



FrancisGol

FrancisGol

**Especificación de Requisitos
Software**

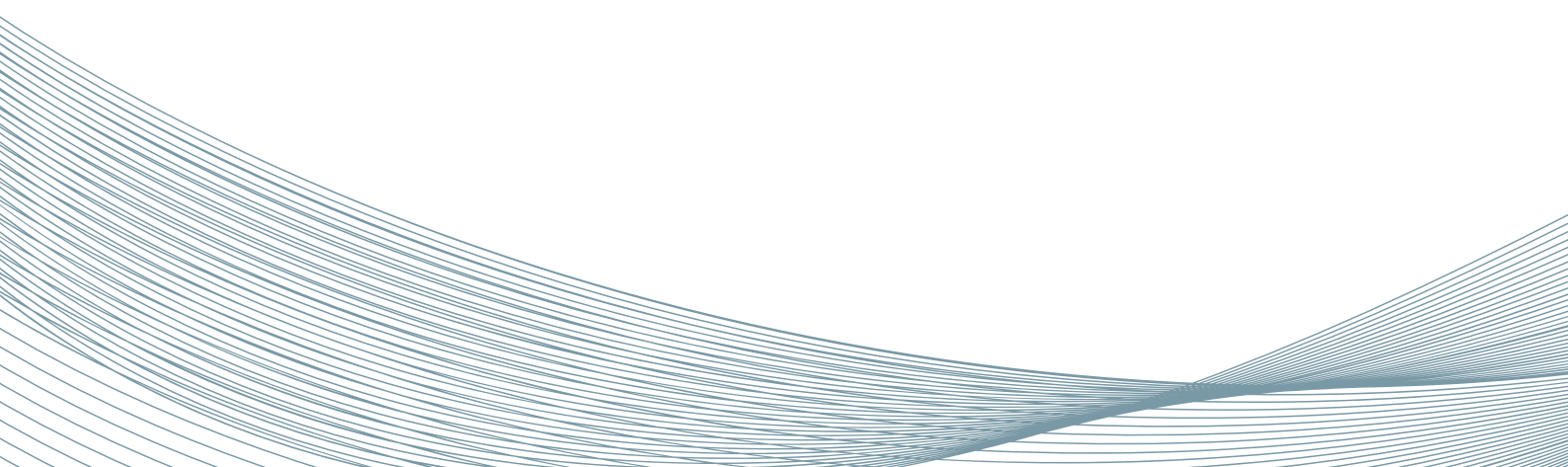
Presentado por:

Francisco Rodríguez Belmonte



índice

1 - Análisis software	1
1.1 - Requisitos funcionales	1
1.2 - Requisitos no funcionales	3
1.3 - Casos de uso	4
2 - Diseño lógico	9
2.1 - Arquitectura MVC	9
2.2 - Modelo Entidad-Relación	12
2.3 - Diagrama de clases	13





Análisis software

Requisitos funcionales

RF-1. Visualización

RF-1.1. Visualizar datos de partidos.

RF-1.1.1. Visualizar estadísticas del partido.

RF-1.1.2. Visualizar resultados.

RF-1.1.3. Visualizar alineaciones.

RF-1.1.4. Visualizar eventos del partido.

RF-1.2 Visualizar competiciones.

RF-1.2.1. Visualizar la clasificación de la competición.

RF-1.2.2. Visualizar las jornadas de la competición.

RF-1.2.3. Visualizar los equipos de la competición.

RF-1.2.4. Visualizar datos de competiciones por año.

RF-1.3. Visualizar fichajes.

RF-1.3.1. Visualizar fichajes de equipos.

RF-1.4. Visualización de favoritos.

RF-1.4.1. Visualizar datos de la competición o equipo favorito.

RF-1.4.2. Visualizar eventos del partido.

RF-1.4.3. Visualizar fichajes de equipos favoritos.



Requisitos funcionales

RF-1.5. Visualizar equipos.

RF-1.5.1. Visualizar la clasificación del equipo por competición.

RF-1.5.2. Visualizar estadísticas del equipo por competición.

RF-1.5.3. Visualizar plantilla del equipo.

RF-1.5.4. Visualizar los últimos fichajes del equipo.

RF-1.5.5. Visualizar datos de del equipo por año.

RF-1.6. Visualización de jugadores.

RF-1.6.1. Visualización de datos del futbolista.

RF-1.6.2. Visualización de estadísticas del futbolista.

RF-1.6.3. Visualizar datos de del futbolista por año.

RF-2. Gestión de favoritos.

RF-2.1. Añadir equipos/competiciones en favoritos.

RF-2.2. Eliminar equipos/competiciones de favoritos.

RF-3. Gestión de plantillas.

RF-3.1. Crear alineación de un equipo.

RF-3.2. Borrar alineaciones propias.

RF-3.3. Editar alineaciones propias.

RF-3.4. Visualizar alineaciones de otros usuarios.

RF-4. Gestión de usuario.

RF-4.1. Crear usuario.

RF-4.2. Editar usuario.

RF-4.3. Borrar usuario.

RF-4.4. Iniciar sesión.

RF-4.5. Cerrar sesión



Requisitos no funcionales

RF-1. Usabilidad

RF-1.1. Mostrará mensajes de error

RF-1.2. Intuitiva y fácil de usar

RF-1.3. Buena legibilidad

RF-1.4. Fácil de navegar entre las distintas páginas

RF-1.5. Capaz de llegar a la misma información desde diferentes puntos

RF-2. Funcionalidad.

RF-2.1. Las plantillas solo se podrán crear con equipos existentes

RF-2.2. Para acceder a un jugador hay que acceder a su equipo

RF-2.3. Para crear plantillas hay que estar registrado

RF-3. Seguridad

RF-3.1. contraseñas de usuarios cifrada

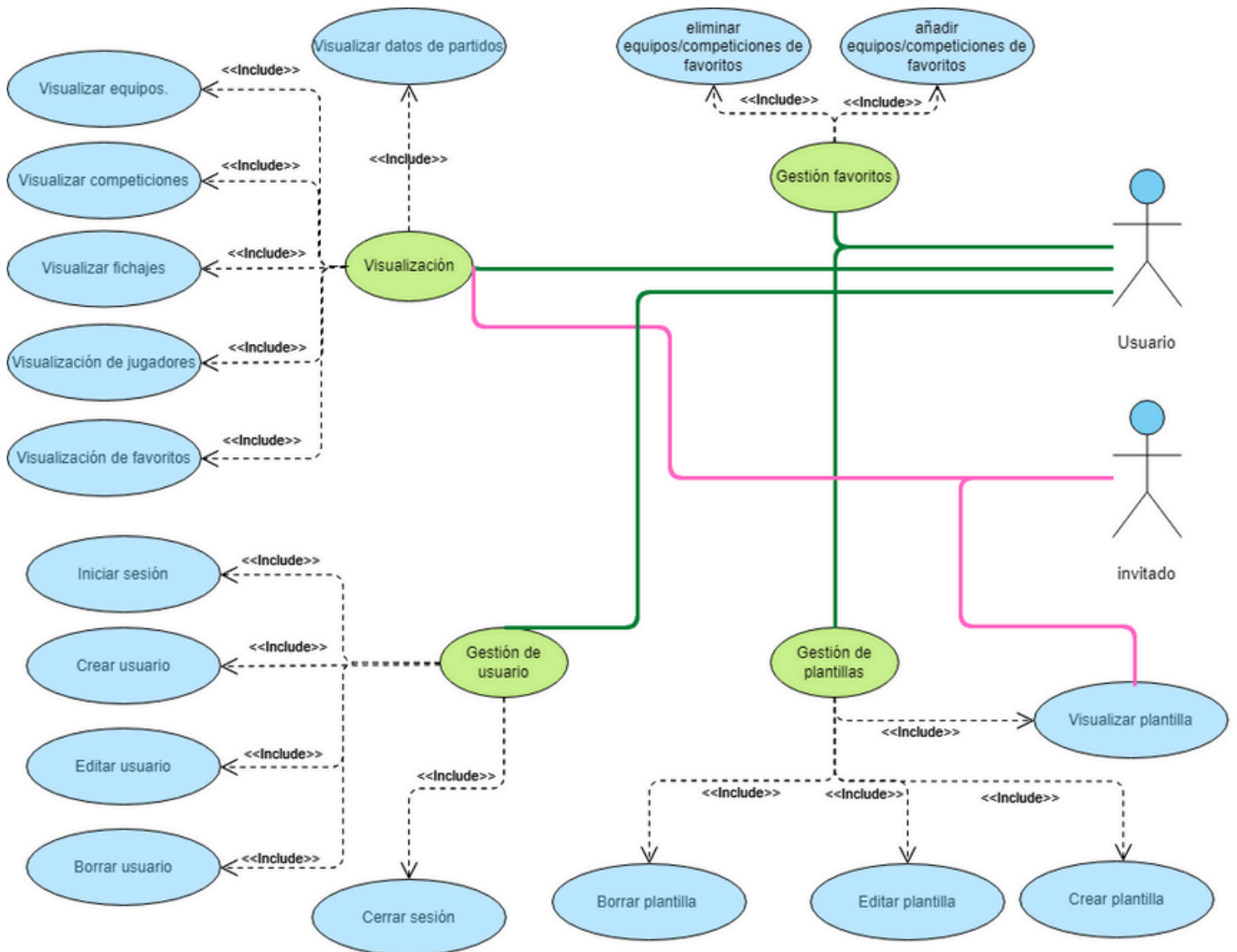
RF-3.1. La página será segura con certificado SSL

RF-4. Disponibilidad

RF-4.1. Se podrá acceder a la página a través de francisgol.com

Casos de uso

Diagrama de casos de uso





Casos de uso

Descripción de los casos de uso

CU-1	Visualización de información de la página
Descripción	Permite visualizar tanto a usuarios como a invitados las competiciones, equipos, plantillas, jugadores y partidos.
Actores	Usuarios e invitados
Precondiciones	Se debe acceder a las secciones de de la página correspondiente a lo que se quiere visualizar
Postcondiciones	Al acceder el usuario o invitado puede ver la información
Escenario principal	<ol style="list-style-type: none">1. Los usuarios o invitados acceden a la página2. Viendo los partidos del momento3. En la parte superior habrá secciones4. Al pulsar en una sección le llevará a la página correspondiente5. Dentro de las secciones habrán más apartados que les permitirá navegar por la página.



Casos de uso

Descripción de los casos de uso

CU-2	Gestión de favoritos
Descripción	El usuario puede guardar en favoritos equipos y competiciones para poder acceder más fácil.
Actor	Usuario
Precondiciones	Se debe registrar e iniciar sesión para poder dar favoritos.
Postcondiciones	Al dar favoritos las secciones serán personalizadas en base a los equipos y competiciones favoritas.
Escenario principal	<ol style="list-style-type: none">1. El usuario se registra.2. Durante el registro selecciona equipos y competiciones favoritas.3. Visita las secciones de partidos/competiciones/fichajes que estarán personalizadas4. Puede seguir añadiendo y quitando equipos y competiciones en favoritos.



Casos de uso

Descripción de los casos de uso

CU-3	Gestión de plantillas
Descripción	El usuario puede crear borrar y editar plantillas de equipos.
Actor	Usuario
Precondiciones	Se debe registrar e iniciar sesión para poder crear, editar y borrar plantillas
Postcondiciones	Al crear una plantilla se vuelve pública para que todo el mundo la vea.
Escenario principal	<ol style="list-style-type: none">1. El usuario se registra.2. Inicia sesión.3. Se dirige a la sección principal de plantillas4. Puede visualizar o crear plantillas.5. Crea una plantilla6. Se publica y ya puede editarla o borrarla

Casos de uso

Descripción de los casos de uso

CU-4	Gestión de usuario
Descripción	El invitado puede crearse una cuenta para obtener privilegios en la aplicación y el usuario puede editar o borrar la cuenta creada
Actor	Usuario e invitado
Precondiciones	Se debe acceder a la página de registro de FrancisGol
Postcondiciones	Al crearse una cuenta se le desbloquea el uso de creación de plantillas y la personalización de información
Escenario principal	<ol style="list-style-type: none"> 1. El invitado accede a la página de registro. 2. Se registra e inicia sesión 3. Como usuario puede acceder al apartado de la cuenta y modificar los datos del usuario o borrar la cuenta

Diseño lógico

Arquitectura MVC

La Arquitectura MVC se basa en 3 partes

- Los controladores: Se encargan de hacer de intermediario entre la vista y el modelo, pide datos a los modelos y devuelve los resultados en las vistas.
- Los modelos: Llevan toda la lógica del programa. es el que hace los cálculos, bucles... Da información al controlador
- Las vistas: Llevan el código HTML y se encarga de mostrar el contenido de la página.

En FrancisGol se va a utilizar este modelo ya que permite tener el código más estructurado.

Lo ideal es ponerle el mismo nombre a la vista, al modelo y al controlador para así saber que archivo pertenece a cada página.

En la próxima página se verá que archivos voy a utilizar para hacer el modelo vista controlador.

Diseño lógico

Arquitectura MVC

Los archivos serán uno por cada página:

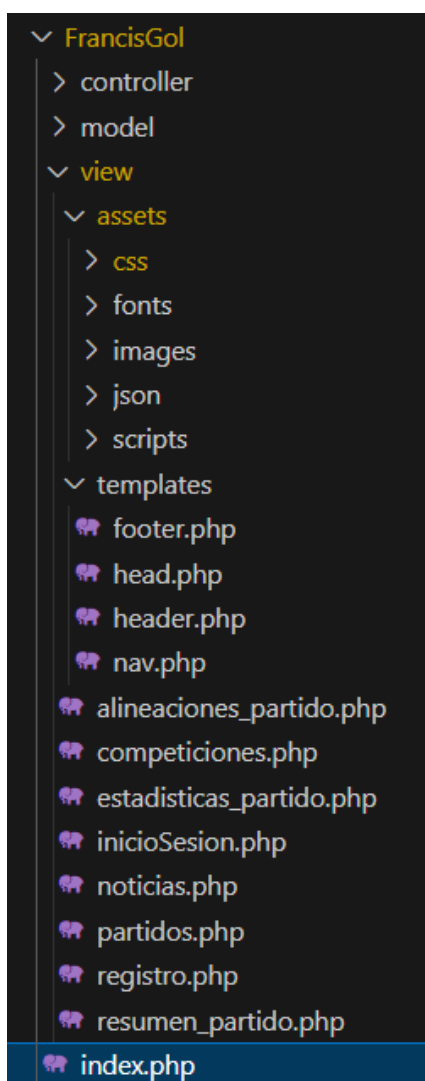
partidos.php
resumen_partido.php
estadísticas_partido.php
alineaciones_partido.php
competiciones.php
clasificacion_competicion.php
jornadas_competicion.php
equipos_competicion.php
fichajes.php
mis_plantillas.php
plantillas_usuarios.php
crear_plantillas.php
inicio_sesion.php
registro.php
competiciones_equipo.php
estadísticas_equipo.php
plantilla_equipo.php
fichajes_equipo.php
datos_jugador.php
estadísticas_jugador.php
cuenta_favoritos.php
cuenta_editar.php

Diseño lógico

Arquitectura MVC

Eso serían las páginas disponibles en la web pero siempre habrán más archivos, como modelos para recoger solicitudes de las APIs o controladores para cerrar sesión o realizar alguna operación como editar o borrar algo.

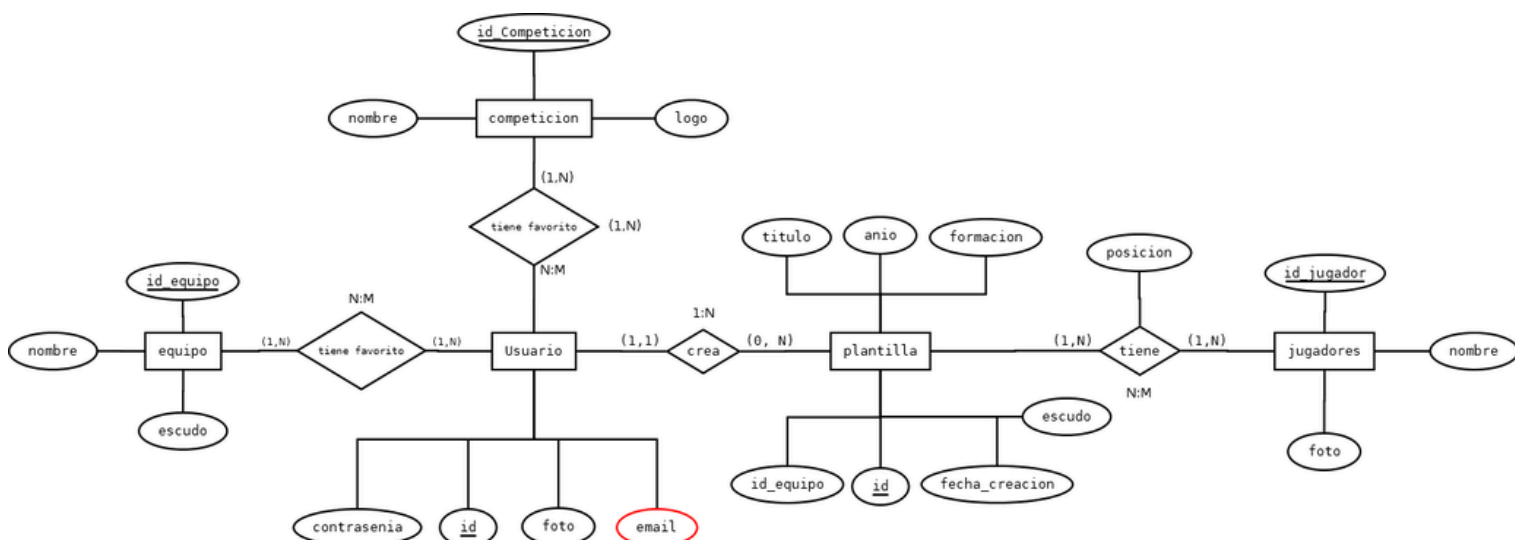
Al terminar quedaría algo así:



Diseño lógico

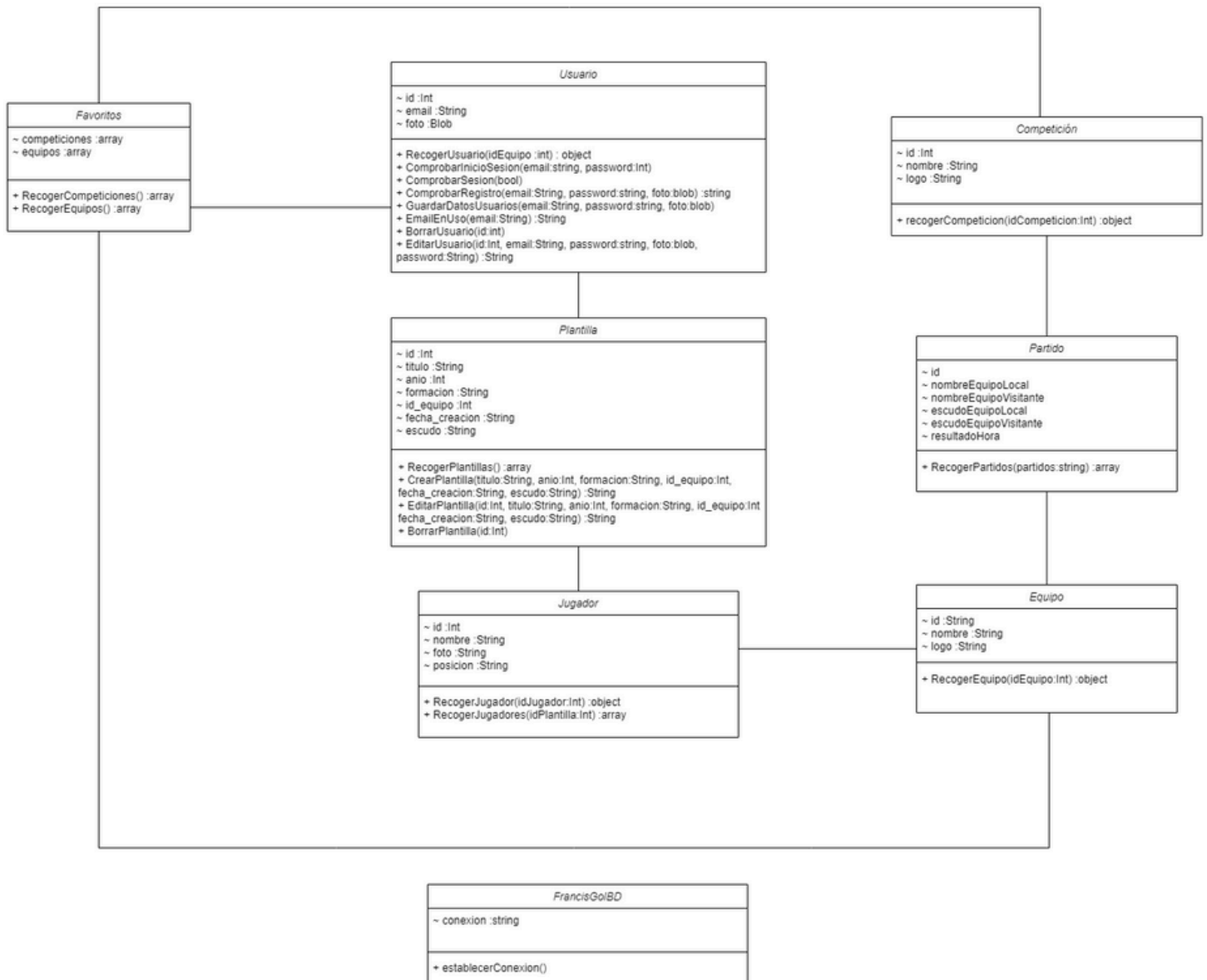
Modelo Entidad-Relación

En la BBDD se va a almacenar lo necesario para ahorrar solicitudes a la API ya que toda la información que llega no se va a almacenar ya que es información que va continuamente cambiando, en el caso de las plantillas la información es estática y en el caso de competiciones o equipos favoritos guardo lo necesario para mostrarlos sin tener que hacer una solicitud a la API.



Diseño lógico

Diagrama de clases





Fin

