NEXTGENMMA

## Propuesta de Proyecto DAM

DESARROLLO DE UNA APLICACIÓN DE GESTIÓN

## Diego Gil Sanz

25209587T

## Desarrollo de aplicaciones Multiplataforma

Curso académico 2023-25

## Tutor de Proyecto

José Manuel Guallar

Contenido

[Propuesta de Proyecto DAM 1](#_Toc198242683)

[Diego Gil Sanz 1](#_Toc198242684)

[Desarrollo de aplicaciones Multiplataforma 1](#_Toc198242685)

[Tutor de Proyecto 1](#_Toc198242686)

[1. Justificación y planificación del proyecto 3](#_Toc198242687)

[1.1.- Descripción del proyecto 3](#_Toc198242688)

[1.2.- Justificación de la tecnología utilizada 3](#_Toc198242689)

[1.3.- Planificación Gantt 4](#_Toc198242690)

[Primera semana 4](#_Toc198242691)

[1.4.- Planificación Scrum 6](#_Toc198242692)

[1.5.- Prototipado 9](#_Toc198242693)

[2. Plan de empresa 9](#_Toc198242694)

[2.1.- Oportunidades de negocio y necesidades demandadas 9](#_Toc198242695)

[2.2.- Relación de productos o servicios a prestar 9](#_Toc198242696)

[2.3.- Constitución de la empresa 9](#_Toc198242697)

[2.4.- Recursos 9](#_Toc198242698)

[3. Análisis y diseño de los requisitos de la aplicación 9](#_Toc198242699)

[3.1.- Herramientas escogidas 9](#_Toc198242700)

[3.2.- Diagrama de casos de uso 10](#_Toc198242701)

[3.4.- MER 11](#_Toc198242702)

[4. Desarrollo de la aplicación 12](#_Toc198242703)

[4.1.- Control de usuarios 12](#_Toc198242704)

[4.2.- Desarrollo e implementación de la aplicación 26](#_Toc198242705)

[MAPEOS 27](#_Toc198242706)

[5. Pruebas e informes de la aplicación 43](#_Toc198242707)

[5.1.- Estrategia de pruebas 43](#_Toc198242708)

[5.2.- Pruebas realizadas 43](#_Toc198242709)

[5.3.- Informes 43](#_Toc198242710)

[6. Implantación y documentación 44](#_Toc198242711)

[6.1.- Exportación y distribución de la aplicación 44](#_Toc198242712)

[6.2.- Manual de instalación 44](#_Toc198242713)

[6.3.- Manual de usuario 44](#_Toc198242714)

[6.4.- Repositorio de la aplicación 44](#_Toc198242715)

[6.5.- JavaDoc 44](#_Toc198242716)

[7.- Bibliografía 45](#_Toc198242717)

[8.- Anexos 46](#_Toc198242718)

# 1. Justificación y planificación del proyecto

### 1.1.- Descripción del proyecto

NEXTGENMMA es una aplicación que busca facilitar toda la gestión de la federación española de mma, busca facilitar las inscripciones de nuevos clubes, gimnasio y ligas.

Las federaciones no evolucionan y los peleadores si, los peleadores quieren lo mejor y se merecen lo mejor.

Nuestra aplicación aportara una solución a los tiempos de espera y el desplazamiento de los trabajadores a las oficinas, contaremos con una función de enviar solicitudes a el correo de la federación, para llevar a cabo todas las solicitudes de forma mas rápida y sin necesidad de llevar horarios de las oficinas.

Podremos eliminar, modificar y añadir los peleadores gracias a la ventana de administración.

También es capaz de mostrar diferentes anuncios/publicaciones para que los usuarios estén bien informados.

### 1.2.- Justificación de la tecnología utilizada

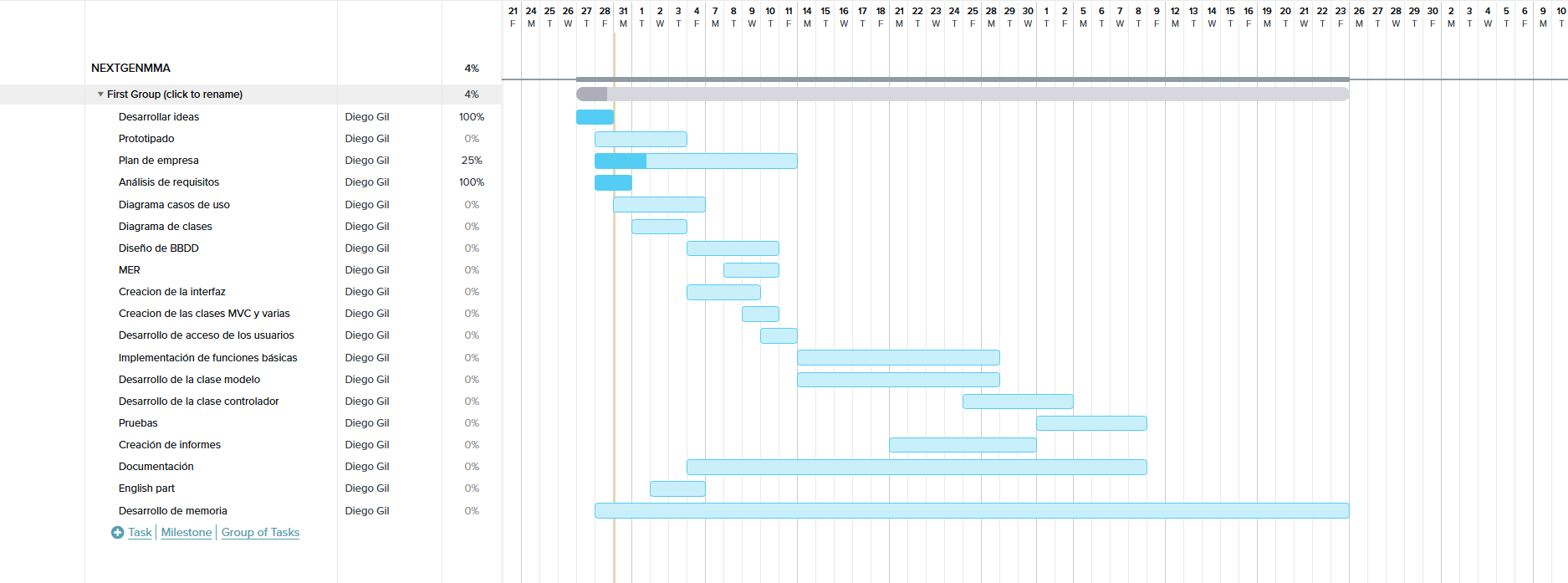
He elegido hibernate debido a su facilidad de mapeo, facilidad de gestión de datos y porque es mas sencillo crear código, me permite simplificar las consultas y operaciones, también tiene una cualidad muy importante y es que es rápida, da mas fluidez a la aplicación.

He elegido Java como lenguaje debido a que estoy acostumbrado a crear proyectos con el, tengo un control suficiente para crear programas con diversas funciones y que sea muy completo, además, es el lenguaje que mas he usado.

Estoy usando el entorno de desarrollo de IntelIJ debido a que la interfaz es muy comoda y me resulta mas fácil desarrollar código, además, me resulta un entorno de desarrollo muy rápido con una gran cantidad de herramientas y posibilidades

### 1.3.- Planificación Gantt

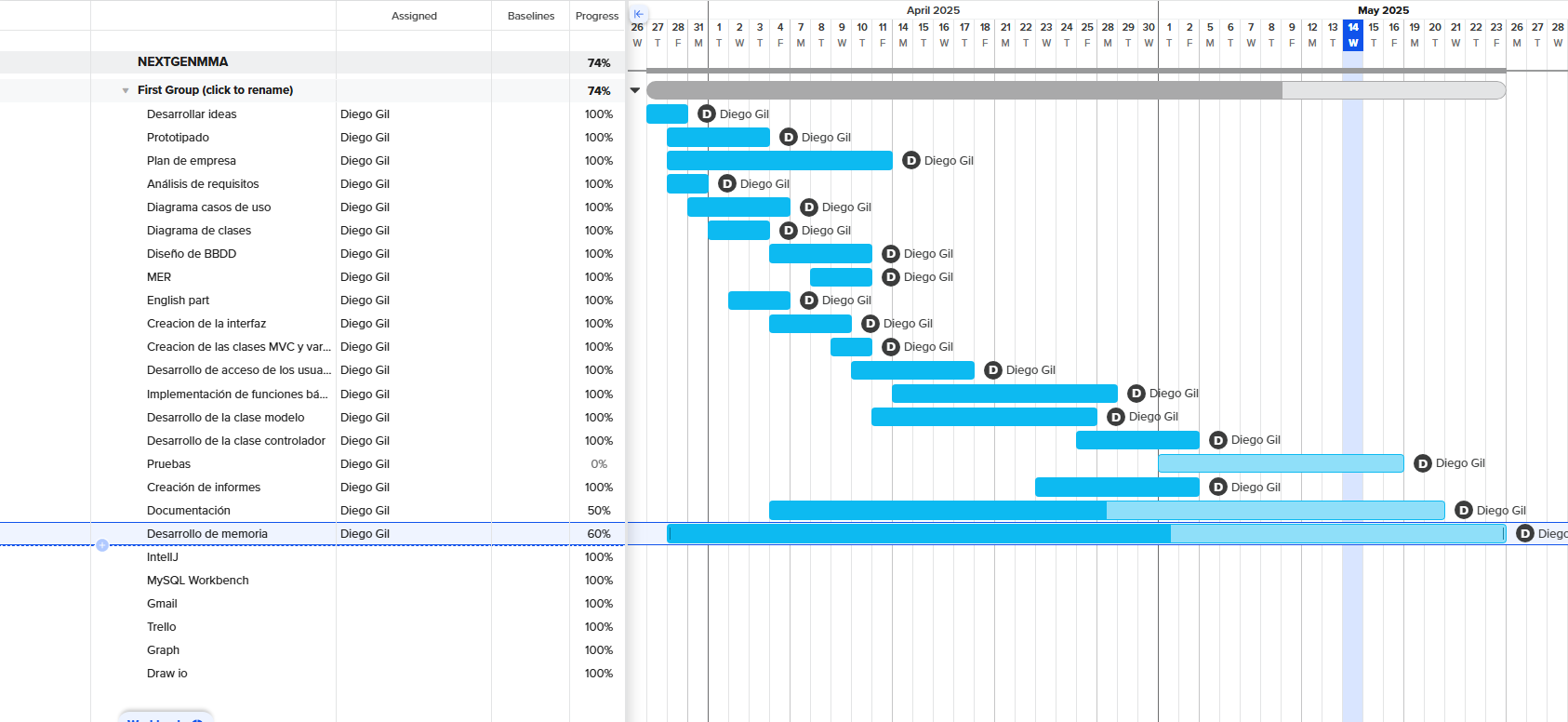
### Primera semana



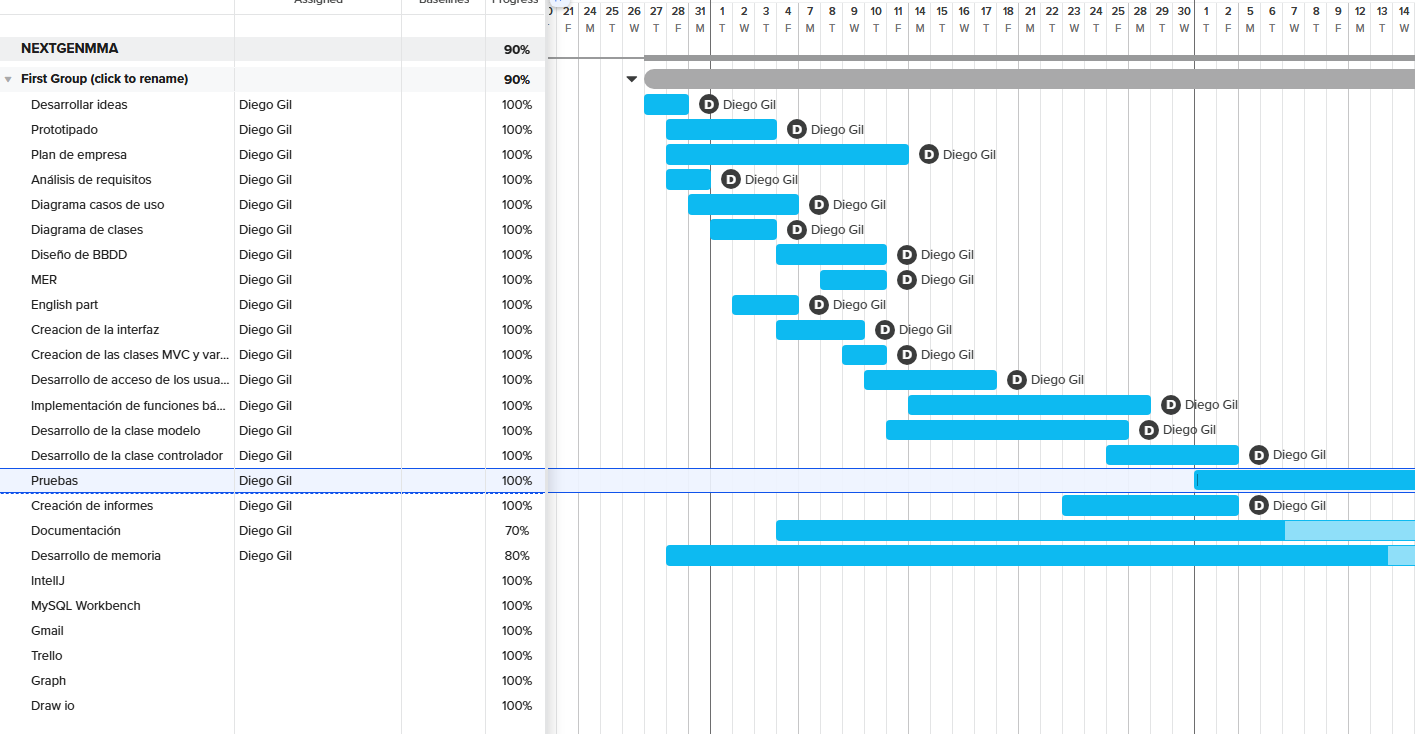
Primeras 3 semanas



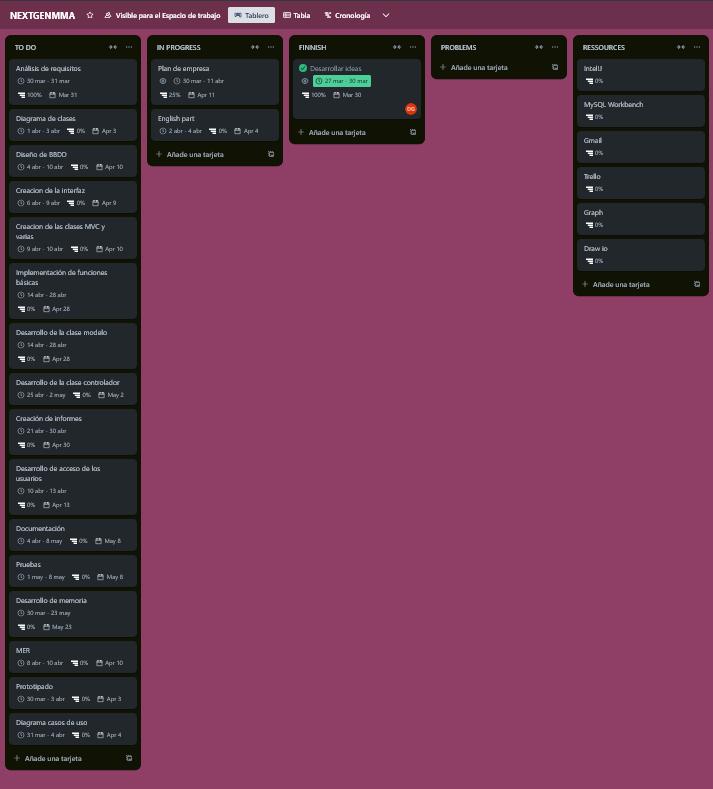
Progreso de 5 semanas



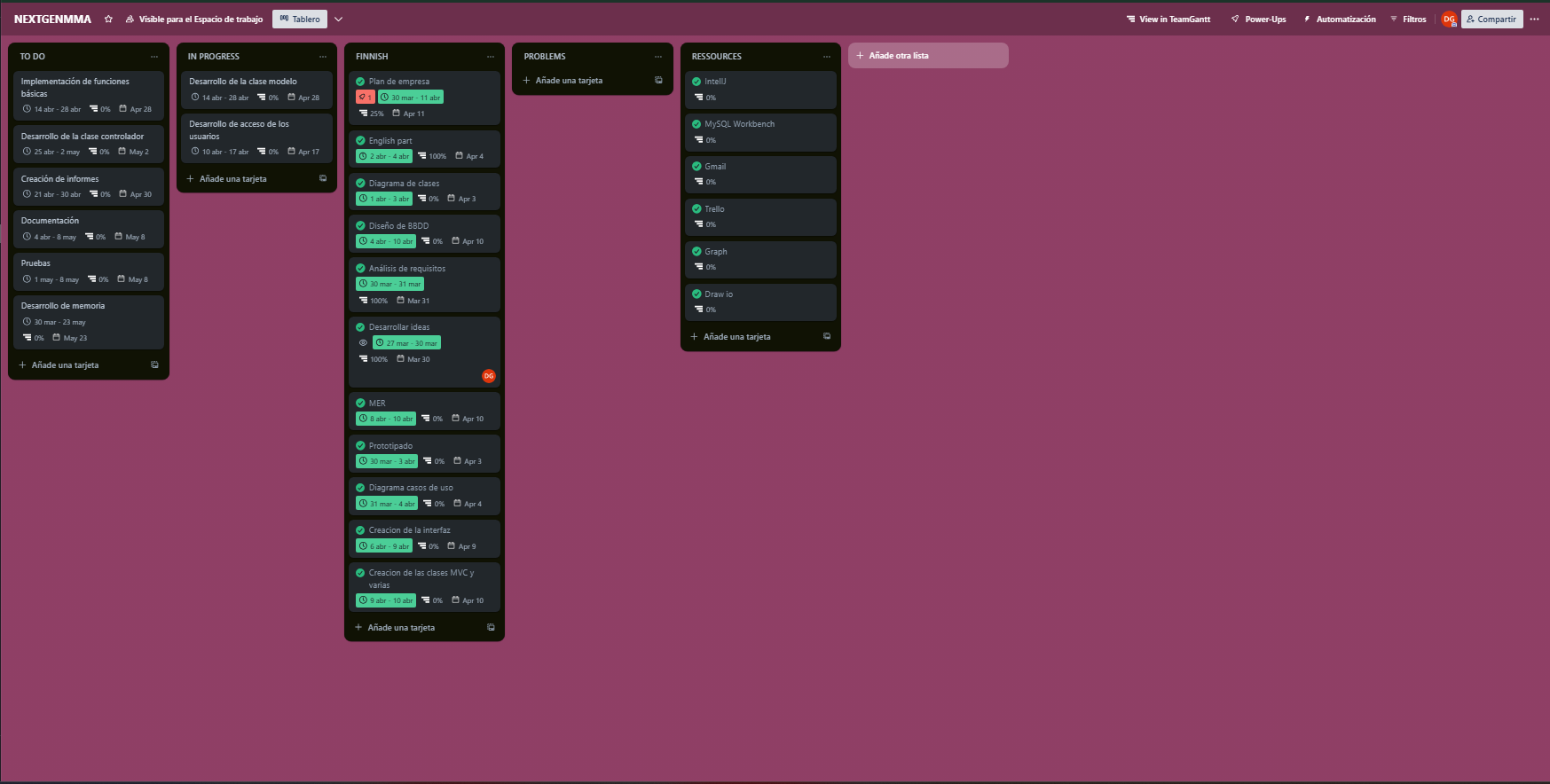
Progreso 7 semanas



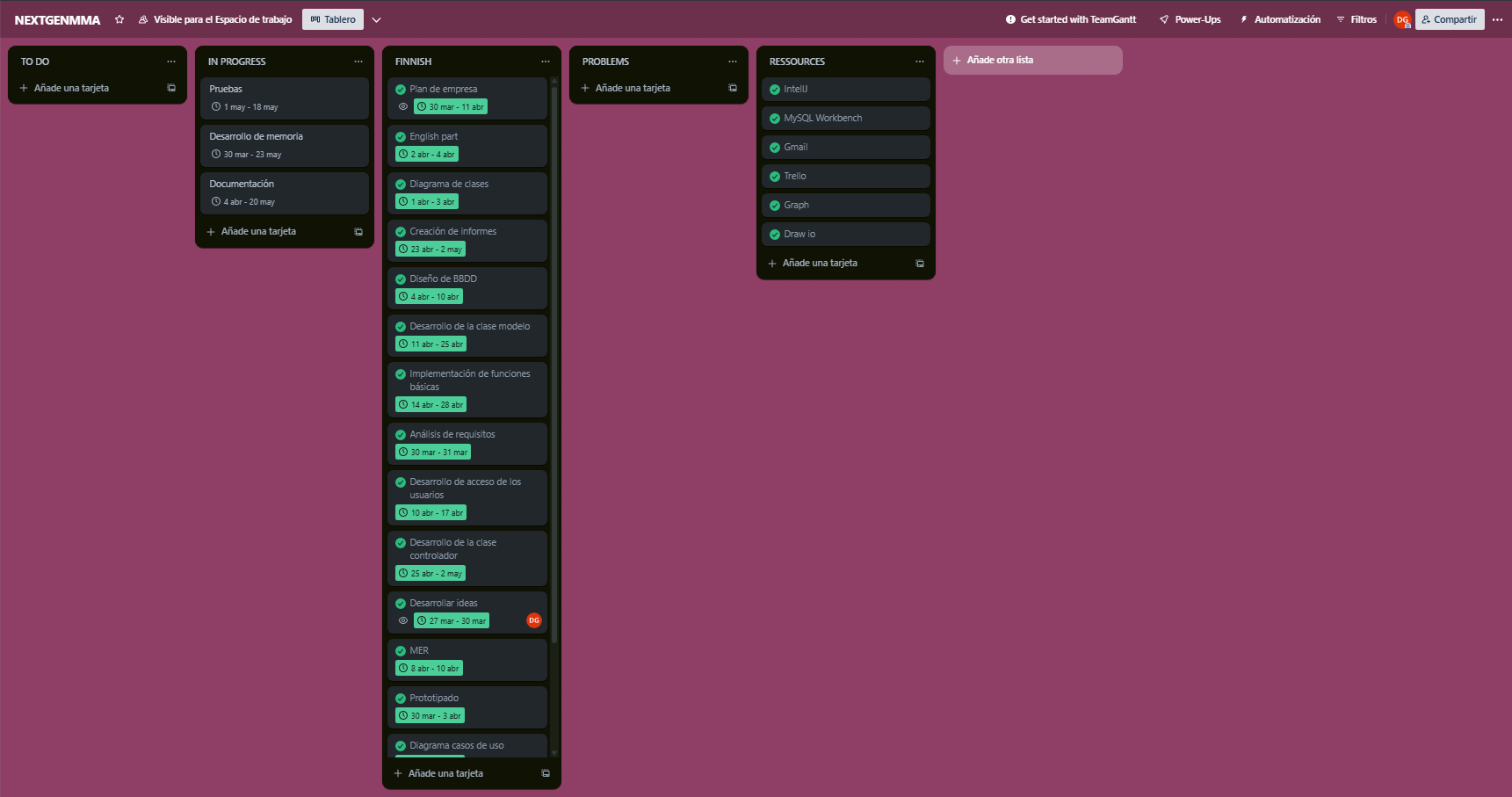
### 1.4.- Planificación Scrum

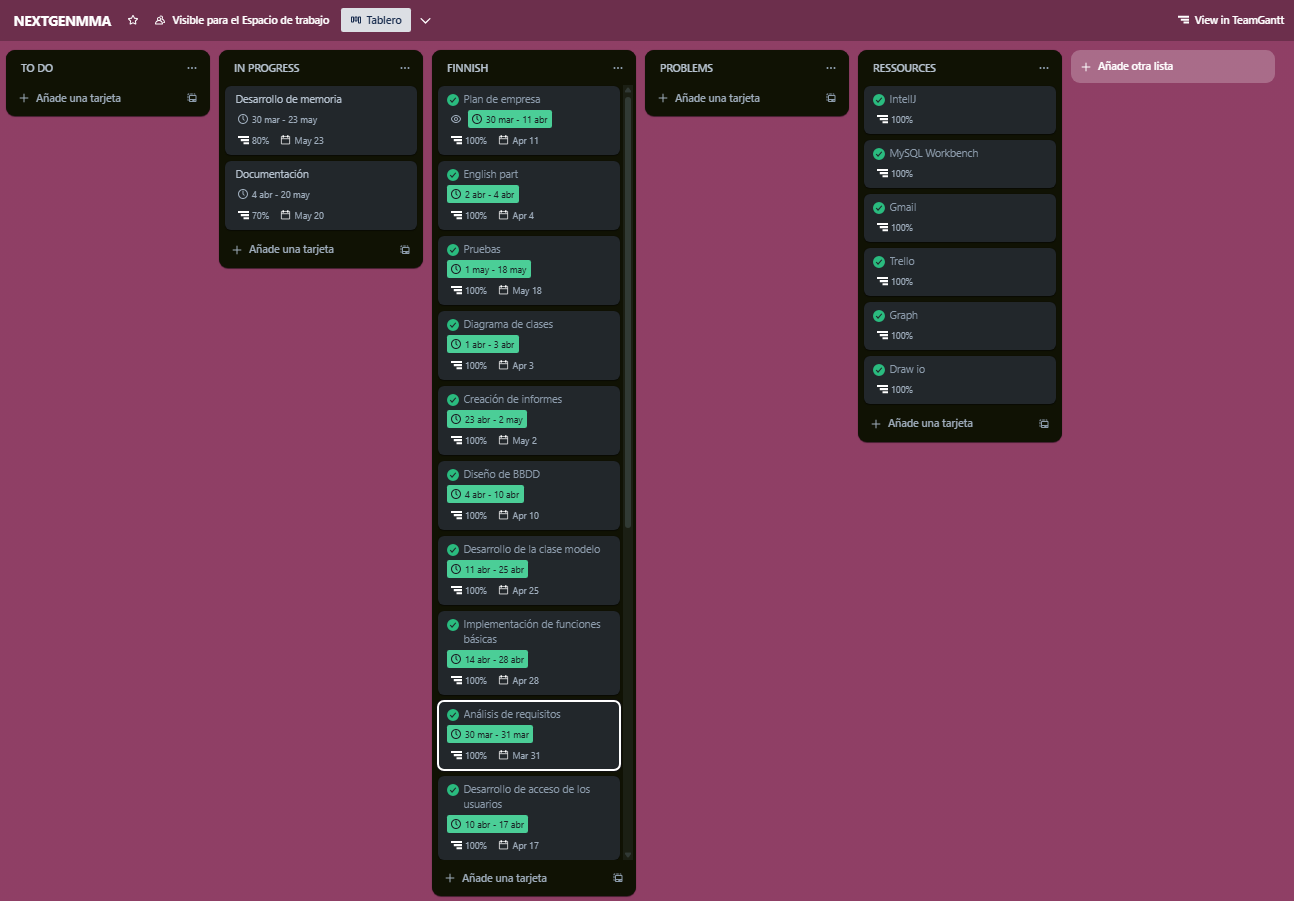


Primeras 3 semanas



Progreso hasta semana 5



Progreso 7 semanas

### 1.5.- Prototipado

https://www.figma.com/design/QT8KhX3sQUVjjTT8GcUW1d/TFG-MMA?node-id=1-1403&t=coS1ciNZcFbV1opu-1

# 2. Plan de empresa

### 2.1.- Oportunidades de negocio y necesidades demandadas

#### 

#### Justificación de la oportunidad de negocio

Las federaciones de los múltiples deportes que se practican en España no usan sus aplicaciones y en algunos casos dejándolas descuidadas, como miembro de un club de artes marciales mixtas, he podido observar que más que nunca, en esta era informática necesitamos evolucionar a la hora de hacer tramites. Por lo que sé, no hay muchas empresas interesadas en poder rehacer algunas de las páginas web de las federaciones.

Al no tener mucha competencia, podemos aprovechar para negociar con cada federación y que logren apostar por nosotros, personas que hemos practicado estos deportes y sabemos cómo lo queremos y lo que necesitamos

#### Descripción de la localización de la empresa.

La localización elegida seria Zaragoza, una ciudad en la que los miembros de la empresa estamos familiarizados.

Apostaremos por el talento local, el cual está en auge, cada día son más los jóvenes que se aventuran en el mundo del desarrollo de aplicaciones y Aragón no se queda atrás. Sabemos que, en Madrid, Barcelona, Valencia, son ciudades grandes que cuentan con una gran cantidad de talento joven, pero nosotros queremos apostar por los jóvenes de Aragon para brindarles una oportunidad de elevar su carrera a un nivel superior sin necesidad de tener que salir de su tierra, además somos una empresa con valores ultra competitivos, buscamos crecer como empresa y llegar a demostrar que Aragón tiene un potencial único en el sector. No solo nos basamos en el talento joven, también nos hemos asegurado de que Zaragoza nos brinde alguna ayuda. Podremos acceder a deducciones fiscales gracias a programas que ayudan a impulsar a las pequeñas empresas, hay otro plan de ayudas que nos ayuda a reducir el IBI, nos aporta un gran porcentaje de descuento a la hora de pagar el IBI y el IAE.

#### Estudio de nuestro cliente y segmentación de mercado

**Clientes**

Nuestros clientes objetivos son las federaciones deportivas, principalmente la federación española de MMA, pero buscamos también expandirnos a diversas federaciones de los diferentes deportes que se practican en España

### 2.2.- Relación de productos o servicios a prestar

#### Producto/servicio

El producto principal que ofrecemos es una aplicación de gestión que permitirá a los administradores añadir, modificar y eliminar nuevos gimnasios, peleadores, ligas o entrenadores. Permitimos que los gimnasios tengan más facilidad para comunicarse con la federación gracias a la función de mandar un mensaje directo a soporte para tratar la solicitud. Las personas que no pertenezcan a ningún club podrán acceder a la sección de POST para estar al día de las ultimas noticias de las MMA.

#### Precio

La aplicación principalmente será gratis para que todos los clubes y federaciones la puedan usar. Al ser una aplicación gratis los fans de las MMA podrán usarla para estar actualizados en todo momento, pero, si no eres fan, podrás descubrir el mundo de las artes marciales mixtas cómodamente y de forma totalmente gratuita.

#### Promoción

Ofrecemos una aplicación gratuita, no podemos hacer rebajas en nuestro precio por lo tanto las federaciones que decidan usar nuestra aplicación, les ofrecemos un curso formativo gratuito para que sus trabajadores sean totalmente eficientes.

#### Publicidad

NEXTGENMMA se dará a conocer a través de las redes sociales, publicando contenido sobre las novedades de las MMA y diversas curiosidades. Crearemos contenido de humor para los jóvenes y al final de cada video promocionaremos nuestra aplicación para que el público una vez entretenido, pueda descubrirnos y tener más posibilidades de llegar a ellos.

#### Distribución

Nuestra aplicación se podrá obtener principalmente desde nuestra página web, pero, también facilitamos a los usuarios que nos siguen en nuestras redes sociales poder obtenerla. Tendremos un enlace de descarga directa de la última versión en la biografía de nuestras RRSS para que sea más rápido y cómodo poder descubrirnos.

### 2.3.- Constitución de la empresa

#### Tipo de sociedad elegido explicando las razones

NEXTGENMMA es una sociedad limitada, ¿Por qué?, una sociedad limitada nos brinda muchas ventajas al ser una empresa pequeña y con poco capital, nos aseguramos protección gracias a la responsabilidad limitada, no tenemos que responder con el capital personal y es una gran ventaja al ser una empresa pequeña que está comenzando, minimizamos posibles daños y aportamos una imagen profesional, asegurando a los posibles inversores o clientes que no hay un gran riesgo de perdidas.

Somos conscientes de que al principio tenemos que pagar 3500 € aproximadamente para cubrir todos los gastos iniciales y de mantenimiento de la sociedad, pero tener la seguridad de que no tenemos que poner el capital personal es lo importante.

Otro punto muy importante es la escalabilidad que permite, empezaremos siendo pocos socios, pero en el futuro nos gustaría ser más, para crecer y desarrollarnos, la sociedad limitada nos permite crecer.

Por ultimo y no menos importante, los impuestos, podremos aprovechar las deducciones y nos beneficiaremos de un tipo impositivo reducido los primeros años.

#### Nombra la relación de trámites mercantiles, fiscales, laborales y locales que debe realizar la empresa

* **TRAMITES MERCANTILES**
  + **Certificación negativa del nombre**
    - * Sirve para comprobar que el nombre no está en uso y está disponible
      * Se solicita en el Registro Mercantil Central
      * Costo de 16€
  + **Redacción de Estatutos Sociales**
    - * Es el Documento que define la actividad de la empresa, también contiene el reparto de las participaciones y cómo funciona
  + **Deposito del capital social**
    - Al ser una sociedad limitada tenemos que tener ingresado mínimo 3000€ en una cuenta de banco a nombre de la empresa
  + **Escritura de constitución ante el notario**
    - * Todos los miembros deben firmar en la escritura publica
      * Costo= 150-300€
  + **Inscripción en el Registro Mercantil**
    - * Da personalidad jurídica a la empresa
      * Costo=200-300
* **TRAMITES FISCALES**
  + **Obtención del NIF provisional**
    - Se solicita en la Agencia Tributaria (hacienda)
    - Hay que tenerlo antes de registrar la sociedad
  + **Alta en el censo de empresarios**
    - Necesitamos estar de alta para declarar los impuestos
  + **Obtención del NIF definitivo**
    - Una vez registrados en el registro mercantil tenemos que solicitarlo
  + **Declaración del impuesto de Actividades Económicas (PROXIMAMENTRE)**
    - Si la empresa llega a generar más de 1 millón de euros anuales tendríamos que declararlo, pero de momento al ser nueva y sin generar mucho estamos exentos
  + **Alta en el impuesto de sociedades**
    - La empresa tributa por el Impuesto de Sociedades
  + **Declaración trimestral y anual de impuestos**
    - IVA
    - Retenciones de IRPF
    - Impuesto de sociedades
* **TRAMITES LABORALES**
  + **Inscripción de la empresa en la Seguridad Social**
    - **NEXTGENMMA** tendrá contratados varios trabajadores por lo tanto tiene que estar inscrita
  + **Alta del administrador en el RETA**
    - Al principio al ser el único administrador de la sociedad, tendré que estar dado de alta, en un futuro si se divide la administración podría ser que no hiciera falta

**Alta de trabajadores en la Seguridad Social**

* + - Les daremos de alta en la RED de la tesorería general de la seguridad social
  + **Prevención de riesgos laborales**
    - Al tener trabajadores estamos obligados a tener un servicio de prevención de riesgos laborales

Por el momento estos son los tramites que tenemos que realizar, no necesitamos Tramites Locales debido a que cada trabajador realizara su actividad desde su domicilio, estos nos ahorran una gran cantidad de dinero, en un futuro cuando decidamos montar oficinas sí que tendríamos que pagar las **tasas municipales**, **Registrarnos en el padrón municipal** y obtener la **Licencia de actividad y apertura.**

### 2.4.- Recursos

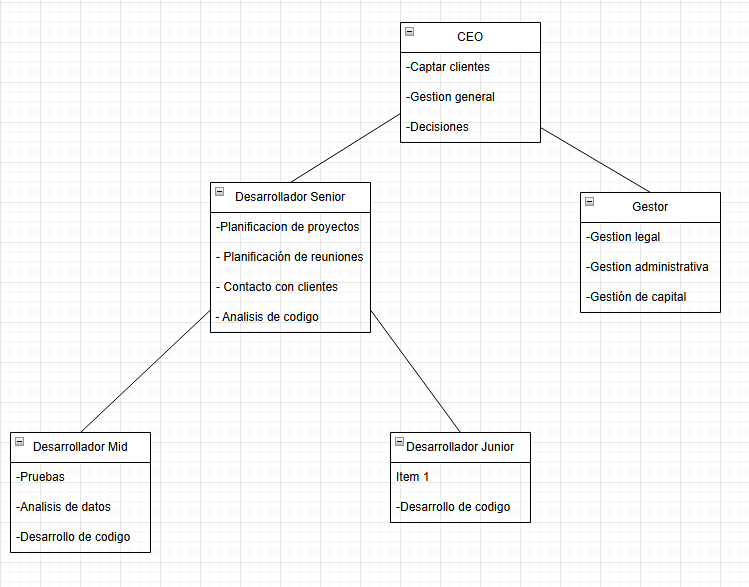
#### Recursos Materiales/Técnicos

|  |  |  |
| --- | --- | --- |
| ***ELEMENTO*** | ***MENSUAL*** | ***ANUAL*** |
| **Licencias IntelIJ**  **(4 licencias)** | 83€ | 996€ |
| **GitHub team**  **(4 cuentas)** | 16€ | 192€ |
| **GitHub copilot**  **(4 cuentas)** | 40€ | 480€ |
| **Servidor en la nube** | 8€ | 96€ |
| **Licencias Trello**  **(4 cuentas)** | 40€ | 480€ |
| ***Total, mensual*** | *187€* |  |
| ***Total, anual*** |  | *2244€* |

|  |  |
| --- | --- |
| ***ELEMENTO*** | ***COSTE*** |
| ***Ordenador sobremesa***  ***(4 unidades)*** | ***4800€*** |
| ***Monitor (4 unidades)*** | ***800€*** |
| ***Periféricos (4 unidades)*** | ***280€*** |
| ***Mesa (4 unidades)*** | ***800€*** |
| ***Sillas (4 unidades)*** | ***400€*** |
| ***TOTAL*** | ***7080€*** |

#### Recursos Humanos

|  |  |  |
| --- | --- | --- |
| ***Concepto*** | ***MENSUAL*** | ***ANUAL*** |
| ***Desarrollador Jr.***  ***(2 desarrolladores)*** | 3640€ | 43680€ |
| **Desarrollador Mid** | 2340€ | 28080€ |
| ***Desarrollador Sr*** | 3250€ | 39.000€ |
| ***Gestor*** | 300€ | 3600€ |
| ***TOTAL*** | ***9530€*** | ***114360€*** |

******

#### Recursos Económicos

#### INVERSION INICIAL

|  |  |
| --- | --- |
| ***CONCEPTO*** | ***COSTE*** |
| ***Recursos humanos*** | 9530€ |
| ***Recursos físicos*** | 7080€ |
| ***Recursos humanos*** | 114360€ |
| ***Fondos SL*** | 3000€ |
| ***Notaria*** | 200€ |
| ***Obtención del NIF*** | 20-50€ |
| ***INVERSION INICIAL*** | ***134.200€*** |

# 3. Análisis y diseño de los requisitos de la aplicación

### 3.1.- Herramientas escogidas

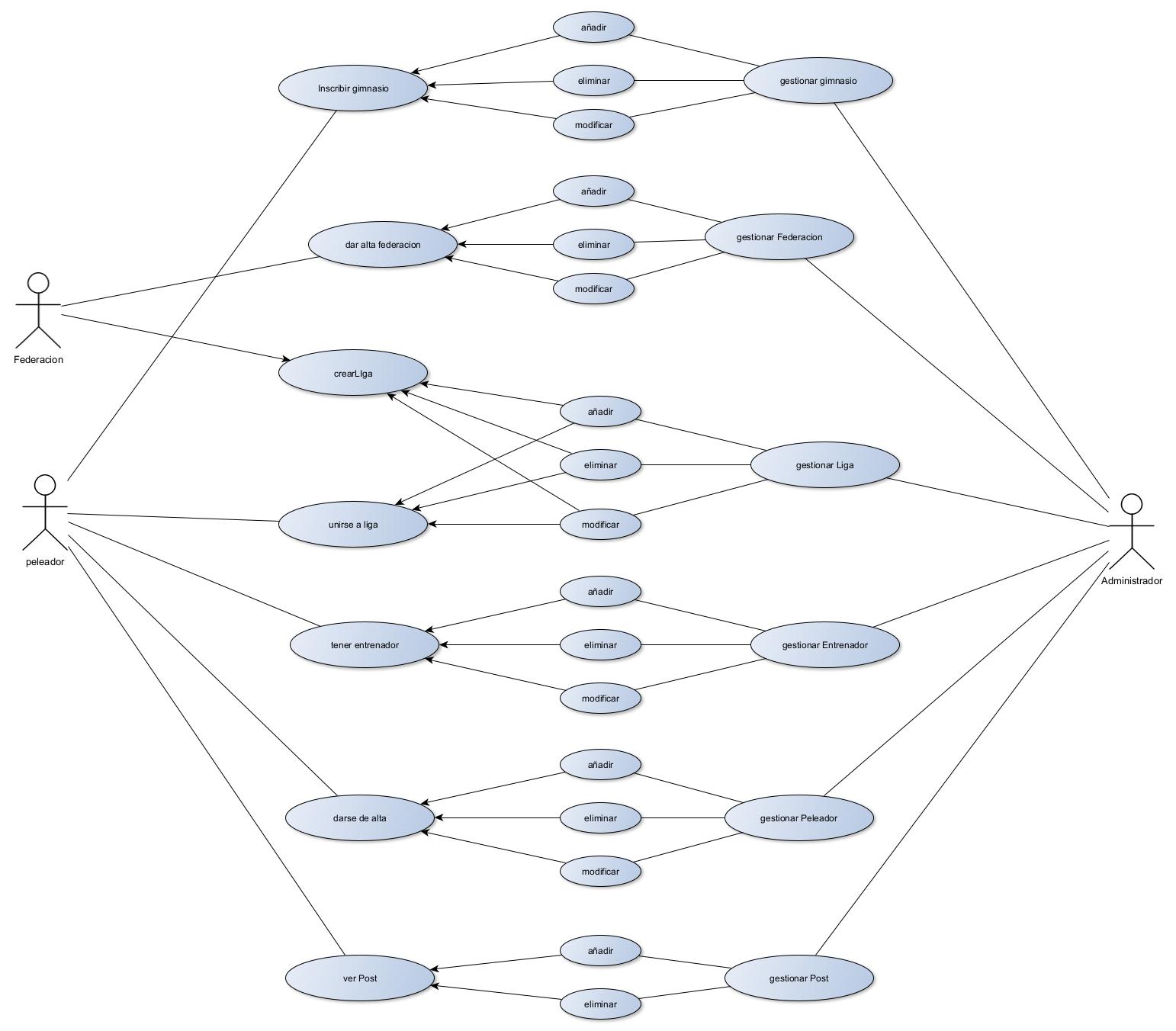
Interfaz de desarrollo= IntelIJ IDEA

Conexión de BBDD= Hibernate

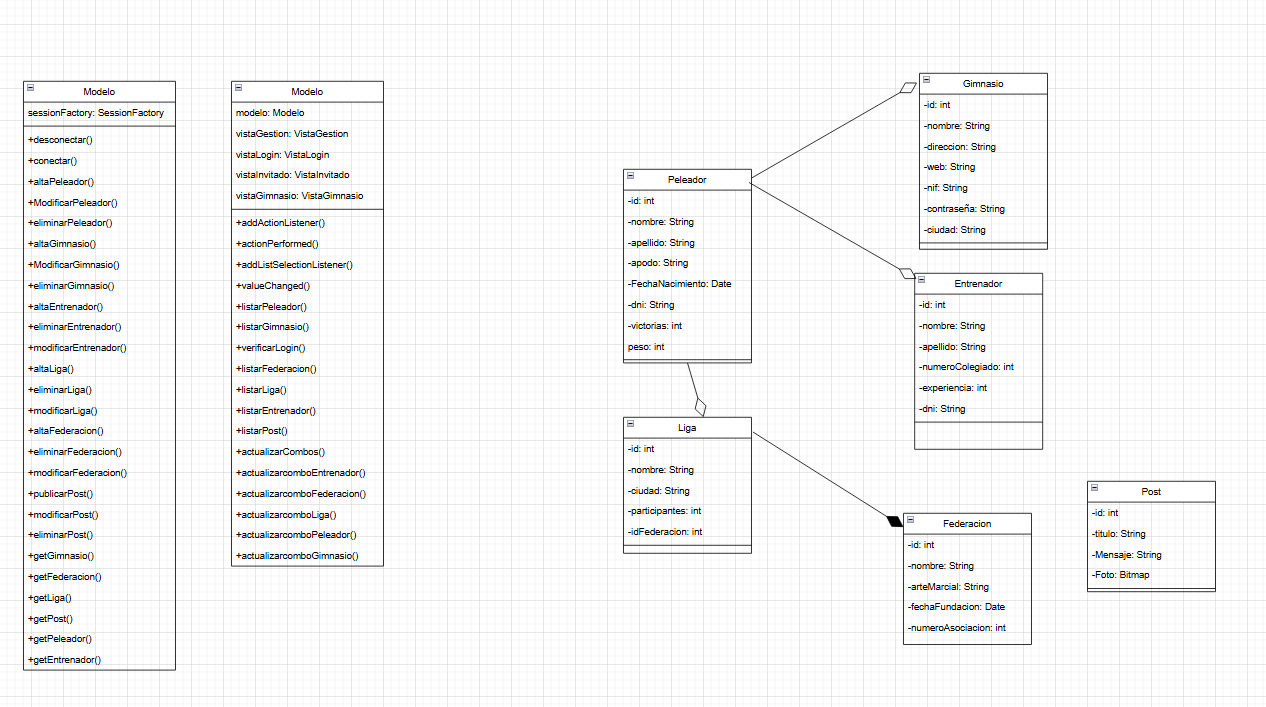
Herramienta de desarrollo de bases de datos= MySQL Workbench

Tipo de bases de datos= MySQL

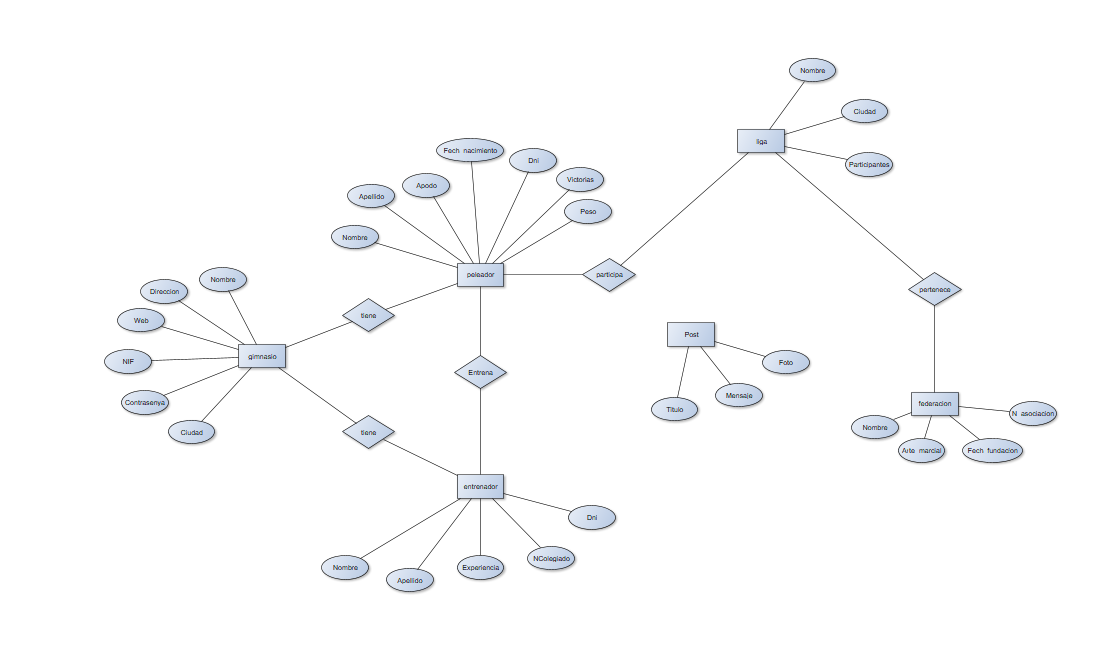
### 3.2.- Diagrama de casos de uso



3.3.- Diagrama de clases



### 3.4.- MER



# 4. Desarrollo de la aplicación

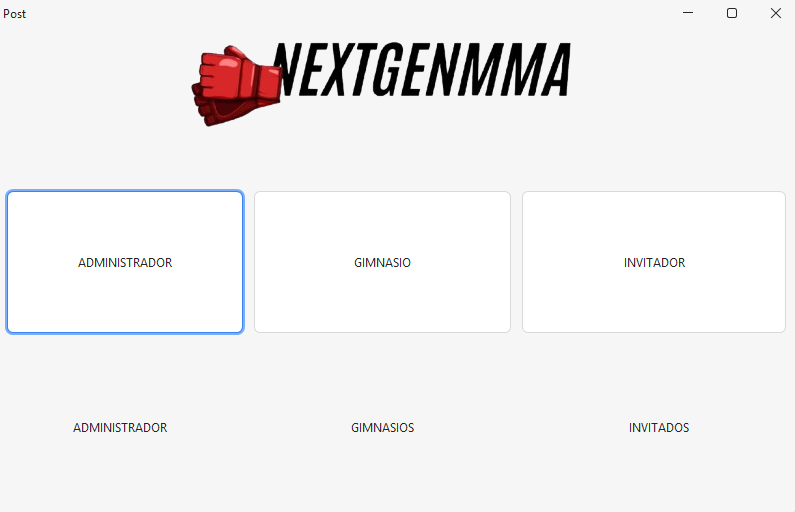
## 4.1.- Control de usuarios

Tengo 3 tipos de usuarios:

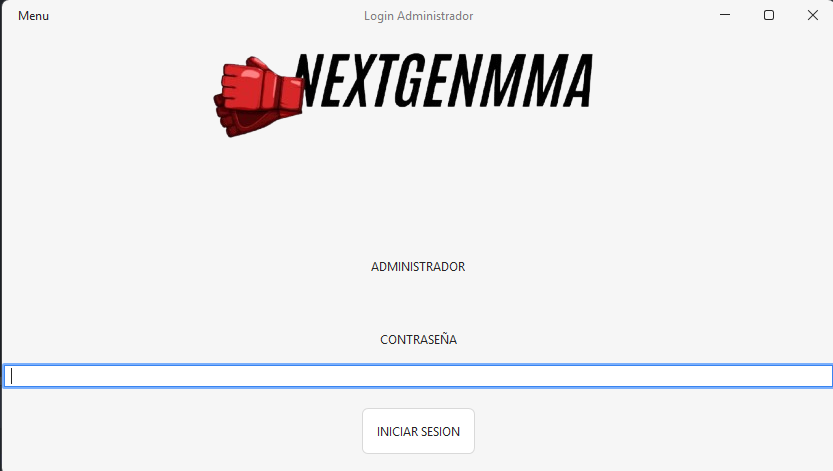
* **Administrador**
* **Gimnasio**
* **Invitado**

Este es el código que utilizo para controlar los inicios de sesión en los diferentes usuarios

case "administrador":  
 vistaUsers.setVisible(false);  
 vistaLoginAdmin.setVisible(true);  
 break;  
case "gimnasio":  
  
 vistaUsers.setVisible(false);  
 comboGimnasioLogin();  
 vistaLoginGimnasio.setVisible(true);  
  
 break;  
case "invitado":  
 vistaUsers.setVisible(false);  
 vistaPost.setVisible(true);  
 inicializarPost(modelo.getPosts());  
 break;  
case "login admin":  
 String hashGimnasioContraseñaLoginAdmin;  
 hashGimnasioContraseñaLoginAdmin = DigestUtils.sha256Hex(vistaLoginAdmin.administradorContraseñaTxt.getText());  
 Admin admin=modelo.getAdmin();  
 if (modelo.verificarContrasena(admin,hashGimnasioContraseñaLoginAdmin)) {  
 vistaLoginAdmin.setVisible(false);  
 actualizarListas();  
 actualizarCombos();  
 vistaGestion.pack();  
 vistaGestion.setVisible(true);  
  
 }else {  
 JOptionPane.showMessageDialog(vistaLoginAdmin, "Contraseña incorrecta", "Error", JOptionPane.ERROR\_MESSAGE);  
 }  
 break;  
case "login gimnasio":  
 String hashGimnasioContraseñaLogin;  
 hashGimnasioContraseñaLogin = DigestUtils.sha256Hex(vistaLoginGimnasio.loginGimnasioContraseñaTxt.getText());  
 if (modelo.verificarContrasenaGimnasio((Gimnasio) vistaLoginGimnasio.loginGimnasioCombo.getSelectedItem(), hashGimnasioContraseñaLogin)) {  
 vistaLoginGimnasio.setVisible(false);  
  
 vistaGimnasio.setVisible(true);  
 vistaGimnasio.solicitudPeleadoresActivosTxt.setText(String.valueOf(modelo.peleadoresActivos((Gimnasio) vistaLoginGimnasio.loginGimnasioCombo.getSelectedItem())));  
  
  
 }else {  
 JOptionPane.showMessageDialog(vistaLoginGimnasio, "Contraseña incorrecta", "Error", JOptionPane.ERROR\_MESSAGE);  
 }  
  
 break;

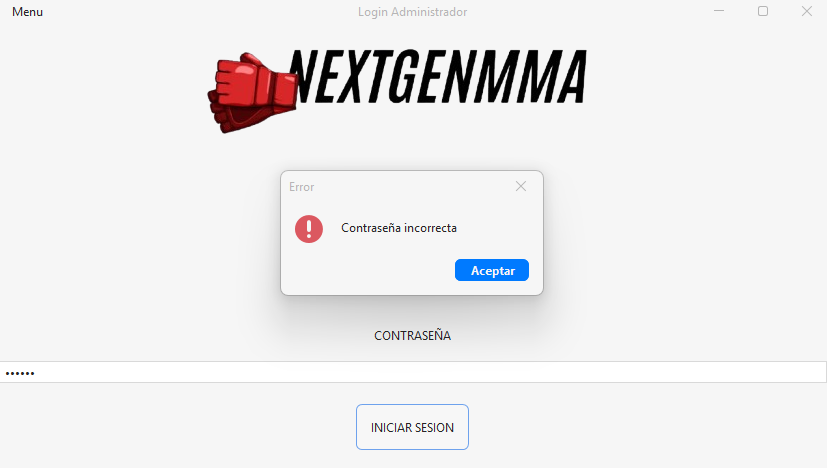
****

**ADMINISTRADOR**

****

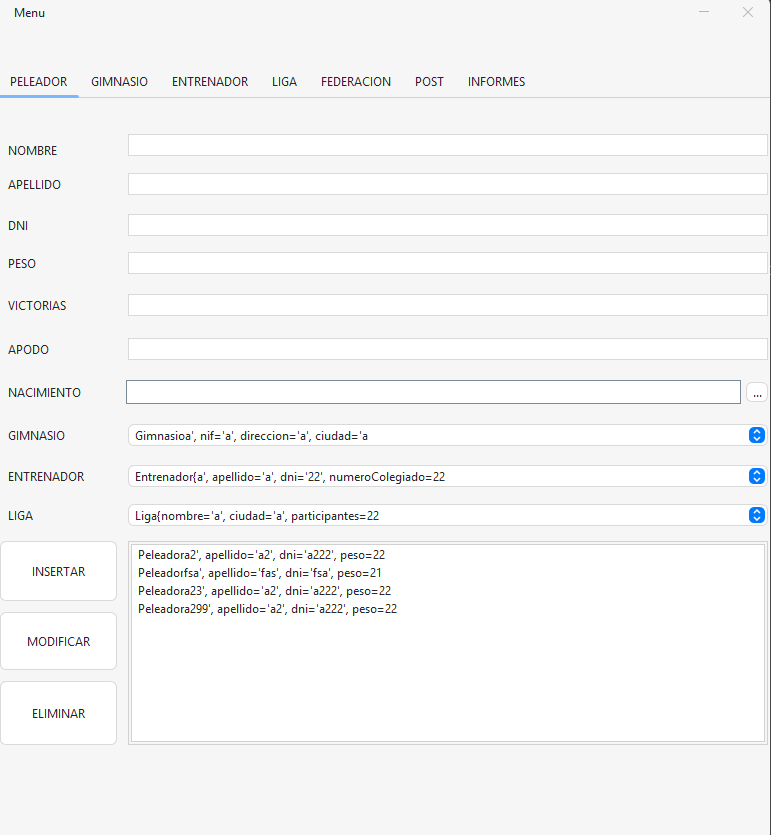
Contara con un login tradicional con usuario fijo, el administrador introducirá la contraseña y accederá a la pestaña de Gestion si la contraseña o de lo contrario le saldrá un mensaje de error.

(CONTRASEÑA ERRONEA)



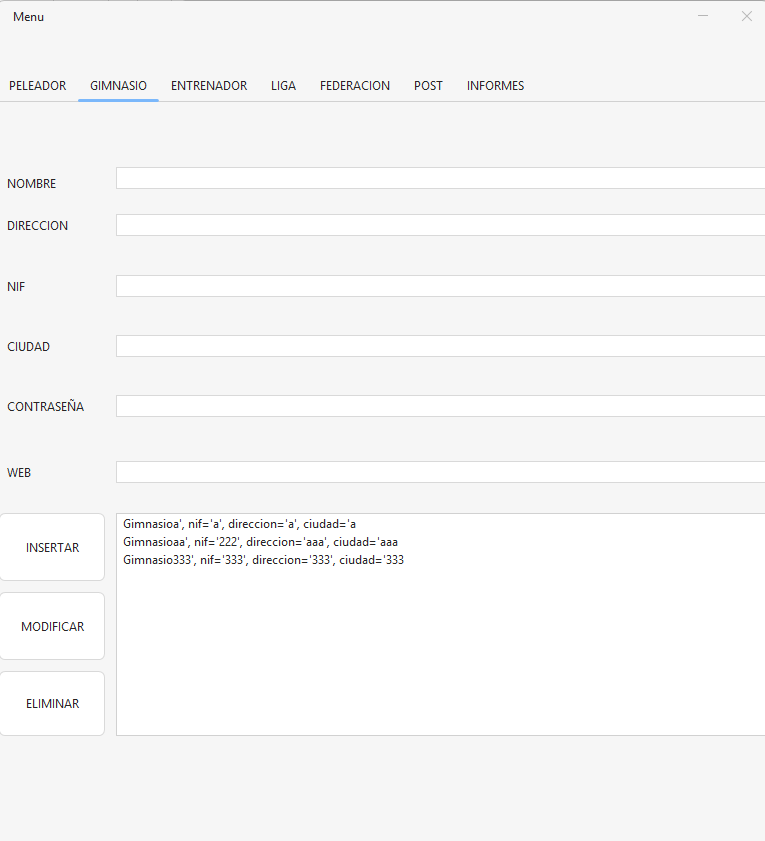
(CONTRASEÑA CORRECTA)

**PESTAÑA PELEADOR**



Aquí el administrador podra eliminar, insertar o modificar los peleadores.

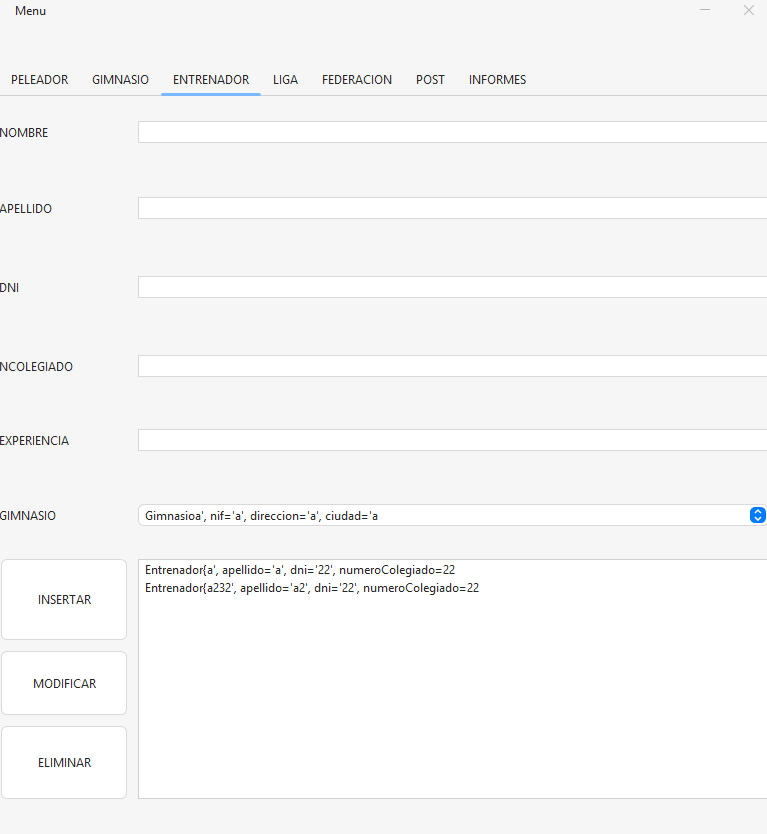
**PESTAÑA GIMNASIO**

****

Aquí el administrador podra eliminar, insertar o modificar los Gimnasios

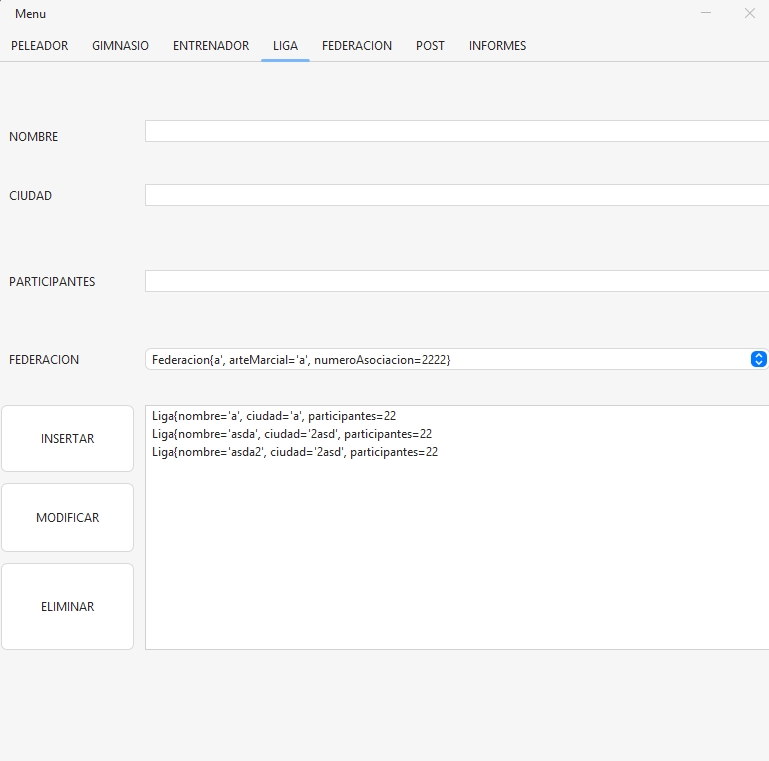
La contraseña se hashea automáticamente.

**PESTAÑA ENTRENADOR**

****

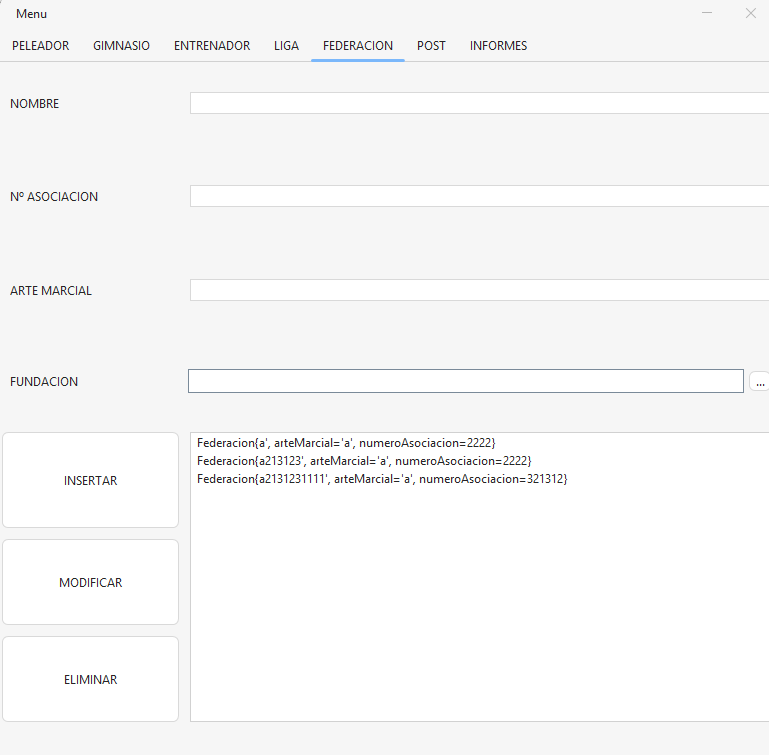
Aquí el administrador podrá eliminar, insertar o modificar los Entrenadores.

**PESTAÑA LIGA**

****

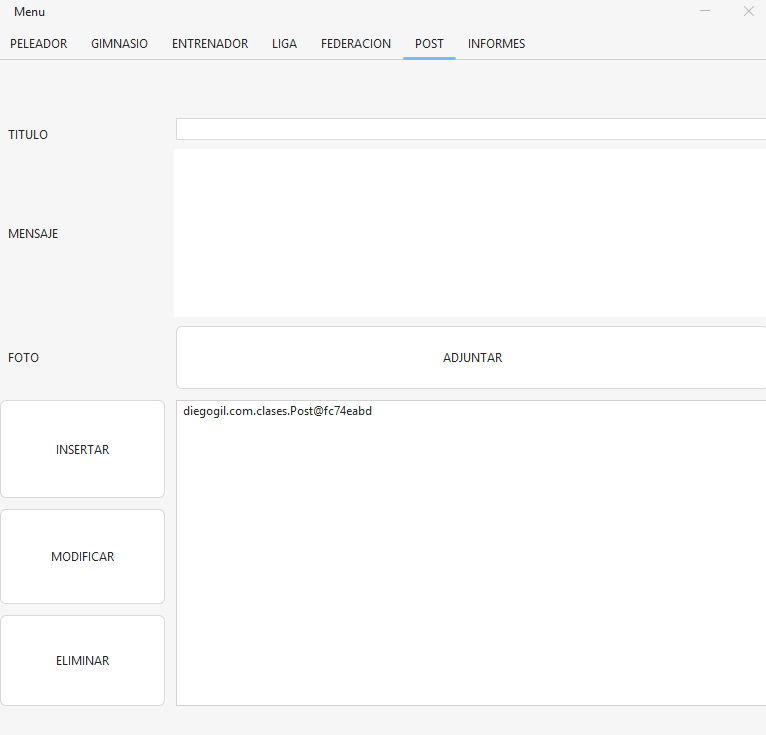
Aquí el administrador podrá eliminar, insertar o modificar las Ligas.

**PESTAÑA FEDERACION**

****

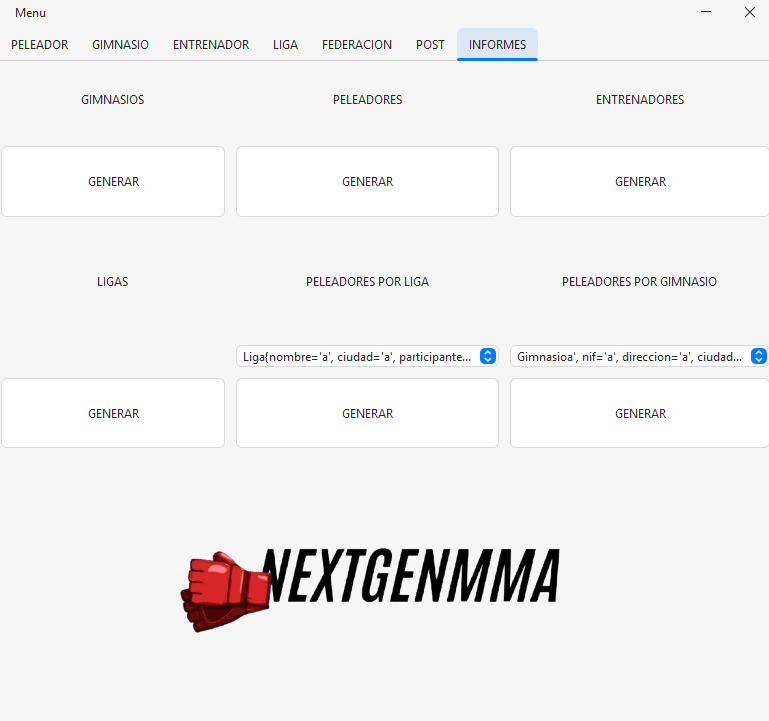
Aquí el administrador podrá eliminar, insertar o modificar las Federaciones.

**PESTAÑA POST**

****

Aquí el administrador podrá eliminar, insertar o modificar los POST.

**PESTAÑA INFORMES**

****

**USUARIO GIMNASIO**

**Login**

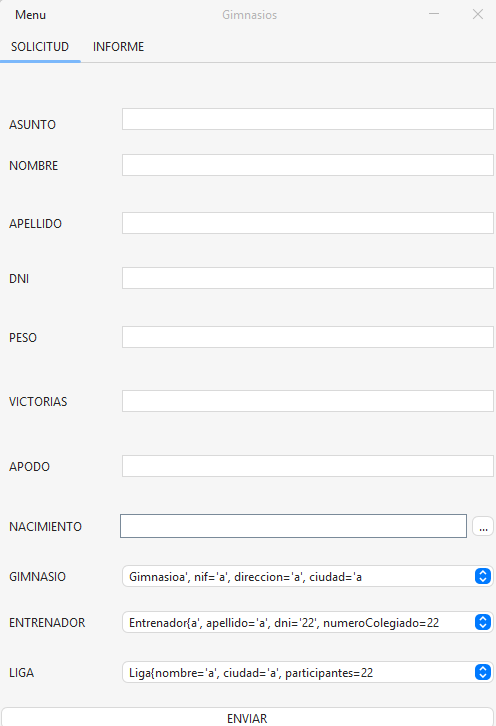
Contara con un login tradicional con varios usuarios, se seleccionara el gimnasio, este introducirá la contraseña y accederá a la pestaña de características de gimnasio si la contraseña o de lo contrario le saldrá un mensaje de error.

(**CONTRASEÑA INCORRECTA**)



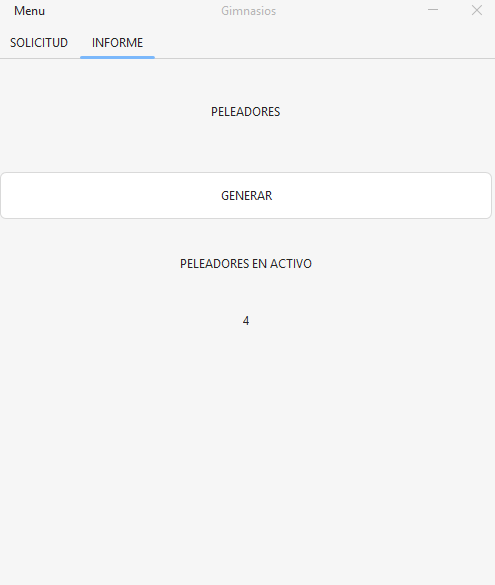
**(CONTRASEÑA CORRECTA)**

**PANEL DE SOLICITUD DE CORREO ELECTRONICO**



Aquí el Gimnasio podra enviar un correo electrónico para realizar acciones sobre sus peleadores

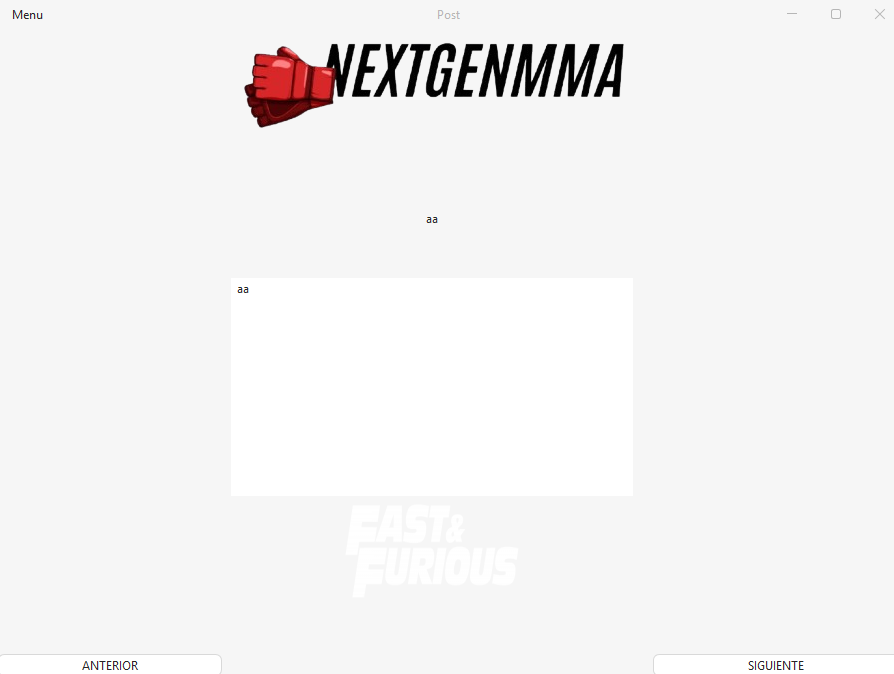
**PANEL DE INFORMES**

****

En la pestaña de informes el Gimnasio será capaz de generar un informe de todos los peleadores que están en su gimnasio además de ver los peleadores que posee.

**INVITADO**

**PANEL DE INVITADOS/POST**

****

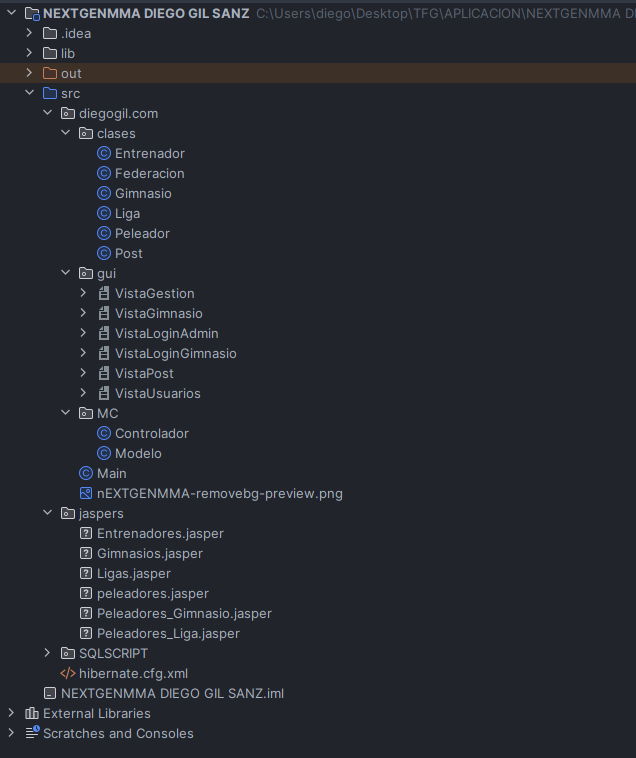
En esta ventana, los invitados podrán ver todos los POST que la publicación ha publicado ordenados del mas nuevo al mas viejo.

## 4.2.- Desarrollo e implementación de la aplicación

La Estructura de Proyecto cuenta con un total de 6 clases mapeadas, 6 Forms con sus respectivas clases para conformar la interfaz, 1 clase modelo donde alberga todo el código de inserción en la base de datos y funciones de generación de informes además de la conexión y la función de enviar correos electrónicos.

Cuento con 6 jasperreport 2 de ellos les paso parámetros a través de código y se encargan de mostrar información.

Por ultimo y de los mas importante, el main, donde llamo a las clases para que funcione correctamente



### MAPEOS

#### PELEADOR





* **Private int idPeleador:**  
  El id del peleador es el identificador único por el cual identificaremos a un peleador en la BBDD, se almacena como numero entero, este no lo tendremos que añadir manualmente debido a que es un campo autoIncrement.
* **Private String nombre:**Es el nombre completo del peleador, se almacena como una cadena de caracteres.
* **Private String apellido:**Aquí almacenaremos los apellidos del peleador, se almacenaran como una cadena de caracteres.
* **Private String dni:**En esta columna almacenaremos el documento de identidad del peleador, se admiten NIF, NIE y visados pero el nombre es dni ya que al principio la aplicación esta pensada para empezar en España y lo que mas hay es DNI, se almacena como cadena de caracteres.
* **Private int peso:**En el campo de peso se almacenara el peso redondeado como numero entero y en formato de kilogramos, la aplicación esta pensada para gestionar la federación y no necesitamos pesos precisos a diferencia de las ligas, este se almacenara como entero en la BBDD.
* **Private int victorias:**

En el campo de victorias se almacenara el numero de victorias del peleador como numero entero.

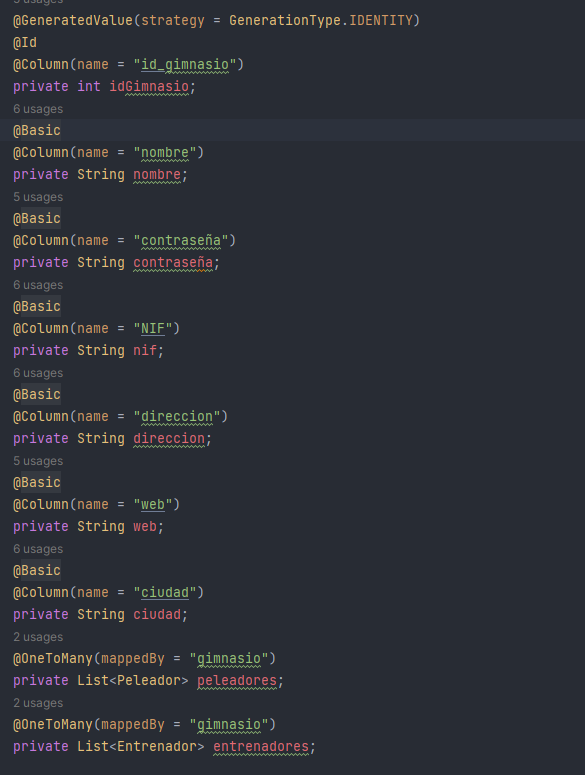
* **Private String apodo:**En el campo apodo se almacena el apodo del peleador como cadena de String, como carácter general es el nombre que recibirá en la presentación.
* **Private Date fecha\_nacimiento:**  
  En el campo de fecha de nacimiento se almacenara la fecha seleccionada en formato de YYYY-MM-DD.
* Un peleador puede tener solo un entrenador
* Un peleador solo puede estar en un gimnasio
* Un peleador solo puede estar en una liga.

#### Entrenador



* **Private int id\_entrenador:**  
  El id del entrenador es el identificador único por el cual identificaremos a un entrenador en la BBDD, se almacena como numero entero, este no lo tendremos que añadir manualmente debido a que es un campo autoIncrement.
* **Private String nombre:**Es el nombre completo del entrenador, se almacena como una cadena de caracteres.
* **Private String apellido:**Aquí almacenaremos los apellidos del entrenador, se almacenaran como una cadena de caracteres.
* **Private String dni:**En esta columna almacenaremos el documento de identidad del entrenador, se admiten NIF, NIE y visados pero el nombre es dni ya que al principio la aplicación esta pensada para empezar en España y lo que mas hay es DNI, se almacena como cadena de caracteres.
* **Private int experiencia**En este campo almacenamos los años de experiencia como numero entero.
* **Private int numero\_colegiado**Almacenaremos el numero de colegiado como numero entero ya que son solo números
* Un entrenador puede tener a su cargo varios peleadores pero un peleador solo un entrenador
* Un Entrenador pertenece a un gimnasio pero un gimnasio puede tener varios entrenadores

#### Gimnasio



* **private int idGimnasio;**El id del gimnasio es el identificador único por el cual identificaremos a un gimnasio en la BBDD, se almacena como numero entero, este no lo tendremos que añadir manualmente debido a que es un campo autoIncrement.
* **private String nombre;**Aqui almacenaremos el nombre del gimnasio como una cadena de caracteres
* **private String contraseña;**Guardaremos la contraseña del gimnasio hasheada en la BBDD para aumentar la seguridad.
* **private String nif;**

En este campo guardaremos el NIF del gimnasio como cadena de caracteres

* **private String direccion;**La dirección del gimnasio de almacenara como una cadena de caracteres en la BBDD
* **private String web;**La pagina web se almacenara como una cadena de String
* **private String ciudad;**La ciudad se almacenara como una cadena de String
* Un gimnasio puede tener varios peleadores pero un peleador un solo gimnasio
* Un gimnasio puede tener varios entrenadores pero un entrenador un solo gimnasio

#### Liga



* **private int idLiga;**El id de la liga es el identificador único por el cual identificaremos a una liga en la BBDD, se almacena como numero entero, este no lo tendremos que añadir manualmente debido a que es un campo autoIncrement.
* **private String nombre;**Es el nombre completo de la liga, se almacena como una cadena de caracteres.
* **private String ciudad;**

La ciudad se guardara en la BBDD como una cadena de caracteres

* **private int participantes;**

La cantidad de caracteres se guardaran como un numero entero

* Una liga puede tener varios peleadores pero un peleador una liga
* Una liga pertenece a una federación/promotora y una federación/promotora puede tener varias ligas.

#### Federacion

#### 

* **private int idFederacion;**El id de la federacion es el identificador único por el cual identificaremos a una federacion en la BBDD, se almacena como numero entero, este no lo tendremos que añadir manualmente debido a que es un campo autoIncrement.
* **private String nombre;**Es el nombre completo de la federacion, se almacena como una cadena de caracteres.
* **private String arteMarcial;**

En este campo se almacenara el arte marcial con la que trabaja la federación/promotora

* **private Date fechaFundacion;**

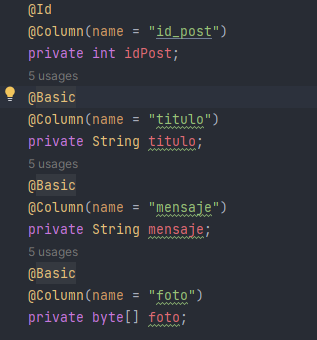
La fecha de fundación se almacenara con formato YYYY/MM/DD

* **private int numeroAsociacion;**

El numero de identificación es la documentación legal que tiene una federación/promotora de identificarse legalmente

* Una federación puede tener varias ligas pero una liga solo una promotora

#### POST



* **private int idPost;**El id del post es el identificador único por el cual identificaremos a un post en la BBDD, se almacena como numero entero, este no lo tendremos que añadir manualmente debido a que es un campo autoIncrement.
* **private String titulo;**En el campo titulo almacenaremos el titulo como una cadena de caracteres
* **private String mensaje;**

El mensaje se almacenara en una cadena de caracteres

* **private byte[] foto;**La imagen se debe guardar como un array de bytes para su correcto almacenamiento

#### Conexión

**LOCAL**

El primer paso es intentar la conexión para comprobar si existe la BBDD, en el caso que exista se desconectara automáticamente y dejara que la conexión mediante hibernate se efectue, en caso de que la BBDD no exista llamara al método encargado de crear la bbdd.

public static Connection connect() {  
 try {  
 if (connection == null || connection.isClosed()) {  
  
 try {  
 connection = DriverManager.getConnection("jdbc:" + sgbd + "://" + ip + ":" + port + "/" + name + "?useSSL=false&allowPublicKeyRetrieval=true", user, password);  
 } catch (SQLException e) {  
 createDatabase();  
 try {  
 connection = DriverManager.getConnection("jdbc:" + sgbd + "://" + ip + ":" + port + "/" + name + "?useSSL=false&allowPublicKeyRetrieval=true", user, password);  
 } catch (SQLException ex) {  
 JOptionPane.showConfirmDialog(null, "Error al conectar a la base de datos. Asegúrate de que el servidor esté en ejecución y la configuración sea correcta.", "Error", JOptionPane.DEFAULT\_OPTION, JOptionPane.ERROR\_MESSAGE);  
 }  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 System.out.println("Conectado a la base de datos: " + name + " en " + ip + ":" + port + " como " + user + " con contraseña: " + password);  
 return connection;  
}

Si la BBDD no esta creada llamara al siguiente método que leera el script de la BBDD adaptado para leerse separando cada sentencia con guiones, una vez creada la BBDD se desconectara y se realizara la conexión mediante hibernate asegurando que puede trabajar en la bbdd

private static void createDatabase() {  
 try {  
 connection = DriverManager.getConnection("jdbc:" + sgbd + "://" + ip + ":" + port + "/?useSSL=false&allowPublicKeyRetrieval=true", user, password);  
 String code = leerFichero(sqlRoute);  
 String[] queries = code.split("--");  
 System.out.println("Ejecutando script SQL para crear la base de datos...");  
 for (String query : queries) {  
 query = query.trim();  
 if (!query.isEmpty()) {  
 try (PreparedStatement statement = connection.prepareStatement(query)) {  
 statement.executeUpdate();  
 }  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 disconnect();  
 }  
}

Una vez nos hemos asegurado de que la BBDD esta creada, conectamos con hibernate

public void conectar() {  
 try {  
 Configuration config = new Configuration();  
 config.configure("hibernate.cfg.xml");  
 config.addAnnotatedClass(Entrenador.class);  
 config.addAnnotatedClass(Gimnasio.class);  
 config.addAnnotatedClass(Peleador.class);  
 config.addAnnotatedClass(Liga.class);  
 config.addAnnotatedClass(Post.class);  
 config.addAnnotatedClass(Federacion.class);  
  
 StandardServiceRegistry ssr = new StandardServiceRegistryBuilder()  
 .applySettings(config.getProperties())  
 .build();  
 sessionFactory = config.buildSessionFactory(ssr);  
 } catch (Exception e) {  
 e.printStackTrace();  
 System.err.println("Error inicializando Hibernate SessionFactory.");  
 }  
}

**SERVIDOR**

En la version de servidor al estar la BBDD ya creada no es necesario comprobar que existe y se puede efectuar la conexión directa

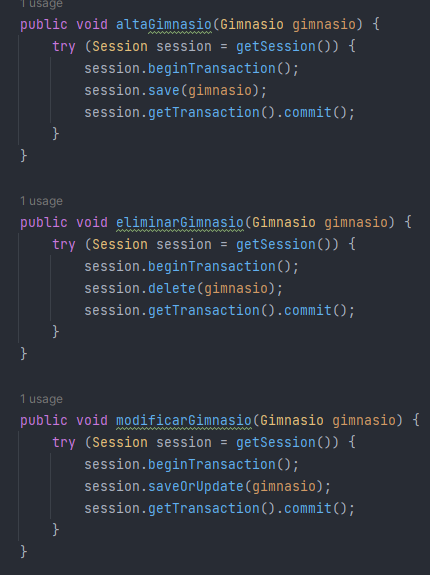


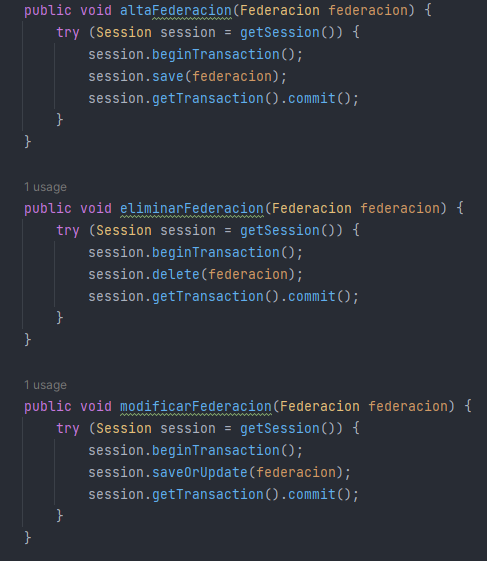
****

#### INSERCION DE DATOS

A la hora de realizar inserciones, modificaciones o eliminaciones, Creamos una Sesion y realizamos la opción deseada

**CONTROLADOR**

* **Insercion**

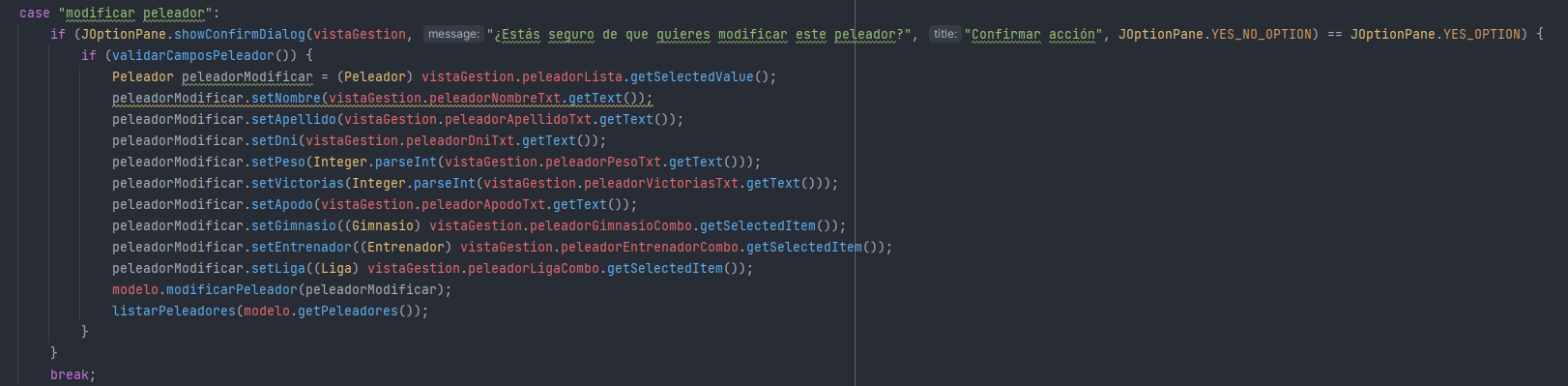
A la hora de insertar cada uno de los elementos, Creamos el objeto, lo rellenamos con los campos que se rellenan en la interfaz grafica.

Este proceso comprueba si hay algun campo vacio u erroneo para corregirlo antes de la insercion para evitar errores, una vez rellenado, le damos al boton de insercion y genera una ventana para verificar la accion, si la respuesta es si, se insertara.

****

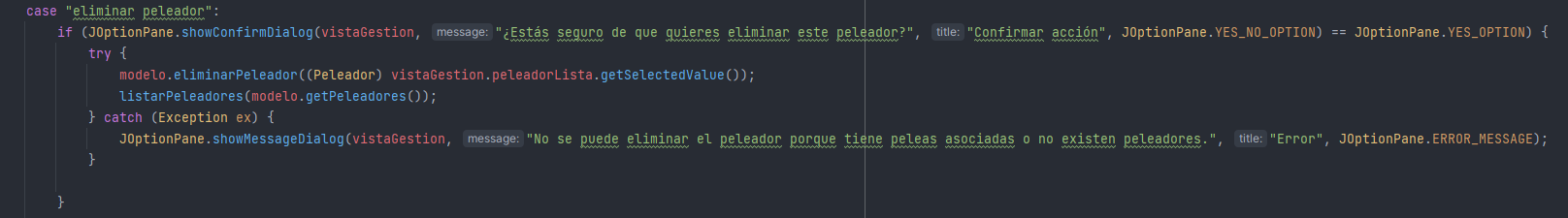
* **Modificacion**

A la hora de la modificacion el usuario necesita seleccionar uno de los objetos de la lista, tiene un control de errores para evitar problemas en la ejecucion, el usuario modifica los datos y una vez clicado en modificacion, saldra una ventana emergente que confirmara la accion



* **Eliminacion**

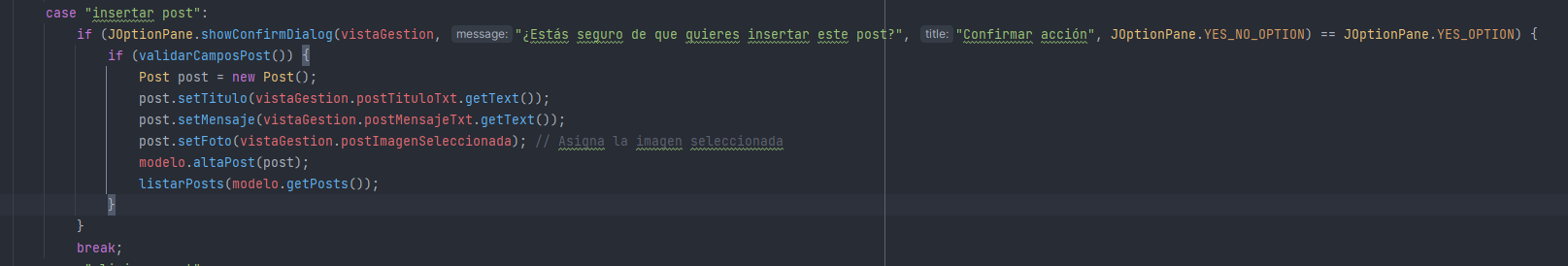
A la hora de eliminar una columna, En cada uno hay que seleccionar en la lista y darle a eliminar, saldra una ventana emergente que buscara confirmar la accion, si la respuesta es si, la elimina



Para insertar un post por lo general es necesaria una imagen, el metodo adjuntar imagen, nos permite seleccionar una imagen filtrada gracias a JFileChooser, una vez seleccionada, redimensionamos la imagen para que corresponda con las medidas de la interfaz, la almacenamos como un array de bytes en formato png y estara lista para la insercion

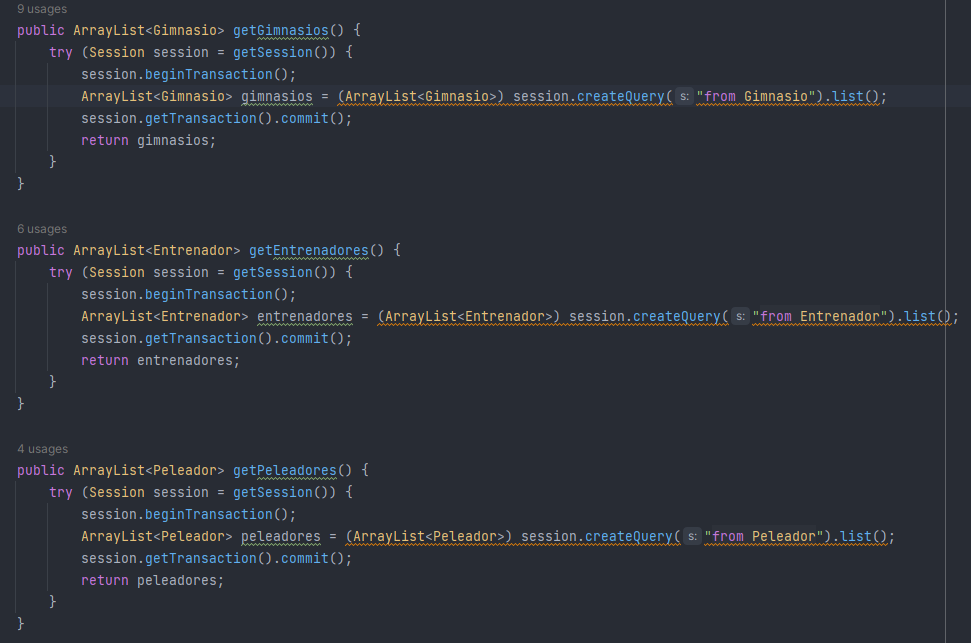


Para insertar un post, creamos el objeto post, lo rellenamos con los campos de la interfaz y la imagen seleccionada, a la hora de insertar, aparecera la ventana de confirmacion y si la respuesta es positiva se insertara



#### Obtencion de datos

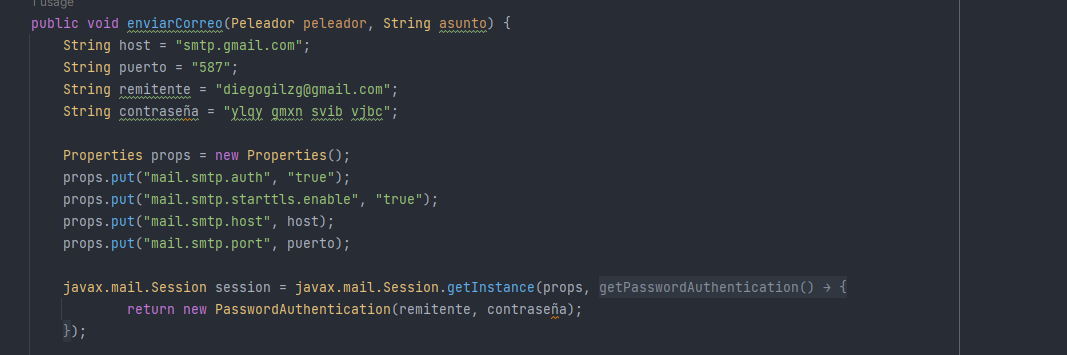
Almacenaremos todos los datos obtenidos en Arraylist según la tabla de la que provengan.



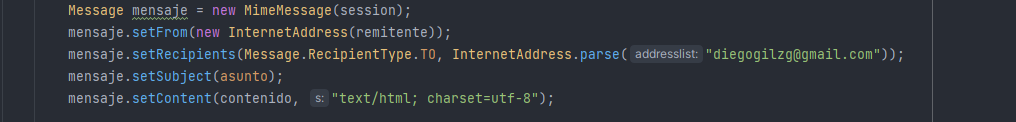


#### Envio de correo

Gracias a la librería javax-mail puedo enviar un mensaje a un correo electrónico, creando una sesión y el mensaje de una forma muy sencilla. Solo hay que indicar el emisor, datos de la cuenta emisora, correo electrónico receptor, rellenar el mensaje por apartador y mandarlo con transport.







# 5. Pruebas e informes de la aplicación

### 5.1.- Estrategia de pruebas

Realizar diferentes situaciones y apuntar en un Excel los resultados

### 5.2.- Pruebas realizadas

La aplicación lleva a cabo 2 tipos de pruebas.

* Pruebas de usabilidad:  
  Mediante un cuestionario que recoge feedback del usuario, el cual ha podido probar/ver la aplicación, puedo verificar que la aplicación es sencilla e intuitiva para que el usuario tenga la mejor experiencia
* Pruebas de caja negra( Pruebas negativas):

He creado un Excel donde indica errores ante situaciones donde hay diferentes entradas de datos que están mal o simplemente que no se realiza, simulando posibles situaciones

### 5.3.- Informes

Texto

# 6. Implantación y documentación

### 6.1.- Exportación y distribución de la aplicación

Texto

### 6.2.- Manual de instalación

Texto

### 6.3.- Manual de usuario

Texto

### 6.4.- Repositorio de la aplicación

Texto

### 6.5.- JavaDoc

Texto

# 7.- Bibliografía

Librería hash

https://commons.apache.org/proper/commons-codec/download\_codec.cgi

# 8.- Anexos

Texto