

## TP FINAL MATERIA JAVA

### Integrantes:

1- 45332, BERRUTI OCTAVIO

### Enunciado general del TP:

#### ***Administracion de club FIFA Ultimate Team.***

El simulador de futbol online mas grande del mundo es el Ultimate Team de FIFA, donde podes crear tu club, a base de compra y venta de jugadores, o jugadores obtenidos a traves de recompensas, estadios, hinchadas, pelotas, equipaciones y muchas cosas mas.

Ésta aplicacion permitira crear, modificar, consultar, y obtener valiosa informacion sobre tu equipo de Ultimate Team, para que se pueda hacer un seguimiento detallado del mismo.

### Links de utilidad:

**Aplicacion:** <https://java-clases.vercel.app/acc/login>

**Backend heroku:** <https://java-tp-oberruti.herokuapp.com/>

**Credenciales user normal que solo ve su club** - Ingreso con cualquier cuenta de google y crea automaticamente un user con un club

**Credenciales super user modo dios ve y modifica todo:**

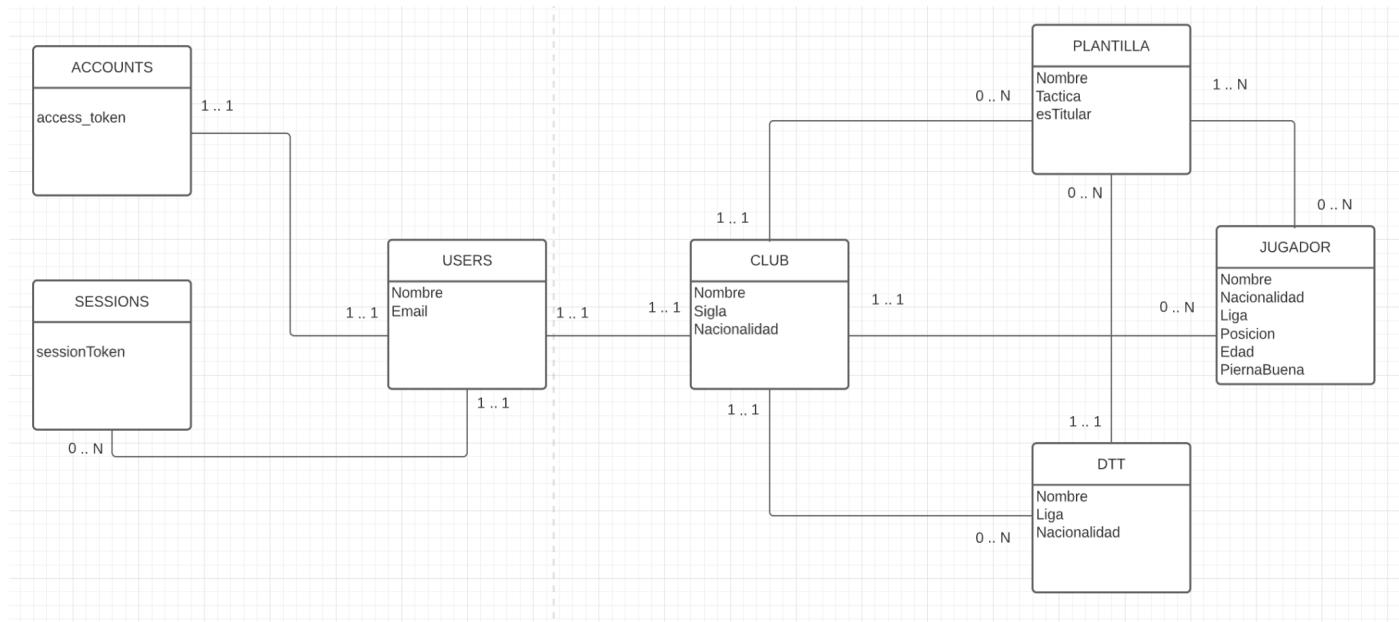
mail: javatpberruti@gmail.com

contraseña: Tpjavaderruti1

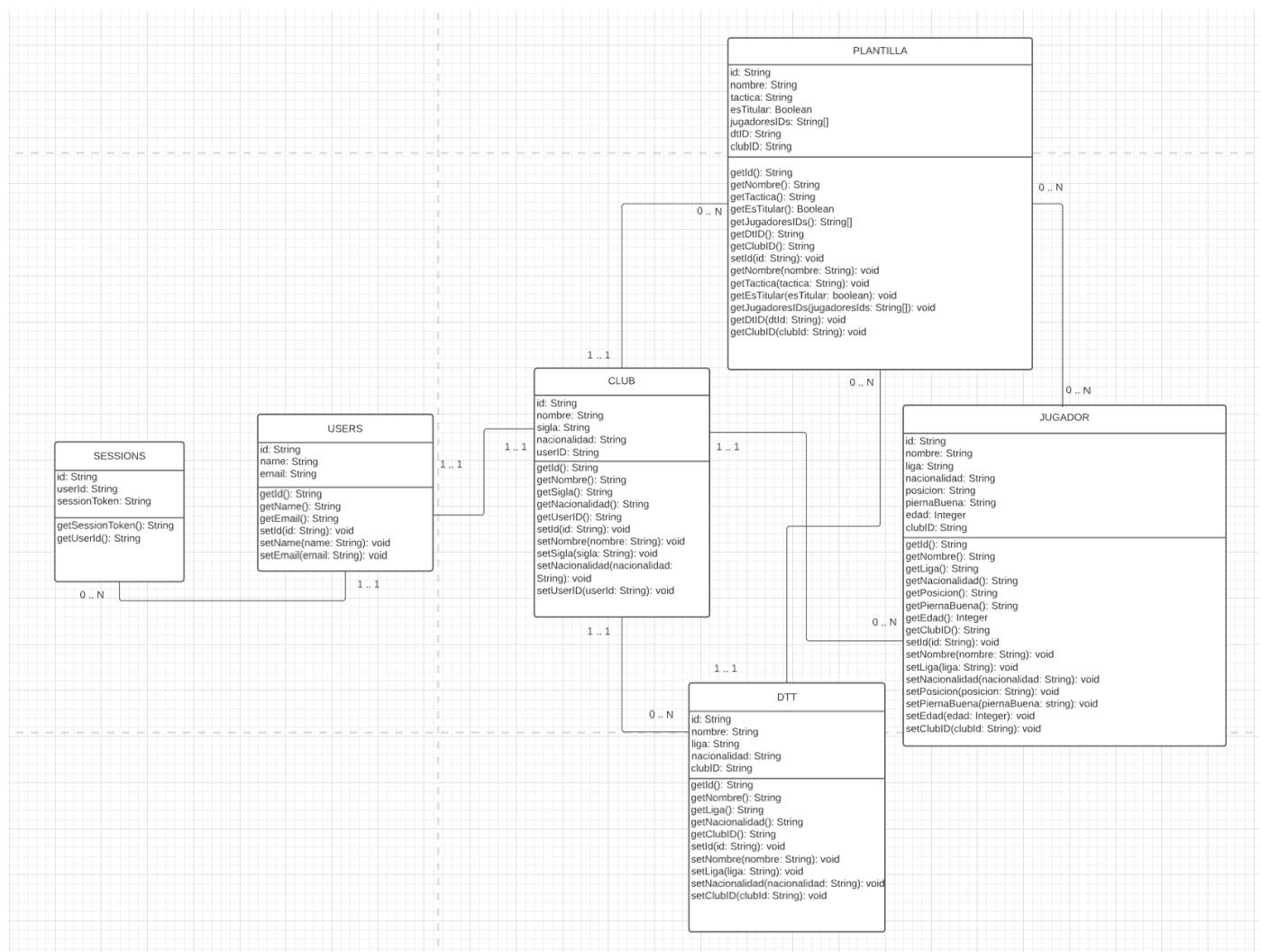
**Repositorio github:** <https://github.com/oberruti/javaClases>

**Codigo del TP final:** <https://github.com/oberruti/javaClases/tree/master/tp>

## Modelo Dominio:



## Diagrama de clases:



## Listado casos de uso:

### **Regularidad**

ABMC simple	Creacion de jugador
	Modificacion de jugador
	Eliminacion de jugador
	Busqueda de jugador
ABMC dependiente	Creacion de plantilla
	Modificacion de plantilla
	Eliminacion de plantilla
	Busqueda de plantilla
CU NO-ABMC	Sugerencia DT ideal por plantilla segun nacionalidad jugadores y DTs
Listado simple	Listado de DTs por nacionalidad
Listado complejo	No Aplica

### **Aprobacion Directa**

ABMC	ABMC Club
	ABMC DT
CU "Complejo" (nivel resumen)	Sugerencia jugador ideal filtrado por plantilla y posicion, basado en las nacionalidades predominantes de la plantilla.
Listado complejo	Jugadores pertenecientes a una plantilla con filtros habilitados
Nivel de acceso	Modo dios(ABMC habilitados para todos los clubes de todos los usuarios)
	Modo user(ABMC habilitado solo para el club del usuario)
Manejo de errores	No requiere detalle
Requerimiento extra obligatorio	No aplica

Publicar el sitio

No requiere detalle

## **CASO DE USO RESUMEN RE-ESTRUCTURADO**

Código y Nombre del CASO DE USO: CURS\_1 Creacion equipo Ultimate Team

#### **Dimensions de clasificación:**

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Resumen	Sin Estructurar	Sistema	Negra	Real	Semántica

**Meta del CASO DE USO: Crear un equipo de Ultimate Team**

**Actor Primario: Usuario**      **Otros:** -

## **PRECONDICIONES (de negocio):**

**PRECONDICIONES (de sistema):** Existe cuenta de google.

**DISPARADOR:** Usuario ingresa a la pagina web del sistema.

## **CAMINO BÁSICO:**

1. Usuario se loguea con su cuenta de google. Sistema valida datos y redirige al Usuario a la pagina principal del mismo.
  2. Usuario se dirige a la seccion “Club”. Sistema valida credenciales y muestra la seccion.
  3. Usuario crea el club. Sistema valida datos, registra y refresca la informacion del club.
  4. Usuario se dirige a la seccion “Jugadores”. Sistema valida credenciales y muestra la seccion.
  5. Usuario crea un jugador. Sistema valida datos, registra y refresca la informacion de los jugadores del club.

**El paso 5 se repite tantas veces como jugadores se deseen crear.**

6. Usuario se dirige a la sección “DTs”. Sistema valida credenciales y muestra la sección.
  7. Usuario crea DT. Sistema valida datos, registra y refresca la información de los DTs del club.

**El paso 7 se repite tantas veces como DTs se deseen crear.**

8. Usuario se dirige a la sección “Plantillas”. Sistema valida credenciales y muestra la sección.
  9. Usuario crea plantilla. Sistema valida datos, registra y refresca la información de las plantillas del club.

**El paso 9 se repite tantas veces como Plantillas se deseen crear.**

## **CAMINOS ALTERNATIVOS:**

## **2.a <Previo> Usuario decide no crear club.**

## **2.a.1 Fin CU.**

**3.a <Previo> Existe club ya creado.**

### **3.a.1. Fin CU.**

**3.a.2. Usuario modifica club. Sistema valida datos, registra y refresca la información del club.**

**3.b <Posterior> Datos ingresados incorrectamente.**

3.b.1. Sistema valida e informa los errores.

3.b.2. Usuario modifica club. Sistema valida datos, registra y refresca la información del club.

**5.a <Previo> Existe jugador ya creado.**

5.a.1. Fin CU.

5.a.2. Usuario modifica jugador. Sistema valida datos, registra y refresca la información de los jugadores.

**5.b <Posterior> Datos ingresados incorrectamente.**

5.b.1. Sistema valida e informa los errores.

5.b.2. Usuario modifica jugador. Sistema valida datos, registra y refresca la información de los jugadores.

**7.a <Previo> Existe DT ya creado.**

7.a.1. Fin CU.

7.a.2. Usuario modifica DT. Sistema valida datos, registra y refresca la información del DT.

**7.b <Posterior> Datos ingresados incorrectamente.**

7.b.1. Sistema valida e informa los errores.

7.b.2. Usuario modifica DT. Sistema valida datos, registra y refresca la información del DT.

**9.a <Previo> Existe plantilla ya creada.**

9.a.1. Fin CU.

9.a.2. Usuario modifica plantilla. Sistema valida datos, registra y refresca la información de la plantilla.

**9.b <Posterior> Datos ingresados incorrectamente.**

9.b.1. Sistema valida e informa los errores.

9.b.2. Usuario modifica plantilla. Sistema valida datos, registra y refresca la información de la plantilla.

**POSTCONDICIONES (de sistema):**

**Éxito:** Club registrado.

**Fracaso:** Club no registrado. Plantilla no registrada. DT no registrado. Jugadores no registrados.

**Éxito alternativo: Plantilla registrada, DT registrado, Club registrado, Jugadores registrados.**

**POSTCONDICIONES (de negocio):**

**Éxito:**

**Fracaso:**

**Éxito alternativo:**

**Reglas de Negocio relacionadas con el casos de uso:**

Paso(s)	Regla de negocio	Tipo
3	Cada club debe tener un nombre, una sigla y una nacionalidad.	
5	Cada jugador debe tener una nacionalidad, una posicion asociada, un nombre, una pierna buena, una edad y una liga.	
7	Cada DT debe tener una nacionalidad, una liga y un nombre.	
9	Cada plantilla tiene que tener solo un DT asociado.	
9	Cada plantilla debe tener un nombre y una tactica asociada.	
9	Solo una plantilla del club puede ser la titular.	

**Código y Nombre del CASO DE USO:** CUU1- Sugerir DT ideal

**Dimensiones de Clasificación:**

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Dialogal

**Meta del CASO DE USO:** Obtener DT ideal para una plantilla.

**ACTORES:**

Primario(s): Usuario

Otros: -

**PRECONDICIONES** (de sistema):

- 1- Cuenta de google creada.
- 2- Plantilla creada.
- 3- DTs creados.
- 4- Jugadores creados.
- 5- Jugadores asociados a la plantilla.

**DISPARADOR:** Usuario se loguea en la aplicacion web.

**FLUJOS (CAMINOS):**

**CAMINO BÁSICO**

1. Usuario se dirige a la seccion “DT IDEAL”
2. El sistema redirige y solicita la seleccion de plantilla.
3. El usuario hace click en el boton desplegable de plantillas.
4. El sistema despliega el listado de plantillas.
5. El usuario hace click en la plantilla seleccionada.
6. El sistema cierra el listado y muestra el DT ideal para la plantilla seleccionada por nacionalidad de DT y Jugadores.

**CAMINOS ALTERNATIVOS**

**POSTCONDICIONES** (de sistema):

Éxito: El sistema muestra el DT ideal para la plantilla seleccionada.

Fracaso:

Éxito Alternativo:

**Código y Nombre del CASO DE USO:** CUU2- Sugerir jugador ideal

**Dimensiones de Clasificación:**

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Dialogal

**Meta del CASO DE USO:** Obtener jugador ideal para una plantilla y una posicion.

**ACTORES:**

Primario(s): Usuario      Otros: -

**PRECONDICIONES** (de sistema):

- 1- Cuenta de google creada.
- 2- Plantilla creada.
- 3- DTs creados.
- 4- Jugadores creados.

**DISPARADOR:** Usuario se loguea en la aplicacion web.

**FLUJOS (CAMINOS):**

*CAMINO BÁSICO*

1. Usuario se dirige a la seccion “JUGADOR IDEAL”
2. El sistema redirige y solicita la seleccion de plantilla y de la posicion.
3. El usuario hace click en el boton desplegable de plantillas.
4. El sistema despliega el listado de plantillas.
5. El usuario hace click en la plantilla seleccionada.
6. El sistema cierra el listado.
7. El usuario hace click en el boton desplegable de posiciones.
8. El sistema despliega el listado de posiciones.
9. El usuario hace click en la posicion seleccionada.
10. El sistema cierra el listado y muestra el jugador ideal para la plantilla y posicion seleccionada.

*CAMINOS ALTERNATIVOS*

**POSTCONDICIONES** (de sistema):

Éxito: El sistema muestra el jugador ideal para la plantilla y posicion seleccionada.

Fracaso: El sistema muestra que no existe jugador ideal para la plantilla y posicion seleccionada.

Éxito Alternativo:

**Código y Nombre del CASO DE USO:** CUU3- Listar DTs por nacionalidad

**Dimensiones de Clasificación:**

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Dialogal

**Meta del CASO DE USO:** Obtener listado de DTs de una nacionalidad.

**ACTORES:**

Primario(s): Usuario

Otros: -

**PRECONDICIONES** (de sistema):

- 1- Cuenta de google creada.
- 2- DTs creados.

**DISPARADOR:** Usuario se loguea en la aplicacion web.

**FLUJOS (CAMINOS):**

*CAMINO BÁSICO*

1. Usuario se dirige a la seccion “LISTADO DTS”
2. El sistema redirige y solicita la seleccion de nacionalidad.
3. El usuario hace click en el boton desplegable de nacionalidades.
4. El sistema despliega el listado de nacionalidades.
5. El usuario hace click en la nacionalidad seleccionada.
6. El sistema cierra el listado y muestra el listado de DTs que tienen la nacionalidad seleccionada por el usuario.

*CAMINOS ALTERNATIVOS*

3.a<durante> El usuario no selecciona nacionalidad.

3.a.1 El sistema muestra el listado de todos los DTs existentes.

**POSTCONDICIONES** (de sistema):

Éxito: El sistema muestra el listado de DTs que tienen la nacionalidad seleccionada.

Fracaso:

Éxito Alternativo: El sistema muestra el listado completo de DTs.

**Código y Nombre del CASO DE USO:** CUU4- Listar jugadores pertenecientes a una plantilla

**Dimensiones de Clasificación:**

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Dialogal

**Meta del CASO DE USO:** Obtener listado de jugadores pertenecientes a una plantilla.

**ACTORES:**

Primario(s): Usuario

Otros: -

**PRECONDICIONES** (de sistema):

- 1- Cuenta de google creada.
- 2- Plantilla creada.
- 3- DTs creados.
- 4- Jugadores creados.

**DISPARADOR:** Usuario se loguea en la aplicacion web.

**FLUJOS (CAMINOS):**

**CAMINO BÁSICO**

1. Usuario se dirige a la seccion “LISTADO JUG”
2. El sistema redirige y solicita la seleccion de plantilla.
3. El usuario hace click en el boton desplegable de plantillas.
4. El sistema despliega el listado de plantillas.
5. El usuario hace click en la plantilla seleccionada.
6. El sistema cierra el listado y muestra el listado de jugadores pertenecientes a la plantilla.

**CAMINOS ALTERNATIVOS**

**POSTCONDICIONES** (de sistema):

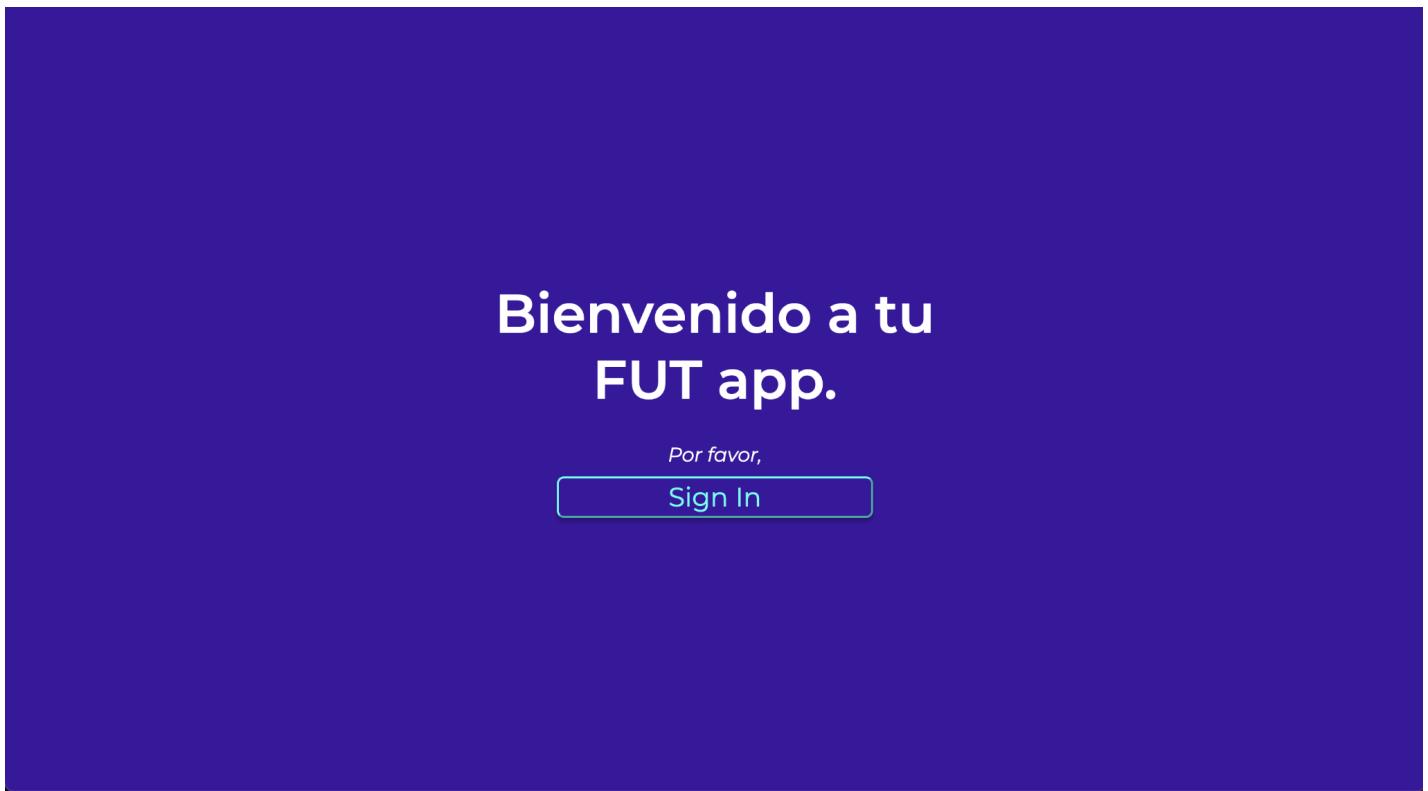
Éxito: El sistema muestra el listado de jugadores asociados a la plantilla seleccionada.

Fracaso:

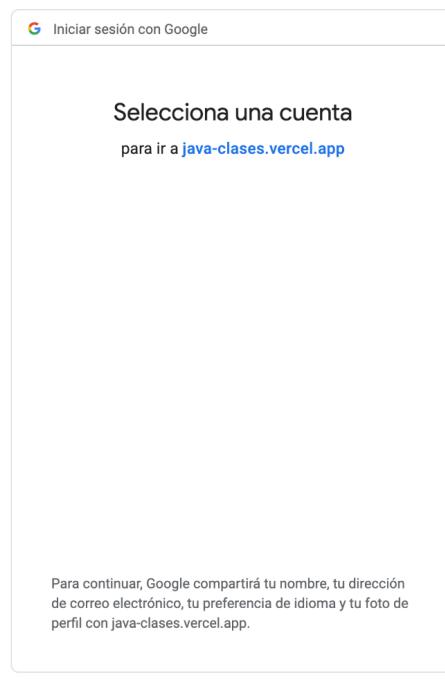
Éxito Alternativo: El sistema muestra que no existen jugadores asociados a la plantilla seleccionada.

## CAPTURAS DE PANTALLA DE TODAS LAS PANTALLAS

### **1- Pantalla login**



### **2- Pantalla logueo con google**



### 3- Pantalla bienvenida - dashboard

DASHBOARD  
CLUB  
PLANTILLA  
JUGADORES  
LISTADO JUG  
JUGADOR IDEAL  
LISTADO DTS  
DT IDEAL  
DTS  
LOGOUT

Bienvenido al gestor de Ultimate Team !

### 4 - Pantalla Club

DASHBOARD  
CLUB  
PLANTILLA  
JUGADORES  
LISTADO JUG  
JUGADOR IDEAL  
LISTADO DTS  
DT IDEAL  
DTS  
LOGOUT

Nombre	Sigla	Nacionalidad
Club Atletico Independiente	CAI	Argentina

[Editar club](#)

## 5- Pantalla Club - Edicion Club

<a href="#">DASHBOARD</a> <a href="#">CLUB</a> <a href="#">PLANTILLA</a> <a href="#">JUGADORES</a> <a href="#">LISTADO JUG</a> <a href="#">JUGADOR IDEAL</a> <a href="#">LISTADO DTS</a> <a href="#">DT IDEAL</a> <a href="#">DTS</a> <a href="#">LOGOUT</a>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;">Nombre</td> <td>Club Atletico Independiente</td> <td style="width: 10%;">Sigla</td> <td>Nacionalidad</td> </tr> <tr> <td colspan="2"><a href="#">Editar club</a></td> <td colspan="2"></td> </tr> <tr> <td colspan="4" style="text-align: center; padding-top: 10px;"> <input type="button" value="Confirmar"/> <input type="button" value="Cancelar"/> </td> </tr> </table>	Nombre	Club Atletico Independiente	Sigla	Nacionalidad	<a href="#">Editar club</a>				<input type="button" value="Confirmar"/> <input type="button" value="Cancelar"/>			
Nombre	Club Atletico Independiente	Sigla	Nacionalidad										
<a href="#">Editar club</a>													
<input type="button" value="Confirmar"/> <input type="button" value="Cancelar"/>													

## 6- Pantalla Plantilla

DASHBOARD	Select	Nombre Search...	Tactica Search...	Es plantilla Titular Search...	Club Search...	DT Search...	Jugadores Search...
<a href="#">CLUB</a>	<input type="checkbox"/>	plantilla uno	433	<input checked="" type="checkbox"/>	Club Atletico Independiente	test dt	Octavio
<a href="#">PLANTILLA</a>	<input type="checkbox"/>	plantilla dos2	532	<input type="checkbox"/>	Club Atletico Independiente	test dt	Octavio
<a href="#">JUGADORES</a>	<input type="checkbox"/>	Plantilla 3	433	<input type="checkbox"/>	Club Atletico Independiente	test dt	
<a href="#">LISTADO JUG</a>	<input type="checkbox"/>	plantilla 4	442	<input type="checkbox"/>	Club Atletico Independiente	test	Octavio Juan
<a href="#">JUGADOR IDEAL</a>	<input type="checkbox"/>	plantilla tress	4321	<input type="checkbox"/>	Club Atletico Independiente	test dt	Octavio Juan
<a href="#">LISTADO DTS</a>	<input type="checkbox"/>	test	442	<input type="checkbox"/>	Club Atletico Independiente	test dt	Octavio
<a href="#">DT IDEAL</a>	<input type="checkbox"/>	test a eliminar	442	<input type="checkbox"/>	Club Atletico Independiente	test dt	
<a href="#">DTS</a>							
<a href="#">LOGOUT</a>		<a href="#">Agregar Plantilla</a>	<a href="#">Editar seleccion</a>	<a href="#">Eliminar seleccion</a>			

## 7- Pantalla Plantilla - Edicion

The screenshot shows a modal dialog for editing a template. At the top are two buttons: 'Confirmar' (Confirm) in green and 'Cancelar' (Cancel) in red. Below them is a text input field labeled 'Nombre' containing 'plantilla 4'. A dropdown menu under 'Tactica' shows '442'. A note says "'Es Titular'" and a warning message in red says 'Ya existe una plantilla titular'. A small gray square icon is present. Below the modal is a dropdown menu set to 'test' and a list of players: 'Octavio' and 'Juan'.

## 8- Pantalla Jugadores

DASHBOARD	Select	Nombre Search...	Liga Search...	Nacionalidad Search...	Posicion Search...	Pierna Buena Search...	Edad Search...	Club Search...
CLUB	<input checked="" type="checkbox"/>	Octavio	sueca	Hungara	delantero	derecha	25	Club Atletico Independiente
PLANTILLA	<input checked="" type="checkbox"/>	Juan	argentina	argentina	defensor	izquierda	22	Club Atletico Independiente
<a href="#">Agregar Jugador</a> <a href="#">Editar seleccion</a> <a href="#">Eliminar seleccion</a>								
<a href="#">JUGADORES</a>								
<a href="#">LISTADO JUG</a>								
<a href="#">JUGADOR IDEAL</a>								
<a href="#">LISTADO DTS</a>								
<a href="#">DT IDEAL</a>								
<a href="#">DTS</a>								
<a href="#">LOGOUT</a>								

## 9- Pantalla Jugadores - Edicion

<a href="#">LISTADO DTS</a> <a href="#">DT IDEAL</a> <a href="#">DTS</a> <a href="#">LOGOUT</a>	<p><b>Nombre</b> Octavio</p> <p><b>Liga</b> sueca</p> <p><b>Nacionalidad</b> Hungara</p> <p><b>Posicion</b></p> <p>delantero</p> <p><b>Pierna Buena</b></p> <p>derecha</p> <p><b>Edad</b> 25</p> <p style="text-align: right;"><a href="#">Confirmar</a> <a href="#">Cancelar</a></p>
--	---

## 10- Pantalla Listado jugadores

<a href="#">DASHBOARD</a> <a href="#">CLUB</a> <a href="#">PLANTILLA</a> <a href="#">JUGADORES</a> <a href="#">LISTADO JUG</a> <b>JUGADOR IDEAL</b> <a href="#">LISTADO DTS</a> <a href="#">DT IDEAL</a> <a href="#">DTS</a> <a href="#">LOGOUT</a>	<p>Por favor seleccione la plantilla para listar sus jugadores</p> <p>plantilla tress - Club Atletico Independiente - 4321</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Nombre</th><th>Liga</th><th>Nacionalidad</th><th>Posicion</th><th>Pierna Buena</th><th>Edad</th><th>Club</th></tr> </thead> <tbody> <tr> <td>Search...</td><td>Search...</td><td>Search...</td><td>Search...</td><td>Search...</td><td>Search...</td><td>Search...</td></tr> <tr> <td>Octavio</td><td>sueca</td><td>Hungara</td><td>delantero</td><td>derecha</td><td>25</td><td>Club Atletico Independiente</td></tr> <tr> <td>Juan</td><td>argentina</td><td>argentina</td><td>defensor</td><td>izquierda</td><td>22</td><td>Club Atletico Independiente</td></tr> </tbody> </table>	Nombre	Liga	Nacionalidad	Posicion	Pierna Buena	Edad	Club	Search...	Octavio	sueca	Hungara	delantero	derecha	25	Club Atletico Independiente	Juan	argentina	argentina	defensor	izquierda	22	Club Atletico Independiente						
Nombre	Liga	Nacionalidad	Posicion	Pierna Buena	Edad	Club																							
Search...	Search...	Search...	Search...	Search...	Search...	Search...																							
Octavio	sueca	Hungara	delantero	derecha	25	Club Atletico Independiente																							
Juan	argentina	argentina	defensor	izquierda	22	Club Atletico Independiente																							

## 11- Pantalla Jugador ideal

DASHBOARD

CLUB

PLANTILLA

JUGADORES

LISTADO JUG

JUGADOR IDEAL

LISTADO DTS

DT IDEAL

DTS

LOGOUT

Por favor seleccione la plantilla para buscar el jugador ideal

plantilla uno - Club Atletico Independiente - 433

Por favor seleccione la posicion para buscar el jugador ideal

defensor

Nombre	Liga	Nacionalidad	Posicion	Pierna Buena	Edad	Club
Juan	argentina	argentina	defensor	izquierda	22	Club Atletico Independiente

## 12- Pantalla Listado DTs

DASHBOARD

CLUB

PLANTILLA

JUGADORES

LISTADO JUG

JUGADOR IDEAL

LISTADO DTS

DT IDEAL

DTS

LOGOUT

Por favor seleccione la nacionalidad para listar sus DTs

argentina

Nombre	Liga	Nacionalidad
Search...	Search...	Search...
test	argentina	argentina

### 13- Pantalla DT Ideal

DASHBOARD

CLUB

PLANTILLA

JUGADORES

LISTADO JUG

JUGADOR IDEAL

LISTADO DTS

DT IDEAL

DTS

LOGOUT

Por favor seleccione la plantilla para buscar el DT ideal

plantilla tress - Club Atletico Independiente - 4321

Nombre	Liga	Nacionalidad
test	argentina	argentina

### 14- Pantalla DTs

DASHBOARD

CLUB

PLANTILLA

JUGADORES

LISTADO JUG

JUGADOR IDEAL

LISTADO DTS

DT IDEAL

DTS

LOGOUT

Select	Nombre Search...	Liga Search...	Nacionalidad Search...	Club Search...
<input type="checkbox"/>	test dt	dt	dt	Club Atletico Independiente
<input checked="" type="checkbox"/>	test	argentina	argentina	Club Atletico Independiente

Agregar DT    Editar seleccion    Eliminar seleccion

Nombre  
test

Liga  
argentina

Nacionalidad  
argentina

Confirmar    Cancelar

## CAPTURAS DE PANTALLA DE CODIGO - CUU 4 - Listar jugadores por plantilla

### **Base de datos:**

#### Database.users

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 801B TOTAL DOCUMENTS: 4 INDEXES TOTAL SIZE: 36KB

**Find** **Indexes** **Schema Anti-Patterns 0** **Aggregation** **Search Indexes ●** **INSERT DOCUMENT**

**FILTER** { field: 'value' } **OPTIONS** **Apply** **Reset**

**QUERY RESULTS: 1-4 OF 4**

```
_id: ObjectId('636f0f3e4c8cd398392636cd')
name: "Octavio Berruti"
email: "octavioberruti8@gmail.com"
image: "https://lh3.googleusercontent.com/a/ALm5wu0vg3SlX_ZIRH3-5Xynln5uGhsyCz..."
emailVerified: null
```

#### Database.sessions

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 456B TOTAL DOCUMENTS: 4 INDEXES TOTAL SIZE: 36KB

**Find** **Indexes** **Schema Anti-Patterns 0** **Aggregation** **Search Indexes ●** **INSERT DOCUMENT**

**FILTER** { field: 'value' } **OPTIONS** **Apply** **Reset**

**QUERY RESULTS: 1-5 OF 5**

```
_id: ObjectId('63708ea87be85a709d32e02c')
sessionToken: "00070a7b-29f9-42ea-b658-fcc14bc35045"
userId: ObjectId('636f0f3e4c8cd398392636cd')
expires: 2022-11-14T06:31:32.668+00:00
```

## Database.club

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 698B TOTAL DOCUMENTS: 4 INDEXES TOTAL SIZE: 36KB

Find

Indexes

Schema Anti-Patterns 0

Aggregation

Search Indexes 0

INSERT DOCUMENT

FILTER

{ field: 'value' }

OPTIONS

Apply

Reset

QUERY RESULTS: 1-4 OF 4

```
_id: ObjectId('636ae03f2cb4d5765c8ce0e3')
nombre: "Club Atletico Independiente"
sigla: "CAI"
nacionalidad: "Argentina"
userID: "636f0f3e4c8cd398392636cd"
_class: "com.javatp.javaTP.database.Club.Club"
```

## Database.plantilla

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 1.76KB TOTAL DOCUMENTS: 7 INDEXES TOTAL SIZE: 36KB

Find

Indexes

Schema Anti-Patterns 0

Aggregation

Search Indexes 0

INSERT DOCUMENT

FILTER

{ field: 'value' }

OPTIONS

Apply

Reset

QUERY RESULTS: 1-7 OF 7

```
_id: ObjectId('636fb9c9a14a8c95ccfc3d87')
nombre: "plantilla uno"
tactica: "433"
esTitular: true
> jugadoresIDs: Array
  clubID: "636ae03f2cb4d5765c8ce0e3"
  _class: "com.javatp.javaTP.database.Plantilla.Plantilla"
  dtID: "637be56aaf2c170c438e420c"
```

## Database.jugador

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 701B TOTAL DOCUMENTS: 3 INDEXES TOTAL SIZE: 36KB

Find

Indexes

Schema Anti-Patterns 0

Aggregation

Search Indexes 0

INSERT DOCUMENT

FILTER

{ field: 'value' }

OPTIONS

Apply

Reset

QUERY RESULTS: 1-3 OF 3

```
_id: ObjectId('636f22f976c958cb4a9f5b11')
nombre: "Octavio"
liga: "sueca"
nacionalidad: "Hungara"
posicion: "delantero"
piernaBuena: "derecha"
edad: 25
clubID: "636ae03f2cb4d5765c8ce0e3"
_class: "com.javatp.javaTP.database.Jugador.Jugador"
```

## **FRONTEND:**

### **Imports:**

```
 1 import { GetServerSideProps } from "next";
 2 import { getToken } from "next-auth/jwt";
 3 import { getSession } from "next-auth/react";
 4 import { useMemo, useState } from "react";
 5 import {
 6   Column,
 7   ColumnDef,
 8   flexRender,
 9   getCoreRowModel,
10   getFilteredRowModel,
11   Table,
12   useReactTable,
13 } from "@tanstack/react-table";
14 import { Layout } from "../../common/components/page";
15 import { COLORS, FONTS } from "../../common/styles/style";
16 import React from "react";
17 import {
18   HorizontalCentered,
19   VerticalStack,
20 } from "../../common/components/flex";
21 import { StyleMap } from "../../common/utils/tsTypes";
22 import { Jugadores, JugadorRow } from "../jugadores";
23 import { DropDownList } from "react-widgets";
24 import ErrorMessage from "../../common/components(ErrorMessage";
25 import { ERRORES } from "../../common/components/page/utils";
```

## Fetching data para la pagina

```
1 export const getServerSideProps: GetServerSideProps = async ({  
2   req,  
3   res,  
4 }: any) => {  
5   const session = await getSession({ req });  
6  
7   if (!session) {  
8     // If not user, redirect to login  
9     return {  
10       props: {},  
11       redirect: {  
12         destination: "/acc/login",  
13         permanent: false,  
14       },  
15     };  
16   }  
17  
18   const token = await getToken({ req, raw: true });  
19  
20   const resClub = await fetch(  
21     `${process.env.BACKEND_URL}/club/query?sessionToken=${token}`  
22   );  
23   const club = await resClub.json();  
24  
25   if (!resClub.ok) {  
26     if (  
27       club.message === ERRORES.NO_CLUB ||  
28       club.message === ERRORES.NO_SESSION  
29     ) {  
30       return {  
31         props: {  
32           criticalError: club.message,  
33         },  
34       };  
35     }  
36   }  
37  
38   const resPlantillas = await fetch(  
39     `${process.env.BACKEND_URL}/plantilla/query?sessionToken=${token}`  
40   );  
41   const plantillas = await resPlantillas.json();  
42  
43   if (!resPlantillas.ok) {  
44     if (  
45       plantillas.message === ERRORES.NO_CLUB ||  
46       plantillas.message === ERRORES.NO_SESSION  
47     ) {  
48       return {  
49         props: {  
50           criticalError: plantillas.message,  
51         },  
52       };  
53     }  
54   }  
55 };
```

```
49         props: {
50             criticalError: plantillas.message,
51         },
52     );
53 }
55
56 const getJugadoresByPlantillaID = async (id: String) => {
57     const jugadoresByPlantillaIdRes = await fetch(
58         `${process.env.BACKEND_URL}/plantilla/${id}/jugadores/query?sessionToken=${token}`
59     );
60     const jugadoresByPlantillaId = await jugadoresByPlantillaIdRes.json();
61
62     if (!jugadoresByPlantillaIdRes.ok) {
63         {
64             return {
65                 criticalError: jugadoresByPlantillaId.message,
66             };
67         }
68     }
69
70     if (jugadoresByPlantillaId) {
71         return jugadoresByPlantillaId;
72     }
73     return [];
74 };
75
76 const getPlantillaByPlantilla = async (plantilla) => {
77     const jugadores = await getJugadoresByPlantillaID(plantilla.id);
78     if (jugadores.criticalError) {
79         return jugadores;
80     }
81     return {
82         id: plantilla.id,
83         nombre: plantilla.nombre,
84         tactica: plantilla.tactica,
85         esTitular: plantilla.esTitular,
86         club: club.nombre,
87         jugadores,
88     };
89 };
90
91 const plantillasYClub = async () => {
92     const plantillasDone = [];
93
94     for (var i = 0; i < plantillas.length; i++) {
95         const plant = await getPlantillaByPlantilla(plantillas[i]);
96         if (plant.criticalError) {
97             return plant;
98         }
99         plantillasDone.push(plant);
100    }
101}
```

```
102     return plantillasDone;
103   };
104
105   const plantillasYClubJson = await plantillasYClub();
106
107   if (plantillasYClubJson.criticalError) {
108     return {
109       props: {
110         criticalError: plantillasYClubJson.criticalError,
111       },
112     };
113   }
114
115   // If user, stay here
116   return {
117     props: {
118       plantillasYClub: plantillasYClubJson,
119     },
120   };
121 };
122
123 export default listadoJugadoresPorPlantillaPage;
124
```

## Definicion de styles para la pagina:

```
 1 const styles: StyleMap = {  
 2   input: {  
 3     height: "25px",  
 4     width: "70%",  
 5     borderBottom: `1px solid ${COLORS.green}`,  
 6     borderTopStyle: "none",  
 7     borderLeftStyle: "none",  
 8     borderRightStyle: "none",  
 9     fontSize: "14px",  
10     marginBottom: "6%",  
11     display: "flex",  
12     alignSelf: "center",  
13     fontFamily: FONTS.comments.fontFamily,  
14     background: COLORS.blue,  
15     color: COLORS.white,  
16     outline: "none",  
17     boxShadow: "none",  
18   },  
19   confirm: {  
20     height: "30px",  
21     border: "2px solid #75cb64",  
22     background: COLORS.blue,  
23     boxShadow: "0 1px 1px rgba(0, 0, 0, 0.25)",  
24     color: "white",  
25     fontSize: "14px",  
26     marginTop: "2%",  
27     marginBottom: "4%",  
28     borderRadius: "5px",  
29     cursor: "pointer",  
30     textAlign: "center",  
31     width: "100px",  
32     fontFamily: FONTS.comments.fontFamily,  
33     marginRight: "100px",  
34   },  
35   cancel: {  
36     height: "30px",  
37     border: "2px solid #cb6464",  
38     background: COLORS.blue,  
39     boxShadow: "0 1px 1px rgba(0, 0, 0, 0.25)",  
40     color: "white",  
41     fontSize: "14px",  
42     marginTop: "2%",  
43     marginBottom: "4%",  
44     borderRadius: "5px",  
45     cursor: "pointer",  
46     textAlign: "center",  
47     width: "100px",  
48     fontFamily: FONTS.comments.fontFamily,
```

```
49 },
50   title: {
51     marginTop: "3%",
52     marginBottom: "3%",
53     textAlign: "center",
54     color: "white",
55     fontSize: "20px",
56     fontFamily: FONTS.comments.fontFamily,
57     borderBottom: `1px solid ${COLORS.green}`,
58     width: "200px",
59     display: "flex",
60     justifyContent: "center",
61     alignSelf: "center",
62   },
63   errorMessage: {
64     fontFamily: FONTS.comments.fontFamily,
65     fontStyle: FONTS.comments.fontSize,
66     fontSize: FONTS.comments.fontSize,
67     fontWeight: FONTS.comments.fontWeight,
68     color: COLORS.rose,
69     textAlign: "center",
70     width: "100%",
71     margin: "10px",
72   },
73   maybeTitle: {
74     maxHeight: "15px",
75     height: "auto",
76     width: "70%",
77     color: COLORS.green,
78     borderTopStyle: "none",
79     borderLeftStyle: "none",
80     borderRightStyle: "none",
81     fontSize: "14px",
82     marginBottom: "0.5%",
83     display: "flex",
84     alignSelf: "center",
85     fontFamily: FONTS.comments.fontFamily,
86     background: COLORS.blue,
87     outline: "none",
88     boxShadow: "none",
89   },
90 };
```

## Definicion de tipos:

```
1 type PlantillaRow = {
2   id: string;
3   nombre: string;
4   tactica: string;
5   esTitular: boolean;
6   club: string;
7   jugadores: Jugadores;
8 };
9
10 type ListadoJugadoresPorPlantillaPageProps = {
11   criticalError?: string;
12   plantillasYClub?: PlantillaRow[];
13 };
14
```

## Definicion de componente de filtrado:

```
1 function Filter({
2   column,
3   table,
4 }: {
5   column: Column<any, any>;
6   table: Table<any>;
7 }) {
8   const firstValue = table
9     .getPrefilteredRowModel()
10    .flatRows[0]?.getValue(column.id);
11
12   return typeof firstValue === "number" ? (
13     <div className="flex space-x-2">
14       <input
15         type="number"
16         value={((column.getFilterValue() as any)?.[0] ?? "") as string}
17         onChange={(e) =>
18           column.setFilterValue([e.target.value, old?.[1]])
19         }
20         placeholder={`Min`}
21         className="w-24 border shadow rounded"
22       />
23       <input
24         type="number"
25         value={((column.getFilterValue() as any)?.[1] ?? "") as string}
26         onChange={(e) =>
27           column.setFilterValue([old?.[0], e.target.value])
28         }
29         placeholder={`Max`}
30         className="w-24 border shadow rounded"
31       />
32     </div>
33   ) : (
34     <input
35       type="text"
36       value=((column.getFilterValue() ?? "") as string)
37       onChange={(e) => column.setFilterValue(e.target.value)}
38       placeholder={`Search...`}
39       className="w-36 border shadow rounded"
40     />
41   );
42 }
```

## **Definicion del componente principal de la pagina:**

```
● ● ●
1 function listadoJugadoresPorPlantillaPage({
2   plantillasYClub = [],
3   criticalError,
4 }: ListadoJugadoresPorPlantillaPageProps) {
5   const COLUMNS = useMemo<ColumnDef<JugadorRow>[]>(
6     () => [
7       {
8         header: "Nombre",
9         accessorKey: "nombre",
10      },
11      {
12        header: "Liga",
13        accessorKey: "liga",
14      },
15      {
16        header: "Nacionalidad",
17        accessorKey: "nacionalidad",
18      },
19      {
20        header: "Posicion",
21        accessorKey: "posicion",
22      },
23      {
24        header: "Pierna Buena",
25        accessorKey: "piernaBuena",
26      },
27      {
28        header: "Edad",
29        accessorKey: "edad",
30      },
31      {
32        header: "Club",
33        accessorKey: "club",
34      },
35    ],
36    []
37  );
38
39 const [errorMessage, setErrorMessage] = useState<string | undefined>();
40 const [plantillaSelected, setPlantillaSelected] = useState<PlantillaRow>();
41 const jugadoresYClub = (plantillaSelected: PlantillaRow) =>
42   plantillaSelected &&
43   plantillaSelected.jugadores.map((jugador) => ({
44     id: jugador.id,
45     nombre: jugador.nombre,
46     liga: jugador.liga,
47     nacionalidad: jugador.nacionalidad,
48     posicion: jugador.posicion,
49     piernaBuena: jugador.piernaBuena,
50     edad: jugador.edad.toString(),
51     club: plantillaSelected.club,
52   }));
53
54 const [data, setData] = useState([]);
55
56 const columns = useMemo(() => COLUMNS, []);
57
58 const onChange = (value) => {
59   setErrorMessage(undefined);
60   const plantillaSelected = plantillasYClub.find(
61     (plantilla) => plantilla.id === value.id
62   );
63   setPlantillaSelected(plantillaSelected);
64   const jugadores = jugadoresYClub(plantillaSelected);
65   if (jugadores.length > 0) {
66     setData(jugadores);
67   } else {
68     setErrorMessage("No existen jugadores en esta plantilla.");
69   }
70 };
71
72 const table = useReactTable({
73   data,
74   columns,
75   getCoreRowModel: getCoreRowModel(),
76   getFilteredRowModel: getFilteredRowModel(),
77 });
78
```

**Definicion del componente de error principal:**

```
1 if (criticalError) {  
2     return (  
3         <Layout>  
4             <ErrorMessage message={criticalError} />  
5         </Layout>  
6     );  
7 }
```

## Definicion del componente tabla y listado:

```
1 return (
2     <Layout>
3         <VerticalStack>
4             <HorizontalCentered style={{ width: "100%" }}>
5                 <div
6                     style={{
7                         ...styles.maybeTitle,
8                         justifyContent: "center",
9                         marginTop: "20px",
10                    }}
11                >
12                    Por favor seleccione la plantilla para listar sus jugadores
13                </div>
14            </HorizontalCentered>
15            <div
16                style={{
17                    minWidth: "50%",
18                    width: "50%",
19                    height: "100px",
20                    display: "flex",
21                    alignContent: "center",
22                    alignItems: "center",
23                    alignSelf: "center",
24                }}
25            >
26                <DropdownList
27                    data={plantillasYClub.map((object) => ({
28                        id: object.id,
29                        value: `${object.nombre} - ${object.club} - ${object.tactica}`,
30                    }))}
31                    dataKey="id"
32                    textField="value"
33                    value={plantillaSelected}
34                    onChange={onChange}
35                />
36            </div>
37        <VerticalStack>
38            {errorMessage ? (
39                <div style={styles.errorMessage}>{errorMessage}</div>
40            ) : (
41                <div className="container">
42                    <table
43                        style={{
44                            borderCollapse: "collapse",
45                            width: "100%",
46                        }}
47                >
48                    <thead>
49                        {table.getHeaderGroups().map((headerGroup) =>
50                            <tr key={headerGroup.id}>
51                                {headerGroup.headers.map((header) => (
52                                    <th
53                                        key={header.id}
54                                        style={{
55                                            border: '1px solid ${COLORS.green}',
56                                            textAlign: "left",
57                                            padding: "8px",
58                                            color: "white",
59                                        }}
60                                    >
61                                        {flexRender(
62                                            header.column.columnDef.header,
63                                            header.getContext()
64                                        )}
65                                        {header.column.getCanFilter() ? (
66                                            <div>
67                                                <Filter column={header.column} table={table} />
68                                            </div>
69                                        ) : null}
70                                    </th>
71                                )));
72                            </tr>
73                        )));
74                    </thead>
75                    <tbody>
76                        {table.getRowModel().rows.map((row) => {
77                            return (
78                                <tr key={row.id}>
79                                    {row.getVisibleCells().map((cell) => {
80                                        return (
81                                            <td
82                                                style={{
83                                                    border: '1px solid ${COLORS.green}',
84                                                    textAlign: "left",
85                                                    padding: "8px",
86                                                    color: "white",
87                                                }}
88                                                key={cell.id}
89                                            >
90                                                {flexRender(
91                                                    cell.column.columnDef.cell,
92                                                    cell.getContext()
93                                                )}
94                                            </td>
95                                        );
96                                    )));
97                                </tr>
98                            )));
99                        });
100                    </tbody>
```

## **BACKEND:**

### **Club controller:**

```
1 package com.javatp.javaTP.database.Club;
2
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.web.bind.annotation.PostMapping;
7 import org.springframework.web.bind.annotation.CrossOrigin;
8 import org.springframework.web.bind.annotation.GetMapping;
9 import org.springframework.web.bind.annotation.PathVariable;
10 import org.springframework.web.bind.annotation.RequestBody;
11 import org.springframework.web.bind.annotation.RequestMapping;
12 import org.springframework.web.bind.annotation.RequestParam;
13 import org.springframework.web.bind.annotation.RestController;
14
15 import com.javatp.javaTP.database.Sessions.Sessions;
16 import com.javatp.javaTP.database.Sessions.SessionsController;
17 import com.javatp.javaTP.database.Sessions.SessionsRepository;
18 import com.javatp.javaTP.exception.ApiRequestException;
19
20 // @GetMapping
21 // public Optional<Club> club(@RequestParam(name = "id", required = false, defaultValue = "hola" ) String id ) {
22
23 @RestController
24 @RequestMapping("/club")
25 public class ClubController {
26     @Autowired
27     private ClubRepository clubRepository;
28
29     @Autowired
30     private SessionsRepository sessionsRepository;
31
32     @Autowired
33     private SessionsController sessionsController;
34
35     private Sessions getSessionByToken(String sessionToken ) {
36         try {
37             Sessions session = sessionsRepository.getSessionBySessionToken(sessionToken).get();
38             return session;
39         } catch(RuntimeException e) {
40             throw new ApiRequestException("Error - Usted no esta autenticado");
41         }
42     }
43
44     @CrossOrigin("*")
45     @GetMapping("/query")
46     public Club club(@RequestParam(name = "sessionToken", required = true ) String sessionToken ) {
47         Sessions session = getSessionByToken(sessionToken);
48         try {
49             Club club = clubRepository.getClubByUserId(session.getUserId()).get();
50             return (Club) club;
51         } catch (RuntimeException eClub) {
52             throw new ApiRequestException("Error - No existe club asociado");
53         }
54     }
55 }
```

## **Jugador Repository:**

```
1 package com.javatp.javaTP.database.Jugador;
2
3 import java.util.ArrayList;
4 import java.util.Optional;
5
6 import org.springframework.data.mongodb.repository.MongoRepository;
7 import org.springframework.data.mongodb.repository.Query;
8
9 //examples https://javatechonline.com/spring-boot-mongodb-query-examples/
10 public interface JugadorRepository extends MongoRepository<Jugador, String> {
11
12     @Query("{id :?0}")
13     Optional<Jugador> getJugadorById(String id);
14
15     ArrayList<Jugador> findByClubID(String ClubID);
16
17     Optional<Jugador> findByIdAndClubID(String id, String ClubID);
18
19 }
```

```
● ● ●
1 public class Jugador {
2
3     @Id
4     protected String id;
5
6     protected String nombre;
7     protected String liga;
8     protected String nacionalidad;
9     protected String posicion;
10    protected String piernaBuena;
11    protected Integer edad;
12
13
14    protected String clubID;
15
16    public Jugador() {}
17
18    public Jugador(String nombre, String liga, String nacionalidad, String piernaBuena, Integer edad,
19    String clubID) {
20        this.nombre = nombre;
21        this.liga = liga;
22        this.nacionalidad = nacionalidad;
23        this.piernaBuena = piernaBuena;
24        this.edad = edad;
25        this.clubID = clubID;
26    }
27
28    public String getId() {
29        return this.id;
30    }
31
32    public void setId(String id) {
33        this.id = id;
34    }
35
36    public String getNombre() {
37        return this.nombre;
38    }
39
40    public void setNombre(String nombre) {
41        this.nombre = nombre;
42    }
43
44    public String getLiga() {
45        return this.liga;
46    }
47
48    public void setLiga(String liga) {
49        this.liga = liga;
50    }
51
52    public String getNacionalidad() {
53        return this.nacionalidad;
54    }
55
56    public void setNacionalidad(String nacionalidad) {
57        this.nacionalidad = nacionalidad;
58    }
59
60    public String getPosition() {
61        return this.posicion;
62    }
63
64    public void setPosition(String posicion) {
65        this.posicion = posicion;
66    }
67
68    public String getPiernaBuena() {
69        return this.piernaBuena;
70    }
71
72    public void setPiernaBuena(String piernaBuena) {
73        this.piernaBuena = piernaBuena;
74    }
75
76    public Integer getEdad() {
77        return this.edad;
78    }
79
80    public void setEdad(Integer edad) {
81        this.edad = edad;
82    }
83
84    public String getClubID() {
85        return this.clubID;
86    }
87
88    public void setClubID(String clubID) {
89        this.clubID = clubID;
90    }
91
92    @Override
93    public String toString() {
94        return String.format(
95            "Jugador[id=%s, nombre='%s', liga='%s', nacionalidad='%s', posicion='%s', piernaBuena='%s', edad='%s',
96            clubID='%s']",
97            id, nombre, liga, nacionalidad, posicion, piernaBuena, edad, clubID);
98    }
99 }
```

## Plantilla Controller:

```
1 package com.javatp.javaTP.database.Plantilla;
2
3
4
5 import java.util.ArrayList;
6 import java.util.Arrays;
7 import java.util.List;
8 import java.util.Optional;
9 import java.util.stream.*;
10
11 import org.springframework.beans.factory.annotation.Autowired;
12 import org.springframework.web.bind.annotation.PostMapping;
13 import org.springframework.web.bind.annotation.CrossOrigin;
14 import org.springframework.web.bind.annotation.DeleteMapping;
15 import org.springframework.web.bind.annotation.GetMapping;
16 import org.springframework.web.bind.annotation.PathVariable;
17 import org.springframework.web.bind.annotation.RequestBody;
18 import org.springframework.web.bind.annotation.RequestMapping;
19 import org.springframework.web.bind.annotation.RequestParam;
20 import org.springframework.web.bind.annotation.RestController;
21
22 import com.javatp.javaTP.database.Club.Club;
23 import com.javatp.javaTP.database.Club.ClubRepository;
24 import com.javatp.javaTP.database.Dtt.Dtt;
25 import com.javatp.javaTP.database.Dtt.DttRepository;
26 import com.javatp.javaTP.database.Jugador.Jugador;
27 import com.javatp.javaTP.database.Jugador.JugadorRepository;
28 import com.javatp.javaTP.database.Sessions.Sessions;
29 import com.javatp.javaTP.database.Sessions.SessionsController;
30 import com.javatp.javaTP.database.Sessions.SessionsRepository;
31 import com.javatp.javaTP.exception.ApiRequestException;
32
33 //GetMapping
34 // public Optional<Plantilla> plantilla(@RequestParam(name = "id", required = false, defaultValue = "hola" ) String id ) {
35
36 @RestController
37 @RequestMapping("/plantilla")
38 public class PlantillaController {
39
40     @Autowired
41     private PlantillaRepository plantillaRepository;
42
43     @Autowired
44     private SessionsRepository sessionsRepository;
45
46     @Autowired
47     private ClubRepository clubRepository;
48
49     @Autowired
50     private JugadorRepository jugadorRepository;
51
52     @Autowired
53     private DttRepository dttRepository;
54
55     @Autowired
56     private SessionsController sessionsController;
57
58     private Sessions getSessionByToken(String sessionToken ) {
59         try {
60             Sessions session = sessionsRepository.getSessionBySessionToken(sessionToken).get();
61             return session;
62         } catch(RuntimeException e) {
63             throw new ApiRequestException("Error - Usted no esta autenticado");
64         }
65     }
66
67     private Club getClubBySessionToken(String sessionToken ) {
68         Sessions session = getSessionByToken(sessionToken);
69         try {
70             Club club = clubRepository.getClubByUserId(session.getUserId()).get();
71             return club;
72         } catch(RuntimeException e) {
73             throw new ApiRequestException("Error - No existe club asociado");
74         }
75     }
76
77     @CrossOrigin("*")
78     @GetMapping("/query")
79     public ArrayList<Plantilla> getPlantillaes(@RequestParam(name = "sessionToken", required = true ) String sessionToken ) {
80         Club club = getClubBySessionToken(sessionToken);
81         try {
82             return (ArrayList<Plantilla>) plantillaRepository.findByClubID(club.getId());
83         } catch(RuntimeException e) {
84             throw new ApiRequestException("Error - No existen plantillas para ese club");
85         }
86     }
}
```

```
 1  private Plantilla getPlantillaByIdAndClubId(String id, String clubId) {
 2      try {
 3          Plantilla plantilla = plantillaRepository.findByIdAndClubID(id, clubId).get();
 4          return plantilla;
 5      } catch (RuntimeException e) {
 6          throw new ApiRequestException("Error - No se encontro la plantilla");
 7      }
 8  }
 9
10  @CrossOrigin("*")
11  @GetMapping(path = "/{id}/jugadores/query")
12  public ArrayList<Jugador> getJugadoresByPlantillaId(@PathVariable("id") String id, @RequestParam(
13      name = "sessionToken", required = true ) String sessionToken ) {
14      if (id.isEmpty()) {
15          throw new ApiRequestException("Error - Parametros incorrectos");
16      }
17      Club club = getClubBySessionToken(sessionToken);
18      Plantilla plantilla = getPlantillaByIdAndClubId(id, club.getId());
19      String[] jugadoresIds = plantilla.getJugadoresIDs();
20      ArrayList<Jugador> jugadores = new ArrayList<Jugador>();
21      try {
22          for (int i=0; i<jugadoresIds.length; i++) {
23              Jugador jugador = jugadorRepository.getJugadorById(jugadoresIds[i]).get();
24              jugadores.add(jugador);
25          }
26          return jugadores;
27      } catch(RuntimeException e) {
28          throw new ApiRequestException("Error - No se encontraron jugadores");
29      }
}
```

## Sessions repository

```
1 package com.javatp.javaTP.database.Sessions;
2
3 import java.util.Optional;
4
5 import org.springframework.data.mongodb.repository.MongoRepository;
6 import org.springframework.data.mongodb.repository.Query;
7
8 //examples https://javatechonline.com/spring-boot-mongodb-query-examples/
9 public interface SessionsRepository extends MongoRepository<Sessions, String> {
10
11     @Query("{sessionToken :?0}")
12     Optional<Sessions> getSessionBySessionToken(String sessionToken);
13
14 }
```

## Sessions clase

```
1 package com.javatp.javaTP.database.Sessions;
2
3 import org.springframework.data.annotation.Id;
4
5
6 public class Sessions {
7
8     @Id
9     protected String id;
10
11    private String sessionToken;
12
13    private String userId;
14
15    public Sessions() {}
16
17
18    public String getSessionToken() {
19        return this.sessionToken;
20    }
21
22    public String getUserId() {
23        return this.userId;
24    }
25 }
```