

## **Parcial Arquitectura**

Juan Sebastian Dueñas Robayo

833539

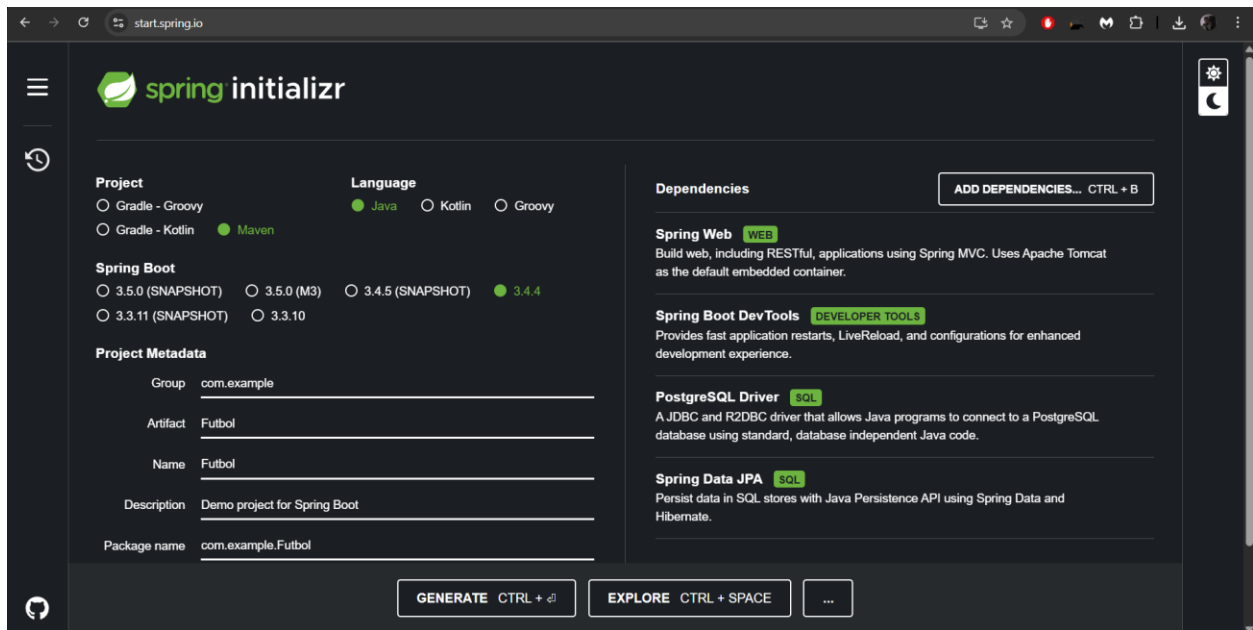
Corporación Universitaria Minuto de Dios

Ing. William Alexander Matallana Porras

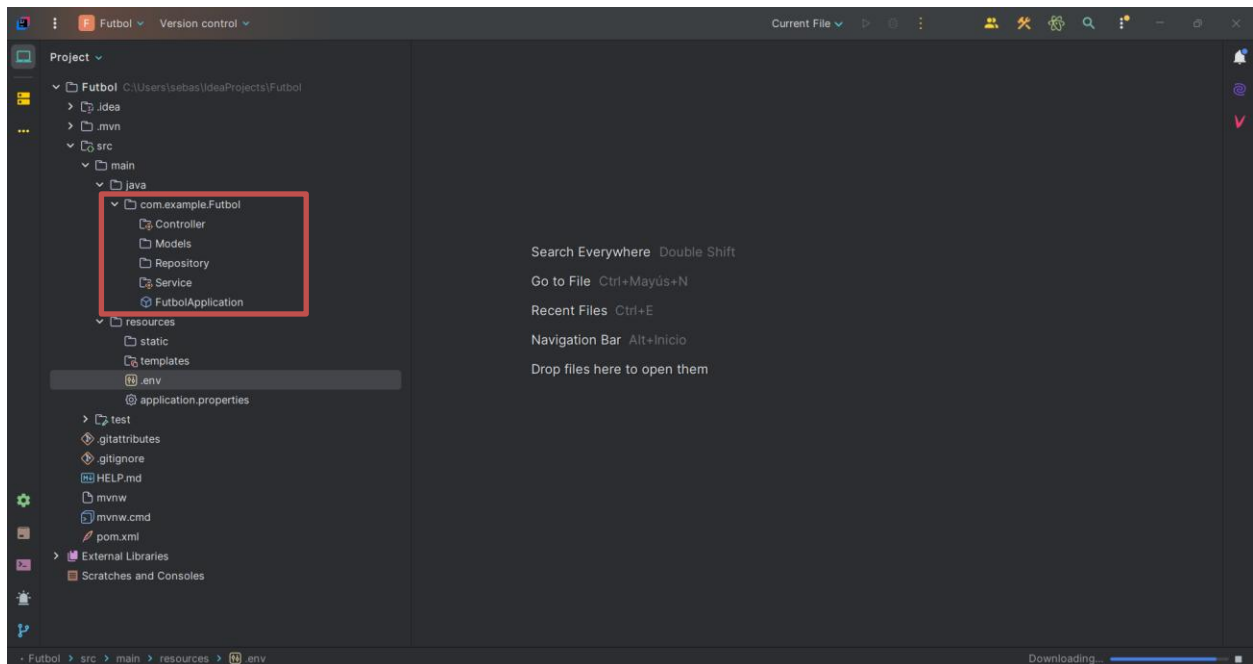
Arquitectura de Software

2025

## 1. Empezamos creando el proyecto con la página de Spring Initializr

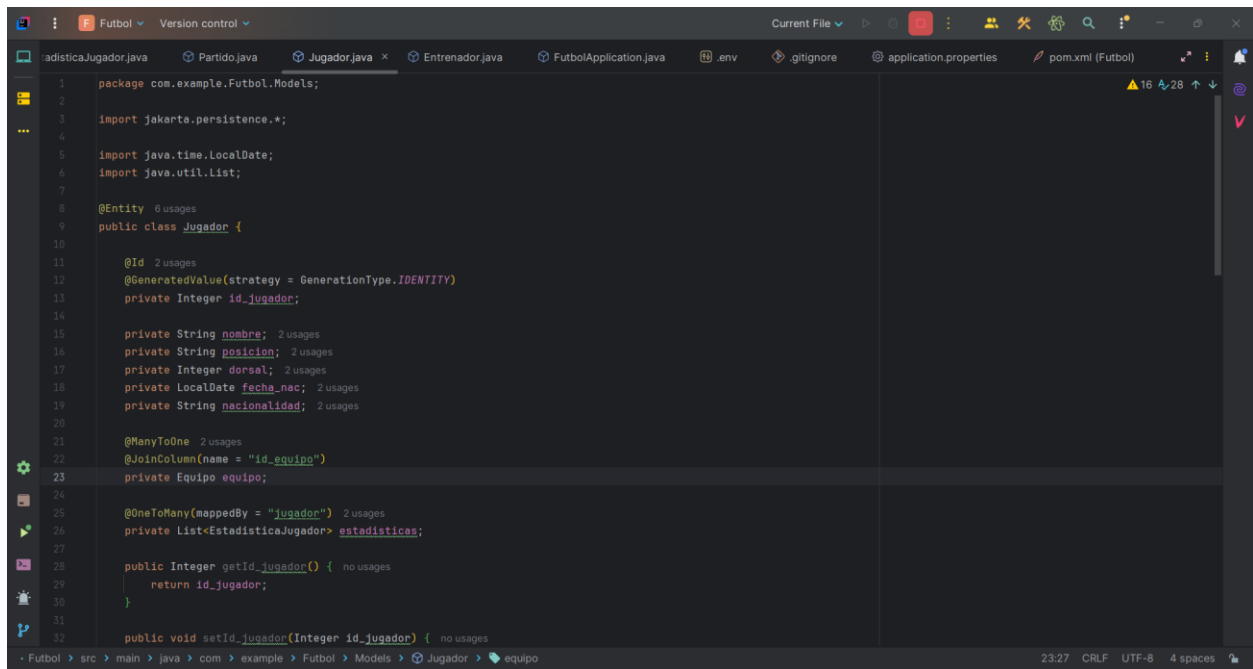


## 2. Descomprimos la carpeta y la abrimos con nuestro IntelliJ, dentro del paquete com.example.Futbol, crearemos las carpetas de models, service, controller y repository



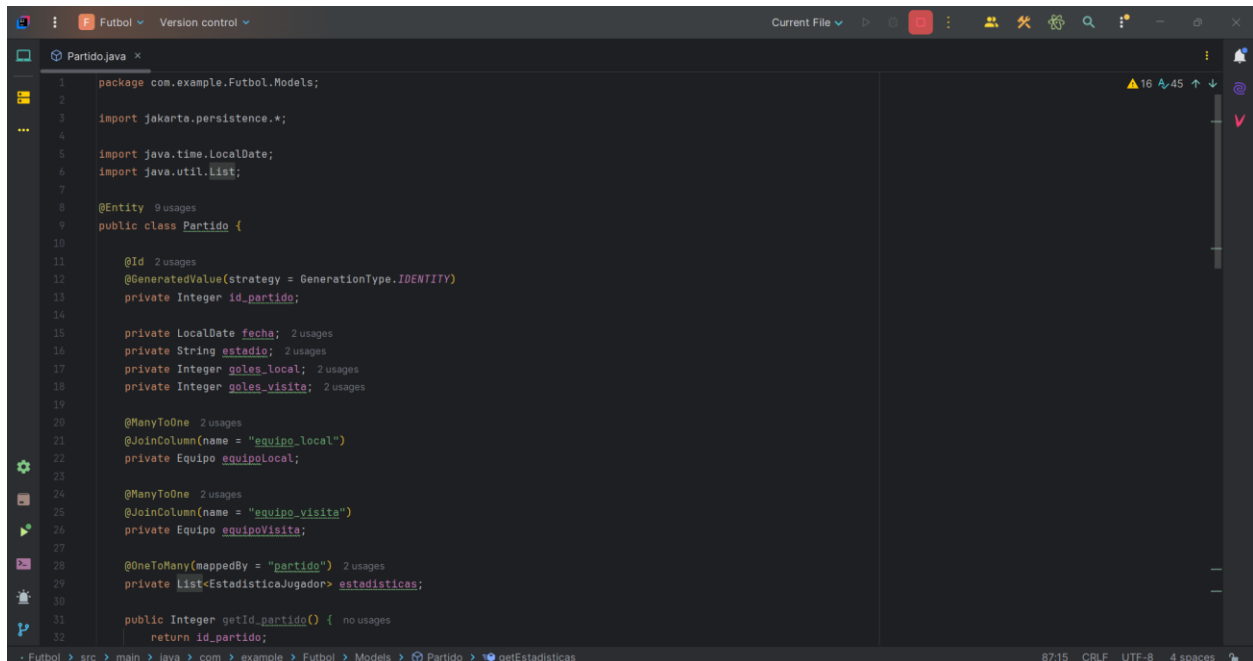
3. Dentro de la carpeta model, creamos cada uno de los modelos según la información del pdf

## Jugador



```
1 package com.example.Futbol.Models;
2
3 import jakarta.persistence.*;
4
5 import java.time.LocalDate;
6 import java.util.List;
7
8 @Entity
9 public class Jugador {
10
11     @Id
12     @GeneratedValue(strategy = GenerationType.IDENTITY)
13     private Integer id_jugador;
14
15     private String nombre;
16     private String posicion;
17     private Integer dorsal;
18     private LocalDate fecha_nac;
19     private String nacionalidad;
20
21     @ManyToOne
22     @JoinColumn(name = "id_equipo")
23     private Equipo equipo;
24
25     @OneToMany(mappedBy = "jugador")
26     private List<EstadisticaJugador> estadisticas;
27
28     public Integer getId_jugador() {
29         return id_jugador;
30     }
31
32     public void setId_jugador(Integer id_jugador) {
33         this.id_jugador = id_jugador;
34     }
35 }
```

## Partido



```
1 package com.example.Futbol.Models;
2
3 import jakarta.persistence.*;
4
5 import java.time.LocalDate;
6 import java.util.List;
7
8 @Entity
9 public class Partido {
10
11     @Id
12     @GeneratedValue(strategy = GenerationType.IDENTITY)
13     private Integer id_partido;
14
15     private LocalDate fecha;
16     private String estadio;
17     private Integer goles_local;
18     private Integer goles_visitado;
19
20     @ManyToOne
21     @JoinColumn(name = "equipo_local")
22     private Equipo equipoLocal;
23
24     @ManyToOne
25     @JoinColumn(name = "equipo_visitado")
26     private Equipo equipoVisitado;
27
28     @OneToMany(mappedBy = "partido")
29     private List<EstadisticaJugador> estadisticas;
30
31     public Integer getId_partido() {
32         return id_partido;
33     }
34
35     public void setId_partido(Integer id_partido) {
36         this.id_partido = id_partido;
37     }
38 }
```

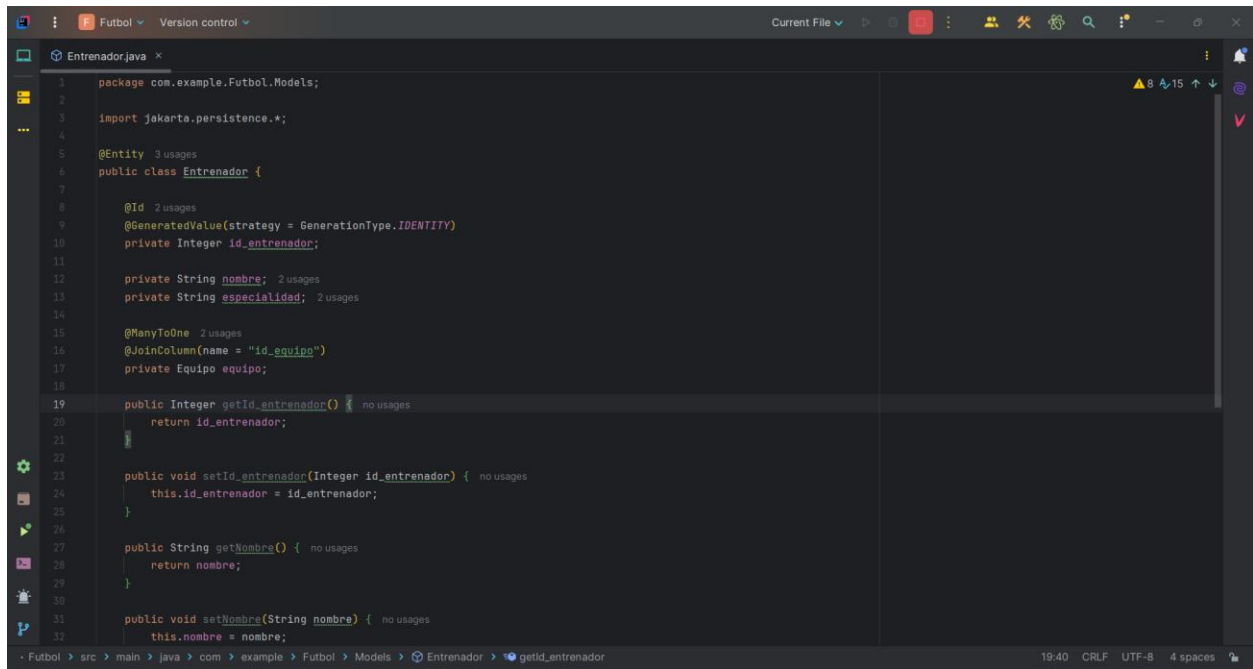
## Estadística de Jugador

```
1 package com.example.Futbol.Models;
2
3 import jakarta.persistence.*;
4
5 @Entity
6 public class EstadisticaJugador {
7
8     @Id
9     @GeneratedValue(strategy = GenerationType.IDENTITY)
10    private Integer id_estadistica;
11
12    private Integer minutos_jugados;
13    private Integer goles;
14    private Integer asistencias;
15    private Integer tarjetas_amarillas;
16    private Integer tarjetas_rojas;
17
18    @ManyToOne
19    @JoinColumn(name = "id_jugador")
20    private Jugador jugador;
21
22    @ManyToOne
23    @JoinColumn(name = "id_partido")
24    private Partido partido;
25
26    public Integer getId_estadistica() {
27        return id_estadistica;
28    }
29
30    public void setId_estadistica(Integer id_estadistica) {
31        this.id_estadistica = id_estadistica;
32    }
33 }
```

## Equipo

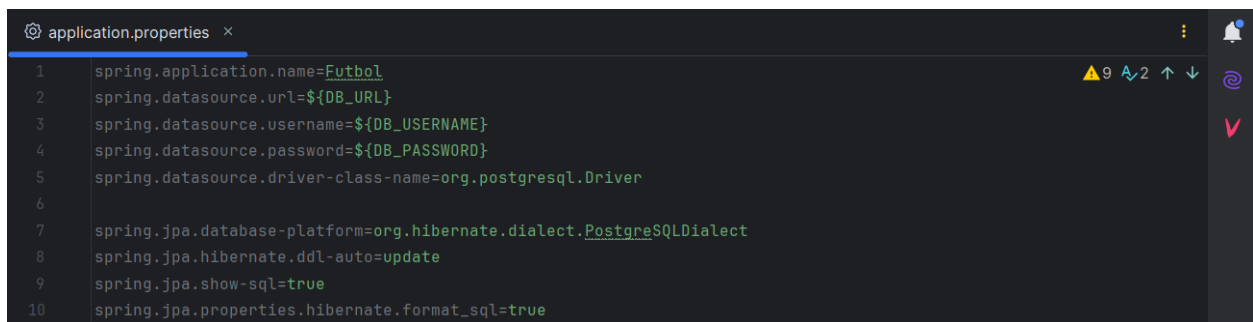
```
1 package com.example.Futbol.Models;
2
3 import jakarta.persistence.*;
4
5 import java.time.LocalDate;
6 import java.util.List;
7
8 @Entity
9 public class Equipo {
10
11     @Id
12     @GeneratedValue(strategy = GenerationType.IDENTITY)
13    private Integer id_equipo;
14
15    private String nombre;
16    private String ciudad;
17    private LocalDate fundacion;
18
19    @OneToMany(mappedBy = "equipo", cascade = CascadeType.ALL)
20    private List<Jugador> jugadores;
21
22    @OneToMany(mappedBy = "equipo", cascade = CascadeType.ALL)
23    private List<Entrenador> entrenadores;
24
25    @OneToMany(mappedBy = "equipoLocal")
26    private List<Partido> partidosLocal;
27
28    @OneToMany(mappedBy = "equipoVisita")
29    private List<Partido> partidosVisita;
30
31    public Integer getId_equipo() {
32        return id_equipo;
33    }
34 }
```

## Entrenador



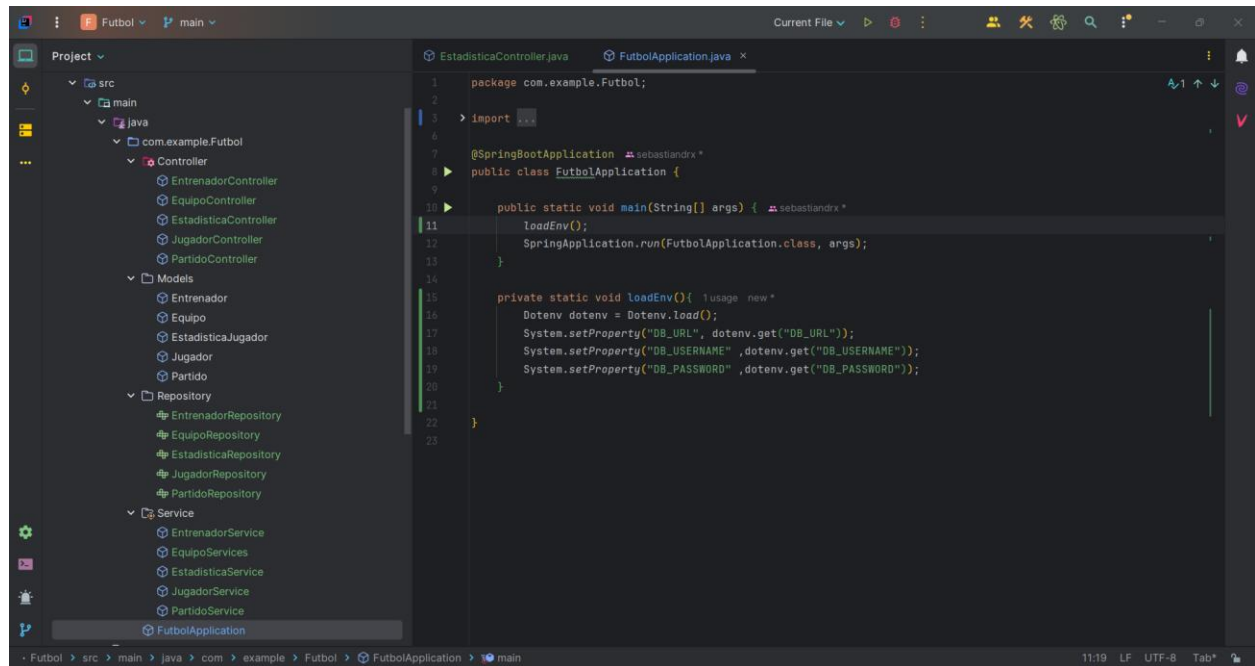
```
1 package com.example.Futbol.Models;
2
3 import jakarta.persistence.*;
4
5 @Entity
6 public class Entrenador {
7
8     @Id
9     @GeneratedValue(strategy = GenerationType.IDENTITY)
10    private Integer id_entrenador;
11
12    private String nombre;
13    private String especialidad;
14
15    @ManyToOne
16    @JoinColumn(name = "id_equipo")
17    private Equipo equipo;
18
19    public Integer getId_entrenador() {
20        return id_entrenador;
21    }
22
23    public void setId_entrenador(Integer id_entrenador) {
24        this.id_entrenador = id_entrenador;
25    }
26
27    public String getNombre() {
28        return nombre;
29    }
30
31    public void setNombre(String nombre) {
32        this.nombre = nombre;
33    }
34}
```

4. Dentro del apartado de application.properties colocaremos la información de la conexión de la base de datos a supabase































```
1 spring.application.name=Futbol
2 spring.datasource.url=${DB_URL}
3 spring.datasource.username=${DB_USERNAME}
4 spring.datasource.password=${DB_PASSWORD}
5 spring.datasource.driver-class-name=org.postgresql.Driver
6
7 spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
8 spring.jpa.hibernate.ddl-auto=update
9 spring.jpa.show-sql=true
10 spring.jpa.properties.hibernate.format_sql=true
```


5. Dentro del FutbolApplication cargaremos una función llamada loadEnv para que recorra nuestras variables y permita arrancar nuestra conexión



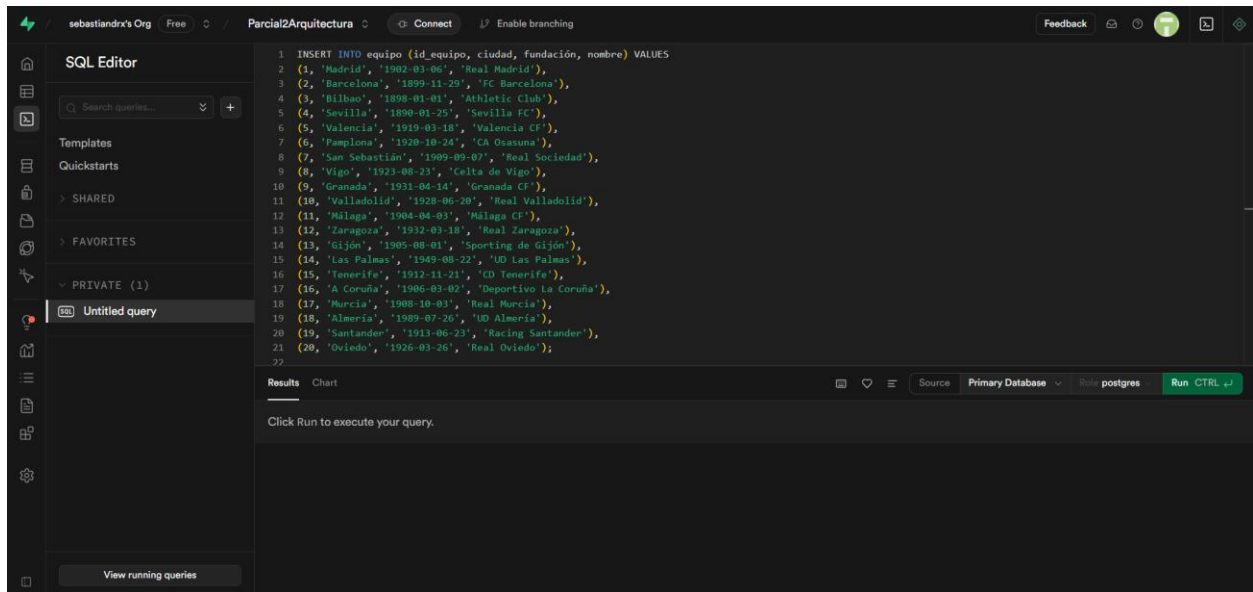
6. Luego de crear todos nuestros controladores, servicios y repositorios nos dijimos a insertar los registros por cada tabla

Project ▾

- ▾  src
  - ▾  main
    - ▾  java
      - ▾  com.example.Futbol
        - ▾  Controller
          -  EntrenadorController
          -  EquipoController
          -  EstadisticaController
          -  JugadorController
          -  PartidoController
        - ▾  Models
          -  Entrenador
          -  Equipo
          -  EstadisticaJugador
          -  Jugador
          -  Partido
        - ▾  Repository
          -  EntrenadorRepository
          -  EquipoRepository
          -  EstadisticaRepository
          -  JugadorRepository
          -  PartidoRepository
        - ▾  Service
          -  EntrenadorService
          -  EquipoServices
          -  EstadisticaService
          -  JugadorService
          -  PartidoService

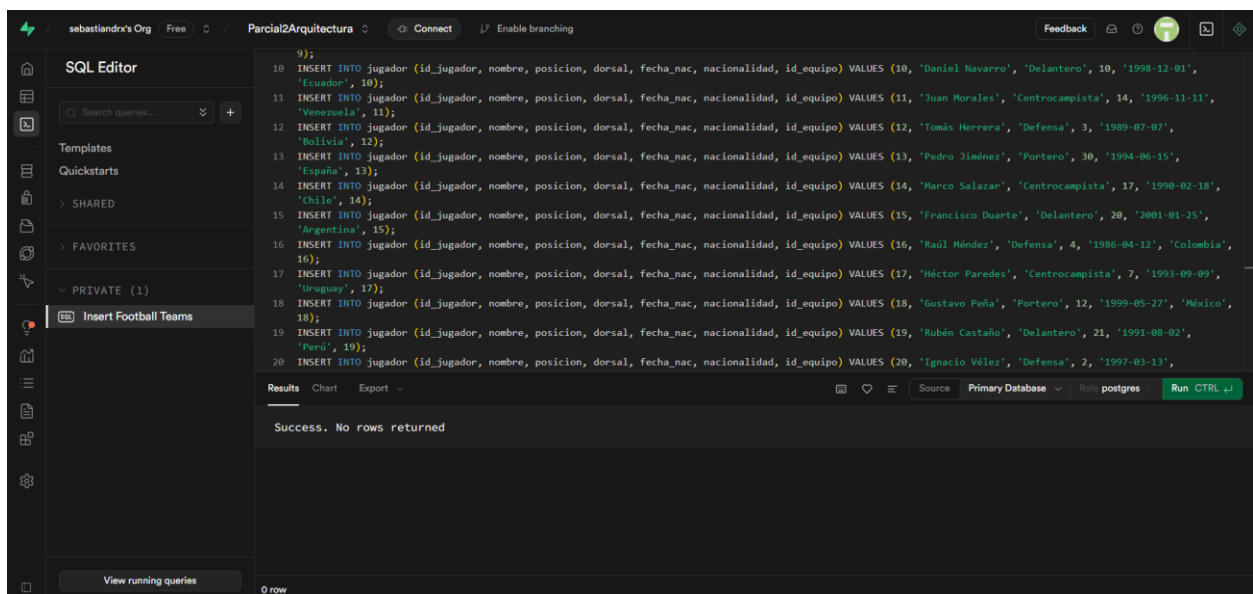
 FutbolApplication

## 7. En SQL EDITOR colocaremos los datos que vamos a insertar en cada tabla



The screenshot shows the SQL Editor interface with a sidebar on the left containing navigation icons and a list of queries. The main editor area displays an SQL INSERT statement for a table named 'equipo'. The query lists 20 teams with their IDs, names, cities, founding years, and names. The results pane at the bottom is empty, showing a message to click 'Run' to execute the query.

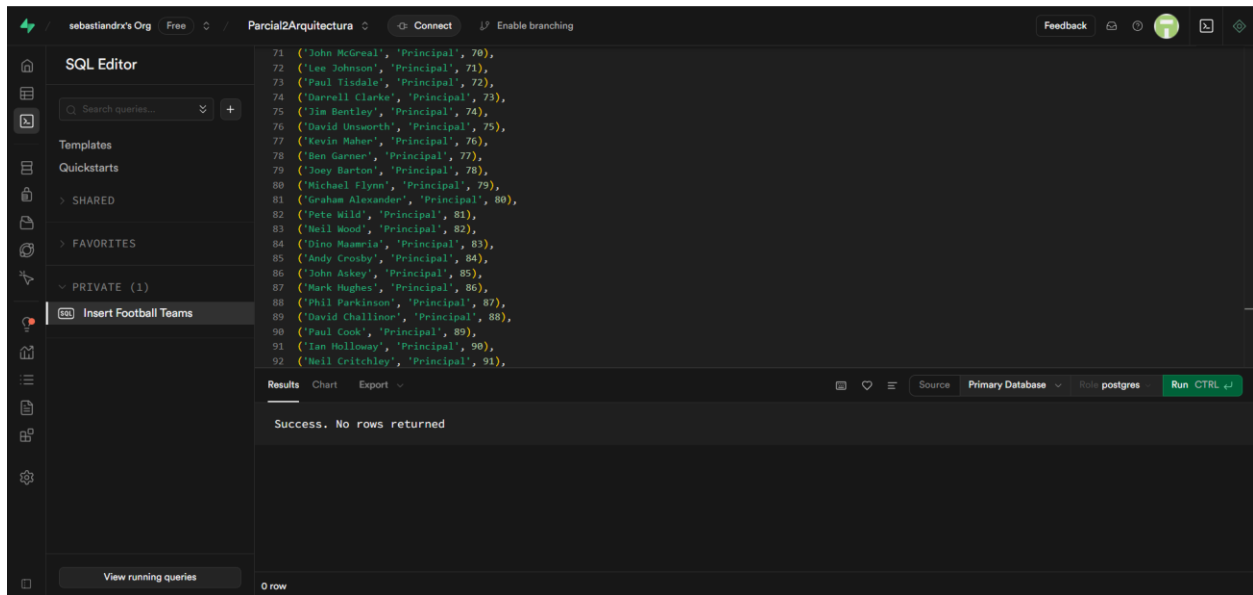
```
1 INSERT INTO equipo (id_equipo, ciudad, fundación, nombre) VALUES
2 (1, 'Madrid', '1902-03-06', 'Real Madrid'),
3 (2, 'Barcelona', '1899-11-29', 'FC Barcelona'),
4 (3, 'Bilbao', '1898-01-01', 'Athletic Club'),
5 (4, 'Sevilla', '1890-01-25', 'Sevilla FC'),
6 (5, 'Valencia', '1919-03-18', 'Valencia CF'),
7 (6, 'Pamplona', '1920-10-24', 'CA Osasuna'),
8 (7, 'San Sebastián', '1909-09-07', 'Real Sociedad'),
9 (8, 'Vigo', '1923-08-23', 'Celta de Vigo'),
10 (9, 'Granada', '1931-04-14', 'Granada CF'),
11 (10, 'Valladolid', '1928-06-20', 'Real Valladolid'),
12 (11, 'Málaga', '1984-04-03', 'Málaga CF'),
13 (12, 'Zaragoza', '1932-01-18', 'Real Zaragoza'),
14 (13, 'Gijón', '1905-08-01', 'Sporting de Gijón'),
15 (14, 'Las Palmas', '1949-08-22', 'UD Las Palmas'),
16 (15, 'Tenerife', '1912-11-21', 'CD Tenerife'),
17 (16, 'A Coruña', '1906-03-02', 'Deportivo La Coruña'),
18 (17, 'Murcia', '1908-10-03', 'Real Murcia'),
19 (18, 'Almería', '1989-07-26', 'UD Almería'),
20 (19, 'Santander', '1913-06-23', 'Racing Santander'),
21 (20, 'Oviedo', '1926-03-26', 'Real Oviedo');
22
```



The screenshot shows the SQL Editor interface with a sidebar on the left. The main editor area displays an SQL INSERT statement for a table named 'jugador'. The query lists 20 players with their IDs, names, positions, jersey numbers, birth dates, nationalities, and team IDs. The results pane at the bottom shows 'Success. No rows returned'.

```
9);
10 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (10, 'Daniel Navarro', 'Delantero', 10, '1998-12-01', 'Ecuador', 10);
11 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (11, 'Juan Morales', 'Centrocampista', 14, '1996-11-11', 'Venezuela', 11);
12 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (12, 'Tomás Herrera', 'Defensa', 3, '1989-07-07', 'Bolivia', 12);
13 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (13, 'Pedro Jiménez', 'Portero', 30, '1994-06-15', 'España', 23);
14 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (14, 'Marco Salazar', 'Centrocampista', 17, '1990-02-18', 'Chile', 14);
15 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (15, 'Francisco Duarte', 'Delantero', 20, '2001-01-25', 'Argentina', 15);
16 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (16, 'Raul Méndez', 'Defensa', 4, '1986-04-12', 'Colombia', 16);
17 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (17, 'Máximo Paredes', 'Centrocampista', 7, '1993-09-09', 'Uruguay', 17);
18 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (18, 'Gustavo Peña', 'Portero', 12, '1999-05-27', 'México', 18);
19 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (19, 'Rubén Castaño', 'Delantero', 21, '1991-08-02', 'Perú', 19);
20 INSERT INTO jugador (id_jugador, nombre, posicion, dorsal, fecha_nac, nacionalidad, id_equipo) VALUES (20, 'Ignacio Vélez', 'Defensa', 2, '1997-03-13', 'Argentina', 15);
```





8. Luego de insertar los registros, lanzamos la conexión y vamos a probar con postman cada una de ellas

## 9. CONSULTAS NATIVAS

- Obtener los resultados de todos los partidos indicando los nombres de los equipos: <http://localhost:8080/partidos/resultados>
- Obtener los jugadores que han marcado más de X goles  
<http://localhost:8080/jugadores/goleadores?minGoles=0>
- Obtener todos los jugadores de un equipo específico:  
<http://localhost:8080/jugadores/equipo/1>
- Obtener el número total de goles marcados por un equipo en todos sus partidos: <http://localhost:8080/estadisticas/goles/equipo/1>