Sistema de Torneos e Competencias Deportivas

**Nome Alumno/a:**

***Jorge Sobrino Mojón***

**Curso: *2º DAM* Materia:** ***Proxecto Final Módulo Acceso a Datos***

**Docente:** ***Javier Feijóo López***

***🗐 Proxecto: https://github.com/Jsobrino98/ProyectoFinalAD***

Contido

[1. Introdución 2](#_Toc189548783)

[2. Deseño da Base de Datos – Modelo EER 2](#_Toc189548784)

[3. Tecnoloxías Empregadas e Configuración da Contorna 2](#_Toc189548785)

[4. Deseño do Backend – API REST con Spring Boot 2](#_Toc189548786)

[5. Implementación de Autenticación e Seguridade 3](#_Toc189548787)

[6. Desenvolvemento do Backend con Thymeleaf 3](#_Toc189548788)

[7. Documentación da API e Probas en Postman 3](#_Toc189548789)

[8. Conclusións e Valoración Persoal 3](#_Toc189548790)

[9. Anexos (Opcional) 3](#_Toc189548791)

# 1. Introdución

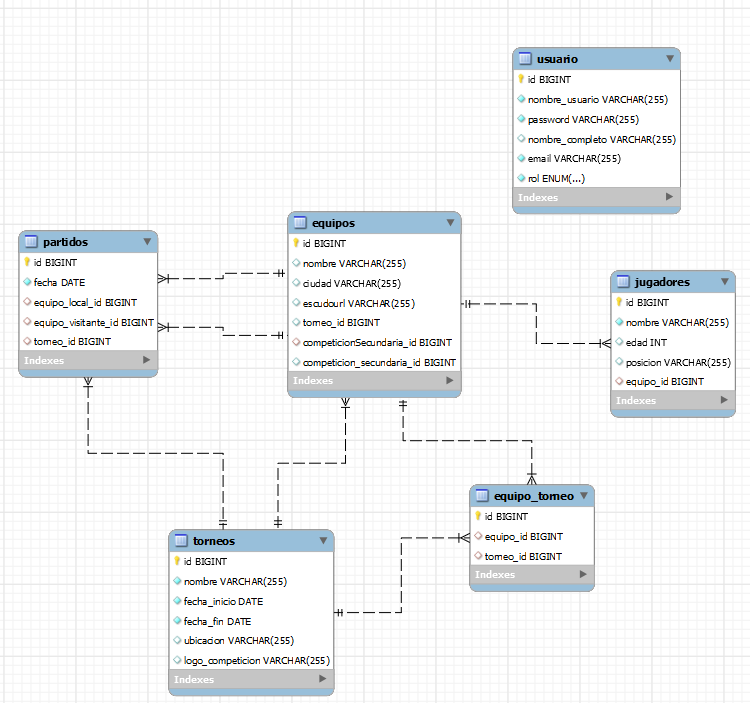
Breve presentación do proxecto, respondendo a preguntas clave:

* Cal é a finalidade do sistema?
* Que problema resolve?
* Quenes son os usuarios principais?
* Resumen das tecnoloxías empregadas.

# 2. Deseño da Base de Datos – Modelo EER

Explicación do deseño da base de datos, incluíndo:

* Modelo Entidade-Relación Estendido (EER) cunha imaxe.



**Equipos**

* **id**: Identificador único del equipo.
* **nombre**: Nombre del equipo.
* **ciudad**: Ciudad del equipo.
* **escudourl**: Imagen del escudo

Relaciones:

* Relacionado con jugadores a través de equipo\_id.
* Relacionado con partidos como equipo\_local\_id y equipo\_visitante\_id.

**Jugadores**

* **id**: Identificador único del jugador.
* **nombre**: Nombre del jugador.
* **edad**: Edad del jugador.
* **equipo\_id**: Identificador del equipo al que pertenece (clave foránea a equipos).

**Torneos**

* **id**: Identificador único del torneo.
* **nombre**: Nombre del torneo.
* **fecha\_inicio**: Fecha de inicio del torneo.
* **fecha\_fin**: Fecha de finalización del torneo.
* **ubicacion**: Pais donde se juega.
* **logo\_competicion:** Imagen de la competición**.**

Relaciones:

* Relacionado con partidos a través de torneo\_id.

**Partidos**

* **id**: Identificador único del partido.
* **torneo\_id**: Identificador del torneo (clave foránea a torneos).
* **equipo\_local\_id**: Identificador del equipo local (clave foránea a equipos).
* **equipo\_visitante\_id**: Identificador del equipo visitante (clave foránea a equipos).
* **fecha**: Fecha del partido.

Relaciones:

* Relacionado con resultados a través de partido\_id.

**Usuarios**:

* **id**: Identificador único del usuario.
* **nombre\_usuario**: Nombre único del usuario para autenticación.
* **password**: Contraseña encriptada del usuario.
* **nombre\_completo**: Nombre completo del usuario.
* **email**: Correo electrónico único del usuario.
* **rol**: Rol del usuario en el sistema (ADMIN o NORMAL).

Relaciones:

* Relacionado con publicaciones a través de usuario\_id.
* Relacionado con pedidos (en caso de un sistema de compras) a través de usuario\_id.

# 3. Tecnoloxías Empregadas e Configuración da Contorna

* Linguaxe de programación: **Java 17**
* Framework: **Spring Boot .**
* Base de datos: **MySQL.**
* ORM: **Spring Data JPA.**
* Autenticación: **Spring Security + JWT.**
* Frontend para administradores: **Thymeleaf + Bootstrap.**
* Ferramentas de documentación: **Swagger, Postman.**
* Xestión de dependencias: **Maven**.

Incluír pasos para configurar o contorno:

* IDE recomendado (**IntelliJ, VisualStudioCode**).

# 4. Deseño do Backend – API REST con Spring Boot

Explicación da arquitectura do backend:

**AD.SistemaTorneosCompeticions**

**│── config/** # Configuración general (Swagger, CORS, JPA, etc.)

**│── controllers**/ # Controladores REST (endpoints de la API)

**│── models/** # Modelos de datos (entidades JPA)

**│── models/DTO/** # Objetos de transferencia de datos (DTOs)

**│── repositories/** # Repositorios para acceso a la base de datos (JPA)

**│── security/** # Seguridad (JWT, autenticación, filtros)

**│── services/** # Servicios con la lógica de negocio

**│── Application.java** # Clase principal de Spring Boot

**Controllers**: Los controladores manejan las peticiones HTTP y se comunican con los servicios.

**Models**: Las entidades JPA representan las tablas de la base de datos.

**Repositories**: Los repositorios son interfaces que acceden a la base de datos usando **Spring Data JPA.**

**Services**: Los servicios contienen la lógica de negocio.

**Security**: Maneja autenticación y autorización con **JWT**.

**Config:** Aquí se encuentra la configuración del **Swagger**.

# 5. Implementación de Autenticación e Seguridade

Explicación da seguridade no sistema:

### **Registro e Inicio de Sesión con JWT**

* **Registro:** Se almacena el usuario en la base de datos con la contraseña encriptada usando BCrypt.
* **Inicio de sesión:** Se valida el usuario y contraseña, y si son correctos, se genera un token JWT.
* **Uso del token:** El cliente debe enviar el JWT en el encabezado Authorization en cada petición protegida.

### **Protección de Endpoints con Roles (USER, ADMIN)**

* **Definición de roles:** Los usuarios pueden tener roles como ADMIN o NORMAL.
* **Restricción de acceso:** En SecurityConfig, se define qué endpoints requieren qué roles (hasRole("ADMIN"), hasAnyRole("NORMAL", "ADMIN")).
* **Filtro JWT:** Se interceptan las peticiones y se verifica el token antes de permitir el acceso.

### **Encriptación de Contraseñas con BCrypt**

* **Uso de BCryptPasswordEncoder:** Se encripta la contraseña antes de guardarla en la base de datos.
* **Comparación segura:** Cuando el usuario inicia sesión, la contraseña ingresada se compara con la encriptada en la base de datos usando matches().

# 6. Desenvolvemento do Backend con Thymeleaf

Explicación do panel de administración con Thymeleaf:

* Vistas e plantillas HTML.
* Formularios de inicio de sesión e xestión de datos.
* Controladores MVC e servizos para a web.

# 7. Documentación da API e Probas en Postman

Incluir exemplos de endpoints documentados con Swagger e Postman:

* Capturas de Swagger UI.
* Explicación da colección de Postman e exemplos de respostas.

# 8. Conclusións e Valoración Persoal

Reflexión sobre o desenvolvemento do proxecto:

### **Dificultades y Soluciones**

Lo más complicado fue implementar la seguridad con JWT y Spring Security, ya que hubo problemas con la configuración y dependencias cíclicas. Se resolvió organizando mejor los componentes de seguridad y asegurando que cada uno tuviera una función clara. También se usó BCrypt para encriptar contraseñas y se definieron bien los permisos de acceso según el rol del usuario.

### **Mejoras para Futuras Versiones**

* Ampliar la base de datos con más información sobre torneos, equipos y jugadores.
* Mejorar la gestión de usuarios, permitiendo recuperar contraseñas o cambiar roles.
* Optimizar la seguridad y la autenticación para mayor eficiencia.

# 9. Anexos (Opcional)

Se o proxecto inclúe documentación adicional, achegar:

* Estrutura JSON de respostas da API.
* Código SQL para crear a base de datos.
* Instrucións adicionais de despregue nun servidor.de despliegue en un servidor.