



universidad  
de león



# **Escuela de Ingenierías Industrial, Informática y Aeroespacial**

## **GRADO EN INGENIERÍA INFORMÁTICA**

Sistemas de Información de Gestión y Business Intelligence

Memoria de la aplicación web: FutbolFichajes

Nombre del Alumno:

Aitor Del Río Ferreras

# **Resumen**

Debido a la situación actual vivida por la pandemia, el mundo del futbol, al igual que en otros campos, sufre los estragos del virus en el tema económico. Muchos equipos se ven en la situación de tener que mirar mucho su situación económica para realizar nuevas incorporaciones en sus plantillas y para mantener la plantilla que tiene por lo que ofrecemos un sistema que ayudará a poder buscar jugadores de varias formas para poder utilizar mejor sus presupuestos.

# Índice

Glosario de símbolos o Terminología	4
Introducción	5
Desarrollo	6
1. <i>DESCRIPCIÓN DEL PROBLEMA</i>	6
2. <i>HERRAMIENTAS UTILIZADAS</i>	7
3. <i>COMPOSICIÓN DE LA APLICACIÓN</i>	9
1. Base de datos	10
2. Backend	13
3. Frontend	20
4. <i>ANALISIS DE RESULTADOS</i>	22
• Mostrar jugadores aleatorios:	23
• Mostrar jugadores en base a filtros:	24
• Mostrar los mejores jugadores:	27
• Mostrar los jugadores promesas:	28
5. <i>DAFO</i>	28
6. <i>LÍNEAS DE FUTURO</i>	30
7. <i>LECCIONES APRENDIDAS</i>	31
8. <i>INSTALACION POR PASOS</i>	32
Conclusión	37
Bibliografía y referencias	38

# Glosario de símbolos o Terminología

Frontend: Parte exterior de una aplicación.

Backend: Parte interna de una aplicación que trabaja con los datos.

# Introducción

En este documento se expondrá el trabajo realizado para lograr terminar la aplicación web de recomendación de fichajes.

Primero se realizará una exposición detallada del problema que intentaremos resolver con nuestra aplicación y posteriormente daremos una explicación de su composición y funcionamiento.

La explicación se realizará desde abajo a arriba, es decir, empezaremos explicando la parte de persistencia de datos, donde analizaremos como hemos utilizado la tecnología neo4j.

Seguidamente explicaremos como hemos utilizado la tecnología node js para realizar el backend, que será la capa que hará de intermediario recibiendo peticiones del frontend y trasladándolas a la base de datos y después, cuando reciba los datos. Se los devolverá al frontend.

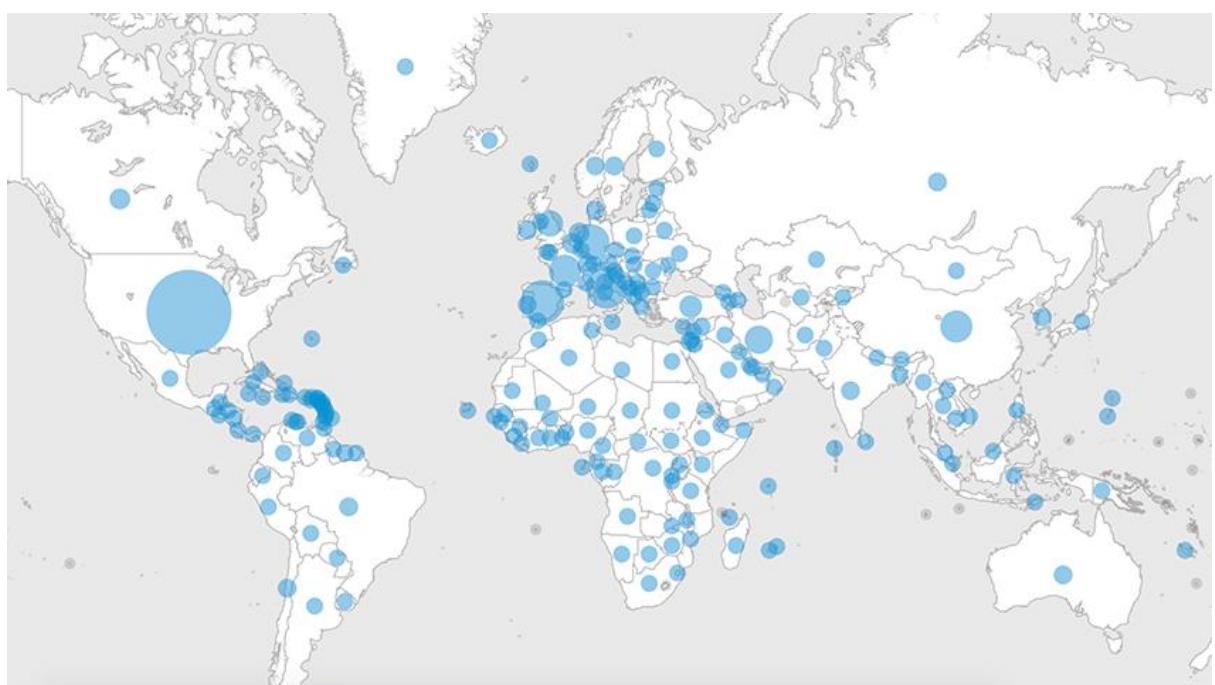
Acto seguido explicaremos como hemos utilizado la tecnología Vue js y en particular Vuetify para implementar una interfaz de usuario.

Para finalizar, explicaremos el funcionamiento de varios casos de uso de la aplicación web, explicando el recorrido tanto fuera de la aplicación (es decir, las acciones que hacemos y los datos que se nos muestra) y también por dentro (haremos un breve recorrido desde por ejemplo la pulsación de un botón hasta la obtención de los datos).

# Desarrollo

## 1. DESCRIPCIÓN DEL PROBLEMA

Debido a la pandemia que esta asolando a cualquier rincón del planeta, tanto en el tema sanitario, por la perdida de tantas vidas humanas, como en lo económico, con las terribles consecuencias que esta trayendo como puede ser perdidas de trabajo, salarios mas bajos o economías mas bajas a nivel global. En nuestro caso nos vamos a centrar en los estragos que esta sufriendo el mundo del deporte, mas concretamente el mundo del futbol.



En el mundo del futbol, solo la ausencia de publico en las gradas de los estadios ha producido que los equipos vean que su nivel económico ha caído drásticamente hasta el punto de que numerosos equipos han acabado sus años con balances económicos en negativo con números negativamente altos. Debido a esto, los equipos necesitan rentabilizar mas el dinero que tienen, ya que es poco y, por otra parte, necesitan cada año hacer proyectos nuevos para poder reinar tanto en su país, como en su continente y a nivel mundial y para ello, una parte muy importante de estos proyectos son la salida y entrada de nuevos jugadores.

Gracias a nuestra aplicación, permitiremos a los clubes poder consultar datos sobre jugadores los cuales ellos mismos podrán introducir las características que quieran que ellos tengan para que puedan ficharlos pudiendo observar tanto sus características personales como las económicas.

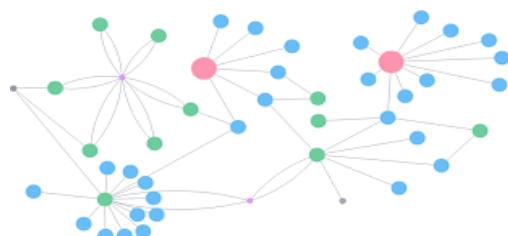
## 2. HERRAMIENTAS UTILIZADAS

Para realizar la aplicación, se nos otorgaba una gran libertad a la hora de elegir lenguaje y tipo de programación (ya sea una aplicación de escritorio o una aplicación web) pero había una sola condición, se debía utilizar el sistema de gestión de bases de datos de grafos NEO4J.

NEO4J permite crear y gestionar bases de datos en forma de grafos, unas bases de datos muy utilizadas en el ámbito empresarial y cada vez más utilizadas a nivel global.

Estas bases de datos nos permiten crear nodos de un tipo de datos, en mi caso nodos de tipo jugador y equipo, y relacionar los nodos por medio de relaciones, aristas que unen dos nodos dándole un significado a la relación. Los nodos pueden tener propiedades, datos que otorgan datos a los nodos con los que luego poder diferenciarlos o para trabajar con los nodos.

Las bases de datos por grafos, a diferencia de las que son relacionalles que utilizan tablas para relacionarse con otras tablas, los nodos no tienen que estar relacionados con otros estrictamente solo por el tipo de nodo que sea, algo que ocurre con las bases de datos relacionalles.



Para usar correctamente NEO4J en nuestro programa, se realizó una búsqueda exhaustiva en una pagina web denominada Kaggle.

Kaggle es una pagina que posee una gran cantidad de datos para proporcionar a los concursantes de las competiciones que propone que suelen basarse en machine learning o en el campo de la ciencia de datos en su mayor medida. Obtuve una base de datos de un popular videojuego denominado “FIFA 21”. Es un archivo con extensión .csv donde los jugadores tenían los datos que necesitaba sin tener campos vacíos o nulos.

id	nombre	nombreCompleto	edad	nacionalidad	puntuacion	potencial
158023	L. Messi	Lionel Messi	33	Argentina	93	93
20801	Cristiano Ronaldo	C. Ronaldo	35	Portugal	92	92
188545	R. Lewandowski	Robert Lewa	31	Poland	91	91
190871	Neymar Jr	Neymar da S	28	Brazil	91	91
192985	K. De Bruyne	Kevin De Bru	29	Belgium	91	91
200389	J. Oblak	Jan Oblak	27	Slovenia	91	93
203376	V. van Dijk	Virgil van Dij	28	Netherlands	90	91
209331	M. Salah	Mohamed Sa	28	Egypt	90	90
231747	K. Mbappé	Kylian Mbap	21	France	90	95
208722	S. Mané	Sadio Mané	28	Senegal	90	90
212831	Alisson	Alisson Ram	27	Brazil	90	91
192448	M. ter Stege	Marc-André	28	Germany	90	93
167495	M. Neuer	Manuel Neu	34	Germany	89	89
200145	Casemiro	Carlos Henrique	28	Brazil	89	89

Aquí tenemos un pequeño ejemplo de los datos que utilizamos para trabajar en la aplicación. El archivo csv viene con datos separados por el separador “;”.

Para poder “jugar” con los datos, esto es, manipularlos y trabajar adecuadamente con ellos, teníamos que introducir los datos del archivo en la base de datos, para ello utilizamos el lenguaje que entiende NEO4J, CYPHER. Este lenguaje nos permite trabajar con las bases de datos de NEO4J de una forma muy cómoda debido a que nos ahorra mucho código respecto a, por ejemplo, sql y además nos proporciona un lenguaje muy similar al lenguaje natural, lo que hace que sea un lenguaje mas sencillo de aprender.

Después de haber hablado de lo referente a la base de datos y las herramientas que conlleva, hablaremos de las herramientas utilizadas en el backend de la aplicación. Para realizar el backend de la aplicación, además de utilizar el lenguaje JavaScript, hemos utilizado NodeJs, que es un entorno de programación en tiempo real utilizado para crear backend en aplicaciones web. Dentro de node js hemos utilizado las siguientes herramientas:

1. Express Js: entorno de programación que nos permite realizar todas las funciones propias de HTTP (get,put,post y delete).
2. Neo4j driver: driver que permite la conexión del backend desde nodejs con la base de datos neo4j y realizar peticiones a datos de la base de datos.

El backend de la aplicación se compone de todo el código necesario para recibir consultas por parte del cliente y se las traslada a la base de datos, realiza la petición y le devuelve los datos la base de datos al backend, que posteriormente se le devolverá al cliente. Para realizar esta aplicación, se utiliza la técnica de programación MVC que después explicaremos mas profundamente.

El frontend este compuesto del código que proporciona una vista con la que el usuario se comunica y le realiza peticiones que posteriormente irán al backend para ser procesadas. Para realizar el frontend se utilizan las siguientes herramientas:

1. VUE js: Framework que permite la creación, de forma sencilla, de aplicaciones web proporcionando gran facilidad de introducción de diferentes librerías para trabajar juntamente con Vue js.
2. Vuetify: Framework que combina vuejs con la estética de material design para poder crear la interfaz de usuario.
3. Axios: Librería de JavaScript que nos permite realizar peticiones HTTP desde el cliente. Lo utilizaremos desde el frontend para realizar las peticiones al backend.

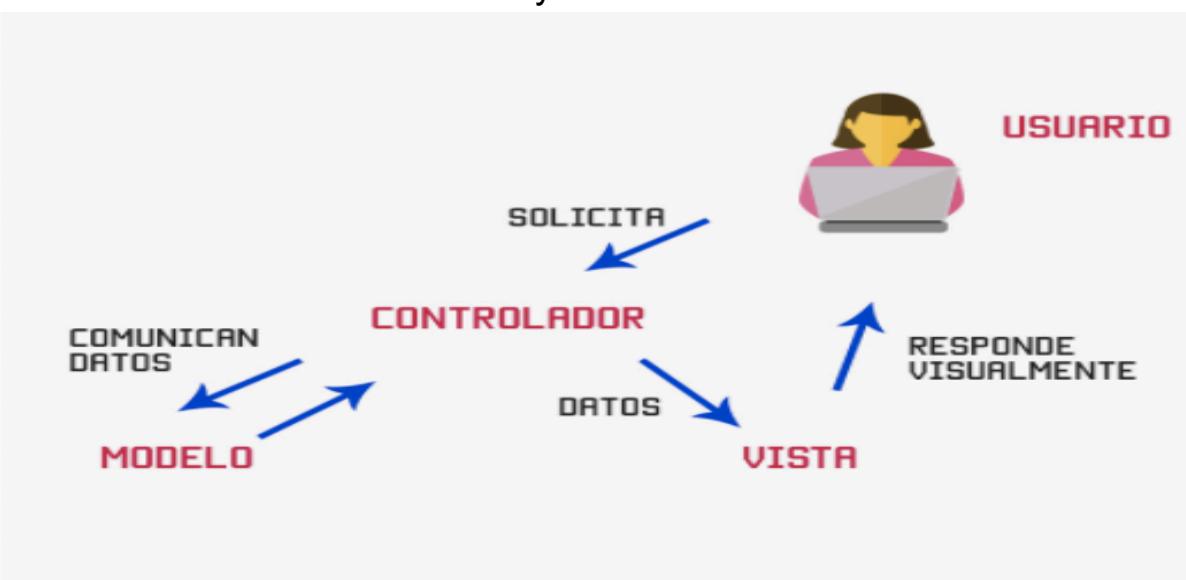
### **3. COMPOSICIÓN DE LA APLICACIÓN**

En esta aplicación, hemos implantado un sistema de programación denominado MVC(modelo vista controlador). En este tipo de sistema, los datos se mueven desde una base de datos o un sistema de

persistencia de datos teniendo una capa entre medias, el modelo.

Explicaremos detenidamente cada parte de este sistema:

- **Modelo:** Es la capa que trabaja con los datos. Esta capa está conectada con la base de datos y es la que trabaja con los datos, es decir, esta capa es quien realiza peticiones a la base de datos para obtener datos, según unos filtros o para realizar cambios en datos dentro de la base de datos.
- **Vista:** Es la capa que contiene el código para visualizar la interfaz de usuario con la que el usuario interactuará para obtener algún tipo de salida. En la vista el usuario introducirá unos datos y a través del controlador, se le pasará al modelo y es quien pedirá los datos oportunos.
- **Controlador:** Su función es recibir las ordenes que el usuario introduce por medio de la interfaz de usuario, en su defecto consola de comandos, y trasladárselas al modelo para que realice las funciones pertinentes con los datos y después se las devuelva al controlador y este a la vista.



## 1. Base de datos

Para la base de datos, hemos utilizado el sistema gestor de bases de datos de grafos Neo4j, que nos permite generar bases de datos en forma de grafos a partir de un archivo con formato csv. La base de datos está compuesta por 19.422 nodos los cuales son de dos tipos:

- **Jugador:** representan un jugador de fútbol de un equipo.  
Todos los nodos de este tipo tienen las siguientes propiedades:

- Edad: Representa la edad de un jugador.
  - Equipo: Representa el equipo donde juega.
  - Id: Es un numero identificador del jugador.
  - Nacionalidad: Representa al país al que pertenece el jugador.
  - Nombre: nombre común o a partir del cual se le conoce al jugador.
  - NombreCompleto: nombre completo del jugador.
  - Posición: Posición en la que el jugador juega en el campo (la abreviatura esta en ingles, en el frontend se traduce al castellano).
  - Potencial: Es la puntuación máxima que podrá alcanzar un jugador en un futuro.
  - Precio: precio, en euros, que cuesta un jugador.
  - Puntuación: es la puntuación actual de un jugador la cual puede mejorar o empeorar en un futuro.
  - Salario: Dinero que cobra un jugador, en euros, por semana.
- Equipo: representa el equipo donde una serie de jugadores juega. Las propiedades que tiene son las siguientes:
  - Id: Identificador el equipo.
  - Nombre: nombre completo del equipo.

Para poder obtener estos datos, se obtienen a partir del archivo, con extensión csv que se encuentra en la carpeta etc “baseDatosFifaCompleto.csv”.

Para cargar este fichero en la base de datos se haría introduciendo este comando:

```

load csv with headers from 'file:///baseDatosFifaCompleto.csv'
as row FIELDTERMINATOR ';'
with row create(j:Jugador)
set j={
    nombre:row.nombre,
    potencial:row.potencial,
    edad:row.edad,
    equipo:row.equipo,
    puntuacion:row.puntuacion,
    salario:row.salarioSemana,
    precio:row.precio,
    posicion:row.posicion,
    foto:row.foto,
    nombreCompleto:row.nombreCompleto,
    nacionalidad:row.nacionalidad,
    id:row.id
}
merge (e:Equipo{nombre:row.equipo})
merge(j)-[:JUEGA_EN]->(e)

```

Este comando realiza las siguientes funciones:

1. Carga el fichero csv con las cabeceras con el comando “load csv headers from...” y le dice a la base de datos que los campos están separados por “;” con la sentencia “FIELDTERMINATOR ‘;’” y guardamos cada fila en la palabra row.
2. Crea un nodo de tipo Jugador llenando cada propiedad con los datos que hay en cada columna de la fila row.
3. Busca si hay un nodo de tipo Equipo con el campo que hay en la columna equipo del jugador.
  - a. Si el equipo no existe, crea un nodo de tipo Equipo con el nombre del equipo y crea una relación JUEGA\_EN desde el nodo Jugador al nodo Equipo.
  - b. Si el equipo existe, toma la referencia del nodo Equipo que contiene el nombre del equipo y realiza la relación JUEGA\_EN desde el nodo Jugador al nodo Equipo.

La base de datos cuenta con 18.741 relaciones que unen los nodos de tipo Jugador a los nodos de tipo Equipo con una relación de tipo JUEGA\_EN.

## 2. Backend

Para implementar el backend, como antes hemos mencionado, hemos utilizado la herramienta Node.js y hemos programado con JavaScript utilizando los drivers de express.js y neo4j.

Todo el backend se encuentra en código en el fichero que se encuentra en la carpeta Backend del proyecto app.js mientras que en el resto de las carpetas de dicho directorio se encuentran las dependencias que necesita el fichero para ejecutarse.

A continuación, vamos a explicar, función por función las tareas que realiza la base de datos:

- dameJugadoresPromesas:

```
app.get("/dameJugadoresPromesas", (req, res) => {
  const session = driver.session();
  var jugadores = [];
  var query = "match(j:Jugador) where j.edad<='22' return j
  order by j.potencial desc limit 15";
  //res.send("Holaaa");
  const resultadoPromesa = session.run(query).subscribe({
    onNext: function (result) {
      //console.log(result.get(0));
      jugadores.push(result.get(0).properties);
    },
    onCompleted: function () {
      res.send(jugadores);
      //session.close();
    },
    onError: function (error) {
      //console.log(error + " erroooooooooooooor");
    }
  })
  //res.send(query);
});
```

En esta función se realizan los siguientes pasos para alcanzar nuestro objetivo: poder mandar al frontend una lista con los diez jugadores cuyo potencial sea mayor y que su edad este entre 15 y 22 años:

Una vez que tenemos la consulta, se la mandamos a la base de datos con el método run de la session que teníamos abierta. Este método nos devuelve 3 situaciones:

onNext: nos devuelve fila a fila la tabla con todos los nodos desde la base de datos. Lo que hago es guardar cada fila, cogiendo sus propiedades, en la lista que voy a devolver al frontend.

onCompleted: lo devuelve cuando ya termine de devolver todos los datos referentes a la consulta por lo que dentro de esta función podremos devolver al frontend la lista con los datos devueltos de la base de datos.

onError: es devuelto cuando se produce un error con la base de datos.

- dameLosMejores:

```
app.get("/dameLosMejores", (req, res) => {
  const session = driver.session();
  var lista = [];
  var query = "match(j:Jugador) return j order by j.puntuacion desc limit 10";
  //res.send(query);
  const resultadoPromesa = session.run(query).subscribe({
    onNext: function (result) {
      lista.push(result.get(0).properties);
    },
    onCompleted: function () {
      res.send(lista);
      session.close();
    },
    onError: function (error) {
      console.log(error + " erroooooooooor");
    }
  });
});
```

La finalidad de esta función es la de devolver los mejores jugadores de la base de datos. Para devolver los mejores, lo que hacemos es decirle que devuelve los jugadores que mejor puntuación tienen devolviendo los 10 mejores jugadores. Para ello, se realizan las mismas funciones que la función anterior ya que guarda en la lista las filas que va devolviendo la base de datos y cuando acaba, la función devuelve la lista al frontend.

- dameJugadoresAleatorios:

```

app.get("/dameJugadoresAleatorios", (req, res)=>{
    const session = driver.session();
    var lista=[];
    var min=60;
    var max=95;
    var potencial=Math.floor((Math.random()*(max+1)-min)+min)
    );
    var query="match(j:Jugador) where j.potencial='"+potencial
    +"' return j order by j.puntuacion desc limit 10";
    console.log(/*query*/"Estoy en funcion dame jugadores
    aleatorios");
    const resultadoPromesa = session.run(query).subscribe({
        onNext: function (result) {
            lista.push(result.get(0).properties);
        },
        onCompleted: function () {
            res.send(lista);
            session.close();
            //res.send(query);
        },
        onError: function (error) {
            console.log(error + " errorooooooooooooor");
        }
    })
});

```

La finalidad de esta función es la de devolver jugadores sin una dinámica fija mas que elegir un numero aleatorio, que representara el potencial y en base a ese numero aleatorio, se le pedirá a la base de datos que devuelva los jugadores, como en los casos anteriores, en una lista.

- dameNacionalidades:

```

app.get("/dameNacionalidades", (req, res) =>{
    const session = driver.session();
    console.log("Estoy en dame nacionalidades");
    var nacionalidades=[];
    var query="match(j:Jugador) return distinct j.nacionalidad"
    const resultadoPromesa = session.run(query).subscribe({
        onNext: function (result) {
            //console.log(result.get(0));
            nacionalidades.push(result.get(0));
        },
        onCompleted: function () {
            //console.log(nacionalidades.length);
            res.send(nacionalidades);
            session.close();
            /*res.send(query);*/
        },
        onError: function (error) {
            console.log(error + " erroooooooooooooor");
        }
    })
});

```

Esta función se utiliza para poder tener en un desplegable en el frontend todas las nacionalidades que hay en la base de datos. Al igual que en las funciones anteriores, se envía una consulta a la base donde le decimos que queremos las nacionalidades distintas para que no haya repeticiones y cuando esas nacionalidades están guardadas en la lista, podemos enviársela al frontend.

- dameEquipos:

```

app.get("/dameEquipos", (req, res) =>{
    const session = driver.session();
    console.log("Estoy en dame Equipos");
    var equipos=[];
    var query="match(j:Jugador) return distinct j.equipo";
    const resultadoPromesa = session.run(query).subscribe({
        onNext: function (result) {
            //console.log(result.get(0));
            equipos.push(result.get(0));
        },
        onCompleted: function () {
            //console.log(nacionalidades.length);
            res.send(equipos);
            session.close();
            /*res.send(query);*/
        },
        onError: function (error) {
            console.log(error + " erroooooooooor");
        }
    })
});

```

Al igual que la función dameNacionalidades, el objetivo de esta función es el de poder mostrar, en un desplegable del frontend, los distintos equipos que hay en la base de datos por lo que le pedimos en la consulta que devuelva equipos para que no se produzca repeticiones y que después envíe la lista con los equipos.

- dameJugadoresConFiltros:

```

app.get("/dameJugadoresConFiltros", (req,res)=>{
  console.log("ESTAMOS EN DAME Jugadores con filtros");

  const session = driver.session();

  var nacionalidad=req.query.nacionalidad;
  var posicion=req.query.posicion;
  var equipo=req.query.equipo;
  var puntuacion=req.query.puntuacion;
  var potencial=req.query.potencial;
  var salario=req.query.salario;
  var precio=req.query.precio;
  var edad=req.query.edad;

  var lista=[];
  var query = "match(j:Jugador) ";
  //console.log("LA NACIONALIDAD ES:"+nacionalidad);
  //console.log("LA EDAD MAXIMA ES:"+edad);
  //res.send("Servidor contesta edad maxima:"+edad);

  if(nacionalidad!="cualquiera"){
    query+="where j.nacionalidad='"+nacionalidad+"'";
  }
  if(posicion!="cualquiera"){
    if(query.includes("where")){
      query+="AND j.posicion='"+posicion+"' ";
    }else{
      query+="where j.posicion='"+posicion+"' ";
    }
  }
  if(equipo!="cualquiera"){
    if(query.includes("where")){
      query+="AND j.equipo='"+equipo+"' ";
    }else{
      query+="where j.equipo='"+equipo+"' ";
    }
  }
}

```

```

        if(puntuacion!="cualquiera"){
            if(query.includes("where")){
                query+="AND j.puntuacion <= '"+puntuacion+"' ";
            }else{
                query+="where j.puntuacion <= '"+puntuacion+"' ";
            }
        }
        if(potencial!="cualquiera"){
            if(query.includes("where")){
                query+="AND j.potencial<='"+potencial+"' ";
            }else{
                query+="where j.potencial<='"+potencial+"' ";
            }
        }
        if(salario!="cualquiera"){
            if(query.includes("where")){
                query+="AND j.salario<='"+salario+"' ";
            }else{
                query+="where j.salario<='"+salario+"' ";
            }
        }
        if(precio!="cualquiera"){
            if(query.includes("where")){
                query+="AND j.precio<='"+precio+"' ";
            }else{
                query+="where j.precio<='"+precio+"' ";
            }
        }
        if(edad!="cualquiera"){
            if(query.includes("where")){
                query+="AND j.edad<='"+edad+"' ";
            }else{
                query+="where j.edad<='"+edad+"' ";
            }
        }
        if(query.includes("where")==false){
            var num=Math.floor((Math.random()*(97+1)-70)+70));
            query+="where j.potencial<='"+num+"' ";
            //console.log("NUMERO:"+num);
        }
    }

    query+="return j order by j.puntuacion desc limit 15";
    const resultadoPromesa = session.run(query).subscribe({
        onNext: function (result) {
            console.log(result.get(0).properties);
            lista.push(result.get(0).properties);
        },
        onCompleted: function () {
            if(lista.length==0){
                lista.push("Según esos filtros, no hay jugadores");
            }
            res.send(lista);
            console.log(lista);
            console.log(query);
            session.close();
            //res.send(query);
        },
        onError: function (error) {
            console.log(error + " erroooooooooor");
        }
    })
});

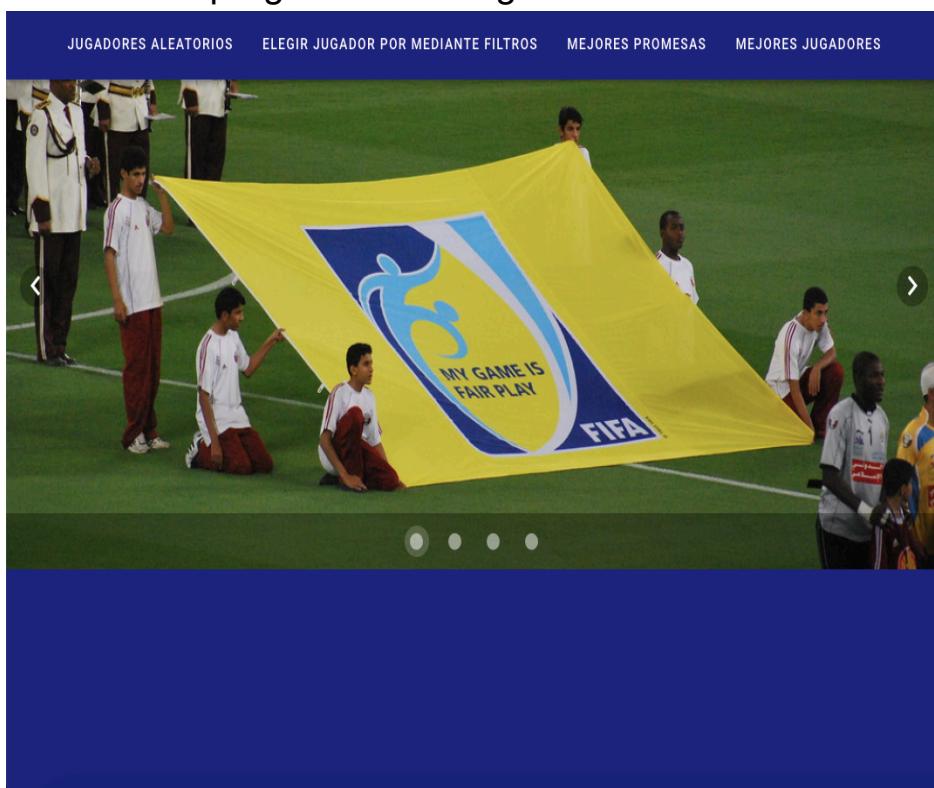
```

Esta función es la que mas trabajo a nivel de desarrollo y a nivel computacional genera. En esta función, la finalidad es la de devolver una serie de jugadores según unos criterios que el propio usuario define desde el frontend.

### 3. Frontend

El frontend ha sido creado gracias a la herramienta denominada Vuetify para crear interfaces web de usuario para que el usuario pueda interactuar con la interfaz. Vuetify se usa a partir de Vue Js ya que es el framework original que lo hace funcionar. Para el frontend, también se utiliza JavaScript y axios para permitir las peticiones desde Frontend al Backend y viceversa utilizando peticiones HTTP.

La interfaz que presentamos es muy sencilla para que el usuario pueda utilizarla sin ningún tipo de problema. La primera imagen que tenemos cuando iniciamos el programa es la siguiente:

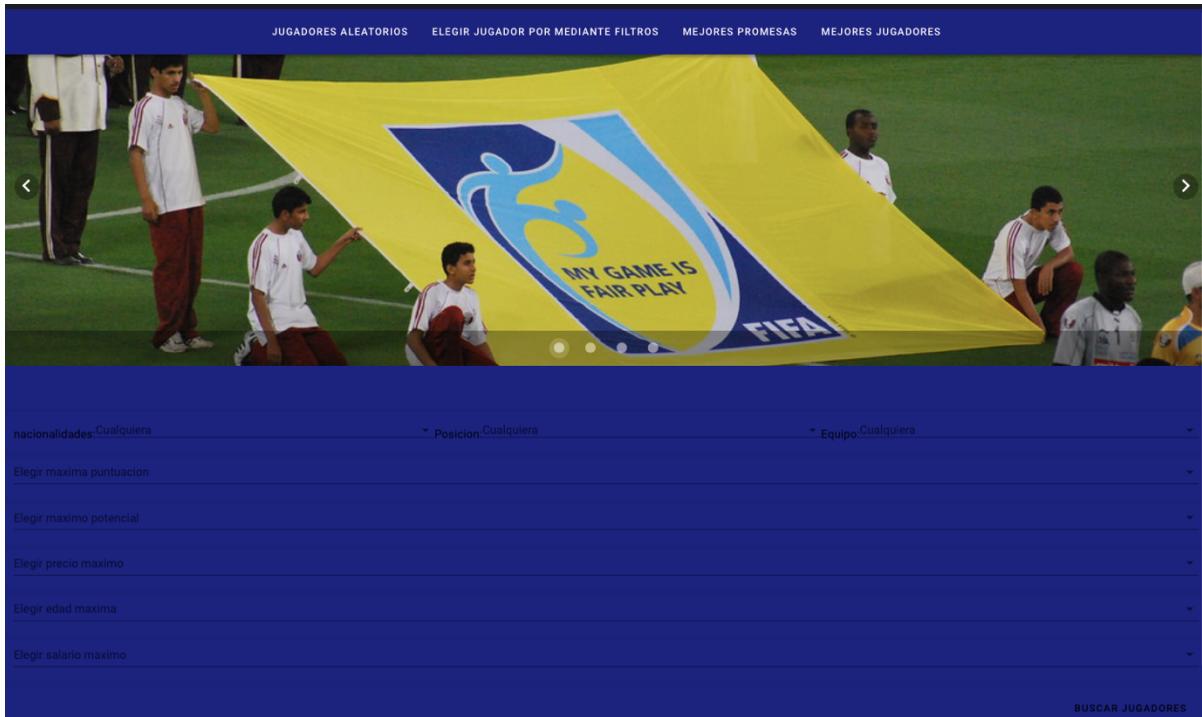


Como podemos observar, la interfaz es de color azul y se divide, en un primer lugar en 3 partes como son la barra de navegación que tenemos arriba con los 4 botones, el carrousel de imágenes que ofrezco y debajo irán las diferentes funcionalidades que ofrezco.

Según se pulse cada botón, excepto el botón “Elegir jugadores por filtros” que explicaremos mas adelante, son botones que y nos dan la funcionalidad directamente mostrándonos los datos de los jugadores directamente. Los datos de los jugadores se mostrarán en unos tipos de objetos denominados “Cartas”. Gracias a estas cartas, podremos, en

espacios pequeños, incluir la información de cada jugador seleccionado. Pulsado cualquiera de los botones, se nos devolverá una serie de cartas con información de los jugadores. Tal y como esta planteada la interfaz, es posible mostrar la información de dos funcionalidades, esto es, puedo pulsar “Mejores Jugadores” y después “Mejores Promesas” y se me mostraran ambos tipos de jugadores.

Hablaremos un poco de como esta creada la parte que involucra los filtros. Es la parte mas sensible de la aplicación debido a que filtraremos datos según una gran cantidad de parámetros (todos los que elija el usuario introducir). Según pulsamos el botón “Mostrar por parámetros”, nos encontramos con esta imagen:



Podemos observar que tenemos una serie de parámetros que será opción del usuario introducir los que quiera, introducir todos o no introducir ninguno. Pasamos a explicar cada uno para saber el funcionamiento de cada uno:

- Nacionalidad: desplegable que nos muestra las distintas nacionalidades que tienen los jugadores que se encuentran en la base de datos.
- Posiciones: Desplegable que nos muestra la abreviatura (en castellano) de las posiciones que hay en el campo en futbol 11.
- Equipos: desplegable que nos muestra los distintos equipos que podemos encontrar en la base de datos.

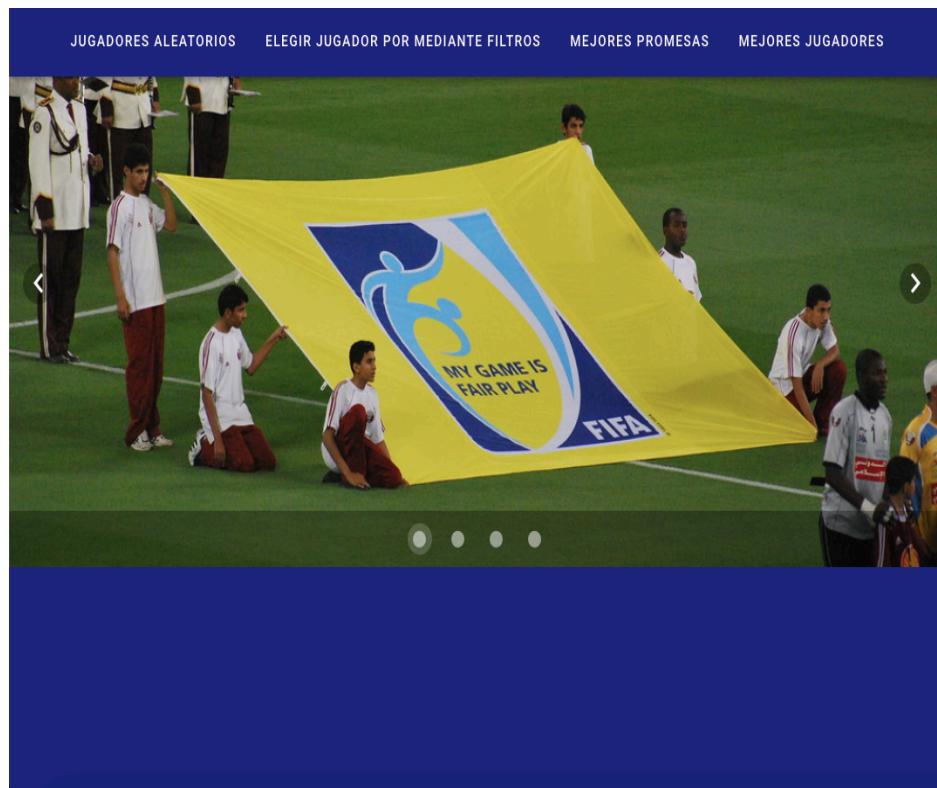
En los sucesivos parámetros posibles, se realiza una doble funcionalidad, esto es, primero nos encontramos con un desplegable que nos muestra la opción Si o la opción No para saber si queremos utilizar ese parámetro en concreto. Si la opción elegida es “Si”, se nos mostrara una slider que nos permitirá introducir un numero determinado según la opción. El desplegable del que hablamos anteriormente seguirá estando presente por si el usuario se arrepintiera y quisiera no utilizar este parámetro.

- Elegir máxima puntuación: nos permite elegir la puntuación mas alta que queremos que tenga el jugador o los jugadores que se nos muestre.
- Elegir máximo potencial: nos permitirá elegir el potencial máximo que tendrán los jugadores que se nos presenten.
- Elegir precio máximo: nos permitirá elegir el precio máximo que se podría gastar en un jugador y, por ende, mostraría el precio mas alto que podría costar un jugador.
- Elegir edad máxima: nos permite introducir la edad mas alta que puede tener un jugador.
- Elegir máximo salario: nos permite introducir cual debe ser el salario mas alto que podrá tener un jugador como mucho.

Una vez elegidos los parámetros, podemos pulsar el botón “Buscar Jugadores” y nos mostrara, como máximo 15 jugadores si es que se encuentran mas de 15, o en su defecto, se mostrara un mensaje de error en caso de haber introducido parámetros que no se asemejen a los datos de algún jugador.

#### **4. ANALISIS DE RESULTADOS**

Cuando se ejecuta la aplicación, podemos encontrar esta pagina de inicio:



Cada uno de los botones que tenemos en la barra de arriba, es una función que desarrolla la aplicación. A continuación, vamos a explicar cada función que contiene la aplicación web utilizando ejemplos en el caso de mostrar la base de datos con filtros.

- Mostrar jugadores aleatorios:

Su función es la de mostrar, bajo un parámetro que elige el backend sin que el frontend lo sepa, una serie de jugadores de forma aleatoria.

Puede ser muy práctico en el caso de que un entrenador necesite un jugador sin tener en cuenta las características, simplemente quiere un jugador. La ejecución del programa nos daría esta salida:

<b>Nombre completo:</b> Sean Scannell	<b>Nombre completo:</b> Flamur Kastrati	<b>Nombre completo:</b> Pascal Sohm
ID:186422 Nombre Completo:Sean Scannell Nacionalidad:Republic of Ireland Precio:.450.000 Salario:3.000 Puntuacion:64 Potencial:64 Edad:29 Equipo Actual:Grimsby Town Posicion:CB/Defensa central	ID:190518 Nombre Completo:Flamur Kastrati Nacionalidad:Kosovo Precio:.475.000 Salario:1.000 Puntuacion:64 Potencial:64 Edad:28 Equipo Actual:Kristiansund BK Posicion:CAM/Mediocentro ofensivo	ID:238686 Nombre Completo:Pascal Sohm Nacionalidad:Germany Precio:.475.000 Salario:3.000 Puntuacion:64 Potencial:64 Edad:28 Equipo Actual:SG Dynamo Dresden Posicion:CB/Defensa central
<b>Nombre completo:</b> Adrián Bone	<b>Nombre completo:</b> Soo Beom Kim	<b>Nombre completo:</b> Kennedy Igboananike
ID:212022 Nombre Completo:Adrián Bone Nacionalidad:Ecuador Precio:.300.000 Salario:500 Puntuacion:64 Potencial:64 Edad:31 Equipo Actual:Emelec Posicion:RM/Medio derecho	ID:202292 Nombre Completo:Soo Beom Kim Nacionalidad:Korea Republic Precio:.375.000 Salario:2.000 Puntuacion:64 Potencial:64 Edad:29 Equipo Actual:Gangwon FC Posicion:CB/Defensa central	ID:197941 Nombre Completo:Kennedy Igboananike Nacionalidad:Nigeria Precio:.400.000 Salario:1.000 Puntuacion:64 Potencial:64 Edad:31 Equipo Actual:IK Sirius Posicion:GK/Portero

Podemos observar que nos devuelve una serie de jugadores con la característica que la puntuación es la misma en todos los jugadores por lo que podemos interpretar que el parámetro aleatorio es la puntuación.

- Mostrar jugadores en base a filtros:

En este caso, podemos obtener los jugadores introduciendo una serie de parámetros en un apartado que se nos muestra cuando pulsamos el botón que nos abre los filtros. Dichos filtros son los siguientes:

nacionalidades: Cualquiera ▾ Posicion: Cualquiera ▾ Equipo: Cualquiera ▾

Elegir maxima puntuacion ▾

Elegir maximo potencial ▾

Elegir precio maximo ▾

Elegir edad maxima ▾

Elegir salario maximo ▾

BUSCAR JUGADORES

Debido a que podemos introducir múltiples combinaciones de parámetros, vamos a poner 3 ejemplos para mostrar como muestra las salidas:

- 1) Mostrar Jugadores que sean españoles: Para realizar dicha tarea, lo único que necesitamos es seleccionar, en el desplegable, que la nacionalidad sea “Spain”. Realizando este paso, obtenemos la siguiente salida:

Cualquiera	Posicion:Cualquiera	Equipo.Cualquiera
nacionalidades: Spain	Elegir maxima puntuacion	Elegir maximo potencial
	Elegir maximo precio	Elegir edad maxima
	Elegir salario maximo	
<b>Nombre completo:Sergio Ramos García</b>	<b>Nombre completo:Sergio Busquets Burgos</b>	
ID:155862	ID:189511	
Nombre Completo:Sergio Ramos García	Nombre Completo:Sergio Busquets Burgos	
Nacionalidad:Spain	Nacionalidad:Spain	
Precio:24.500.000	Precio:38.000.000	
Salario:.300.000	Salario:.240.000	
Puntuacion:89	Puntuacion:87	
Potencial:89	Potencial:87	
Edad:34	Edad:31	
Equipo Actual:Real Madrid	Equipo Actual:FC Barcelona	
Posicion:CB/Defensa central	Posicion:CDM/Mediocentro defensivo	
<b>Nombre completo:David Josué Jiménez Silva</b>	<b>Nombre completo:Jordi Alba Ramos</b>	
ID:168542	ID:189332	
Nombre Completo:David Josué Jiménez Silva	Nombre Completo:Jordi Alba Ramos	
Nacionalidad:Spain	Nacionalidad:Spain	
Precio:22.500.000	Precio:32.000.000	
Salario:.58.000	Salario:.220.000	
Puntuacion:86	Puntuacion:86	
Potencial:86	Potencial:86	
Edad:34	Edad:31	
Equipo Actual:Real Sociedad	Equipo Actual:FC Barcelona	
Posicion:CAM/Mediocentro ofensivo	Posicion:LB/Lateral izquierda	

- 2) Mostrar jugadores españoles que tengan una puntuación igual o menor a 85 y que su salario sea igual o menos que 112.000 euros semanales:

Cualquiera  
nacionalidades: Spain ▾ Posicion:Cualquiera ▾ Equipo:Cualquiera ▾

Maxima puntuacion  si Elegir maxima puntuacion

Elegir maximo potencial

Elegir precio maximo

Elegir edad maxima

Maximo Salario  si Elegir salario maximo

**BUSCAR JUGADORES**

<b>Nombre completo:</b> Álvaro Borja Morata Martín	<b>Nombre completo:</b> Pablo Sarabia García
ID:201153	ID:198950
Nombre Completo:Álvaro Borja Morata Martín	Nombre Completo:Pablo Sarabia García
Nacionalidad:Spain	Nacionalidad:Spain
Precio:26.000.000	Precio:18.000.000
Salario:.100.000	Salario:.100.000
Puntuacion:82	Puntuacion:81
Potencial:83	Potencial:81
Edad:27	Edad:28
Equipo Actual:Juventus	Equipo Actual:Paris Saint-Germain
Posicion:ST/Delantero centro	Posicion:CB/Defensa central

<b>Nombre completo:</b> Héctor Junior Firpo Adamés
ID:241184
Nombre Completo:Héctor Junior Firpo Adamés
Nacionalidad:Spain
Precio:15.500.000
Salario:.110.000
Puntuacion:79
Potencial:85
Edad:23
Equipo Actual:FC Barcelona
Posicion:GK/Portero

- 3) Mostrar a Leo Messi: Para este caso, sabemos que Messi tiene 33 años, su nacionalidad es argentina y juega en el FC Barcelona por lo que, introduciendo esos parámetros, deberá aparecer en la salida de pantalla:

Cualquiera  
nacionalidades: Argentina ▼ Posicion: Cualquiera ▼ Equipo: FC Barcelona ▼

Elegir maxima puntuacion  
no

Elegir maximo potencial

Elegir precio maximo

Maxima edad ————— Elegir edad maxima  
si

Elegir salario maximo  
no

**BUSCAR JUGADORES**

**Nombre completo:Lionel Messi**

ID:158023  
Nombre Completo:Lionel Messi  
Nacionalidad:Argentina  
Precio:67.500.000  
Salario:.560.000  
Puntuacion:93  
Potencial:93  
Edad:33  
Equipo Actual:FC Barcelona  
Posicion:RW/Extremo derecho

- Mostrar los mejores jugadores:

Esta función nos permite que la aplicación nos muestre cuales son los mejores jugadores en base a la puntuación que tenga. Nos mostrara de golpe los 10 mejores jugadores que estén en la base de datos en orden de puntuación.

<b>Nombre completo:Lionel Messi</b> ID:158023 Nombre Completo:Lionel Messi Nacionalidad Argentina Precio: 67.500.000 Salario: .560.000 Puntuacion:93 Potencial:93 Edad:33 Equipo Actual:FC Barcelona Posicion:RW/Extremo derecho	<b>Nombre completo:C. Ronaldo dos Santos Aveiro</b> ID:20801 Nombre Completo:C. Ronaldo dos Santos Aveiro Nacionalidad:Portugal Precio:46.000.000 Salario: .220.000 Puntuacion:92 Potencial:92 Edad:35 Equipo Actual:Juventus Posicion:ST/Delantero centro	<b>Nombre completo:Jan Oblak</b> ID:200389 Nombre Completo:Jan Oblak Nacionalidad:Slovenia Precio:75.000.000 Salario: .125.000 Puntuacion:91 Potencial:93 Edad:27 Equipo Actual:Atlético Madrid Posicion:GK/Portero	<b>Nombre completo:Kevin De Bruyne</b> ID:192985 Nombre Completo:Kevin De Bruyne Nacionalidad:Belgium Precio:87.000.000 Salario: .370.000 Puntuacion:91 Potencial:91 Edad:29 Equipo Actual:Manchester City Posicion:CAM/Mediocentro ofensivo	
<b>Nombre completo:Neymar da Silva Santos Jr.</b> ID:190871 Nombre Completo:Neymar da Silva Santos Jr. Nacionalidad:Brazil Precio: 90.000.000 Salario: .270.000 Puntuacion:91 Potencial:91 Edad:28 Equipo Actual:Paris Saint-Germain Posicion:LW/Extremo izquierdo	<b>Nombre completo:Robert Lewandowski</b> ID:188545 Nombre Completo:Robert Lewandowski Nacionalidad:Poland Precio:80.000.000 Salario: .240.000 Puntuacion:91 Potencial:91 Edad:31 Equipo Actual:FC Bayern München Posicion:ST/Delantero centro	<b>Nombre completo:Virgil van Dijk</b> ID:203376 Nombre Completo:Virgil van Dijk Nacionalidad:Netherlands Precio:75.500.000 Salario: .210.000 Puntuacion:90 Potencial:91 Edad:28 Equipo Actual:Liverpool Posicion:CB/Defensa central	<b>Nombre completo:Sadio Mané</b> ID:208722 Nombre Completo:Sadio Mané Nacionalidad:Senegal Precio:78.000.000 Salario: .250.000 Puntuacion:90 Potencial:90 Edad:28 Equipo Actual:Liverpool Posicion:LW/Extremo izquierdo	<b>Nombre completo:Kylian Mbappé</b> ID:231747 Nombre Completo:Kylian Mbappé Nacionalidad:France Precio: 105.500.000 Salario: .250.000 Puntuacion:90 Potencial:95 Edad:21 Equipo Actual:Paris Saint-Germain Posicion:ST/Delantero centro
<b>Nombre completo:Mahomed Salah</b> ID:209331 Nombre Completo:Mahomed Salah Nacionalidad:Egypt Precio: 78.000.000 Salario: .250.000 Puntuacion:90 Potencial:90 Edad:28 Equipo Actual:Liverpool Posicion:RW/Extremo derecho				
<b>Nombre completo:Matthijs de Ligt</b> ID:235243 Nombre Completo:Matthijs de Ligt Nacionalidad:Netherlands Precio: 49.500.000 Salario: .30.000.000 Puntuacion:85 Potencial:92 Edad:20 Equipo Actual:Juventus Posicion:CB/Defensa central	<b>Nombre completo:João Félix Sequeira</b> ID:245444 Nombre Completo:João Félix Sequeira Nacionalidad:Portugal Precio:32.000.000 Salario: .51.000 Puntuacion:81 Potencial:93 Edad:21 Equipo Actual:Atlético Madrid Posicion:CAM/Mediocentro ofensivo	<b>Nombre completo:Jadon Sancho</b> ID:233049 Nombre Completo:Jadon Sancho Nacionalidad:England Precio:69.500.000 Salario: .82.000 Puntuacion:87 Potencial:93 Edad:21 Equipo Actual:Bayern München Posicion:CAM/Mediocentro ofensivo	<b>Nombre completo:Vinicius José de Oliveira Júnior</b> ID:238744 Nombre Completo:Vinicius José de Oliveira Júnior Nacionalidad:Brazil Precio:27.500.000 Salario: .95.000 Puntuacion:80 Potencial:93 Edad:17 Equipo Actual:Real Madrid Posicion:ST/Delantero centro	<b>Nombre completo:Kai Havertz</b> ID:235790 Nombre Completo:Kai Havertz Nacionalidad:Germany Precio: 57.000.000 Salario: .105.000 Puntuacion:85 Potencial:93 Edad:21 Equipo Actual:Chelsea Posicion:CAM/Mediocentro ofensivo
<b>Nombre completo:Francisco Mota Castro Trincão</b> ID:244778 Nombre Completo:Francisco Mota Castro Trincão Nacionalidad:Portugal Precio:20.000.000 Salario: .105.000 Puntuacion:78 Potencial:91 Edad:20 Equipo Actual:FC Barcelona Posicion:RM/Medio derecho	<b>Nombre completo:Marcus Rashford</b> ID:230985 Nombre Completo:Erling Haaland Nacionalidad:Norway Precio:41.500.000 Salario: .100.000 Puntuacion:84 Potencial:92 Edad:19 Equipo Actual:Bayern München Posicion:ST/Delantero centro	<b>Nombre completo:Lautaro Martínez</b> ID:231621 Nombre Completo:Lautaro Martínez Nacionalidad:Italy Precio:41.500.000 Salario: .34.000 Puntuacion:85 Potencial:92 Edad:21 Equipo Actual:Milan Posicion:GK/Portero	<b>Nombre completo:Trent Alexander-Arnold</b> ID:231281 Nombre Completo:Trent Alexander-Arnold Nacionalidad:England Precio:60.000.000 Salario: .110.000 Puntuacion:87 Potencial:92 Edad:21 Equipo Actual:Liverpool Posicion:RB/Lateral derecho	<b>Nombre completo:Sandro Tonali</b> ID:241096 Nombre Completo:Sandro Tonali Nacionalidad:Italy Precio:18.500.000 Salario:20.000.000 Puntuacion:77 Potencial:91 Edad:20 Equipo Actual:Milan Posicion:CAM/Mediocentro ofensivo
<b>Nombre completo:Rodrygo Silva de Goes</b> ID:248812 Nombre Completo:Rodrygo Silva de Goes Nacionalidad:Brazil Precio:21.000.000 Salario: .90.000 Puntuacion:79 Potencial:90 Edad:20 Equipo Actual:Real Madrid Posicion:ST/Delantero centro				

- Mostrar los jugadores promesas:

Esta función nos permite que la aplicación web nos muestre los mejores jugadores del futuro, esto es, en base a la edad del jugador (jugadores muy jóvenes) y en base al potencial que tengan (valor futuro que puede tener su puntuación), nos devolverá una serie de jugadores como veremos a continuación:

<b>Nombre completo:Kylian Mbappé</b> ID:231747 Nombre Completo:Kylian Mbappé Nacionalidad:France Precio: 105.500.000 Salario: .160.000 Puntuacion:90 Potencial:95 Edad:21 Equipo Actual:Paris Saint-Germain Posicion:ST/Delantero centro	<b>Nombre completo:João Félix Sequeira</b> ID:245444 Nombre Completo:João Félix Sequeira Nacionalidad:Portugal Precio:32.000.000 Salario: .51.000 Puntuacion:81 Potencial:93 Edad:21 Equipo Actual:Atlético Madrid Posicion:CAM/Mediocentro ofensivo	<b>Nombre completo:Jadon Sancho</b> ID:233049 Nombre Completo:Jadon Sancho Nacionalidad:England Precio:69.500.000 Salario: .82.000 Puntuacion:87 Potencial:93 Edad:21 Equipo Actual:Bayern München Posicion:CAM/Mediocentro ofensivo	<b>Nombre completo:Vinicius José de Oliveira Júnior</b> ID:238744 Nombre Completo:Vinicius José de Oliveira Júnior Nacionalidad:Brazil Precio:27.500.000 Salario: .95.000 Puntuacion:80 Potencial:93 Edad:17 Equipo Actual:Real Madrid Posicion:ST/Delantero centro	<b>Nombre completo:Kai Havertz</b> ID:235790 Nombre Completo:Kai Havertz Nacionalidad:Germany Precio: 57.000.000 Salario: .105.000 Puntuacion:85 Potencial:93 Edad:21 Equipo Actual:Chelsea Posicion:CAM/Mediocentro ofensivo
<b>Nombre completo:Matthijs de Ligt</b> ID:235243 Nombre Completo:Matthijs de Ligt Nacionalidad:Netherlands Precio: 49.500.000 Salario: .30.000.000 Puntuacion:85 Potencial:92 Edad:20 Equipo Actual:Juventus Posicion:CB/Defensa central	<b>Nombre completo:Erling Haaland</b> ID:230985 Nombre Completo:Erling Haaland Nacionalidad:Norway Precio:41.500.000 Salario: .100.000 Puntuacion:84 Potencial:92 Edad:19 Equipo Actual:Bayern München Posicion:ST/Delantero centro	<b>Nombre completo:Gianluigi Donnarumma</b> ID:231621 Nombre Completo:Gianluigi Donnarumma Nacionalidad:Italy Precio:41.500.000 Salario: .34.000 Puntuacion:85 Potencial:92 Edad:21 Equipo Actual:Milan Posicion:GK/Portero	<b>Nombre completo:Trent Alexander-Arnold</b> ID:231281 Nombre Completo:Trent Alexander-Arnold Nacionalidad:England Precio:60.000.000 Salario: .110.000 Puntuacion:87 Potencial:92 Edad:21 Equipo Actual:Liverpool Posicion:RB/Lateral derecho	<b>Nombre completo:Sandro Tonali</b> ID:241096 Nombre Completo:Sandro Tonali Nacionalidad:Italy Precio:18.500.000 Salario:20.000.000 Puntuacion:77 Potencial:91 Edad:20 Equipo Actual:Milan Posicion:CAM/Mediocentro ofensivo
<b>Nombre completo:Francisco Mota Castro Trincão</b> ID:244778 Nombre Completo:Francisco Mota Castro Trincão Nacionalidad:Portugal Precio:20.000.000 Salario: .105.000 Puntuacion:78 Potencial:91 Edad:20 Equipo Actual:FC Barcelona Posicion:RM/Medio derecho	<b>Nombre completo:Marcus Rashford</b> ID:231677 Nombre Completo:Marcus Rashford Nacionalidad:England Precio:53.000.000 Salario: .150.000 Puntuacion:85 Potencial:91 Edad:22 Equipo Actual:Manchester United Posicion:RM/Medio derecho	<b>Nombre completo:Lautaro Martínez</b> ID:231678 Nombre Completo:Lautaro Martínez Nacionalidad:Argentina Precio:44.500.000 Salario: .130.000 Puntuacion:84 Potencial:91 Edad:22 Equipo Actual:Inter Posicion:ST/Delantero centro	<b>Nombre completo:Federico Valverde</b> ID:239953 Nombre Completo:Federico Valverde Nacionalidad:Uruguay Precio:36.000.000 Salario: .135.000 Puntuacion:83 Potencial:90 Edad:22 Equipo Actual:Real Madrid Posicion:CM/Mediocentro	<b>Nombre completo:Rodrygo Silva de Goes</b> ID:248812 Nombre Completo:Rodrygo Silva de Goes Nacionalidad:Brazil Precio:21.000.000 Salario: .90.000 Puntuacion:79 Potencial:90 Edad:20 Equipo Actual:Real Madrid Posicion:ST/Delantero centro

## 5. DAFO

- Debilidades:

Cabe mencionar que el algoritmo utilizado no emplea grandes dificultades y emplea algoritmos sencillos, aunque son diferentes para cada función que ofrece el sistema lo que produce que haya mas mantenimiento de la aplicación. Otra debilidad que puede presentar son los colores utilizados a la hora de realizar la interfaz

dado que mucha gente puede no entender porque he realizado esa composición.

- Amenazas:

Esta aplicación, a mi modo de ver, puede ser realizada por mucha gente y mas teniendo en cuenta que la base de datos se puede coger en KAGGLE además de que esta aplicación, como la base de datos no se actualiza, es posible que, como los jugadores van cumpliendo años, no muestre los jugadores correctamente en algún parámetro porque por ejemplo si un jugador tiene 25 años, pero cuando se hizo la base de datos tenia 24, aparecerá con 24 años.

- Fortaleza:

Es una aplicación muy intuitiva y fácil de utilizar con diferentes funciones, lo que hace que el usuario, normalmente un entrenador o dueño de un equipo de futbol, tenga muchas mas posibilidades de encontrar determinados jugadores según los intereses y objetivos que tengan. La interfaz, aunque presenta colores un poco extraños para quien la utiliza, esta creada de una forma simple, lo que hace que, para realizar cualquier función, se tenga a primera vista. El algoritmo de búsqueda tarda un tiempo ínfimo en mostrar los datos, algo muy significativo debido a la cantidad de datos que tiene la base de datos y todo ello es gracias al motor de búsqueda de neo4j.

- Oportunidades:

Contiene una base de datos muy amplia, seguramente con la capacidad de ofrece cualquier jugador que ahora tenga una ficha con un equipo de futbol profesional de cualquier liga. Gracias a que la base de datos que utiliza es la proporcionada como base de un videojuego muy relacionado al organismo que dirige el futbol, cada año, esta base de datos se va a reordenar y actualizar, lo que facilita que, actualizando esta base de datos en el programa, podamos utilizar siempre el mismo programa para recomendar, aunque actualizando ciertas partes.

## **6. LÍNEAS DE FUTURO**

Buscando en páginas de internet, he observado que, a menos que sea en un videojuego, ya sea biwenger, futmondo, comunio, el propio FIFA etc, no existen aplicaciones de recomendación de jugadores a modo real para los entrenadores. Si que es cierto que todos los entrenadores o dirigentes pueden utilizar herramientas, pero no he encontrado como la que hice, aunque posiblemente existan, aunque sea con otras funcionalidades.

Debido a como está la situación actual, incluso los grandes clubes, por supuesto sin desmerecer a los demás clubes demostrando tanta grandeza como los grandes clubes, necesitan saber como manejar su dinero con mucha cautela debido al descenso de ingresos que se está produciendo debido a la crisis sanitaria y económica futura que se producirá.

Podemos observar que la tecnología, hoy en día, cada vez está siendo más utilizada. Casos son las google glass que determinados entrenadores utilizaban para ver datos de sus jugadores.



Más casos son los sistemas de información que se utilizan en el fútbol para el análisis o de alineaciones.

A mi modo de ver y viendo como la tecnología cada vez encuentra más cabida en todos los ámbitos, aplicaciones de este tipo, más complejas diría yo, pueden ser muy utilizadas debido a que se pueden descubrir jugadores con un click y después ojearles para que muchos equipos no se gasten mucho dinero y puedan tener equipos muy buenos o mejorar los equipos que ya tienen.

Dije que aplicaciones mas complejas que esta debido a que yo únicamente utilizo el potencial y la puntuación general que tienen los jugadores aunque si que es contemplable poder introducir muchos mas parámetros e incluso puntuaciones diversas como ritmo, velocidad, visión, colocación que también vienen bien para un entrenador debido a la amplia variedad de alineaciones y de modos de juego por lo que introduciendo también este tipo de parámetros, se pueden realizar filtrados mas estrictos para encontrar justo el jugador que queremos en una posición definida en el campo.

## **7. LECCIONES APRENDIDAS**

Una vez realizado este trabajo, puedo afirmar que soy capaz de aprender nuevas herramientas por mi cuenta, aunque debo dar las gracias a mi profesor de la asignatura por recomendarme el sistema de bases de datos neo4j.

He descubierto un mundo de bases de datos totalmente nuevo, las bases de datos por grafos y el lenguaje cypher que hace que las consultas sean muy sencillas.

Para aprender este lenguaje, tuve que realizar varios tutoriales de la pagina de neo4j además de haber visto varios videos, todos ellos de trabajadores de neo4j.

También estuve leyendo ciertos fragmentos de algún libro, recomendado por el profesor que me ayudo a reforzar los conocimientos que tenia sobre los grafos.

Me he dado cuenta de que soy capaz de aprender cosas nuevas por mi mismo ya que el profesor nos dio la idea y viendo los tutoriales pude aprender en varias semanas como funciona las bases de datos neo4j además de entender como funciona el lenguaje cypher.

Realmente el lenguaje cypher y el SQL tienen muchas cosas en común ya que ambos realizan consultas y peticiones a una base de datos.

Si que es cierto que neo4j y SQL son muy diferentes dado que uno crea bases de datos formadas por tablas que tienen relaciones entre ellas mientras que las bases de datos neo4j, es decir, las bases de datos de grafos unen nodos de cierto tipo de objeto, pero no todos

los nodos de ese objeto tienen porque estar unidos a un nodo en especial.

También he aprendido a organizarme por mi cuenta a la hora de desarrollar una aplicación. He creado mi propia dinámica de desarrollo de software, al menos a la hora de hacer aplicaciones web que estén compuesta por frontend y backend. Mi nueva dinámica sería la siguientes:

- 1) crear la base de datos sin tener en cuenta que tipo sea, es decir, sin tener en cuenta que sea una SQL, no-SQL o de grafos.
- 2)crear el backend con sentencias sencillas para poder conectarlo con la base de datos mediante los drivers necesarios (en este caso, el driver necesario fue el de neo4j).
- 3)crear el frontend para poder conectarlo al backend, en este caso con la herramienta axios y así poder realizar peticiones al backend. Una vez creado todas las conexiones entre los distintos elementos, cuando creaba las peticiones a la base de datos, implementaba el código necesario y posteriormente utilizaba la herramienta postman para comprobar que el código realizado funcionaba correctamente y una vez comprobado esto, podemos hacer las peticiones con el frontend directamente.

Al haber creado esta dinámica, posiblemente ya existente, me ayudaba a llevar cierta rapidez a la hora de realizar el código y comprobar que funcionaba correctamente.

## **8. INSTALACION POR PASOS**

Para poder utilizar mi programa, se necesita realizar una serie de pasos para poder instalarlo y ejecutarlo en el navegador web.

Vamos a explicar el proceso de instalación por orden desde la parte mas baja, que es NEO4J hasta el frontend y posterior ejecución.

- NEO4J: la versión que utilizamos de neo4j es la versión 4.1.3 por lo que, quien utilice el programa explicado, deberá utilizar dicha versión. El nombre de la base de datos ha de ser “FutbolFichajes” y la contraseña ha de ser la misma (a menos que se quiera cambiar, por lo que habrá que cambiarla también en el fichero backend). Cuando ya hemos creado una base de datos, el archivo, que esta en el fichero etc, llamado

“baseDatosFifaCompleto.csv” deberá ser introducido en el fichero import de neo4j y una vez encendido la base de datos, lo único necesario para introducir la base de datos será introducir el siguiente comando:

```
load csv with headers from 'file:///baseDatosFifaCompleto.csv' as row
FIELDTERMINATOR ';'
with row create(j:Jugador)
set j={
    nombre:row.nombre,
    potencial:row.potencial,
    edad:row.edad,
    equipo:row.equipo,
    puntuacion:row.puntuacion,
    salario:row.salarioSemana,
    precio:row.precio,
    posicion:row.posicion,
    foto:row.foto,
    nombreCompleto:row.nombreCompleto,
    nacionalidad:row.nacionalidad,
    id:row.id
}
merge (e:Equipo{nombre:row.equipo})
merge(j)-[:JUEGA_EN]->(e)
```

Después, en el buscador introducimos “localhost:7474” para introducir, como admin “neo4j” y como contraseña “FutbolFichajes” y de esta manera, ya se permitirán las conexiones externas con la base de datos

Una vez introducido este comando, la base de datos ya está lista para ser utilizada.

Tanto para backend como para Frontend, se deberá utilizar una terminal diferente y realizar los pasos que a continuación explicamos.

- Backend: utilizando el comando “cd”, de deberá acceder al directorio de la carpeta backend. Una vez allí, se ejecutará el comando “npm install” para poder instalar en su equipo todas las dependencias necesarias para ejecutar el backend. Una vez termina de instalar todas las dependencias, se ejecutará el

comando “nodemon app.js” y la parte del backend ya estará funcionando. Una vez realizado esto, debería quedarnos así la terminal:

```
● ● ● backend — node < node /usr/local/bin/nodemon app.js — 80x29
Last login: Fri Dec  4 18:44:25 on ttys000
[aitordelrioferreras@MacBook-Air-de-Aitor ~ % cd Desktop/Bussiness-Intelligence/b]
ackend
[aitordelrioferreras@MacBook-Air-de-Aitor backend % npm install
npm WARN backend@1.0.0 No repository field.

audited 218 packages in 1.599s

11 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

          npm update check failed
          Try running with sudo or get access
          to the local update config store via
sudo chown -R $USER:$(id -gn $USER) /Users/aitordelrioferreras/.config

[aitordelrioferreras@MacBook-Air-de-Aitor backend % nodemon app.js
[nodemon] 2.0.6
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
¡Aplicación escuchando en el puerto 3000!
```

- Frontend: utilizando el comando “cd”, accederemos al directorio de la carpeta frontend. Una vez situados, ejecutaremos el comando “npm install” para instalar las dependencias necesarias para ejecutar e instalar el programa. Una vez que están todas las dependencias instaladas, ejecutaremos el comando “npm run serve” para ejecutar el programa. Cuando ejecutamos este comando, la terminal aparece de la siguiente forma:

```

[aitordelrioferreiras@MacBook-Air-de-Aitor ~ % cd Desktop/Bussiness-Intelligence/f]
rontend
[aitordelrioferreiras@MacBook-Air-de-Aitor frontend % npm install
audited 1366 packages in 8.531s

64 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

          npm update check failed
          Try running with sudo or get access
          to the local update config store via
sudo chown -R $USER:$(id -gn $USER) /Users/aitordelrioferreiras/.config

[aitordelrioferreiras@MacBook-Air-de-Aitor frontend % npm run serve
]

> frontend@0.1.0 serve /Users/aitordelrioferreiras/Desktop/Bussiness-Intelligence
/frontend
> vue-cli-service serve

[INFO] Starting development server...
98% after emitting CopyPlugin

[DONE] Compiled successfully in 21110ms
8:29:18

App running at:
- Local:  http://localhost:8080/
- Network: http://192.168.1.38:8080/

Note that the development build is not optimized.
To create a production build, run npm run build.

```

A menos que no se tenga conexión a internet, aparecerán, tanto la dirección localhost como una ip, que será la del propio PC. Para utilizar el programa, habrá que copiar la dirección completa (cualquiera de las dos) en el buscador de internet y se ejecutará la aplicación. Es obligatorio seguir los pasos de forma estricta para poder ejecutar la aplicación.

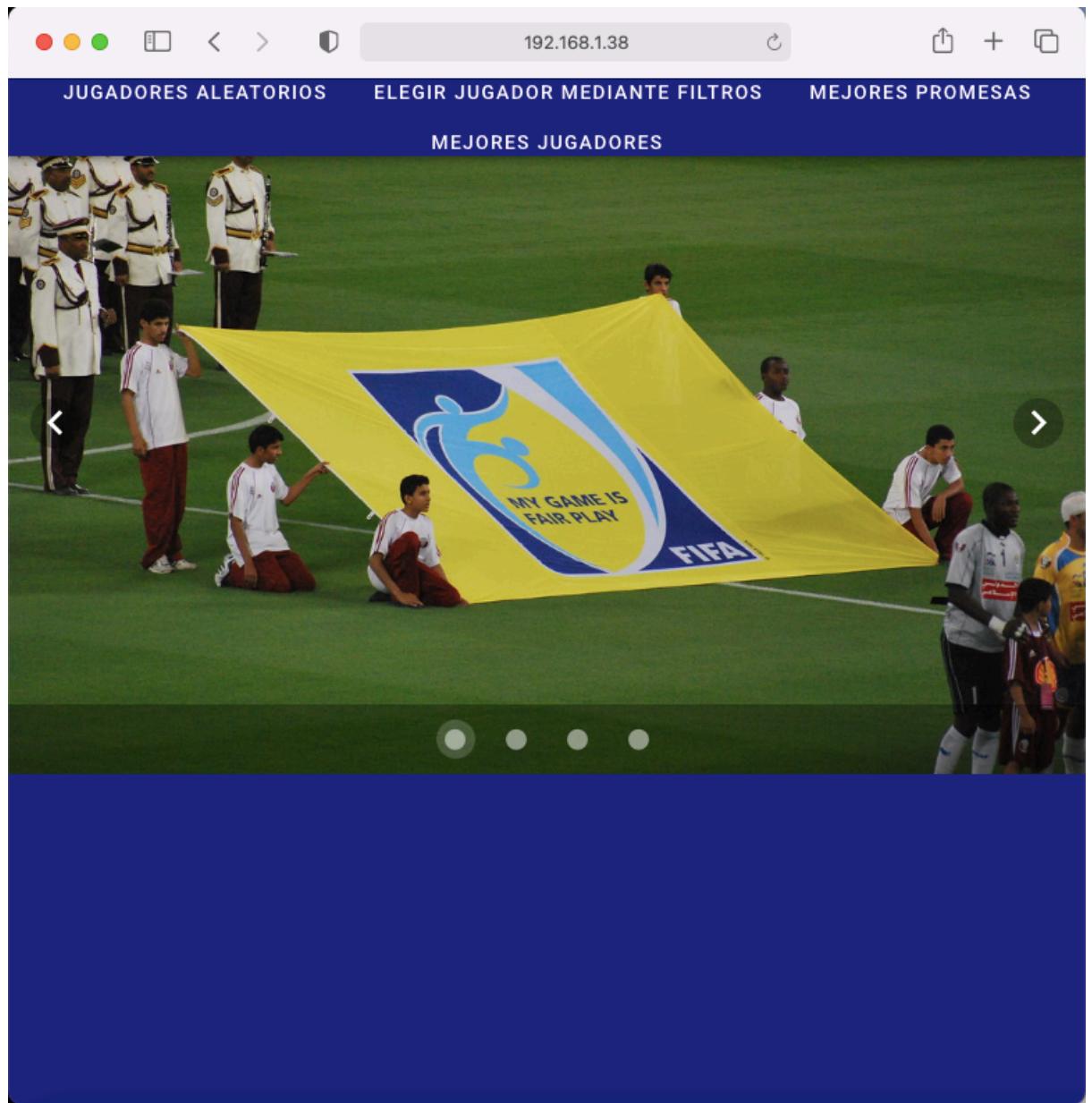


Ilustración 1:imagen principal con IP propia

# Conclusión

Gracias a este trabajo, he sido capaz de, por mi mismo, aprender un nuevo lenguaje como es cypher y un nuevo sistema de bases de datos y a programar y entender que es un sistema de recomendación, para que se utiliza, que tipos de sistemas de recomendación existen y que beneficio se obtienen de ellos en el mundo real.

# Bibliografía y referencias

Graph database: bases de datos para una interconexión eficiente:

<https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/graph-database/#:~:text=A%20diferencia%20de%20las%20de,con%20diferentes%20valores%20de%20atributo>

Plataforma para hacer competiciones de inteligencia artificial:

<https://inlab.fib.upc.edu/es/blog/plataforma-para-hacer-competiciones-de-inteligencia-artificial>

Qué es NodeJS y para qué sirve: <https://openwebinars.net/blog/que-es-nodejs/>

MVC (Model, View, Controller) Explicado.: <https://codigofacilito.com/articulos/mvc-model-view-controller-explicado>

Qué es MVC: <https://desarrolloweb.com/articulos/que-es-mvc.html>

**Link Github:** <https://github.com/adelrf01/Bussiness-Intelligence.git>

**Link OSF:** <https://osf.io/sjw4f/>