



# PROYECTO INTEGRADOR

AdraKode

Rubén Peña, Daniel Correa,  
Ainhoa Blanca, Adrián Arcones

## ÍNDICE

ÍNDICE.....	1
LINKS.....	1
Análisis y diseño de los requisitos Hardware y Software .....	2
Software .....	2
? Java .....	2
? Eclipse.....	2
? Git.....	2
? MySQL .....	2
Hardware.....	3
Planificación general del proyecto. ....	4
Prototipo interfaz gráfica.....	5
Diseño de la interfaz de las ventanas. ....	6
Creación de las clases pertenecientes a la Vista. ....	9
Generación del Modelo Relacional. Normalización .....	9
Creación de la BBDD: tablas, índices, etc .....	9
Inserción de los datos necesarios para la aplicación.....	10
Diagrama de casos de uso. ....	10
Diseño del logo.....	11

## LINKS

<https://github.com/ruben170305/AdraKode>

<https://trello.com/b/JpEzRMrt/adrakode>

<https://github.com/ruben170305/AdraKode.wiki.git>

## Análisis y diseño de los requisitos Hardware y Software

Antes de empezar a diseñar y construir el sistema, necesitaremos especificar el diseño y requisitos del propio programa. En este apartado, separaremos los detalles en dos partes: Hardware y Software.

### Software

Durante el desarrollo del sistema pedido por el Club de Rol de la UEM, utilizaremos varias herramientas que nos permitirán avanzar en el proyecto sin limitaciones y/o problemas.

- **Java**

El lenguaje de programación que usaremos para codificar la aplicación será Java.

Este lenguaje orientado a objetos nos permitirá implementar un sistema CRUD para poder realizar consultas SQL desde la propia aplicación de Java. Esto será útil ya que podremos realizar validaciones en las consultas si ocurre algún error en estas.

Además, utilizaremos un modelo de arquitectura MVC (Modelos, Vistas y Controladores) para separar las responsabilidades de los modelos lógicos, mejorando la depuración y la organización del proyecto.

- **Eclipse**

Utilizaremos Eclipse para implementar las clases encargadas de la interfaz gráfica, así como las clases de la lógica de la aplicación.

Además, nos servirá para conectar la aplicación con la base de datos, en este caso MySQL, mediante el sistema CRUD hecho en Java.

Tendremos en cuenta la jerarquía en el diseño, de tal forma que sean fácilmente identificables los elementos principales de la misma, así como el orden de ejecución.

Respecto a la interfaz gráfica, implementaremos una disposición clara de los contenidos, evitando interfaces sobrecargadas (una disposición de elementos limpia).

- **Git**

Realizaremos el análisis y el diseño de la aplicación empleando técnicas UML.

Crearemos un repositorio de GitHub donde controlaremos las versiones de desarrollo del proyecto.

Gestionaremos las diferentes versiones del software y el trabajo colaborativo, además realizaremos pruebas de testeo sobre los programas.

- **MySQL**

Utilizaremos MySQL para crear la base de datos, definiendo así su estructura y las características de sus elementos según el modelo relacional que se está ejecutando a la vez que este documento.

Diseñaremos modelos lógicos normalizados interpretando diagramas entidad/relación, realizaremos el diseño físico de bases de datos utilizando asistentes, herramientas

gráficas y el lenguaje de definición de datos (SQL), y por último consultaremos y modificaremos la información almacenada.

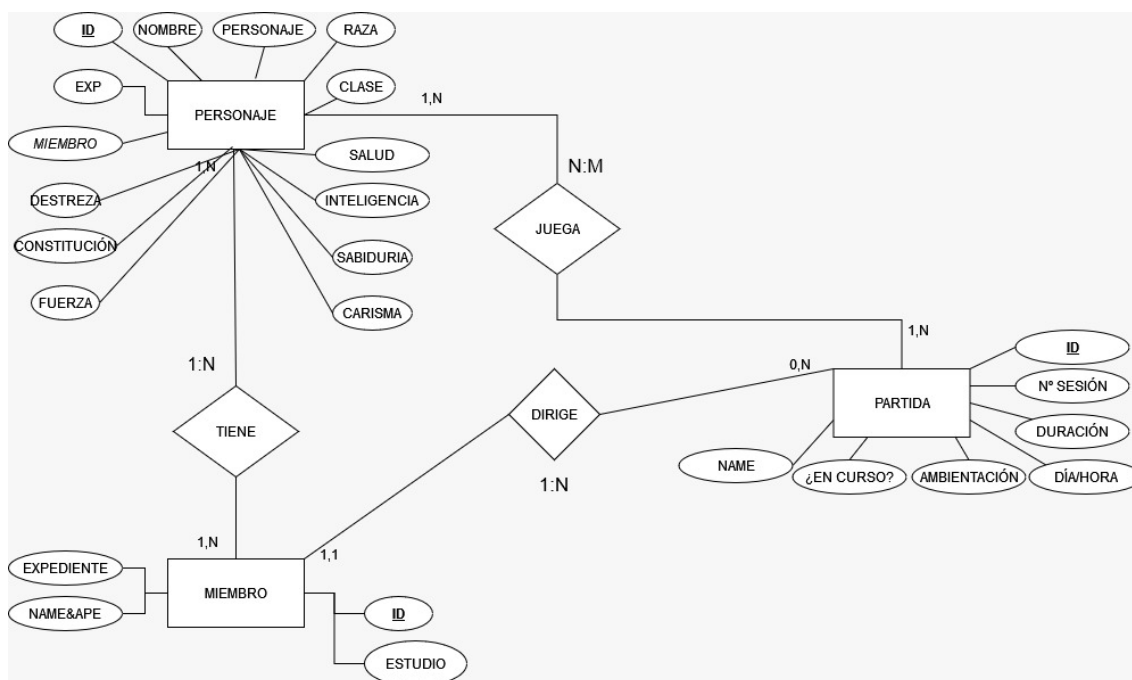
## Hardware

En la parte del Hardware, dividiremos los requisitos entre mínimos y recomendados. Estos requisitos están dirigidos a los desarrolladores de la aplicación, pero también se aplican para los usuarios de esta.

Especificación	Requisitos Mínimos	Requisitos Recomendados
<b>Procesador (CPU)</b>	Intel Core i3 o AMD Ryzen 3, 2.0 GHz o superior	Intel Core i5 o AMD Ryzen 5, 2.5 GHz o superior
<b>Memoria RAM</b>	4 GB	8 GB o más
<b>Espacio en Disco Duro</b>	500 MB de espacio disponible	1 GB de espacio libre
<b>Conexión a Internet</b>	10 Mbps	20 Mbps o superior
<b>Sistema Operativo</b>	Windows 10, macOS Mojave, Ubuntu 20.04	Windows 10 o posterior, macOS Catalina o posterior, Ubuntu 20.04 LTS o posterior
<b>Resolución de Pantalla</b>	1280 x 720 píxeles	1920 x 1080 píxeles o superior
<b>Periféricos</b>	Teclado y ratón	Teclado y ratón

A continuación, se enseña una captura de la información que manejaremos en nuestro proyecto.

En ella aparecen los miembros del club con sus diferentes atributos, los personajes de las partidas y la información necesaria sobre ellas, con unos requisitos mínimos para la interfaz.



## Planificación general del proyecto.

Para diseñar nuestra interfaz gráfica, crearemos tres clases las cuales se llamarán Personajes, Partidas y Miembros del club para asignarlas a cada pestaña de la aplicación.

Los requisitos mínimos para la interfaz deberán ser los siguientes:

- Crearemos una pantalla de login con la que los miembros podrán acceder con usuario y contraseña.
- A la aplicación podrá ser usada por cualquier miembro, pero estos tendrán diferentes permisos si son jugadores o Game Master. Estos se diferenciarán al entrar en el juego.
- Habrá un menú con las opciones de consulta, alta, baja, y modificación para cada opción, dependiendo de quién esté usando el programa.
- El jugador podrá consultar cualquier partida, pero no modificarla
- El jugador podrá editar y borrar sus propios personajes, crear personajes nuevos y consultar cualquier personaje en el sistema.
- El Game Master podrá consultar cualquier partida, y editar aquellas que esté dirigiendo. También podrá crear partidas nuevas (que se crearán con 0 sesiones por defecto, y marcadas como “en curso”)

Dentro de la clase personajes habrá cuatro métodos que serán los siguientes:

- Nuevo personaje: Servirá para añadir los datos de los personajes que vayan a participar.
- Ver personajes: Para poder ver los personajes que han sido añadidos.
- Modificar personaje: Si algún personaje está mal introducido, este se podrá modificar.
- Borrar personaje: En el caso de que se quiera eliminar un personaje, este método nos lo permitirá.

Dentro de la clase partida crearemos los siguientes métodos:

- Buscar partida: Servirá para una vez introducido los participantes que dé comienzo al juego.
- Resultado de la partida: Se podrá observar los resultados que han obtenido los participantes.

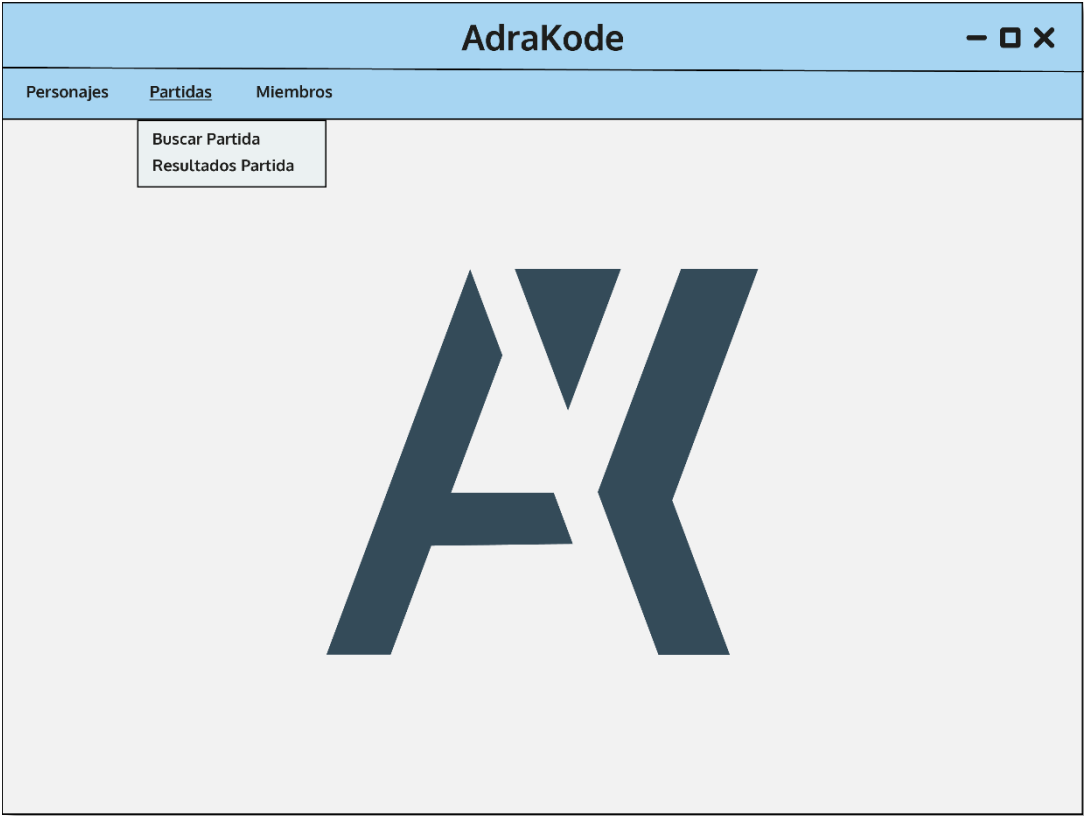
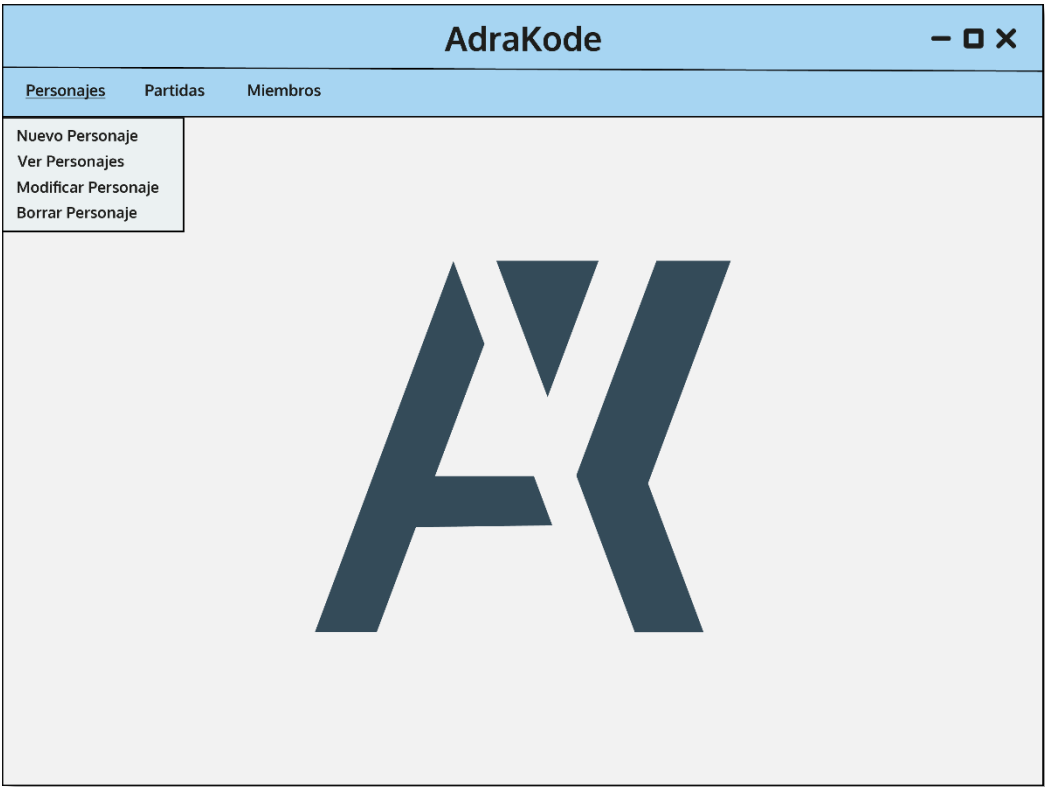
Cada uno de los personajes que se metan, tendrán un identificador único.

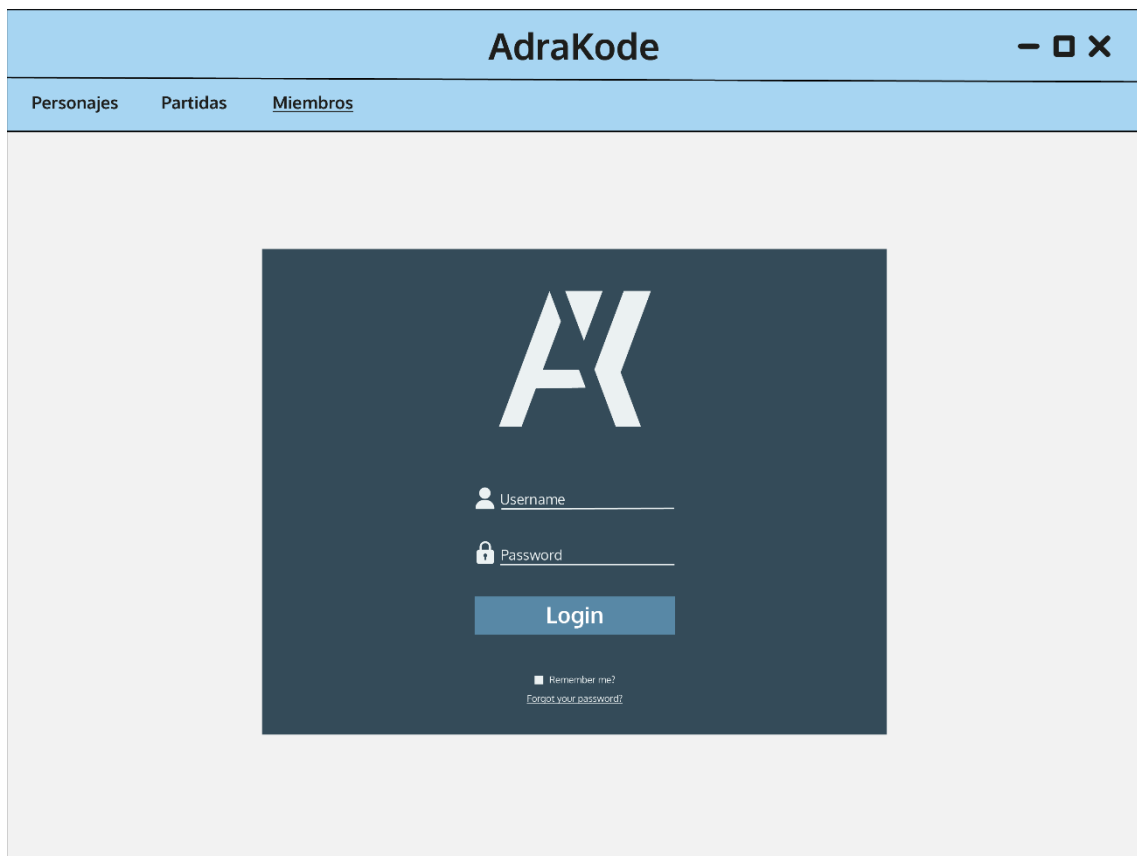
Dentro de la clase miembro crearemos los siguientes atributos:

- Número de expediente
- Nombre y apellidos
- Salud
- Estudios

Cada uno de los miembros que se metan, tendrán un identificador único.

Prototipo interfaz gráfica.






Diseño de la interfaz de las ventanas.



## EDITAR PERSONAJE

Seleccione personaje: Jugador 1 

Raza: \_\_\_\_\_  
Experiencia: \_\_\_\_\_  
Fuerza: \_\_\_\_\_  
Destreza: \_\_\_\_\_  
Constitución: \_\_\_\_\_  
Inteligencia: \_\_\_\_\_  
Sabiduría: \_\_\_\_\_  
Carisma: \_\_\_\_\_

GUARDAR

ELIMINAR

## VER PERSONAJES

### Personaje 1

Raza 1

Experiencia 0 %  
Fuerza 0 %  
Destreza 0 %  
Constitución 0 %  
Inteligencia 0 %  
Sabiduría 0 %  
Carisma 0 %

<

>



SELECCIONAR



# EDITAR PARTIDAS

\*Seleccione partida: [Partida 1](#) v

## PARTIDA 1 [editar](#)






-  Anfitrión [Usuario 1](#) [editar](#)
-  Jugadores [3](#) [editar](#)
-  Duración [15'](#) [editar](#)
-  Fecha [16-04 | 15:00 pm](#) [editar](#)
-  Estado [En espera](#) [editar](#)

GUARDAR

ELIMINAR






# VER PARTIDAS

## PARTIDA 1

-  Anfitrión [Usuario 1](#)
-  Jugadores [4](#)
-  Duración [30'](#)
-  Fecha [13-04 | 16:00 pm](#)
-  Estado [En curso](#)






JUGAR

## PARTIDA 2

-  Anfitrión [Usuario 2](#)
-  Jugadores [3](#)
-  Duración [15'](#)
-  Fecha [16-04 | 15:00 pm](#)
-  Estado [En espera](#)

JUGAR

## PARTIDA 3

-  Anfitrión [Usuario 5](#)
-  Jugadores [2](#)
-  Duración [25'](#)
-  Fecha [12-04 | 11:00 am](#)
-  Estado [Finalizada](#)

JUGAR

## Creación de las clases pertenecientes a la Vista.

- Para el inicio de sesión se han creado dos clases una llamada login y la otra ListenerBotonLogin en las cuales se mostrarán las diferentes características que tendrá para que la vista funcione como lo esperado.
- Para crear personajes y editar personaje solo se ha creado una clase ya que todavía no tiene nada relacionado.
- En el menú se han creado tres clases llamadas PMenuListener, MenuMain y Menu. Hay que tener en cuenta que desde la clase MenuMain es donde se podrán ejecutar el resto de las clases mencionadas anteriormente.
- Por último, se han creado diferentes clases las cuales se usarán para hacer consultas SQL desde Java.

## Generación del Modelo Relacional. Normalización

Hemos pasado el modelo entidad relación al modelo relacional estableciendo las tablas correspondientes a las entidades y las relaciones que lo necesiten.

Además, hemos comprobado que nuestra base está normalizada porque no tiene atributos multivaluados.

MIEMBRO			
ID	EXPEDIENTE	NAME&APE	ESTUDIO

PERSONAJE											
ID	NOMBRE	PERSONAJES	RAZA	EXP	CLASE	DESTREZA	SALUD	CONSTITUCIÓN	SABIDUR	FUERZA	CARISMA

PARTIDA							
ID	NAME	¿EN CURSO?	AMBIENTACIÓN	DÍA/HORA	DURACIÓN	Nº SESIÓN	ID MIEMBRO

JUEGA	
ID PERSONAJE	ID PARTIDA

TIENE	
ID MIEMBRO	ID PERSONAJE

## Creación de la BBDD: tablas, índices, etc

Para la creación de la base de datos se han creado 5 tablas que son las siguientes:

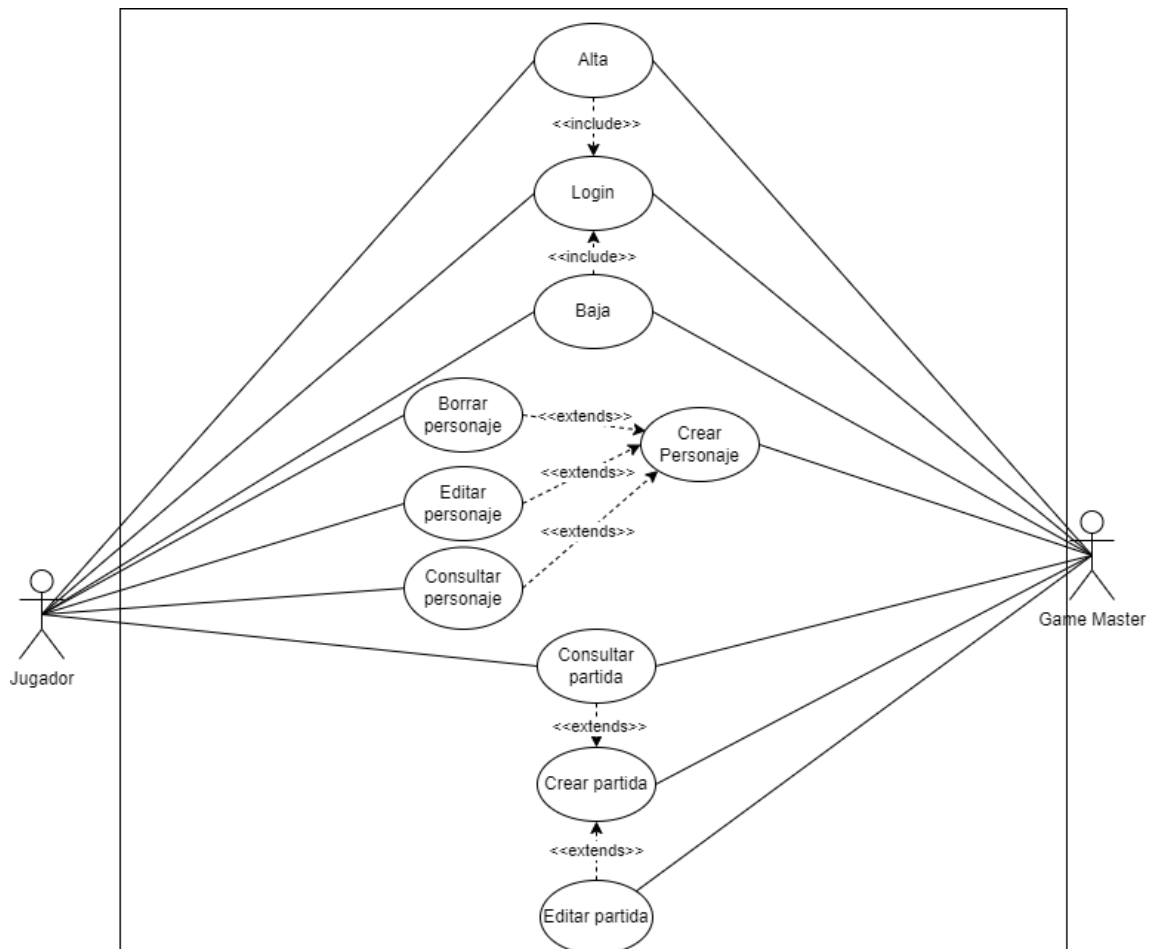
- **Personaje** que contiene: cod, nombre, personaje, raza, clase y expe.
- **Miembro** que contiene: cod, fullname, expediente y estudio.
- **Partida** que contiene: cod, num\_sesion, nombre, dia\_duracion, ambientación, enCurso y controlador. Además, contiene una foreign key con miembro.
- **Utiliza\_personaje** que contiene: id\_miembro e id\_personaje ambas son primary key y además contiene dos foreign key que son id\_miembro para miembro e id\_personaje para personaje.
- **Juega** que contiene: id\_partida, id\_personaje, fuerza, destreza, constitución, inteligencia, sabiduría y carisma. Además, contiene primary key id\_partida y id\_personaje y dos foreign key que son id\_partida para partida e id\_personaje para personaje.

## Inserción de los datos necesarios para la aplicación.

Tenemos las tablas ya creadas a nuestro script y vamos a guardar datos en estas tablas usando inserts para posteriormente cuando necesitemos esa información podamos acceder a ella

## Diagrama de casos de uso.

Para el diseño de nuestro diagrama de casos de uso nos enfocamos en cómo sería nuestra interfaz. De esta forma añadimos para el login inclusiones que consideramos necesarias para entrar al juego y por otro lado también añadimos varias extensiones para crear personaje y crear partida. También se puede apreciar que hay dos actores jugador y game master con sus respectivas operaciones.

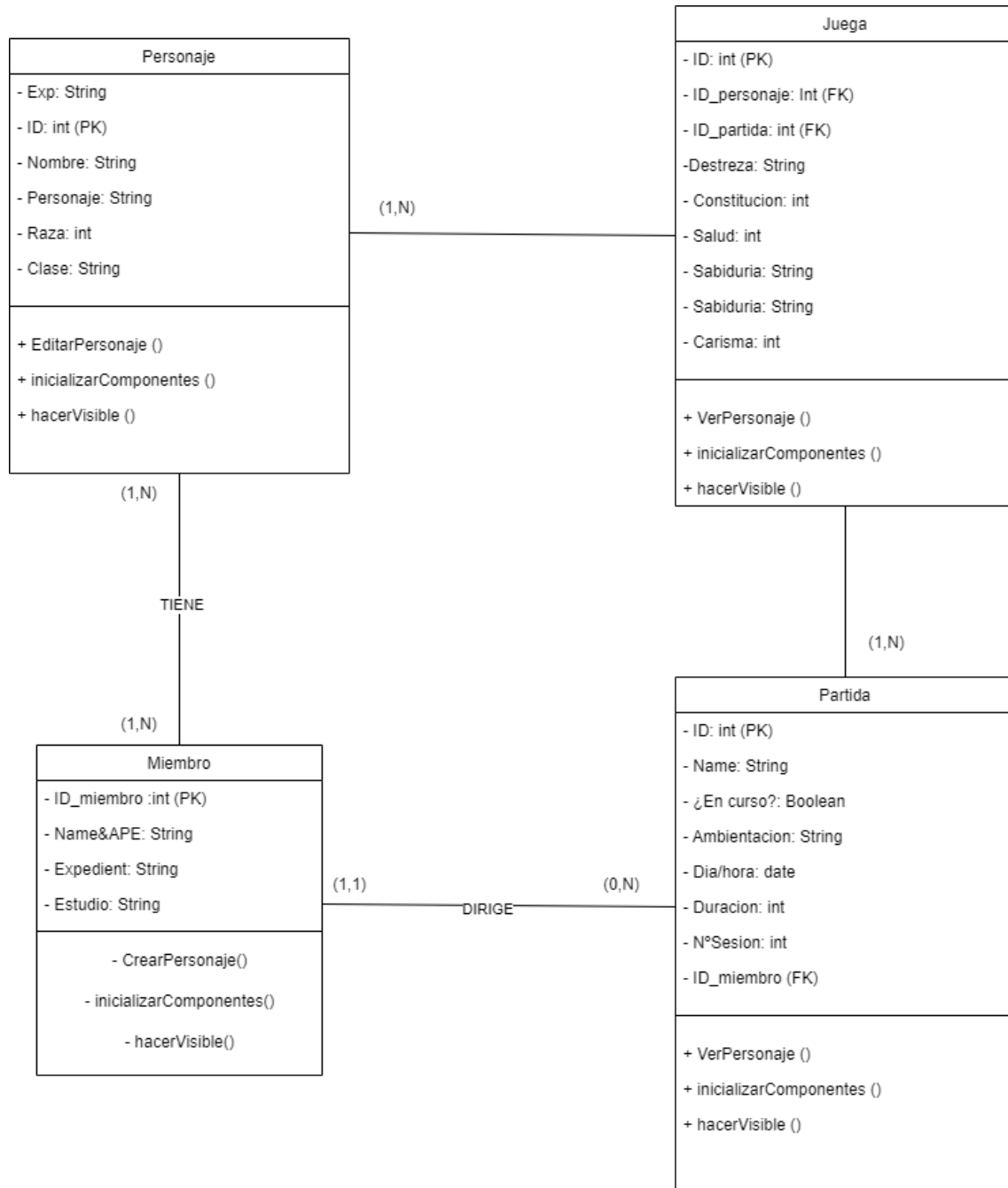


## Diseño del logo.

Para el diseño del logo optamos por una paleta de colores azules combinándolo con los colores neutros blanco y negro, de esta forma hicimos un rediseño de nuestro logo haciendo un contraste de colores que le daba carisma y elegancia.



## Generación del diagrama de clases.



A partir del diagrama de clases:

- Creación de las clases pertenecientes al Modelo

#### MODEL:

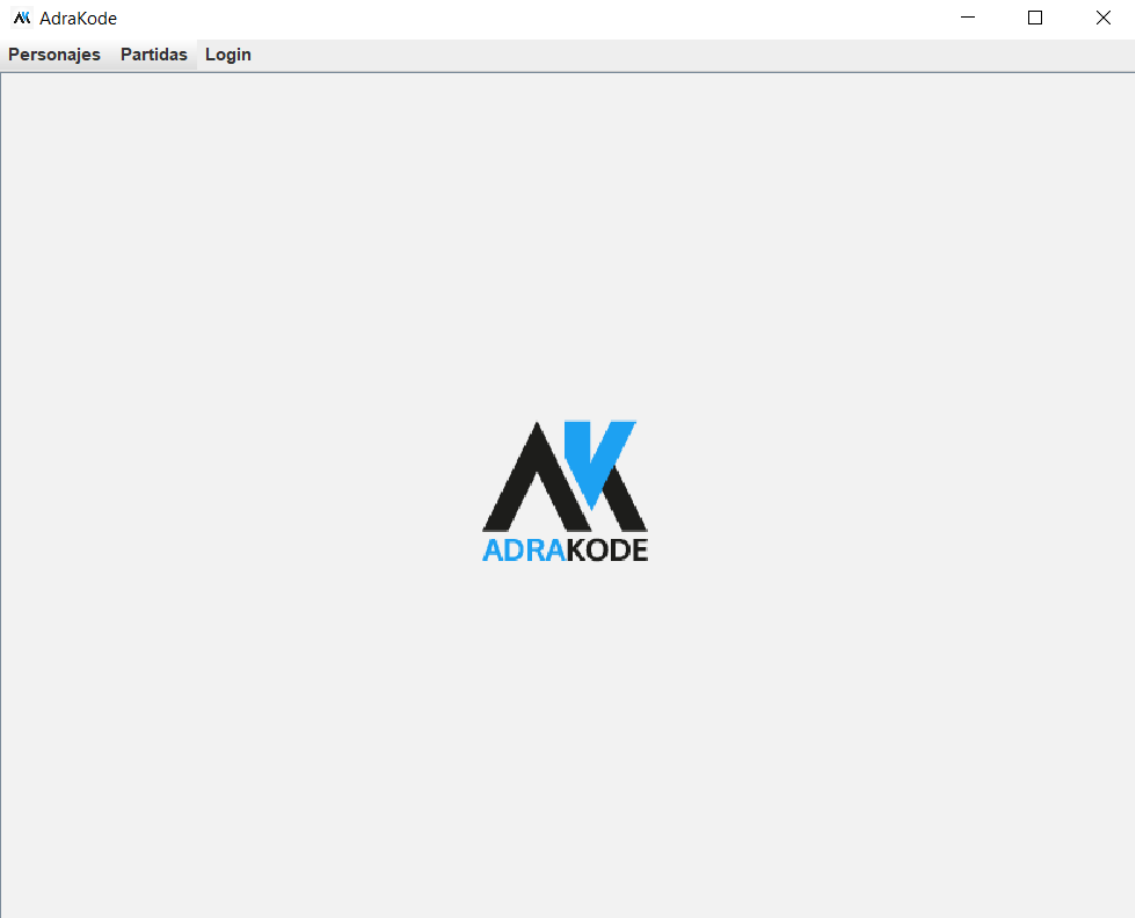
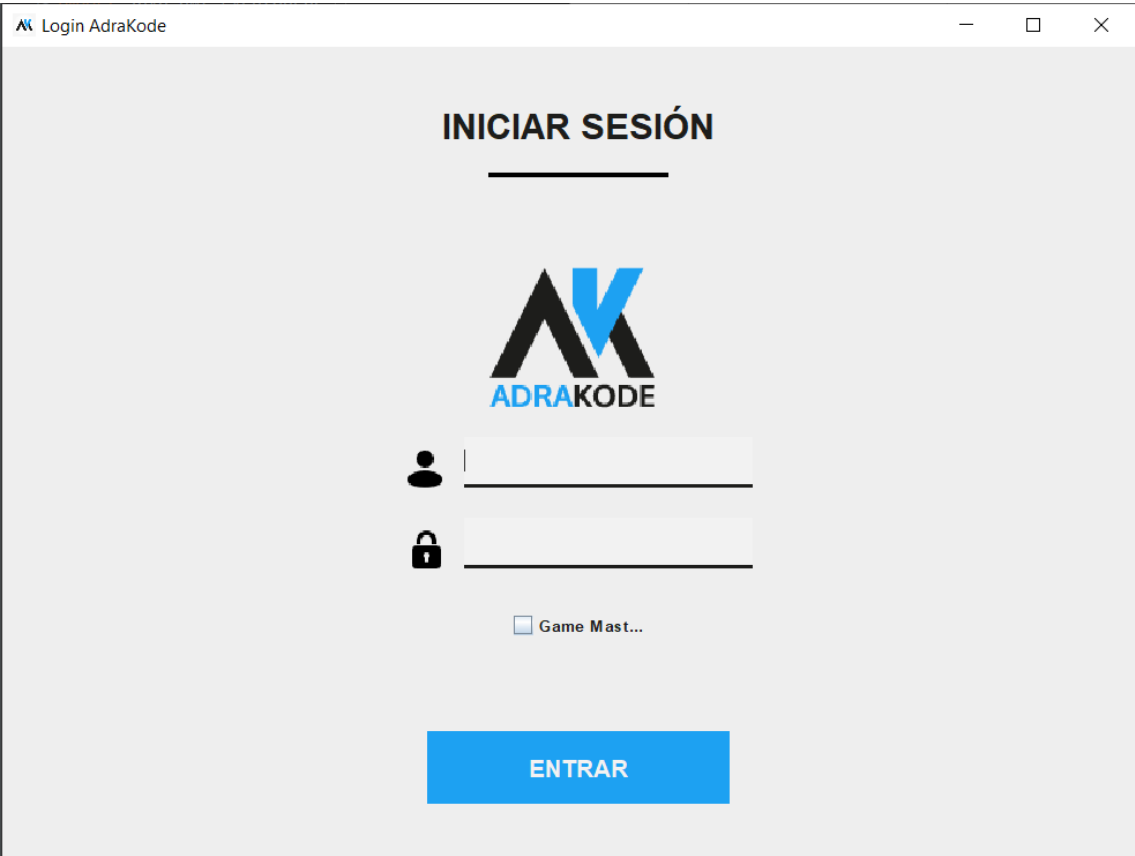
- Delete: Elimina cualquier dato de la base de datos.
- Insert: Inserta cualquier dato a la base de datos.
- Model: Conexión a la base de datos MYSQL la cual tiene atributos que almacenan la información de la conexión y el método `Model_mysql_connect()` para realizar la conexión con la base de datos.
- Query: Clase que sirve para hacer consultas a la base de datos.
- Select: Representa consultas a la base de datos. Tiene un atributo `database_name` que almacena el nombre y dos métodos `create_sql()` y `exec_sql()` los cuales devolverán datos.
- Update: Se utilizará para posibles actualizaciones en la base de datos.

#### LISTENERS:

- EditarPersonajeListener: Botón para editar el personaje
- ListenerBottonLogin: Botón para iniciar sesión en el login.
- LoginListener: clase del listener para comprobar si es correcto. Si lo es, cerrará la ventana de login y mostrará la ventana principal (menú)
- PMenuListener: clase en la cual se introducen la acción de las demás clases para cambiar la ventana según la opción que clickemos.
- VerPersonajesListener: Botón de editar personaje. Redigiremos a la ventana dependiendo del botón.

- Creación de las clases pertenecientes al Control encargadas del acceso a BBDD y del manejo de la aplicación.

Modificaciones de los anteriores sprint



# VER PARTIDAS

## Partida 1



- Anfitrión 1
- 4
- 30'
- 13-04 16:00 pm
- En curso

ID	Anfitrión	Jugadores	Duración	Fecha	Estado	
Partida 1	Usuario 1	4	30'	13-04 16:00 pm	En curso	^
Partida 2	Usuario 2	3	15'	16-04 15:00 pm	En espera	
Partida 3	Usuario 5	2	25'	12-04 11:00 am	Finalizada	
Partida 2	Usuario 2	3	15'	16-04 15:00 pm	En espera	
Partida 3	Usuario 5	2	25'	12-04 11:00 am	Finalizada	
Partida 2	Usuario 2	3	15'	16-04 15:00 pm	En espera	v

JUGAR

# VER PARTIDAS

## Partida 1



- Anfitrión 1
- 4
- 30'
- 13-04 16:00 pm
- En curso

ID	Anfitrión	Jugadores	Duración	Fecha	Estado	
Partida 1	Usuario 1	4	30'	13-04 16:00 pm	En curso	^
Partida 2	Usuario 2	3	15'	16-04 15:00 pm	En espera	
Partida 3	Usuario 5	2	25'	12-04 11:00 am	Finalizada	
Partida 2	Usuario 2	3	15'	16-04 15:00 pm	En espera	
Partida 3	Usuario 5	2	25'	12-04 11:00 am	Finalizada	
Partida 2	Usuario 2	3	15'	16-04 15:00 pm	En espera	v


JUGAR





# EDITAR PARTIDA


Partida: \_\_\_\_\_




 Anfitrión \_\_\_\_\_

 Jugadores \_\_\_\_\_

 Duración \_\_\_\_\_

 Fecha \_\_\_\_\_

 Estado \_\_\_\_\_

Jugador	Raza	Clase	
Jugador 1	Raza 1	Clase 1	^
Jugador 2	Raza 2	Clase 2	
Jugador 3	Raza 3	Clase 3	
Jugador 4	Raza 4	Clase 4	
Jugador 5	Raza 5	Clase 5	
Jugador 6	Raza 6	Clase 6	v


GUARDAR





# CREAR PARTIDA


Partida: \_\_\_\_\_




 Anfitrión \_\_\_\_\_

 Jugadores \_\_\_\_\_

 Duración \_\_\_\_\_

 Fecha \_\_\_\_\_

 Estado \_\_\_\_\_

CREAR



## VER PERSONAJES

PERSONAJE Jugador 1



Raza

Clase



Experiencia

0 %



Fuerza

0 %



Destreza

0 %



Constitución

0 %



Inteligencia

0 %



Sabiduría

0 %



Carisma

0 %



SELECCIONAR

## EDITAR PERSONAJE

PERSONAJE 1



Raza \_\_\_\_\_

Clase \_\_\_\_\_



Experiencia

0



Fuerza

0



Destreza

0



Constitución

0



Inteligencia

0



Sabiduría

0



Carisma

0

GUARDAR



## CREAR PERSONAJE

### PERSONAJE 1



Raza \_\_\_\_\_

Clase \_\_\_\_\_



Experiencia



Fuerza



Destreza



Constitución



Inteligencia



Sabiduría



Carisma

CREAR



Realización del manual de usuario en la wiki de GitHub

<https://github.com/ruben170305/AdraKode.wiki.git>