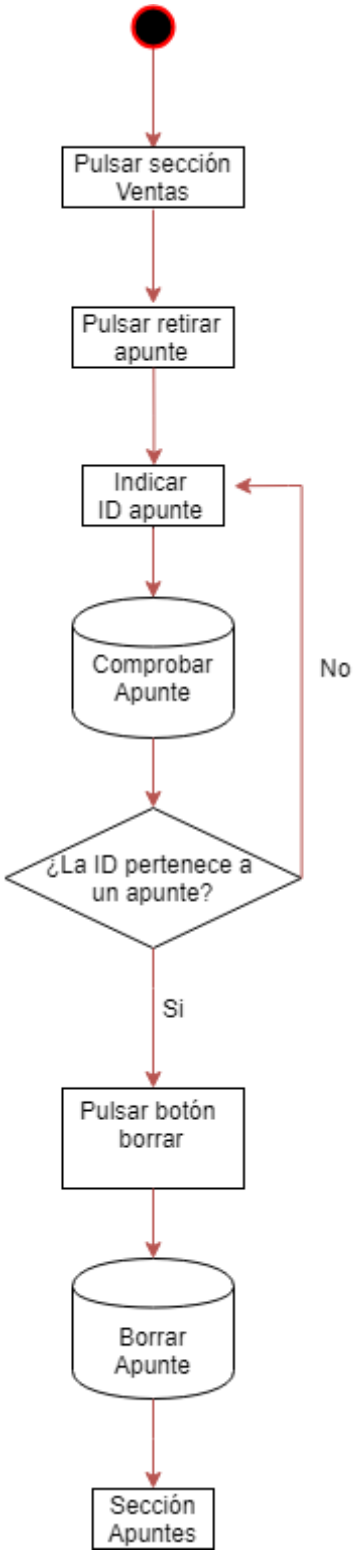
- **Registrar usuario:**

- **Iniciar sesión:**

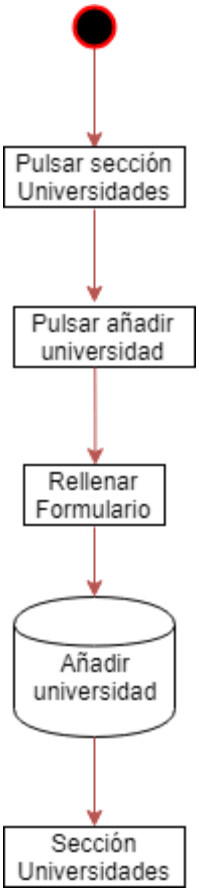
- **Añadir al carro:**



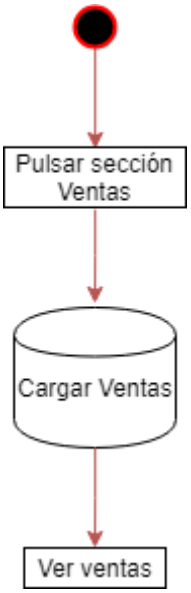
- **Retirar apunte:**



- **Añadir universidad:**

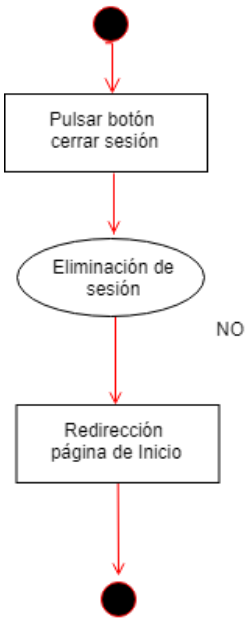


- **Ver productos en ventas:**



- **Modificar Perfil:**

- **Cerrar sesión:**



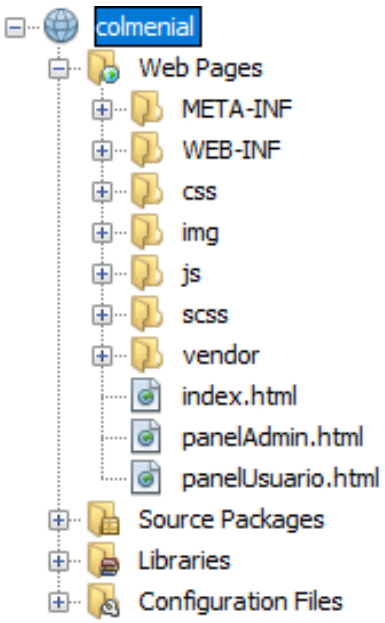
2.2.1Diseño

2.2.2 Estructura de Archivos

2.2.2.1 Front-end

- **Estructura general:**

Como podemos apreciar la estructura general de este proyecto se diferencia en diferentes carpetas que veremos a continuación, diferenciadas sobre el tipo de archivos que aloja en su interior.

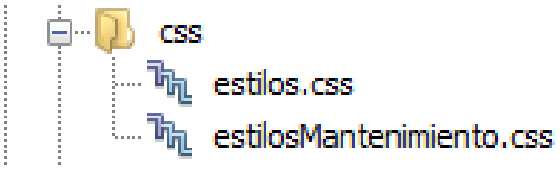


Además, en el directorio raíz, también diferenciamos los archivos HTML de las páginas index y los paneles de usuario y administrador, estos han sido separados en 3 páginas dadas las considerables diferencias entre estas páginas.

*Los subficheros **META-INF** y **WEB-INF,** así como los ficheros **Source Packages**, **Libraries** y **Configuration Files**, son archivos auto-generados por la estructura del proyecto en NetBeans y no han sufrido modificaciones ni se han creado archivos dentro de los mismos, es por ello que no serán explicados.*

- **Carpeta CSS:**

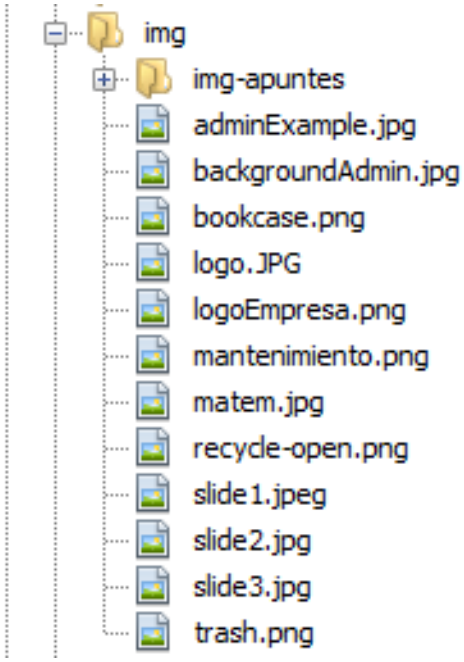
En esta carpeta es donde alojamos el archivo que dotará de estilos a nuestra web, en este caso contamos con dos archivos, dadas las múltiples diferencias entre la Vista Principal y los Paneles de Usuario y Administrador.



Es necesario añadir que este archivo será generado gracias a los archivos ubicados en la carpeta SCSS y a los scripts creados en el metalenguaje SASS.

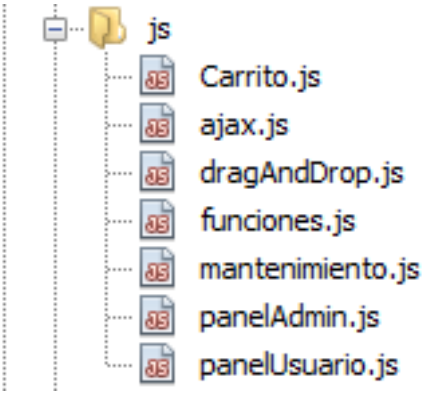
- **Carpeta IMG:**

*Esta carpeta es dónde se ven alojados los archivos pertenecientes a las imágenes que son mostradas en nuestra web, en este fichero encontramos la carpeta **img-apuntes**, que incluye las imágenes de los apuntes publicados por los usuarios.*



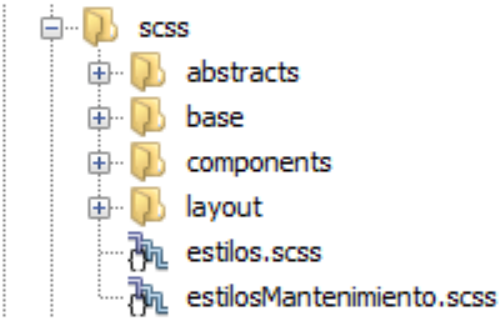
- **Carpeta JS:**

*Esta carpeta es dónde están alojados los archivos de JavaScript, estos archivos son **ajax.js**, encargado de realizar las peticiones AJAX a la API; **funciones.js**, que se encarga de alojar las funciones que pueden ser llamadas por varias páginas, el archivo **dragAndDrop.js** encargado de gestionar el funcionamiento de la función de arrastre para la eliminación de apuntes, el archivo **mantenimiento.js**, encargado de controlar el menú lateral de las páginas de mantenimiento las páginas encargadas de la manipulación del DOM de sus respectivas páginas HTML, **index.js**, **panelAdmin.js** y **panelUsuario.js**.*



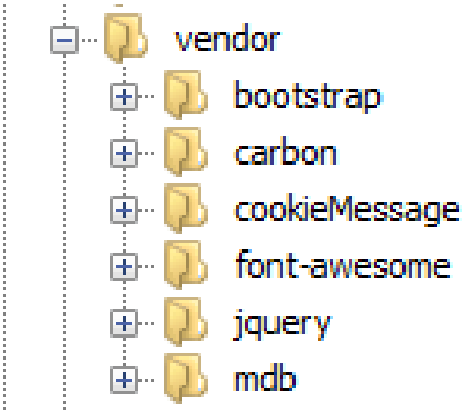
- Carpeta SCSS:

*Esta carpeta aloja los contenidos de los cuales se crearán los archivos CSS, en ella diferenciamos las sub-carpetas **asbstracts**, encargada de la definición de las variables que empleamos; **base**, que contiene la plantilla algunos estándares tipográficos; **components**, que aloja los diferentes componentes creados por nosotros y **layout**, que contiene la distribución y organización de nuestras vistas. Por último, aisladas de estas carpetas encontramos **estilos.scss** y **estilosMantenimiento.scss**, archivos que llaman a los componentes estético que emplearemos en nuestras vistas.*



- Carpeta Vendor:

Incluye las diferentes librerías que hemos empleado en el proyecto.



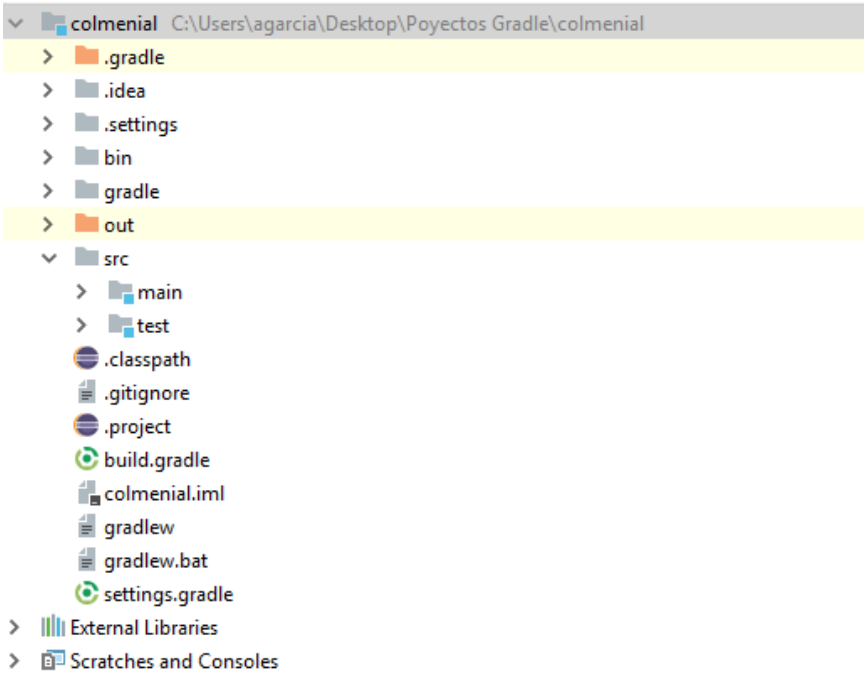
2.2.2.2 Back-end

Para la parte back-end he empleado el sistema Gradle y para obtener una estructura básica para mi proyecto he descargado desde la página Spring Initializr, cuyo link podemos encontrar en el apartado de Bibliografía, una plantilla de estructura.

Estos modelos y plantillas han insertado en mi proyecto una serie de carpetas y archivos en los cuales no voy a entrar a explicar y tan solo explicaré a continuación los ficheros y archivos que han sido creados o modificados de alguna forma por mí.

- **Estructura general:**

*Los archivos y ficheros que vamos a mostrar y explicar son, parte del contenido de la carpeta **src**, el archivo **gradle.build** y el fichero **External Libraries**.*



- External Libraries:

*Es donde el sistema descargará las librerías todas las dependencias que indiquemos en **build.gradle**.*

- Build.gradle:

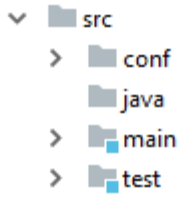
Este archivo es el encargado de gestionar la construcción del proyecto, aquí es donde indicaremos las dependías que queremos utilizar. La inyección de estas dependencias permitirá el funcionamiento de los frameworks, librerías y pluggins que deseamos emplear en nuestro proyecto.

Para inyectar estas dependencias es necesario introducir, en el apartado dependencies las órdenes de compilación obtenidas desde la web MVN Repository, cuyo enlace podemos encontrar en el apartado de Bibliografía.

```
dependencies {
    compile('org.springframework.boot:spring-boot-starter-data-jpa')
    compile('org.springframework.boot:spring-boot-starter-web'
    compile group: 'org.hibernate', name: 'hibernate-gradle-plugin', version: '5.2.17.Final'
    runtime('org.springframework.boot:spring-boot-devtools')
    runtime('mysql:mysql-connector-java')
    testCompile('org.springframework.boot:spring-boot-starter-test')
    compile group: 'org.modelmapper', name: 'modelmapper', version: '1.1.3'


}
```

- SRC:

*En el fichero SRC, como ya hemos indicado, existen algunos ficheros creados por la estructura Gradle, en las cuales no vamos a entrar. El único fichero de este primer nivel de SRC que contiene archivos que han sido modificados o creados es la carpeta **main**.*

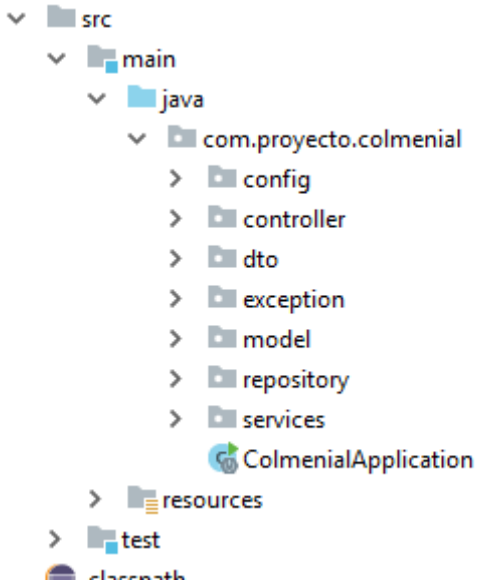


*El fichero **main** contiene las carpetas **java** y **resources**, la primera contiene nuestro proyecto en sí, mientas que en la segunda prepararemos los recursos que emplearemos a lo largo del proyecto.*

En el caso de este proyecto solo se ha utilizado el recurso de la conexión a la base de datos, la cadena de conexión a esta ha sido introducida en el archivo ***applicattion.properties*** *dentro del fichero* ***resources***

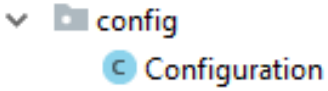
```
spring.datasource.url = jdbc:mysql://localhost:3306/colmenial2?useSSL=false
spring.datasource.username = root
spring.datasource.password = root

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.hibernate.ddl-auto = update
```

En el fichero ***java*** *encontramos el fichero* ***com.proyecto.colmenial****, cuyo nombre indica el nombre del grupo del proyecto y el nombre del propio proyecto, dentro de este fichero encontramos los ficheros que conforman nuestro proyecto.*



En la raíz de este fichero entramos el archivo ***ColmenialApplication****, este archivo es el ejecutable de nuestra aplicación.*

En el fichero ***config****, encontramos el archivo* ***Configuration****.*

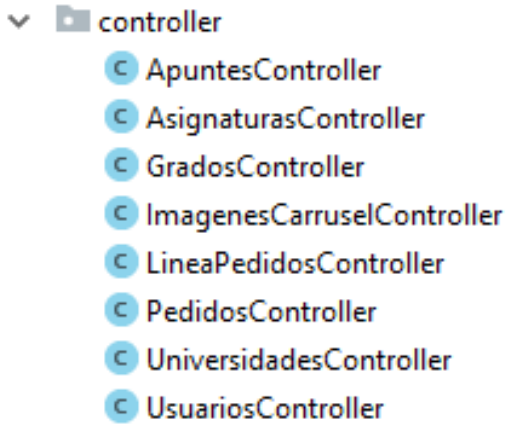


*Este archivo **Configuration** ha sido empleado para cambiar la configuración web del proyecto, con el objetivo de traspasar la seguridad de los navegadores web, ya que nuestro front-end y nuestro back-end se encuentran alojados en servidores web distintos.*

```
public class Configuration {
    @Bean
    public WebMvcConfigurer corsConfigurer() {
        return (WebMvcConfigurerAdapter) addCorsMappings(registry) → {
                registry.addMapping( pathPattern: "/greeting-javaconfig").allowedOrigins("http://localh
        };
    }

    @Bean
    public ModelMapper getModelMapper() { return new ModelMapper(); }

    @Bean
    public GradosServices getGradosService(){return new GradosServiceImpl();}

    @Bean
    public AsignaturasServices getAsignaturasService() { return new AsignaturasServicesImpl(); }

    @Bean
    public ApuntesServices getApuntesService(){return  new ApuntesServicesImpl(); }
}
```

También encontramos la declaración de algunas clases bajo la anotación @Bean, esto se hace para poder acceder a estas clases a lo largo del proyecto con la utilización de la etiqueta @Autowired.

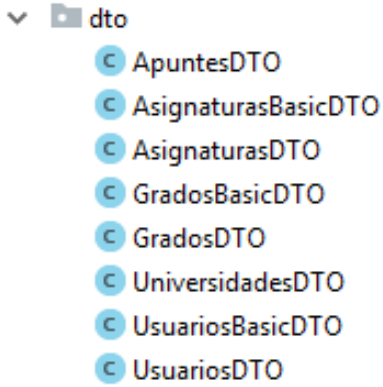
*En el fichero **controller** encontramos los archivos controladores de nuestra API, es por ello que tenemos tantas clases dentro de este fichero como tablas encontramos en la base de datos, exceptuando las tablas de relación entre clases (GradoAsignatura y GradoUniversidad).*



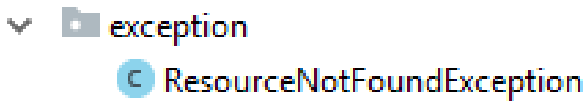
En cada una de las clases empleamos una serie de anotaciones que dan forma a nuestra API (@GetMapping, @PostMapping, @RestController etc…), así como empleamos las ya explicadas anotaciones @Autowired y definimos mediante la anotación @CrossOrigin que queremos que todas las direcciones URL pueden conectarse a nuestra API, esto último también introducido para traspasar la seguridad de los navegadores causado por tener las partes del proyecto en servidores web distintos.

```
import ...
@CrossOrigin(origins = "*")
@RestController
@RequestMapping("/colmenial")
public class ApuntesController {

    @Autowired
    ApuntesRepository apuntesRepository;

    @Autowired
    ApuntesServices apuntesServices;


    @GetMapping("/apuntes")
    public List<ApuntesDTO> getAllApuntes() {

        return apuntesServices.findAll();
    }

    @PostMapping("/apuntes")
    public Apuntes createApunte(@Valid @RequestBody Apuntes apunte) { return apuntesRepository.save(ap

    @GetMapping("/apuntes/id/{id}")
    public ApuntesDTO getApunteById(@PathVariable(value = "id") int id) {

        return apuntesServices.findByID(id);
    }
```

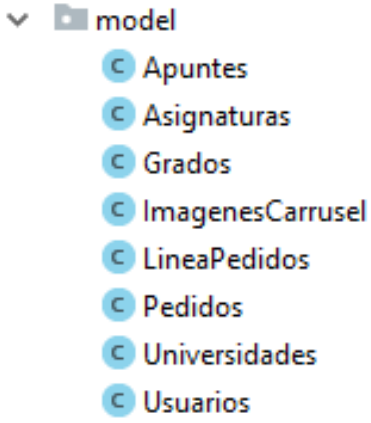
*El fichero **dto** está conformado por una serie de clases auxiliares creadas a causa de las relaciones empleadas en el proyecto, concretamente las relaciones correspondientes a las Foreign Keys de la base de datos, estas clases han sido creadas con el único objetivo de evitar el StackOverflowError, causado por la existencia de bucles infinitos en el proyecto.*



*El fichero **exception** no está implementado actualmente, es una posible mejora para un futuro cercano, el archivo alojado aquí cumplirá la función de controlar las excepciones causadas por un mal uso del API por parte del usuario.*



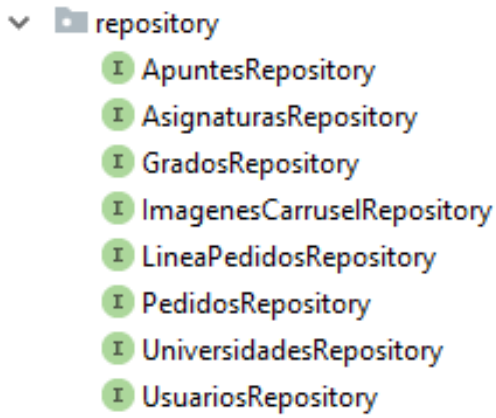
*En el fichero **model** encontramos las clases principales que conforman el proyecto.*



En estas clases es dónde indicamos al proyecto, mediante el uso de anotaciones, la estructura y la forma de nuestra base de datos.

```
@Entity
@Table(name = "grados")
@EntityListeners(AuditingEntityListener.class)

public class Grados implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String nombre;

    @ManyToMany
    @JoinTable()
    private List<Universidades> universidades;

    @ManyToMany
    @JoinTable()
    private List<Asignaturas> asignaturas;

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }

    public String getNombre() { return nombre; }
```

*En el fichero **repository** encontramos las clases utilizadas para la declaración del uso de las funciones de JPA, que será quien nos dé acceso a los datos.*



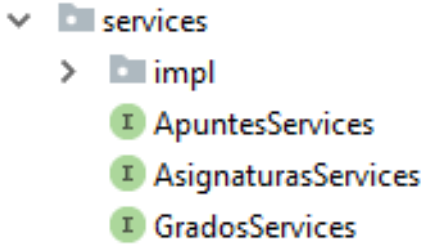
Por defecto, tan solo indicado que nuestra clase es una extensión de JPA, esta nos dotará de unas funciones básicas de consulta en la base de datos.

```
@Repository
public interface ImagenesCarruselRepository extends JpaRepository<ImagenesCarrusel, Integer> {

}
```

Pero también podemos indicarle más funciones JPA declarando funciones en nuestra clase.

```
@Repository
public interface AsignaturasRepository extends JpaRepository<Asignaturas, String> {
    public Asignaturas findByCodigo(String codigo);
    public Asignaturas findByNombre(String nombre);
    public List<Asignaturas> findByGrados_Id(int id);
}
```

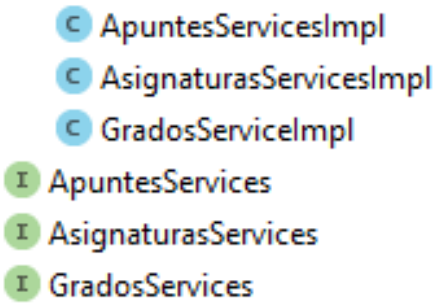
*El fichero **services**, al igual que las clases dto, ha sido creado con el único objetivo de evitar los bucles infinitos y el StackOverflowError, derivados del uso de relaciones entre nuestras clases.*



Incluye interfaces donde declararemos funciones abstractas para tratar estas clases.

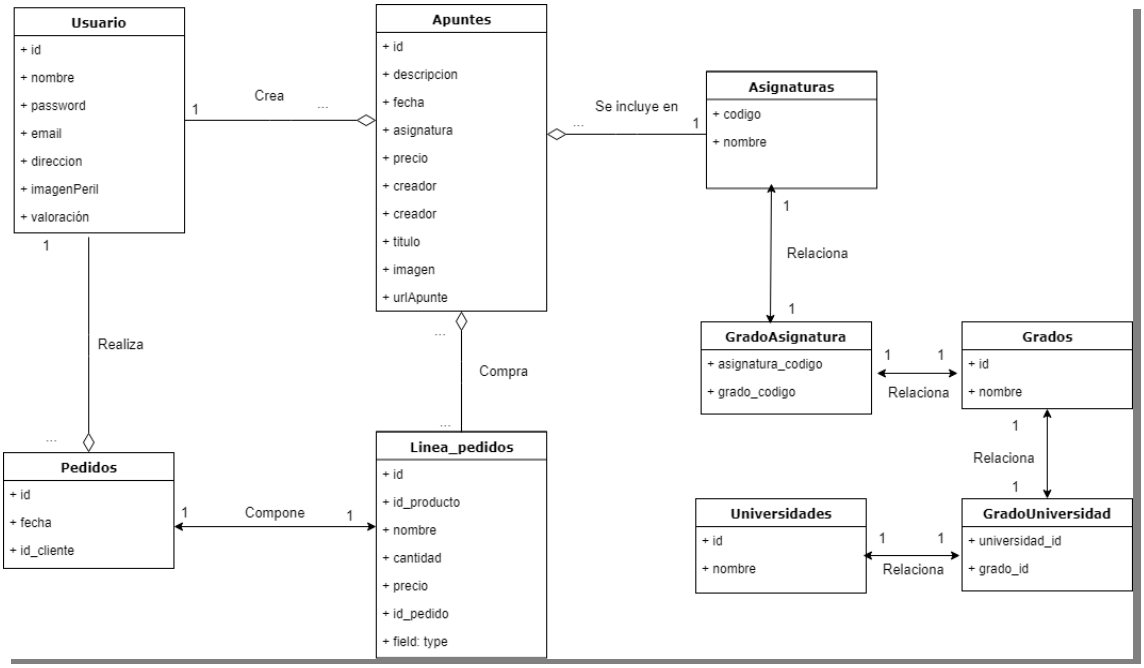
```
public interface GradosServices {
    public GradosDTO findByID(int id);

    public List<GradosDTO> findAll();

    public GradosDTO findByNombre(String nomrbe);

    public List<GradosDTO> findByUniversidades_Id(int id);
}
```

*Así como el fichero **impl**, que contiene las implementaciones de estas funciones.*



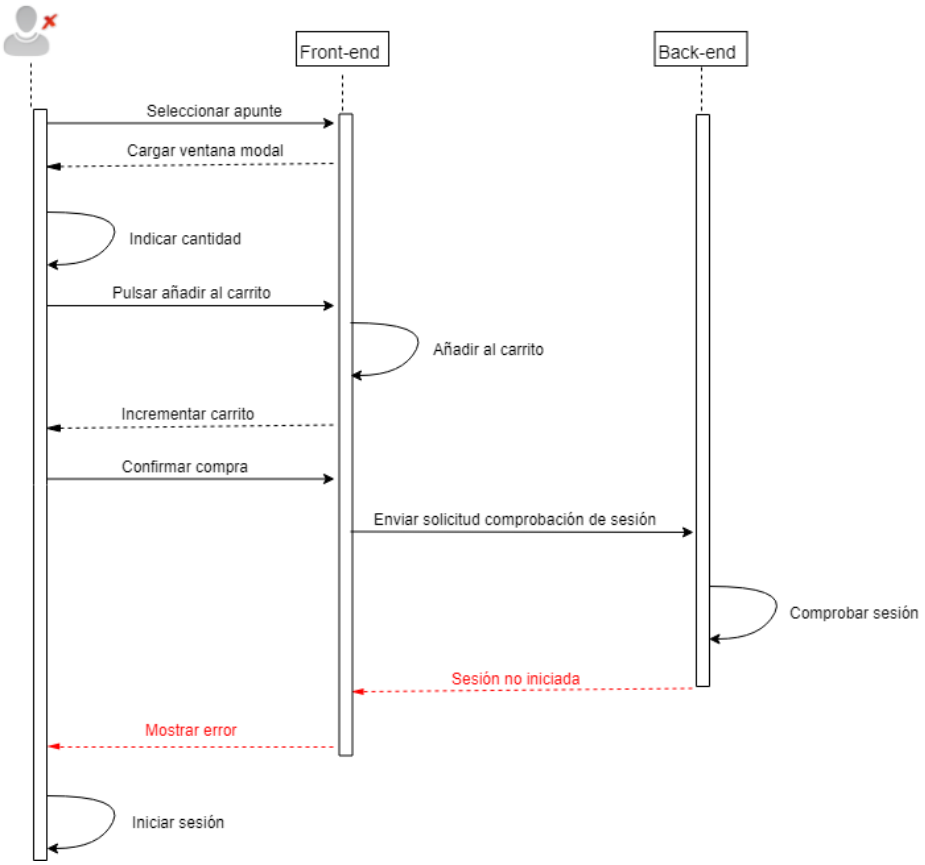
2.2.2 Diagrama de clases

El proyecto no consta de una estructura de clases como tal, aun así, podemos suponer una estructura de clases probable.

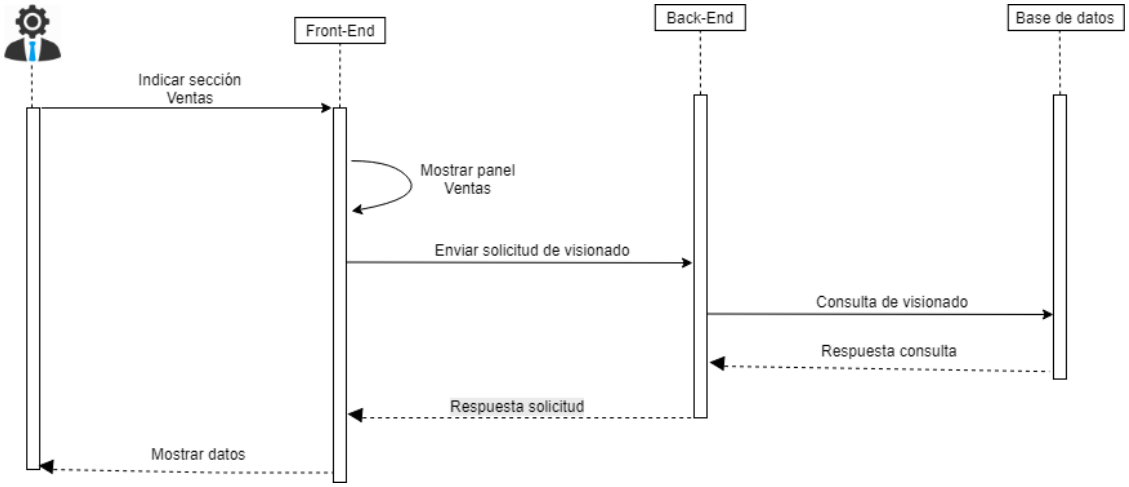


2.2.3Diagramas de secuencias

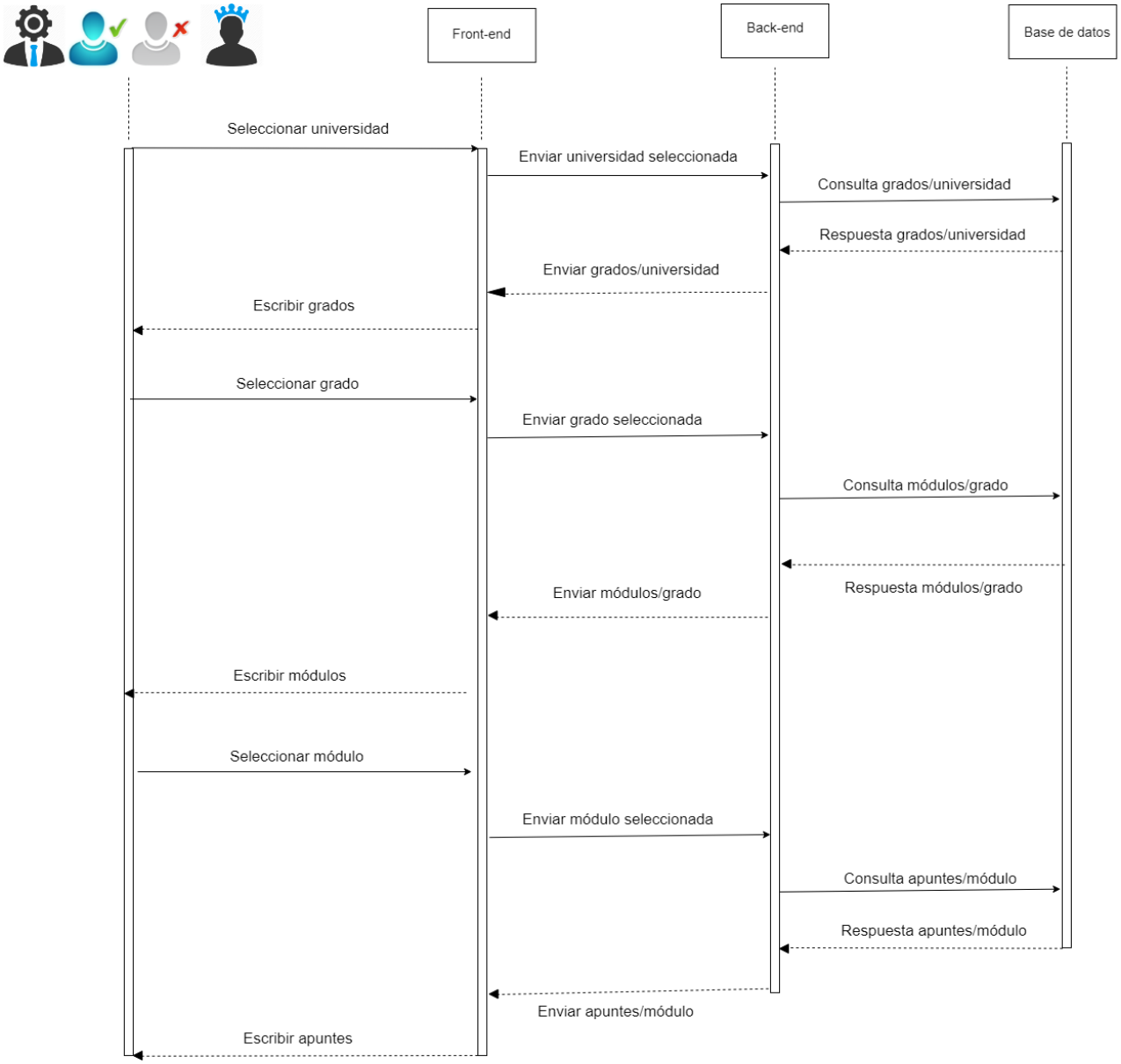
-Añadir al carro, sesión no iniciada:



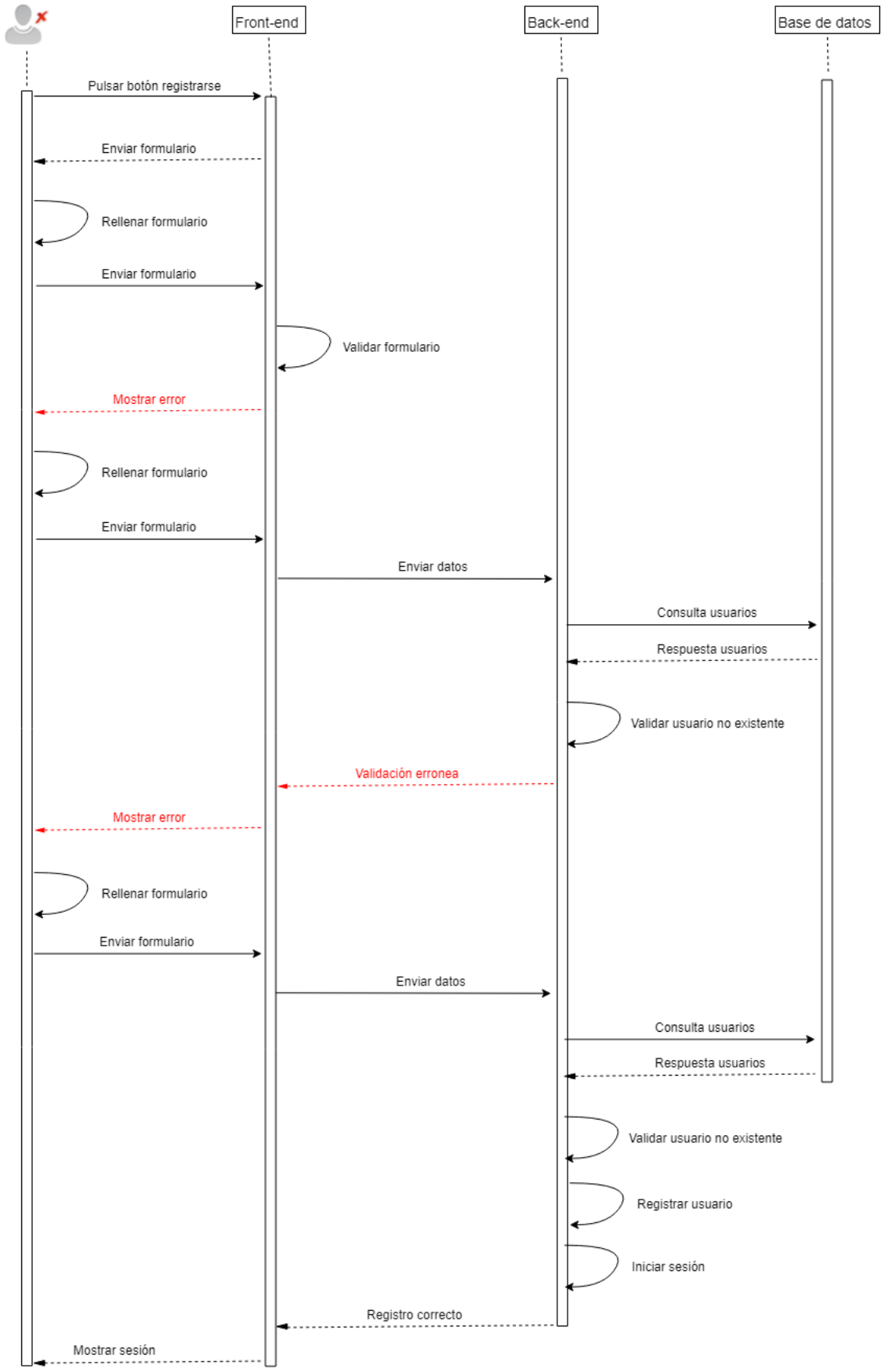
-Ver ventas:



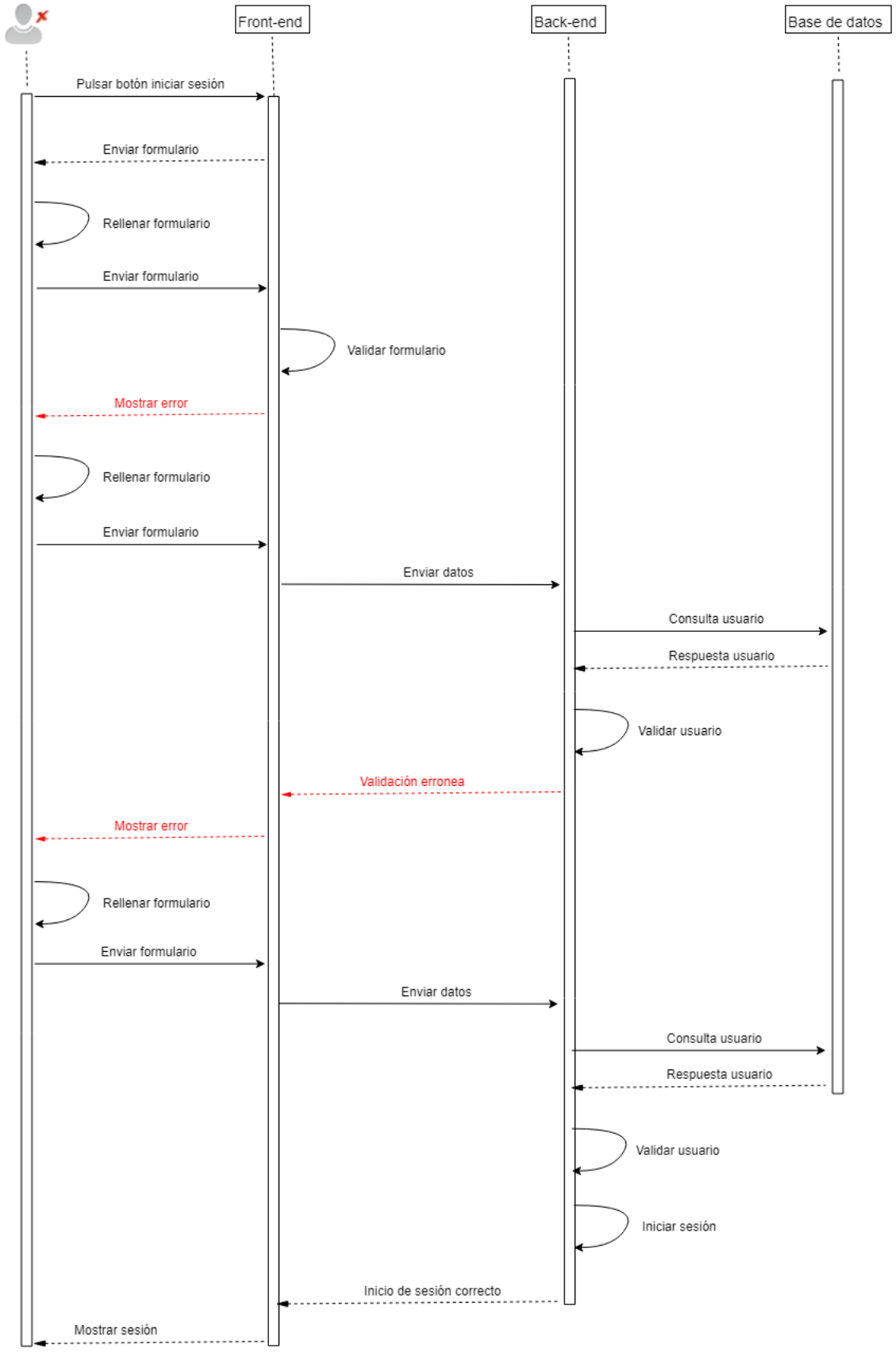
- **Visionado de servicios:**



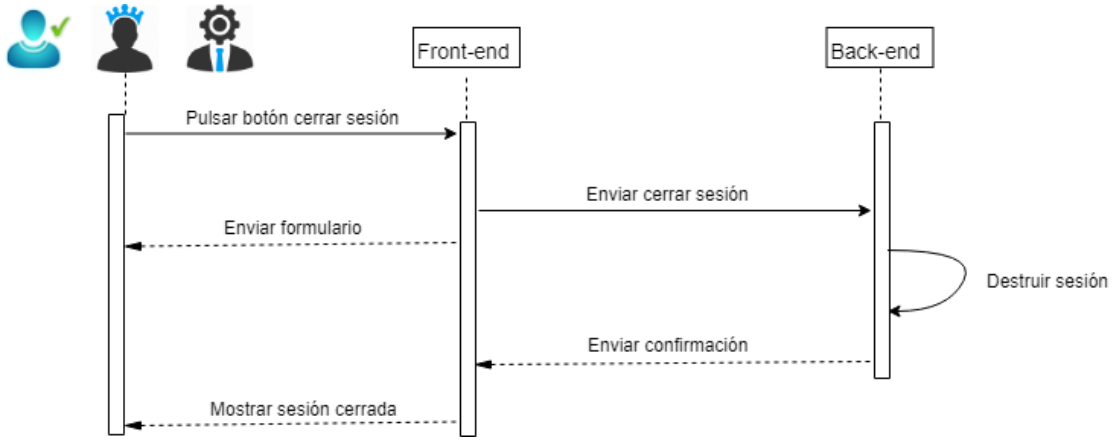
- **Registrar usuario:**



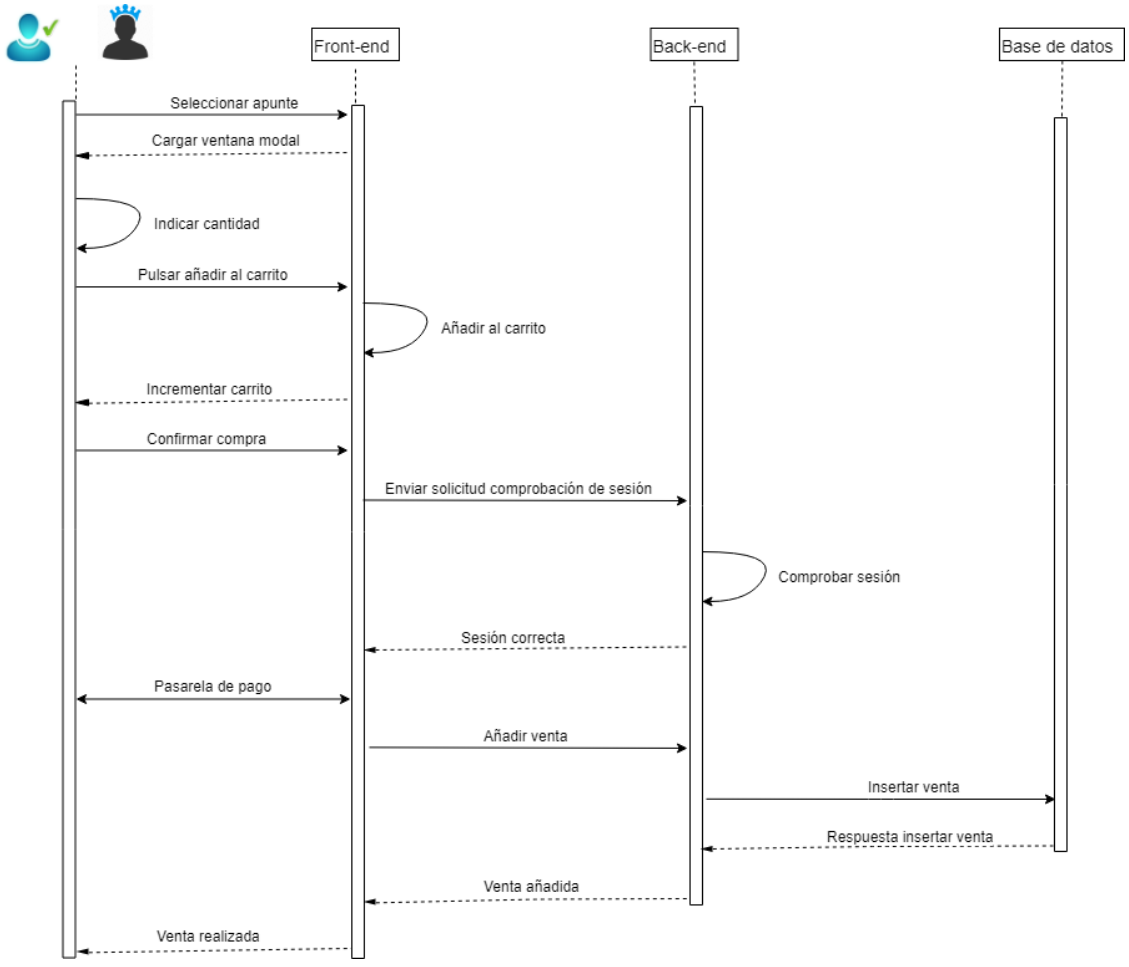
-**Iniciar sesión:**



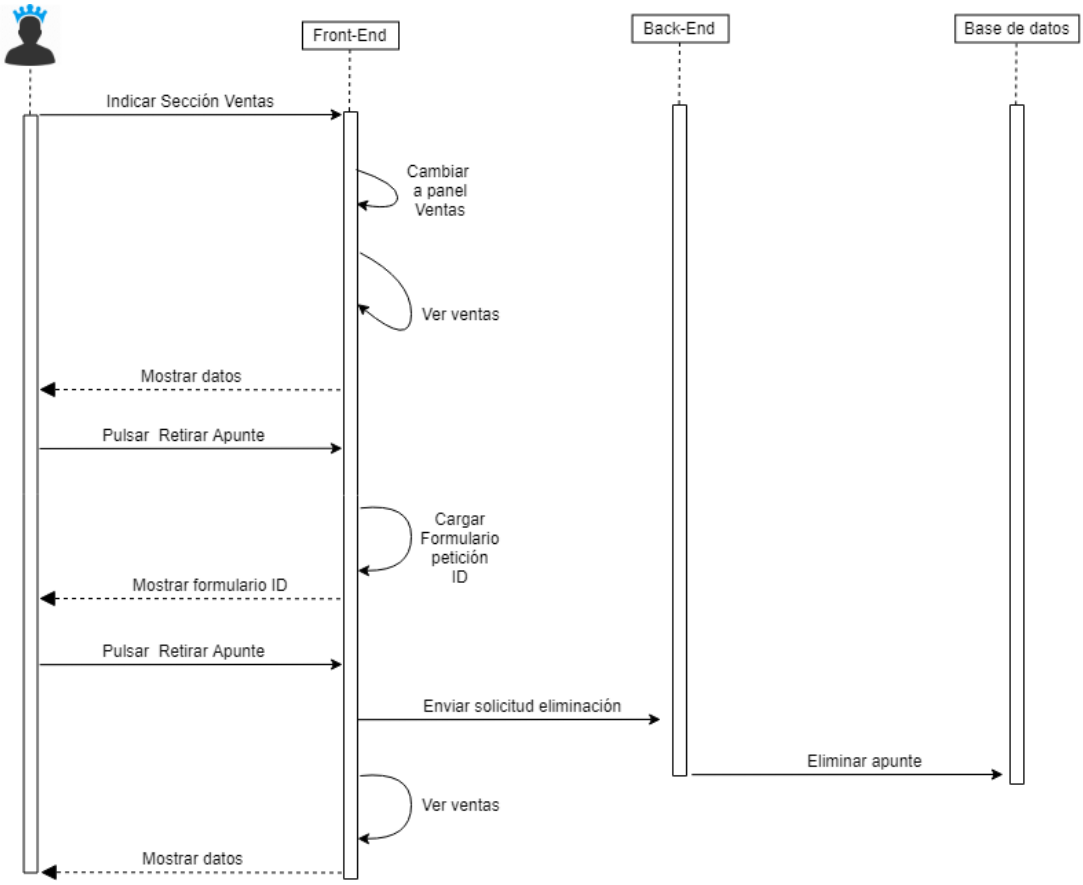
- Cerrar sesión:



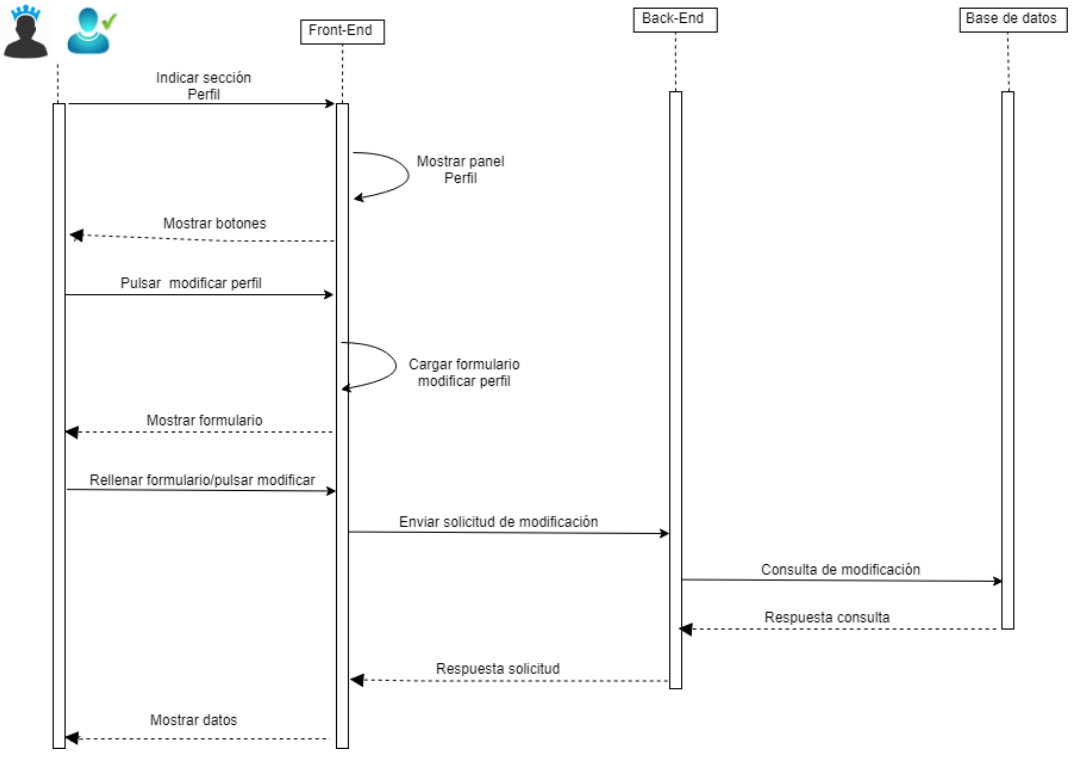
-Añadir al carro, sesión iniciada:



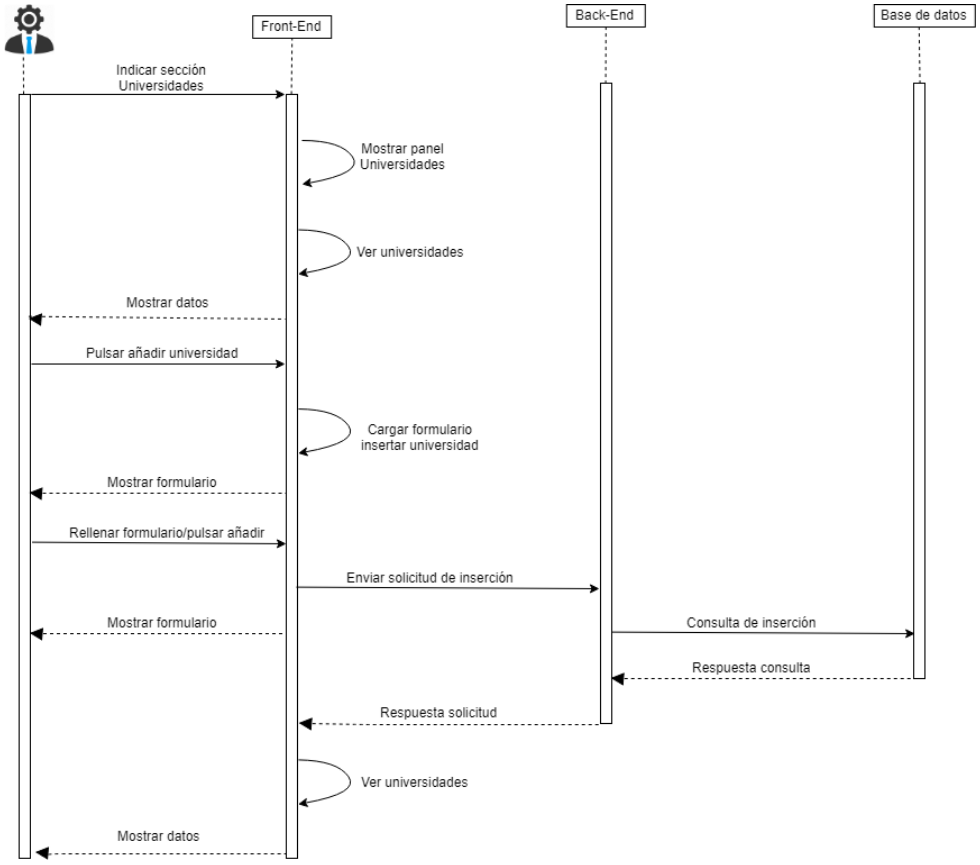
- **Retirar apunte:**



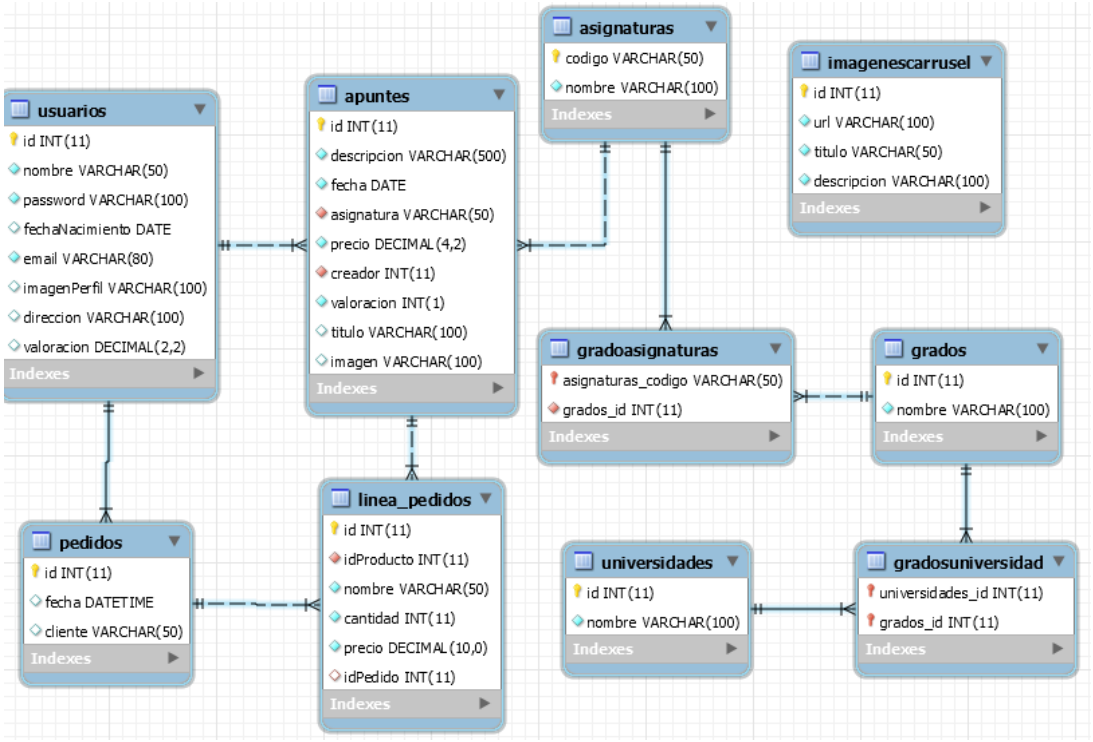
- **Modificar perfil:**



- **Añadir universidad:**



2.2.4 Modelo entidad-relación



2.3 Manual de usuario

En este aparatado haremos, desde los puntos de vista de los diferentes actores, un recorrido por todas las vistas que conforman la aplicación de Colmenial.

Es importante destacar que la aplicación ha sido diseñada de manera que se ajuste visualmente a los diferentes dispositivos desde los cuales los usuarios pueden acceder a ella.

También destacar que se trata de una aplicación en proceso de creación y algunas partes de la misma deben ser pulidas y mejoradas posteriormente a la entrega de este proyecto.

2.3.1 Vista principal

Al acceder a nuestro dominio nos encontramos con esta vista principal e inicial, a la que diferenciamos en dos partes o secciones:

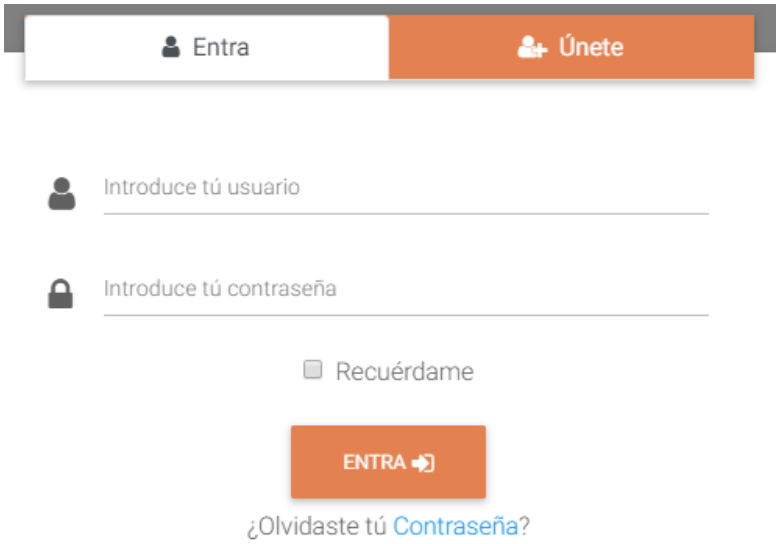
La parte superior, que consta de un carrusel de imágenes una barra superior con el logo de la aplicación y un menú con distintas funcionalidades.

Principalmente estas funcionalidades son las de identificación e interacción con la lista de pedidos, así como un identificador de que nos encontramos en la página de inicio.

3. **Identificación:**
   Podemos acceder a esta opción haciendo click en su referencia en el menú superior, de este modo podremos ver cómo surge una ventana modal en la cual podremos realizar las labores de inicio de sesión o de registro.

   Inicio de sesión:
   Por defecto es la opción que será visible al abrir esta ventana modal, nos solicita un nombre de usuario, aunque también podríamos introducir una dirección email, y una contraseña.

   Podemos marcar la opción de mantener iniciada la sesión, aunque esta es una de las funcionalidades que están pendientes de ser pulidas y mejoradas, al igual que la opción de reestablecer la contraseña que no está implementada.



Si alguno de los datos ofrecidos (contraseña y usuario/email), no coincide con lo almacenado en el sistema, este nos notificará de ello y nos permitirá rectificar el posible error.

Una vez iniciada sesión surgirá desde el lado inferior un mensaje que nos informará de la utilización de cookies por parte de la aplicación y en el menú superior podremos apreciar el cambio de "Identifícate" a el nombre de nuestro usuario, además se añadirá la opción de cerrar sesión.

Registro:

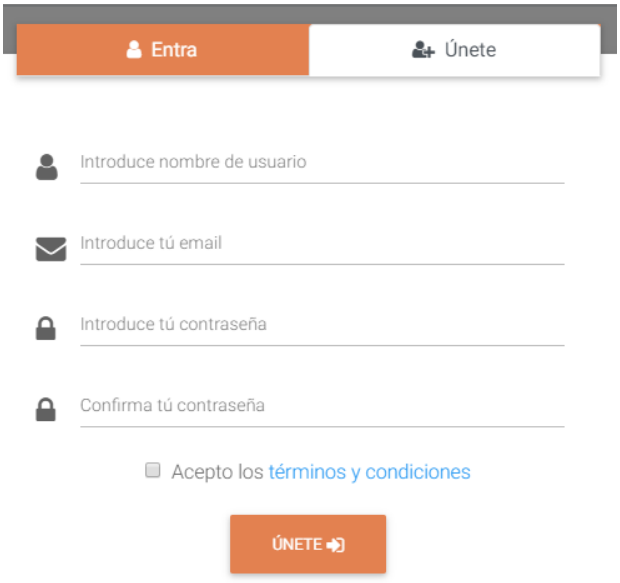
Para acceder a la pestaña de registro debemos hacer click en la referencia "Únete", de esa forma podremos observar que el formulario de acceso cambia y solicita una serie de requisitos.

Estos requisitos no conforman la totalidad de la información recogida por nuestro sistema, pero si la información suficiente necesaria para el funcionamiento en este.

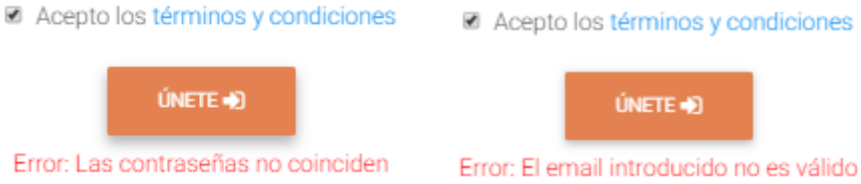
Esta información es el nombre de usuario deseado, una dirección de correo electrónico o email y una contraseña que deberemos introducir dos veces.

Además, para llevar a cabo el registro deberemos aceptar una serie de términos y condiciones, actualmente inexistentes y ficticios, que implementaremos próximamente y en los cuales solicitaremos al usuario permiso para almacenar y tratar la información ofrecida por el.

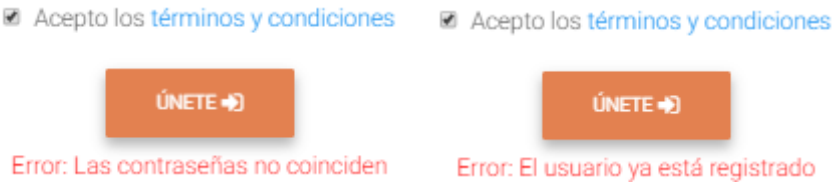
Una vez introducidos los datos requeridos el usuario deberá pulsar el botón "Únete".



Si el email introducido no tiene un formato válido (por ejemplo ejemplo@ejemplo.com), el sistema nos notificará el error, al igual que si las contraseñas introducidas no coinciden entre sí.



Después el sistema comprobará que el nombre de usuario y email introducido no se encuentran a alojados en el sistema, si esta comprobación recibe una respuesta negativa, el sistema notificará al usuario este error.



En caso de que la respuesta a esta validación sea positiva, el sistema almacenará los datos en nuestra base de datos e iniciará la sesión del usuario.

4. **Lista de pedidos:**

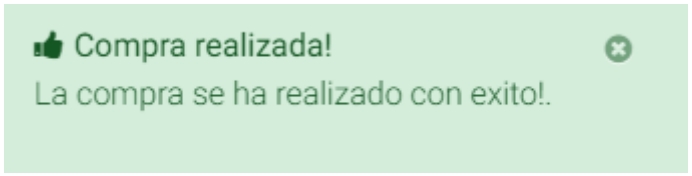
Es la otra funcionalidad de nuestro menú superior, aquí podremos observar los productos que hemos añadido a nuestro carrito, así como modificar la cantidad de estos y eliminarla por completo o llevar a cabo la compra.

Si modificamos la cantidad de producto, podremos ver como el precio total, tanto del producto como de la venta se ve incrementado.

Para eliminar un producto proponemos dos opciones, podemos arrastrar la imagen del apunte hacia la papelera situada en la parte inferior, o podemos hacer click sobre el logo de la papelera que encontramos en la parte derecha, esto es debido a la adaptabilidad de la aplicación, ya que no supimos implementar la funcionalidad drag and drop en dispositivos móviles

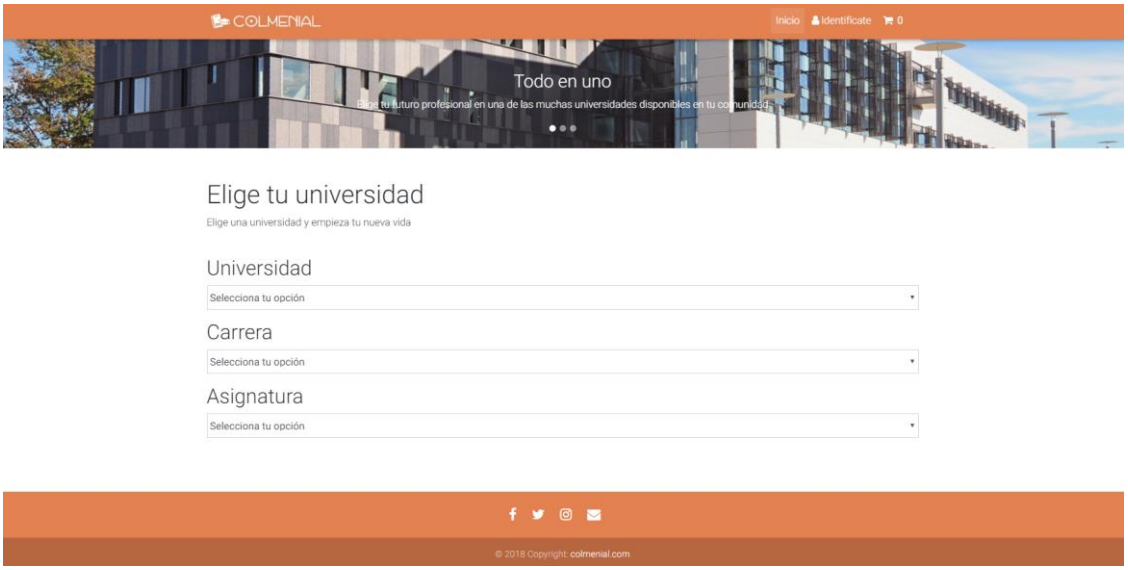
Por último, en la parte inferior encontramos el botón "Realizar compra", el cual hará que el sistema valide la sesión del usuario, mostrando un error si el usuario no ha iniciado sesión previamente.

Si la validación ofrece un resultado no erróneo el usuario accederá a una plataforma de pago actualmente no implementada y una vez finalizado este proceso el sistema mostrará un mensaje confirmando que la compra se ha realizado con éxito.



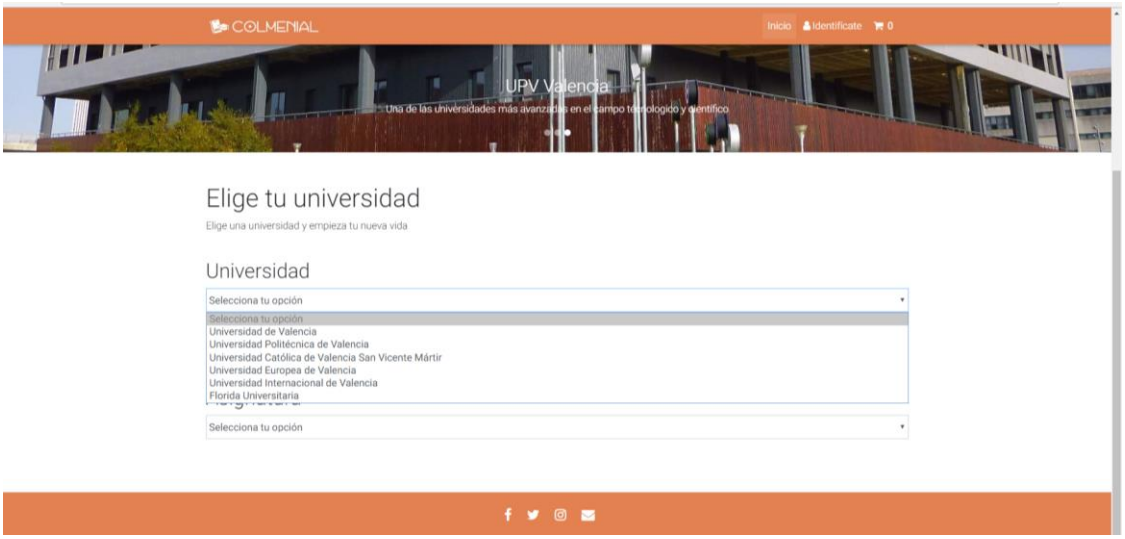
La parte inferior es aquella en la cual se realizarán las labores de selección y visionado de los productos.

También en este apartado encontramos los links de acceso a nuestras redes sociales, así como un link de acceso a nuestra dirección de correo electrónico corporativo, gracias al cual los usuarios de nuestra aplicación pueden entrar en contacto con nosotros.



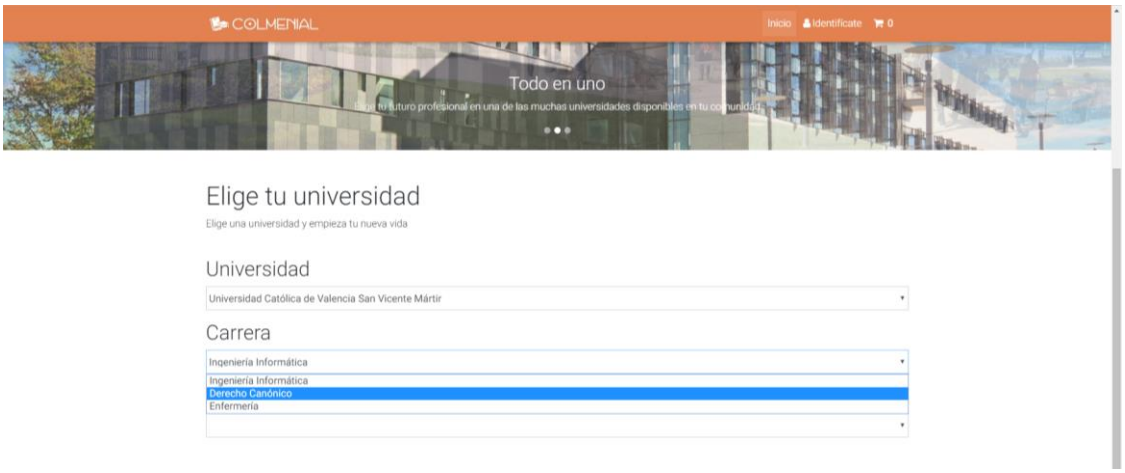
En primer lugar, debemos seleccionar la universidad en la cual deseamos adquirir un apunte. Para ello hacemos click en el desplegable y seleccionamos la universidad.

Actualmente nuestras universidades se limitan a un radio local (Valencia), pero la idea en Colmenial era la de ampliar este radio y acoger universidades de otras localidades.



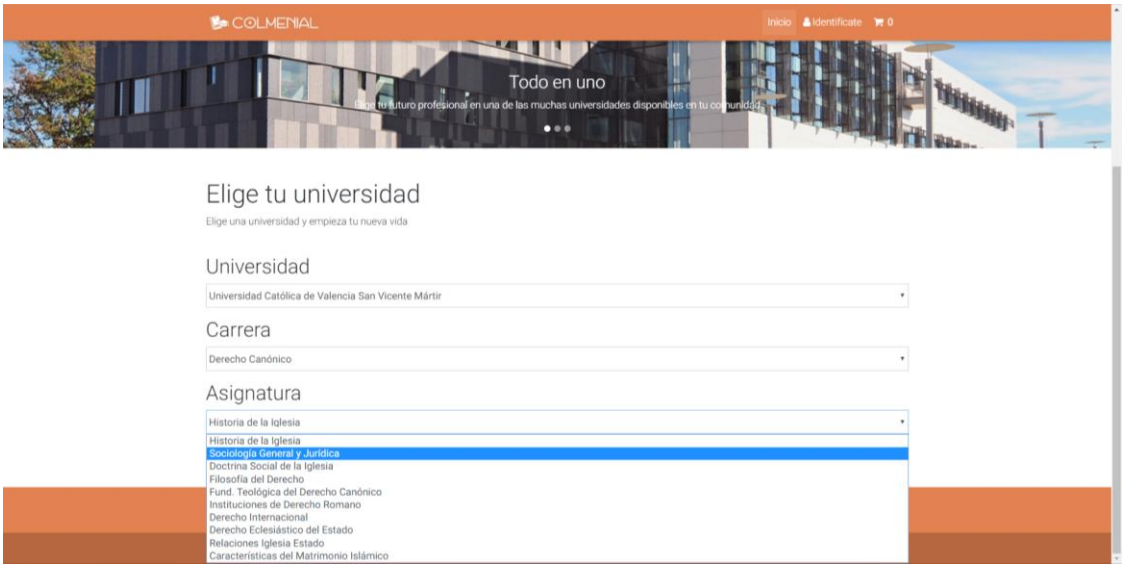
Una vez seleccionada nuestra universidad podremos observar como en el desplegable que hace referencia a las carreras, encontramos grados cursados en esta.

Si hacemos click en este desplegable podremos seleccionar el grado del cual queremos obtener un apunte.

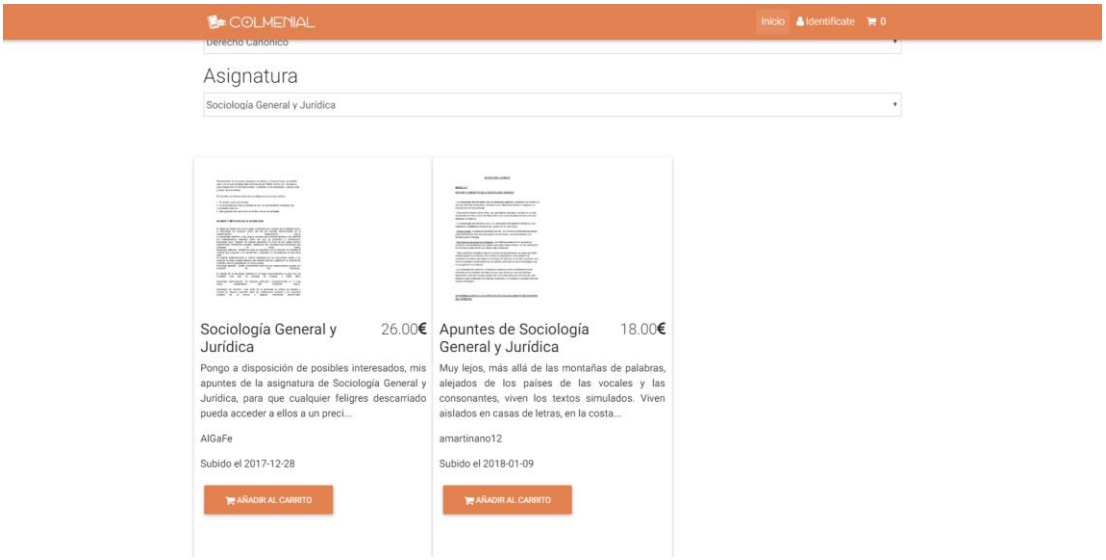


Una vez seleccionada nuestro grado podremos observar como en el desplegable que hace referencia a las asignaturas, encontramos asignaturas cursadas en este grado.

Si hacemos click en este desplegable podremos seleccionar la asignatura de la cual queremos obtener un apunte.

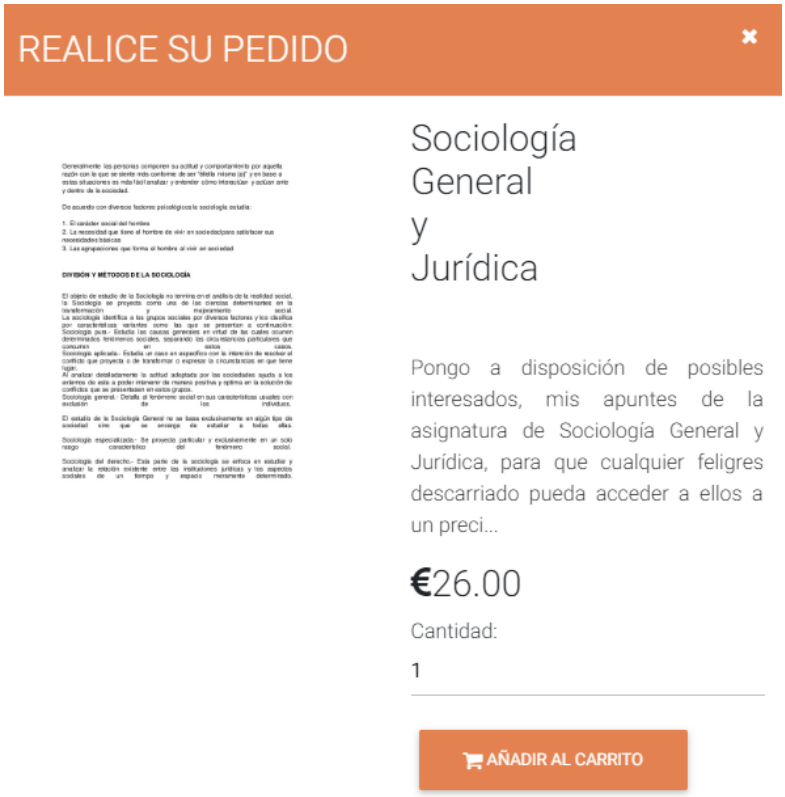


Una vez seleccionada una asignatura veremos como la página aumenta de tamaño, esto es debido a que los apuntes se cargaran inmediatamente después de los desplegables de selección.



De este modo podemos ver información diversa sobre el apunte, el titulo de este, el precio, una descripción, el nombre de usuario del vendedor, la fecha de subida y una imagen del apunte.

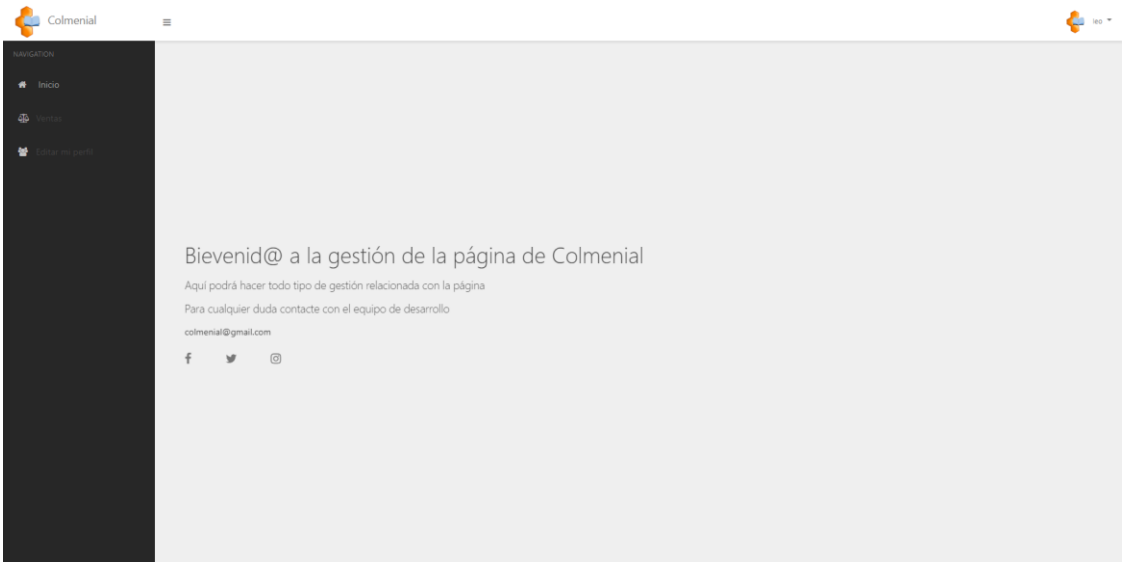
Si pulsamos en el botón "Añadir al carrito" podremos ver más detalladamente la información sobre este producto, así como detallar la cantidad de este que deseamos adquirir.



Una vez pulsado el botón añadir al carrito podremos ver como en la parte superior de la página como el producto ha sido añadido a nuestra lista de pedidos.

2.3.2 Panel de usuario

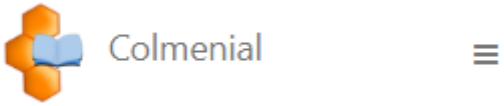
Una vez nos hemos registrado, si hacemos click sobre el nombre de nuestro usuario accederemos a nuestro panel de administración de usuario.



En este panel diferenciamos 3 partes importantes, menú superior, menú izquierdo y panel de visualización.

- **Menú superior**:

  En este menú podemos apreciar dos partes de nuevo, en una de ella se muestra el logo corporativo y al lado un icono de lo que llamaremos menú-hamburguesa, la función de este icono es la de ocultar-o mostrar el menú inferior.



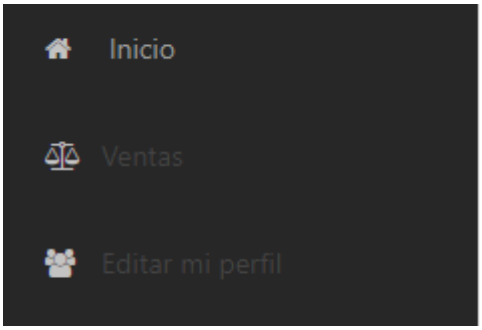
  La otra parte de este menú vuelve a estar compuesta por el logo corporativo y nuestro nombre de usuario, si clickamos sobre este nombre se mostrará un menú desplegable que nos dará la opción de cerrar sesión, si eligiéramos esta, la sesión de usuario actual quedaría eliminada y se nos redirigiría a la Vista Principal.

En el futuro sería interesante añadir más opciones a este menú desplegable.

- **Menú izquierdo**:

  Este menú consta de las opciones de Inicio, Perfil y Ventas, seleccionando una de estas opciones tendremos diferentes opciones con respecto al mantenimiento de la cuenta.

  Es necesario añadir que los roles de usuario premium y usuario normal no están implementados en la actualidad, una vez estén estos implementados el único con la capacidad de realizar las actividades de ventas sería el usuario premium. Además, sería necesario implementar otra opción más a este menú que permitiera visualizar las compras realizadas por el usuario.



- **Panel de visualización**:
  Se encuentra en la parte central de la vista, según la opción señalada en el menú izquierdo nos propondrá unas u otras opciones.

Inicio:

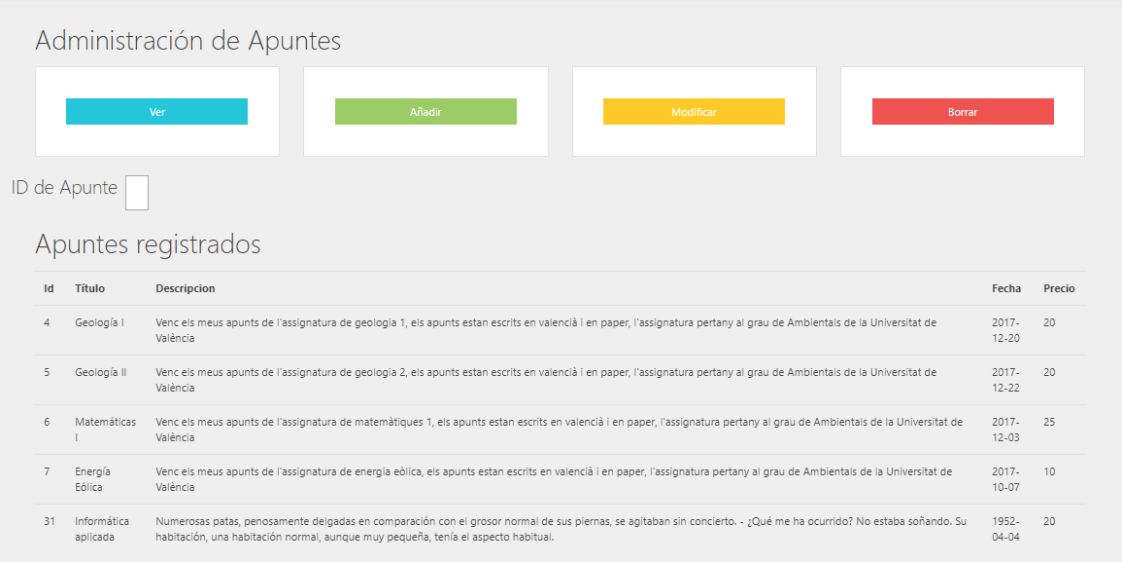
Nos mostrará un mensaje de bienvenida, es la opción marcada por defecto al entrar en el panel de usuario.

Apuntes:

Es importante destacar que, una vez implantados los roles de usuarios correctamente, a esta sección tan solo podrán acceder los usuarios premium.

Mostrará las opciones que el usuario puede realizar sobre sus apuntes publicados.

Además de estas opciones, podemos ver una tabla donde se cargarán los diferentes apuntes puestos a la venta por el usuario y un input donde el usuario deberá introducir la ID del apunte.



Id	Título	Descripcion	Fecha	Precio
4	Geología I	Venc els meus apunts de l'assignatura de geologia 1, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-12-20	20
5	Geología II	Venc els meus apunts de l'assignatura de geologia 2, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-12-22	20
6	Matemáticas I	Venc els meus apunts de l'assignatura de matemàtiques 1, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-12-03	25
7	Energía Eólica	Venc els meus apunts de l'assignatura de energia eòlica, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-10-07	10
31	Informática aplicada	Numerosas patas, penosamente delgadas en comparación con el grosor normal de sus piernas, se agitaban sin concierto. - ¿Qué me ha ocurrido? No estaba soñando. Su habitación, una habitación normal, aunque muy pequeña, tenía el aspecto habitual.	1952-04-04	20

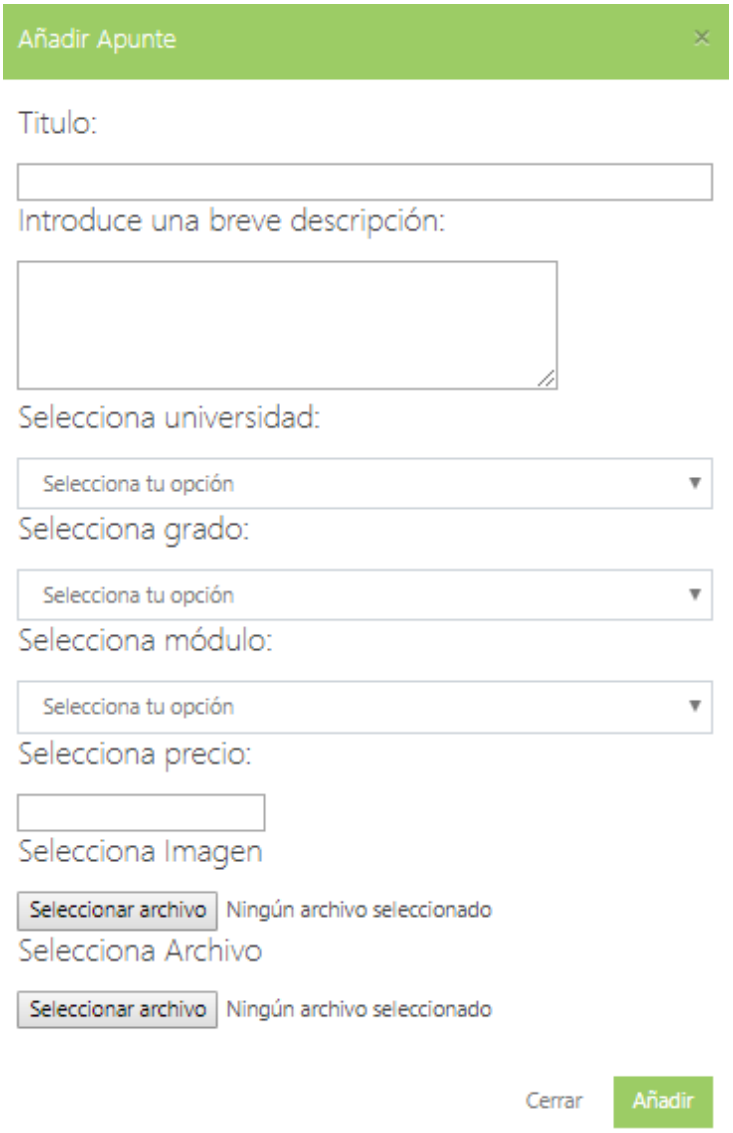
*La función **Ver**, nos permitirá buscar de entre la lista total de los apuntes publicados por el usuario uno en concreto.*

Una vez pulsado este botón la tabla general de los apuntes se verá sustituida por una tabla idéntica pero mostrando exclusivamente el apunte cuya ID sea igual a la facilitada por el usuario.

Además, al lado del input surgirá un botón Ver Todos, pulsando este el usuario volverá a la vista general de todos los apuntes.

*La función **Añadir**, permitirá al usuario poner a la venta un nuevo apunte.*

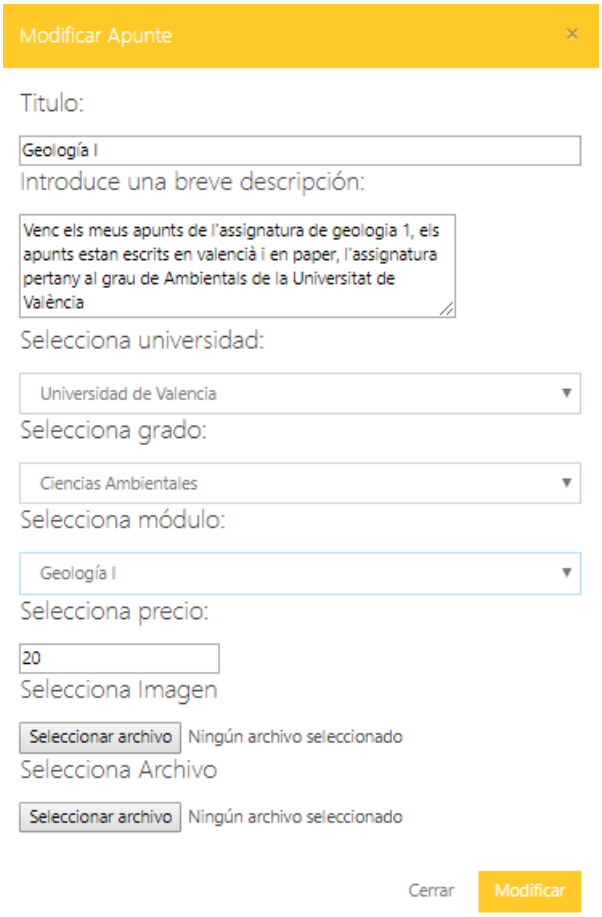
Una vez pulsado este botón surgirá una ventana emergente, en la cual el sistema solicitará al usuario una serie de datos necesarios para la publicación del apunte, el número ID del mismo, así como la fecha de publicación se generarán de forma automática, por lo tanto, no son datos solicitados.



Una vez pulsado el botón añadir, el sistema recargará la vista donde podemos ver la lista total de apuntes, en la cual podemos apreciar el nuevo apunte publicado.

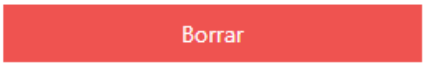
*La función **Modificar** permite al usuario editar la información que el sistema tiene acerca de un apunte publicado por el.*

Una vez pulsado este botón surgirá una ventana emergente, con los mismos campos que en la anterior, en la cual el sistema ya habrá precargado los datos que este tiene almacenados sobre el apunte cuya ID coincide con la facilitada, el usuario podrá modificar estos datos.



Una vez pulsado el botón modificar, el sistema recargará la vista donde podemos ver la lista total de apuntes, en la cual podemos apreciar los cambios realizados sobre el apunte señalado.

*La función **Borrar** se encargará de eliminar el apunte cuya ID coincida con la facilitada por el usuario.*

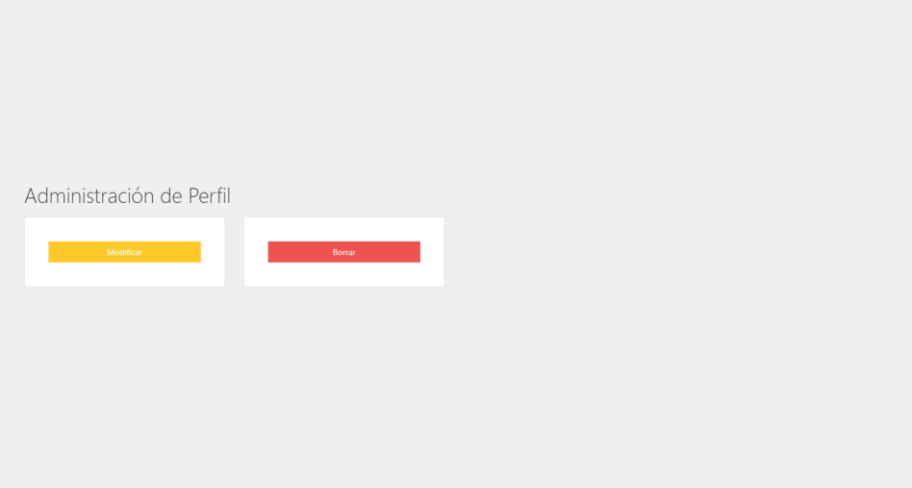


Una vez pulsado este botón, el sistema recargará la vista donde podemos ver la lista total de apuntes, en la cual el usuario podrá apreciar la inexistencia del apunte.

- ***Mi perfil:***

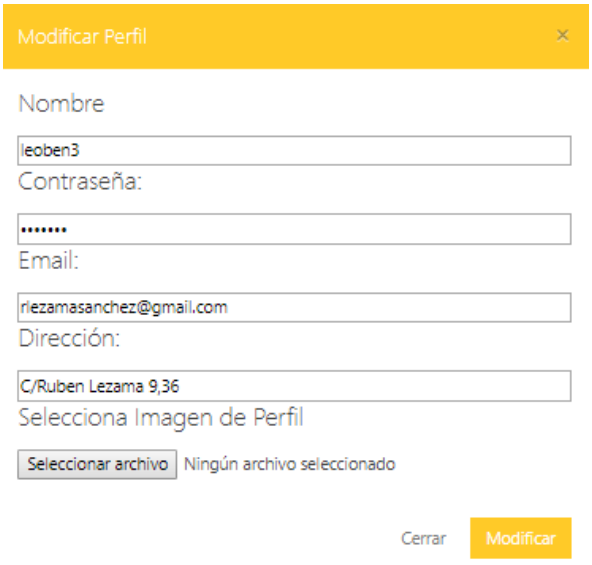
Mostrará las opciones que el usuario puede realizar sobre su perfil.

A esta sección pueden acceder tanto el usuario registrado como el premium, ya que incluyen exclusivamente labores de administración de los datos de usuario recogidos en el sistema.



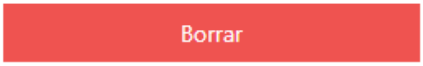
*La función **Modificar** permite al usuario editar su información personal que el sistema mantiene almacenada.*

Una vez pulsado este botón surgirá una ventana emergente, con los mismos campos que el sistema permite editar, en la cual el sistema ya habrá precargado los datos que este tiene almacenados, la ID del usuario es un valor autogenerado y por lo tanto el usuario no puede editarlo.



Una vez pulsado el botón modificar, el sistema recargará la vista general de administración del perfil.

*La función **Borrar** se eliminará el perfil del usuario.*

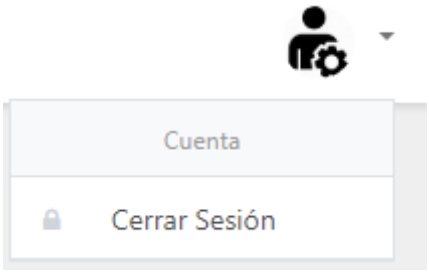


Una vez pulsado el sistema cerrará la sesión de usuario, eliminará el perfil y redirigirá a este a la página principal de ***Colmenial****.*

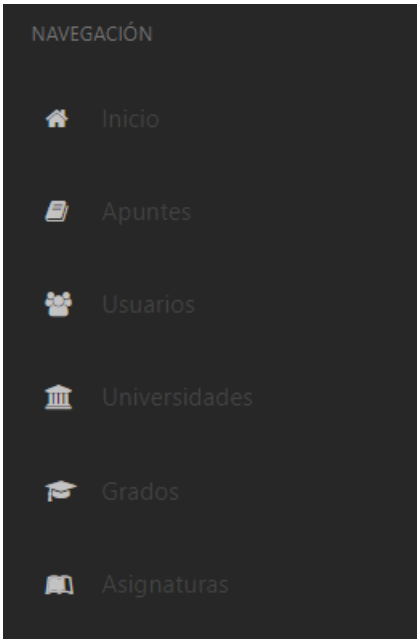
2.3.3 Panel de administrador

De igual modo que en el caso del panel de usuario, el administrador podrá acceder su respectivo panel.

La diferencia principal en el panel superior con respecto a este reside en la ausencia del nombre del usuario y en que el logo corporativo derecho será sustituido por un icono representativo del administrador, la funcionalidad de este sigue siendo la misma, la de abrir el menú donde se incluye el cierre de sesión.



En el menú izquierdo podremos acceder a las secciones donde podremos llevar a cabo las labores de administración de Universidades, Grados, Asignaturas, Apuntes, y Usuarios.



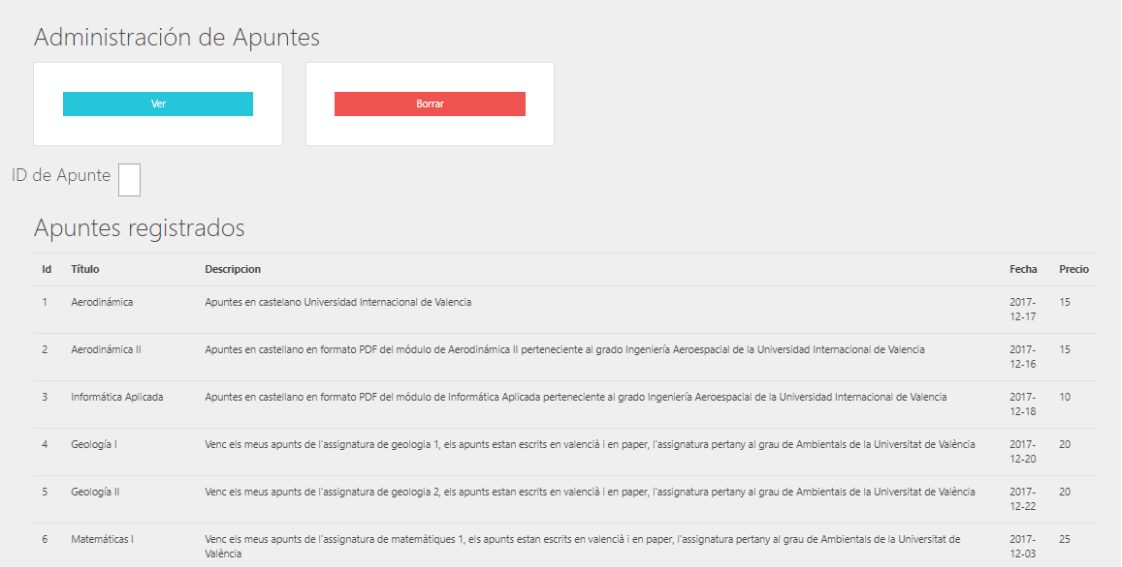
- **Inicio**:

Presentamos en esta sección, al igual que en caso del panel de usuario un mensaje de bienvenida, esta sección, también será la opción cargada por defecto al cargar el panel.

- **Apuntes:**

En esta sección se permite al administrador llevar a cabo labores de moderación, sobre apuntes inadecuados, con contenido ofensivo etc...

El panel está formado por una tabla donde aparecerán todos los apuntes publicados por todos los usuarios, así como el ya utilizado en el Panel de Usuario input que solicita la ID del apunte.



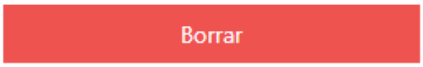
Id	Título	Descripcion	Fecha	Precio
1	Aerodinámica	Apuntes en castelano Universidad Internacional de Valencia	2017-12-17	15
2	Aerodinámica II	Apuntes en castellano en formato PDF del módulo de Aerodinámica II perteneciente al grado Ingeniería Aeroespacial de la Universidad Internacional de Valencia	2017-12-16	15
3	Informática Aplicada	Apuntes en castellano en formato PDF del módulo de Informática Aplicada perteneciente al grado Ingeniería Aeroespacial de la Universidad Internacional de Valencia	2017-12-18	10
4	Geología I	Venc els meus apunts de l'assignatura de geologia 1, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-12-20	20
5	Geología II	Venc els meus apunts de l'assignatura de geologia 2, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-12-22	20
6	Matemáticas I	Venc els meus apunts de l'assignatura de matemàtiques 1, els apunts estan escrits en valencià i en paper, l'assignatura pertany al grau de Ambientals de la Universitat de València	2017-12-03	25

*La función **Ver**, nos permitirá al administrador buscar de entre la lista total de los apuntes publicados por los usuarios uno en concreto.*

Una vez pulsado este botón la tabla general de los apuntes se verá sustituida por una tabla idéntica, pero mostrando exclusivamente el apunte cuya ID sea igual a la facilitada por el administrador.

Además, al lado del input surgirá un botón Ver Todos, pulsando este el administrador volverá a la vista general de todos los apuntes.

*La función **Borrar** se encargará de eliminar el apunte cuya ID coincida con la facilitada por el administrador.*



Una vez pulsado este botón, el sistema recargará la vista donde podemos ver la lista total de apuntes, en la cual el administrador podrá apreciar la inexistencia del apunte.

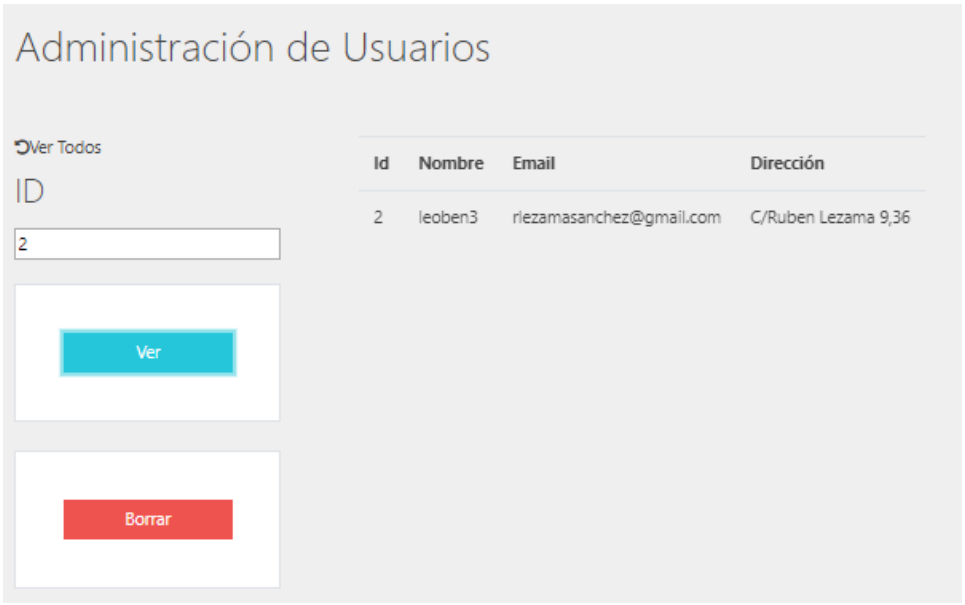
- **Usuarios:**

Al igual que en la sección de ventas, el usuario administrador solamente tendrá poder de moderación antes posibles comportamientos inadecuado, realizados de forma reiterativa por parte de los usuarios.

El panel está formado por una tabla donde aparecerán todos los usuarios que forman parte de la comunidad, así como un input que nos solicita la ID del usuario concreto.

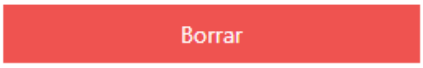
*La función **Ver**, nos permitirá al administrador buscar de entre la lista total de los usuarios uno en concreto.*

Una vez pulsado este botón la tabla general de los usuarios se verá sustituida por una tabla idéntica pero mostrando exclusivamente el usuario cuya ID sea igual a la facilitada por el administrador.



Además, encima del input surgirá un botón Ver Todos, pulsando este el administrador volverá a la vista general de todos los usuarios.

*La función **Borrar** se encargará de eliminar el usuario cuya ID coincida con la facilitada por el administrador, previamente a la eliminación de un usuario, sería conveniente realizar un proceso de contacto con este, realizando avisos con el fin de que el usuario abandone la actitud inadecuada.*



Una vez pulsado este botón, el sistema recargará la vista donde podemos ver la lista total de usuarios, en la cual el administrador podrá apreciar la inexistencia de este.

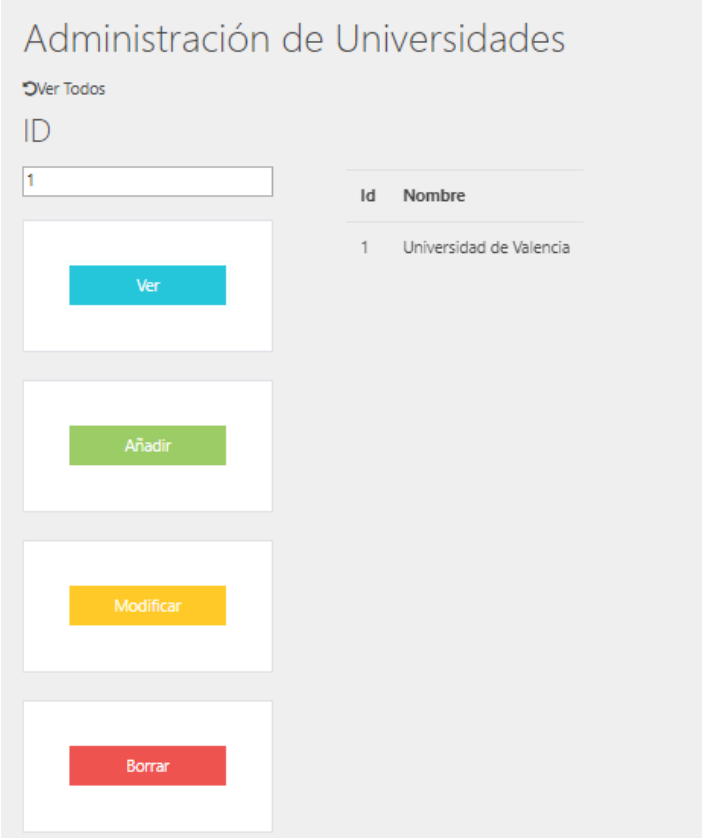
- ***Universidades:***

  Mostrará las opciones que el administrador puede realizar sobre las universidades figurantes en el sistema.

Además de estas opciones, podemos ver una tabla donde se cargarán los diferentes universidades existentes en el sistema y un input donde el administrador deberá introducir la ID del universidad.

*La función **Ver**, nos permitirá buscar de entre la lista total de las universidades almacenadas en el sistema una en concreto.*

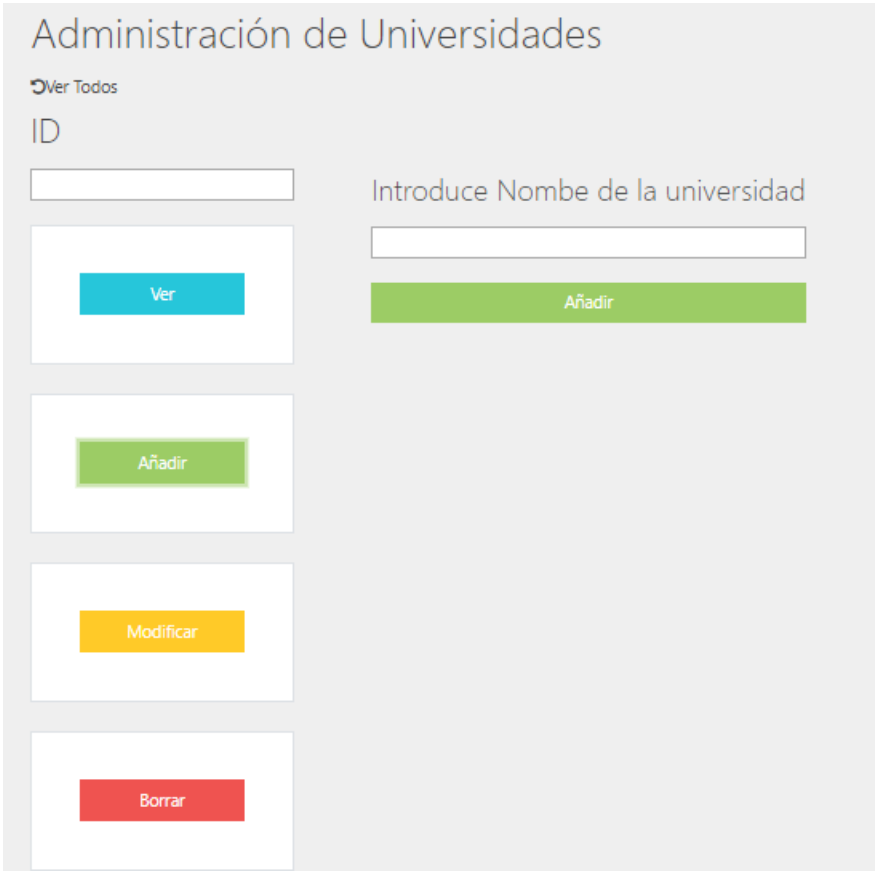
Una vez pulsado este botón la tabla general de universidades se verá sustituida por una tabla idéntica, pero mostrando exclusivamente la universidad cuya ID sea igual a la facilitada por el administrador.



Además, encima del input surgirá un botón Ver Todos, pulsando este el administrador volverá a la vista general de universidades.

*La función **Añadir**, permitirá al administrador poner añadir al sistema una nueva universidad.*

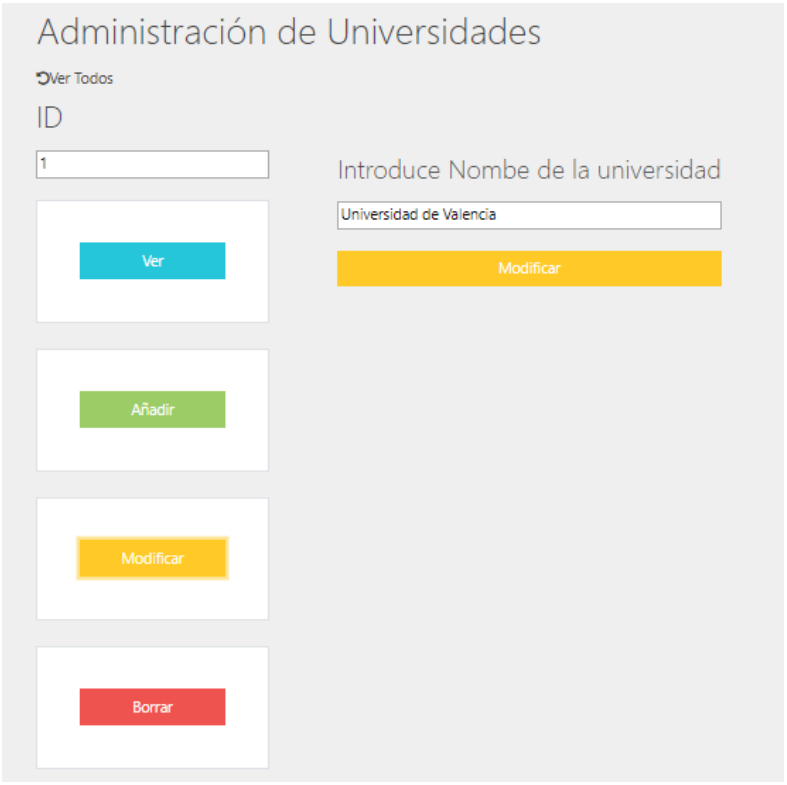
Una vez pulsado este botón la tabla general de universidades se verá sustituida por un formulario que solicita el nombre de la universidad que el administrador desea introducir, la ID es un valor autogenerado y por lo tanto no se solicita.



Una vez pulsado el botón añadir, el sistema recargará la vista donde podemos ver la lista total de universidades, en la cual podemos apreciar la universidad introducida.

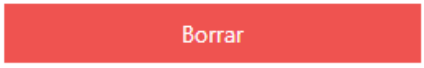
*La función **Modificar** permite al administrador editar el nombre que el sistema tiene almacenado sobre una universidad.*

Una vez pulsado este botón la tabla general de universidades se verá sustituida por un formulario similar al del caso de añadir universidad, pero donde se cargará el nombre que el sistema tiene almacenado sobre la universidad cuya ID coincida con la facilitada por el administrador.



Una vez pulsado el botón modificar, el sistema recargará la vista donde podemos ver la lista total de universidades, en la cual podemos apreciar los cambios realizados sobre la universidad señalada.

*La función **Borrar** se encargará de eliminar la universidad cuya ID coincida con la facilitada por el administrador.*



Una vez pulsado este botón, el sistema recargará la vista donde podemos ver la lista total de universidades, en la cual el administrador podrá apreciar la inexistencia de esta.

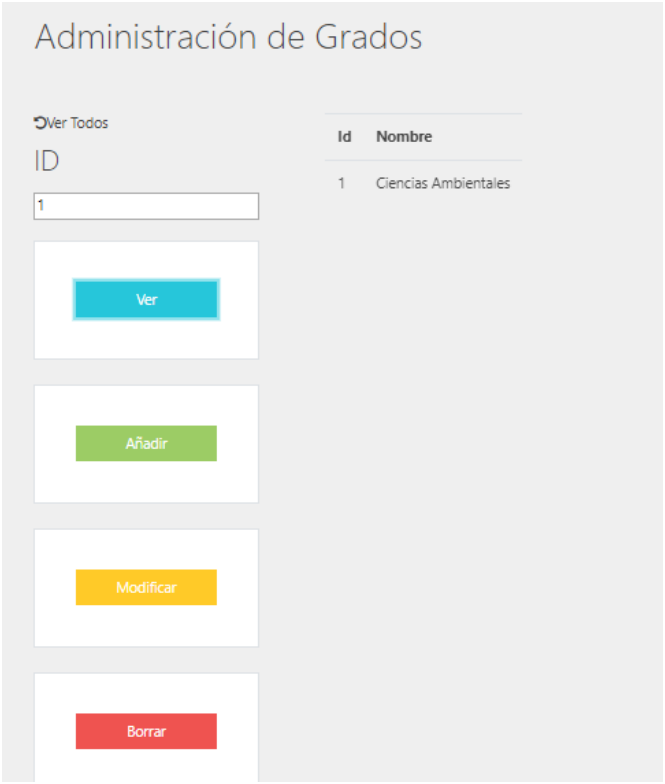
- ***Grados:***

Mostrará las opciones que el administrador puede realizar sobre los grados figurantes en el sistema.

Además de estas opciones, podemos ver una tabla donde se cargarán los diferentes grados existentes en el sistema y un input donde el administrador deberá introducir la ID del grado.

*La función **Ver**, nos permitirá buscar de entre la lista total de los grados almacenados en el sistema uno en concreto.*

Una vez pulsado este botón la tabla general de grados será sustituida por una tabla idéntica pero mostrando exclusivamente el grado cuya ID sea igual a la facilitada por el administrador.



Además, encima del input surgirá un botón Ver Todos, pulsando este el administrador volverá a la vista general de universidades.

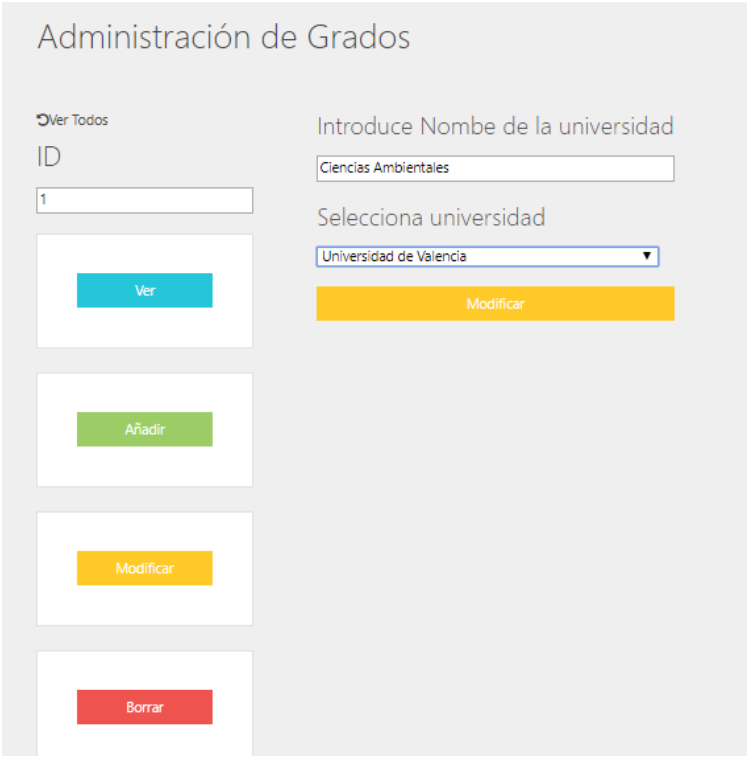
*La función **Añadir**, permitirá al administrador poner añadir al sistema una nueva universidad.*

Una vez pulsado este botón la tabla general de grados será sustituida por un formulario dónde el sistema solicitará al administrador el nombre del grado que desea insertar y la universidad en la cual este se imparte, el número ID se generarán de forma automática.

Una vez pulsado el botón añadir, el sistema recargará la vista donde podemos ver la lista total de grados, en la cual podemos apreciar el grado introducido.

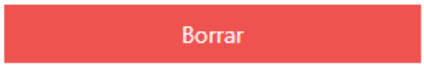
*La función **Modificar** permite al administrador editar la información que el sistema tiene almacenado de un grado.*

Una vez pulsado este botón la tabla general de grados será sustituida por un formulario similar a la explicada anteriormente, en él se cargarán el nombre y la universidad relacionada que el sistema tiene almacenado sobre el grado cuya ID coincida con la facilitada por el administrador.



Una vez pulsado el botón modificar, el sistema recargará la vista donde podemos ver la lista total de grados, en la cual podemos apreciar los cambios realizados sobre el grado señalado.

*La función **Borrar** se encargará de eliminar el grado cuya ID coincida con la facilitada por el administrador.*



Una vez pulsado este botón, el sistema recargará la vista donde podemos ver la lista total de grados, en la cual el administrador podrá apreciar la inexistencia de este.

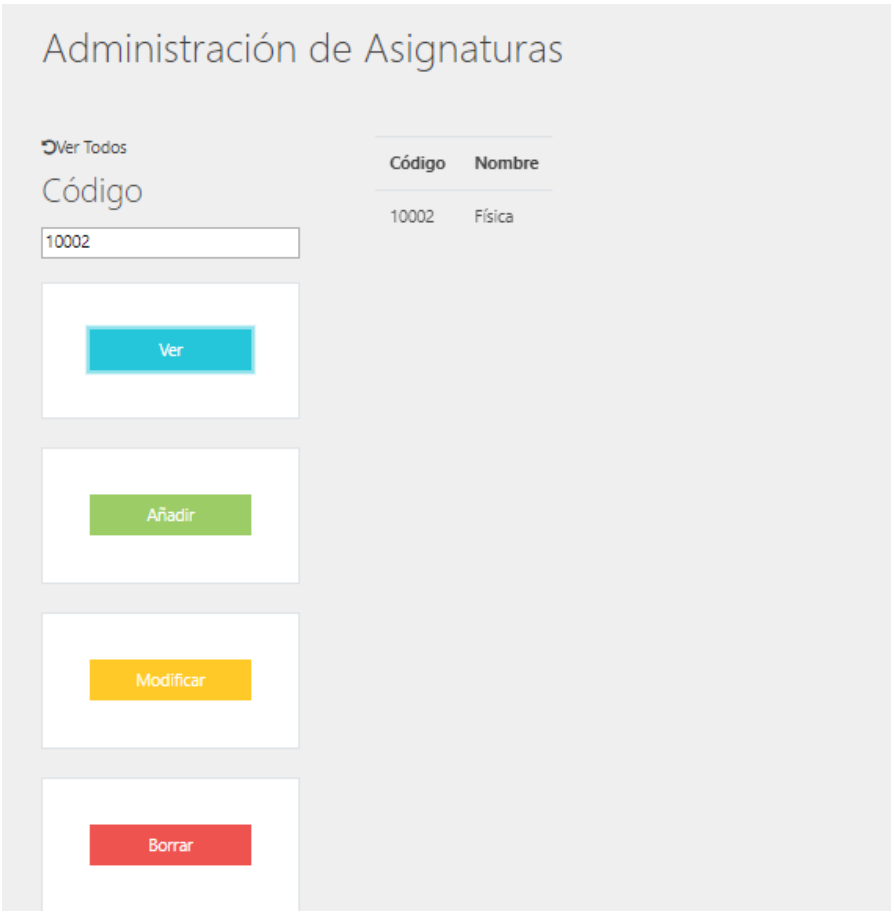
- **Asignatura:**

Mostrará las opciones que el administrador puede realizar sobre las asignaturas figurantes en el sistema.

Además de estas opciones, podemos ver una tabla donde se cargarán las diferentes asignaturas existentes en el sistema y un input donde el administrador deberá introducir el código de la asignatura.

*La función **Ver**, nos permitirá buscar de entre la lista total de las asignaturas almacenadas en el sistema una en concreto.*

Una vez pulsado este botón la tabla general de asignaturas será sustituida por una tabla idéntica, pero mostrando exclusivamente la asignatura cuyo código sea igual al facilitado por el administrador.



Además, encima del input surgirá un botón Ver Todos, pulsando este el administrador volverá a la vista general de asignaturas.

*La función **Añadir**, permitirá al administrador poner añadir al sistema una nueva asignatura.*

Una vez pulsado este botón surgirá una ventana emergente, en la cual el sistema solicitará al administrador el nombre de la asignatura que desea insertar, el código de esta, la universidad imparte y el grado donde se imparte esta asignatura, la lista de grados se generará dinámicamente en función de la universidad seleccionada.

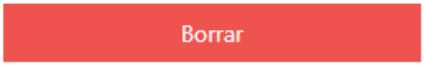
Una vez pulsado el botón añadir, el sistema recargará la vista donde podemos ver la lista total de asignaturas, entre las cuales podemos apreciar la introducida.

*La función **Modificar** permite al administrador editar la información que el sistema tiene almacenado de una asignatura.*

Una vez pulsado este surgirá una ventana modal similar a la explicada anteriormente, en ella se cargarán el nombre y la universidad y grado relacionados que el sistema tiene almacenado sobre la asignatura cuyo código coincida con la facilitada por el administrador.

Una vez pulsado el botón modificar, el sistema recargará la vista donde podemos ver la lista total de asignaturas, en la cual podemos apreciar los cambios realizados sobre la asignatura señalada.

*La función **Borrar** se encargará de eliminar la asignatura cuyo código coincida con el facilitada por el administrador.*



Una vez pulsado este botón, el sistema recargará la vista donde podemos ver la lista total de asignaturas, en la cual el administrador podrá apreciar la inexistencia de esta.

2.4 Herramientas y tecnologías empleadas

2.4.1 Herramientas de desarrollo

Son las herramientas externas que hemos empleado realizar las tareas de programación, depuración, test etc.

Diferenciamos estas herramientas según la utilidad que les hemos dado:

- **Editor de código fuente:**
  Necesario para llevar a cabo parte de la labor de programación, ya que es la aplicación donde he escrito mi código original.

  Antes de realizar la migración a tecnologías JSP, los cambios realizados en esta migración de la parte front-end son mínimos y es por eso que contemplo esta parte desarrollada sobre un editor de código fuente.

  *Utilicé **Visual Studio Code**, desarrollado por Microsoft, ya permite la instalación de múltiples pluggins que facilitan la labor de programación.*

- **Entorno de desarrollo integrado (IDE):**

  Es una aplicación proporciona servicios integrales para facilitar el desarrollo de software.

  *Utilicé **IntelliJ IDEA**, desarrollado por JetBrains, para realizar el proceso de desarrollo de la parte back-end.*

  *Después de desarrollar la parte back-end en JSP decidí utilizar el **Netbeans**, desarrollado por. Apache Software Foundation, para realizar los cambios pertinentes en el apartado front-end.*

- **Sistema gestor de bases de datos:**
  Es el sistema que nos permitirá administrar nuestras bases de datos.
  *En nuestro caso, como ya he citado en el anterior punto, empleamos un servidor de datos **MySQL/MariaDB** y el gestor empleado ha sido **HeidiSQL**.*

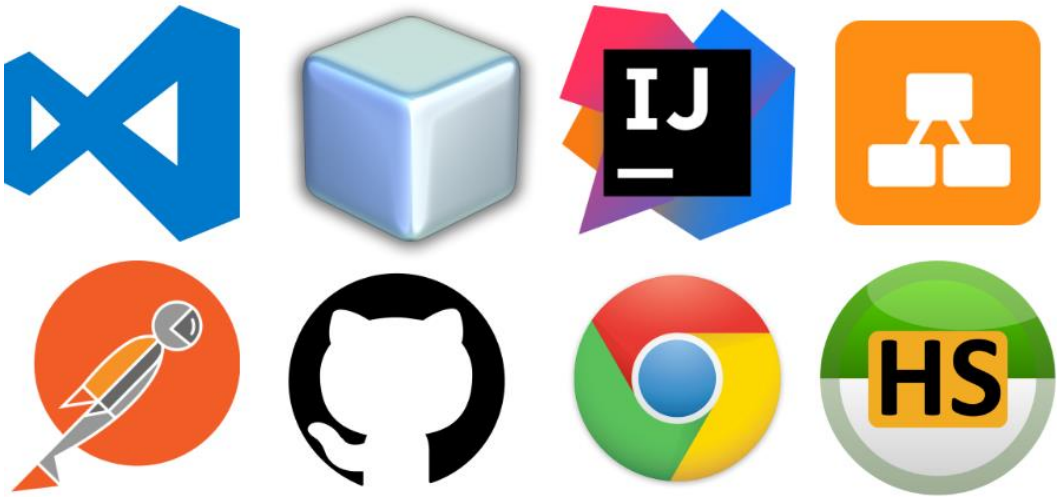
- **Control de versiones:**
  *Permite la gestión de los diversos cambios realizados sobre el proyecto, como sistema de control de versiones hemos empleado Git y como plataforma para este hemos empleado **GitHub**.*

- **Navegación y testeo:**
  Plataformas que hemos empleado para controlar el correcto funcionamiento de nuestra aplicación.

  *En el caso de navegación he utilizado el navegador **Google Chrome** y sus herramientas de inspección de código.*

  *Para realizar un testeo de la API REST creada y comprobar su correcto funcionamiento he utilizado la aplicación **Postman**.*

- **Diseño/creación de diagramas:**
  *Para la creación de los diagramas mostrados anteriormente he utilizado **draw.io**, una aplicación para diagramación gratuita de Google Drive(TM).*



2.4.2 Tecnologías

Son las tecnologías (lenguajes de programación, librerías, frameworks y técnicas de desarrollo), que hemos utilizado para el desarrollo del proyecto.

Diferenciamos estas en función de si pertenecen a la capa de presentación (front-end) o la capa de acceso a datos (back-end).

Tecnologías front-end:

- **JavaScript**: *Lenguaje de programación interpretado que utilizado en la parte de cliente para generar contenido dinámico en páginas web.*
- **HTML5**: *Lenguaje de marcas de hipertexto, que se ha convertido en estándar para la elaboración de páginas web.*
- **CSS3**: *Lenguaje de diseño gráfico que permite la definición y creación de la presentación de un documento estructurado escrito en un lenguaje de marcado.*
- **AJAX**: *Técnica de desarrollo web ejecutada desde el cliente, se utiliza para mantener una conversación asíncrona con el servidor.*
- **SASS**: *Metalenguaje de CSS. Que permite la creación de Scripts que más arde son traducidos a CSS.*
- **Bootstrap**: *Framework utilizado para el diseño de sitios y aplicaciones web. Solo se utiliza en la parte front-end y contiene plantillas de diseño con contenido basado en HTML, CSS y JavaScript.*
- **Jquery**: *Librería de JavaScript que simplifica facilita las tareas de manipulación del DOM.*
- **Font Awesome**: *Librería basada en CSS y LESS que proporciona al desarrollo una gran variedad de iconos que poder implementar en los sitios web.*

Tecnologías back-end:

- **JavaServer Pages (JSP):** *Lenguaje de programación Java destinado a la creación de aplicaciones web dinámicas, para desplegar es necesario la utilización de servidores compatibles con servlets (clases Java que amplían las capacidades de un servidor), en mi caso* ***Apache Tomcat****.*

- **Spring:** *Framework del lenguaje de programación JSP para el desarrollo de aplicaciones.*

- **Spring Boot:** *Tecnología incluida dentro del framework de JSP Spring que simplifica la creación de aplicaciones con Spring.*

- **Java Persistence API (JPA)**: *Framework del lenguaje JSP destinada a no perder las ventajas de la orientación a objetos al interactuar con una base de datos.*

- **Gradle**: *Sistema de automatización de construcción de proyectos.*

- **MySQL**: *Sistema gestor de bases de datos relacional desarrollado bajo licencia GNU GPL por Oracle Corporation, considerada como la base datos de código abierto más popular del mundo y una de las más populares en general junto a Oracle y Microsoft SQL Server.*

- **ModelMapper**: *Librería que permite el mapeo de objetos similares, pero no idénticos.*

3 PROPUESTA DE MEJORA

Hay muchas mejoras que se le podían aplicar a Colmenial, mejoras que o por desconocimiento de tecnologías o por falta de tiempo no se han podido poner en práctica.

- **Limpieza y re-estructuramiento:**

Es la mejora más simple y la considero muy importante. Para mejorar la legibilidad y compresión del código y el proyecto en sí mismo, es necesario que este esté limpio y ordenado, así como la estructura que conforma.

También realizar la implementación de aquellas funcionalidades citadas en puntos anteriores que no hemos podido implementar.

- **Mejora de diseño:**

Especialmente en el caso de los paneles de administración, ya que estos paneles fueron diseñados durante los últimos springs y teníamos menos tiempo para plantear una buena idea para estos que fuera atractiva y funcional.

- **Establecimiento de roles de usuario:**

Actualmente los roles de usuario no están establecidos como tales, no existen dos tipos de usuario, es por eso que en el panel de administración de usuario se pueden realizar labores de usuario premium (mantenimiento de apuntes subidos por un usuario).

- **Mejoras de seguridad:**

Tanto en el front-end se encuentran diferentes errores de seguridad que deberían ser solucionados en un futuro próximo, en el apartado back-en también encontramos errores de seguridad como falta de encriptación de las contraseñas o, a pesar de no ser un error de seguridad en sí mismo, implementar el control de excepciones para posibles usos no previstos de los usuarios.

- ***Establecimiento de términos y condiciones:***

A la hora de que un usuario realice el registro en nuestra página web, deberíamos mantener informado al usuario del uso que se va a dar de su información facilitada, así como obligarle a aceptar estas condiciones en el caso de que quiera usar nuestra aplicación, y de este modo cumplir con la ley de protección de datos.

- ***Redes Sociales:***

Deberíamos retomar el trabajo en el mantenimiento de nuestras redes sociales, actualmente abandonadas, ya que es una buena forma de darnos a conocer y conseguir nuevos usuarios. También deberíamos crear un email corporativo.

- **Implementación total:**

Una mejora posible es la de la implementación de las demás funcionalidades, que como he explicado anteriormente, no se pudieron llevar a cabo por la envergadura de esta el tiempo requerido. Para llevar a cabo esta mejora previamente considero que previamente sería necesaria la migración back-en y front-end.

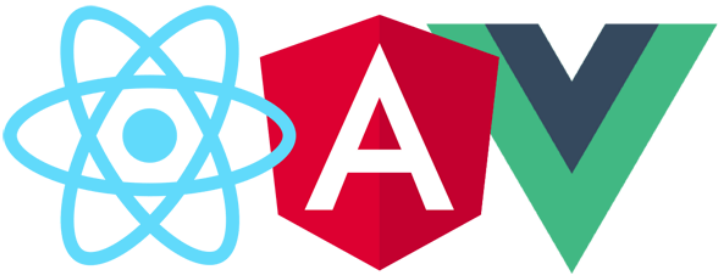
- ***Menú de mantenimiento**:*

  Esta mejora está relacionada con la implantación total del proyecto, actualmente este menú solo contempla la opción de cerrar sesión, una vez realizada la implementación este menú deberá contener más opciones de mantenimiento.

  También sería necesario añadir más paneles de mantenimiento como por ejemplo uno que permita contemplar el historial de las transacciones efectuadas por el usuario, tanto las compras como las ventas, en el caso de tratarse de un usuario premium.

- **Frameworks:**

*Dado el volumen que ocuparía una aplicación como esta con todas las funcionalidades pensadas originalmente, creo que sería necesaria también una migración en la parte front-end, en este caso no cambiaría el lenguaje JavaScript, pero si utilizaría un Framework, en este caso no tengo claro si emplearía **Angular**, **React** o **Vue**.*



- **Otras mejoras:**

*Obviando las funcionalidades existen otras mejoras que me gustaría llevar a cabo, como tener la opción de poder cambiar de idioma en la aplicación, como mínimo dar la opción de poder leer la página en inglés, ya que considero que los estudiantes de intercambio o los estudiantes ERASMUS, podrían estar interesados en formar parte de una comunidad de este tipo. Para ello utilizaría el pluggin basado en Jquery, **translate.js.***

También podría implementar un registro/login complementario al tradicional, instaurado con las principales redes sociales, principalmente estas serían Facebook, Instagram y la cuenta de Google, con el fin de poder hacer más ágil, sencilla y rápida la labor de registro y logeo en nuestra aplicación.

Otra mejora factible es la relacionada con el sistema de valoración e implementar el cálculo de la valoración general de un vendedor basándonos en la media de las valoraciones de los productos vendidos por este.

Por último, una mejora que me gustaría implementar es añadir una pasarela de pago, actualmente el proceso de compra termina cuando añadimos el producto al carrito.

4 Valoración personal

SCRUM es un tipo de metodología ágil muy empleada en el ámbito real de un equipo de programación, por ello considero interesante que se haya visto durante este curso.

Durante el desarrollo del proyecto inicial presentado en clase surgieron algunos problemas derivados de diferencias de opinión en lo que al proyecto respecta, pero finalmente logramos llegar a un punto intermedio y el resultado final considero que es satisfactorio.

Al principio los tiempos eran más pausados y teníamos tiempo de sobra para llevar a cabo los puntos del sprint, pero después conforme se acercaban los exámenes y el proyecto tenía más requerimientos fue más complicado compaginar todo y fuimos más cortos de tiempo.

Me parece que este proyecto inicial ha servido, más allá de para conocer la metodología SCRUM, para realizar culminar un proceso de aprendizaje que ya se ha visto evolucionar durante el desarrollo del resto del curso.

En el sector profesional deberemos llevar a cabo un autoaprendizaje, ya que es un sector donde las tecnologías y herramientas evolucionan muy rápidamente y debemos estar actualizados en estas.

En lo que a la migración de tecnologías back-end respecta, empecé con la idea de realizar este proceso hacía migración Spring/Hibernate, estuve siguiendo diversas guías en internet, pero no acababa de comprender los conceptos, es por ello qué, dado que el tiempo para realizar esta presentación era limitado decidí comenzar el proceso de documentación del proyecto inicial.

Una vez terminada esta documentación, Ramón Martínez, a quien nombro en los agradecimientos me aconsejó realizar la migración hacía tecnologías creadas más recientemente, como Spring Boot y JPA, dado que todavía faltaban 2 semanas para realizar la entrega decidí realizar esta migración.

En este proceso encontré diferentes problemas:

- *Al tener las partes del proyecto front-end y back-end funcionando en diferentes servidores, a la hora de hacer llamadas AJAX al API las entendía como llamadas externas y la seguridad del navegador saltaba, esto se solucionó haciendo una sería de cambios explicados en el apartado perteneciente a la estructura back-end.*

- *Al utilizar relaciones entre clases de muchos a muchos, saltaba el StackOverflowError, causado por la existencia de bucles infinitos, la solución a este caso también se explica en el apartado perteneciente a la estructura back-end.*

En general el proceso me ha parecido satisfactorio y me ha permitido aprender a trabajar con nuevas tecnologías JSP.

5 Bibliografía

- *Guía: Spring*
- *Librerías: Font Awesome, ModelMapper, Jquery*
- *Obtención de plantillas: Bootstrap, Initializr*
- *Repositorios: MVN Repository*
- *Apoyo técnico: Stack Overflow, W3schools*
- *Diagramas: Alicia, Ramos Martín. (2014) Entornos de desarrollo, España, Garceta Grupo Editorial*