Documentación del Proyecto BACK : Gestión de Canchas de Fútbol

Proyecto: Tucanchaya

Estructura de tu proyecto:

mi-proyecto- /

├── controllers/

│ ├── usuarioController.js

│ ├── equipoController.js

│ ├── grupoController.js

│ ├── partidoController.js

│ ├── canchaController.js

│ └── asentamientoController.js

├── models/

│ ├── usuario.js

│ ├── equipo.js

│ ├── grupo.js

│ ├── partido.js

│ ├── cancha.js

│ └── asentamiento.js

├── routes/

│ ├── usuarioRoutes.js

│ ├── equipoRoutes.js

│ ├── grupoRoutes.js

│ ├── partidoRoutes.js

│ ├── canchaRoutes.js

│ └── asentamientoRoutes.js

├── index.js

├── package.json

└── README.md

**1. Alcance del Proyecto**

El proyecto tiene como objetivo desarrollar un sistema para gestionar canchas de fútbol, equipos, grupos, partidos y asentamientos asociados. El sistema permitirá:

* **Gestión de Canchas**: Crear, consultar, actualizar y eliminar canchas de fútbol. Cada cancha tendrá información sobre su dirección, dimensiones, y ubicación.
* **Gestión de Equipos**: Crear, consultar, actualizar y eliminar equipos. Los equipos pueden estar asociados a miembros.
* **Gestión de Grupos**: Crear, consultar, actualizar y eliminar grupos. Los grupos pueden incluir múltiples equipos.
* **Gestión de Partidos**: Crear, consultar, actualizar y eliminar partidos. Los partidos pueden estar asociados a grupos y tienen detalles sobre fecha y ubicación.
* **Gestión de Asentamientos**: Crear, consultar, actualizar y eliminar asentamientos en una cancha. Los asentamientos incluyen información sobre la fecha y la cancha asociada.

**2. Estructura del Proyecto**

La estructura del proyecto está organizada en varias carpetas principales:

* **controllers/**: Contiene los controladores para manejar las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para cada entidad.
* **models/**: Contiene los modelos de Mongoose que definen la estructura de los datos en la base de datos.
* **routes/**: Contiene las rutas para definir los endpoints de la API y su correspondencia con los controladores.

**3. Controladores y Servicios**

**3.1 Controlador de Usuario (usuarioController.js)**

* **Servicios**:
  + **Crear Usuario**: POST /usuarios
  + **Obtener Todos los Usuarios**: GET /usuarios
  + **Obtener Usuario por ID**: GET /usuarios/:id
  + **Actualizar Usuario**: PATCH /usuarios/:id
  + **Eliminar Usuario**: DELETE /usuarios/:id

**3.2 Controlador de Equipo (equipoController.js)**

* **Servicios**:
  + **Crear Equipo**: POST /equipos
  + **Obtener Todos los Equipos**: GET /equipos
  + **Obtener Equipo por ID**: GET /equipos/:id
  + **Actualizar Equipo**: PATCH /equipos/:id
  + **Eliminar Equipo**: DELETE /equipos/:id

**3.3 Controlador de Grupo (grupoController.js)**

* **Servicios**:
  + **Crear Grupo**: POST /grupos
  + **Obtener Todos los Grupos**: GET /grupos
  + **Obtener Grupo por ID**: GET /grupos/:id
  + **Actualizar Grupo**: PATCH /grupos/:id
  + **Eliminar Grupo**: DELETE /grupos/:id

**3.4 Controlador de Partido (partidoController.js)**

* **Servicios**:
  + **Crear Partido**: POST /partidos
  + **Obtener Todos los Partidos**: GET /partidos
  + **Obtener Partido por ID**: GET /partidos/:id
  + **Actualizar Partido**: PATCH /partidos/:id
  + **Eliminar Partido**: DELETE /partidos/:id

**3.5 Controlador de Cancha (canchaController.js)**

* **Servicios**:
  + **Crear Cancha**: POST /canchas
  + **Obtener Todas las Canchas**: GET /canchas
  + **Obtener Cancha por ID**: GET /canchas/:id
  + **Actualizar Cancha**: PATCH /canchas/:id
  + **Eliminar Cancha**: DELETE /canchas/:id

**3.6 Controlador de Asentamiento (asentamientoController.js)**

* **Servicios**:
  + **Crear Asentamiento**: POST /asentamientos
  + **Obtener Todos los Asentamientos**: GET /asentamientos
  + **Obtener Asentamiento por ID**: GET /asentamientos/:id
  + **Actualizar Asentamiento**: PATCH /asentamientos/:id
  + **Eliminar Asentamiento**: DELETE /asentamientos/:id

**4. Modelos**

* **Usuario**: Define la estructura de los datos de un usuario.
* **Equipo**: Define la estructura de los datos de un equipo.
* **Grupo**: Define la estructura de los datos de un grupo.
* **Partido**: Define la estructura de los datos de un partido.
* **Cancha**: Define la estructura de los datos de una cancha.
* **Asentamiento**: Define la estructura de los datos de un asentamiento.

**5. Planes Futuros**

* **Autenticación y Autorización**: Implementar autenticación y autorización para los usuarios.
* **Interfaz de Usuario (UI)**: Desarrollar una interfaz web y móvil para interactuar con el sistema.
* **Notificaciones**: Añadir notificaciones para eventos importantes (e.g., recordatorios de partidos).
* **Integración con Servicios Externos**: Integrar con servicios de mapas para visualización de canchas.

**6. Configuración y Despliegue**

* **Base de Datos**: MongoDB para almacenamiento de datos.
* **Servidor**: Node.js con Express para el servidor de la API.
* **Despliegue**: Considerar plataformas como Heroku, AWS, o Azure para desplegar la aplicación.