

## Documentación del Sistema de Gestión de Atletas

### I. Documentación Técnica

#### 1. Arquitectura y Patrones de Diseño

El sistema está construido siguiendo un diseño de **Arquitectura por Capas** (Layered Architecture).

Capa	Componente Principal	Objetivo	Patrones Aplicados
<b>Presentación (API/UI)</b>	ApiServer , ConsolaUI	Manejar la interacción (REST o Consola) y la entrada/salida de datos.	Arquitectura en Capas
<b>Servicio / Negocio</b>	AtletaService , GestorFinanciero	Implementar la lógica de negocio (reglas de pago, coordinación de CRUD, reportes).	Principio de Responsabilidad Única (SRP)
<b>Persistencia</b>	AtletaDAO , ConexionDB	Abstraer el acceso a la base de datos (MySQL).	<b>Data Access Object (DAO)</b>
<b>Dominio</b>	Atleta , Deporte	Representar los datos y el estado del sistema.	Modelos POJO (Plain Old Java Objects)

#### 2. Especificación de la API REST

El módulo ApiServer utiliza el *framework* **Spark** para levantar un servidor REST en el puerto por defecto 4567.

Método	Endpoint	Descripción	Cuerpo Requerido
<b>GET</b>	/api/v1/atletas	Obtener la lista completa de atletas.	Ninguno
<b>GET</b>	/api/v1/atletas/{id}	Obtener un atleta específico por su ID.	Ninguno
<b>POST</b>	/api/v1/atletas	<b>Crear</b> un nuevo atleta.	Objeto JSON con datos de <b>Atleta</b> y <b>Deporte</b> .
<b>PUT</b>	/api/v1/atletas/{id}	<b>Actualizar</b> un atleta existente.	Objeto JSON completo del <b>Atleta</b> actualizado.
<b>DELETE</b>	/api/v1/atletas/{id}	<b>Eliminar</b> un atleta por su ID.	Ninguno

#### 3. Persistencia de Datos (Base de Datos)

- **Motor:** MySQL.

- **Conexión:** Manejada por la clase ConexionDB que utiliza el conector JDBC.
- **Tabla Principal:** atletas.

Columna	Tipo de Dato	Notas
id	INT	Clave primaria.
nombre , apellido	VARCHAR	Requerido (NOT NULL).
nombre_deporte , especialidad_deporte	VARCHAR	Datos del objeto <code>Deporte</code> .
es_extranjero	BOOLEAN	Utilizado para cálculos en <code>GestorFinanciero</code> .

## II. Manual de Usuario (Modo Consola)

El sistema se puede iniciar en **Modo Consola** si no se proporciona el argumento api al ejecutar Main.java.

### 1. Requisitos y Ejecución

- **Requisitos:** Tener instalado el JDK (Java Development Kit) y los archivos JAR de las librerías (lib/\*) disponibles.
- **Ejecución:** En la línea de comandos, navegue a la carpeta raíz del proyecto y ejecute:

```
java -cp "lib/*;bin" com.proyecto.Main
```

### 2. Menú Principal de Opciones

Al iniciar, se presentará el menú principal:

1. **Registrar Atleta y Entrenamiento:** Permite ingresar los datos de un nuevo atleta, incluyendo su nombre, país, edad, y deporte principal.
2. **Consultar Estadísticas:** Muestra un conteo de la cantidad de atletas agrupados por deporte.
3. **Procesar Pago de Planilla (Gestión Financiera):** Permite ingresar el ID de un atleta para calcular el pago mensual basado en la lógica de `GestorFinanciero`.
4. **Gestión de Persistencia (Guardar/Cargar):** (Opcional, si está implementado) Accede a funcionalidades de importación/exportación.

5. **Generar Reporte CSV:** Crea un archivo reporte\_atletas.csv con todos los datos de los atletas.

6. **Salir:** Finaliza la aplicación.

¡Claro! Con el diagrama de clases (UML) y el código funcional que hemos desarrollado, podemos crear la documentación técnica y un manual de usuario para el **Sistema de Gestión de Atletas**.

---

## Documentación del Sistema de Gestión de Atletas

### I. Documentación Técnica

#### 1. Arquitectura y Patrones de Diseño

El sistema está construido siguiendo un diseño de **Arquitectura por Capas** (Layered Architecture).

Capa	Componente Principal	Objetivo	Patrones Aplicados
<b>Presentación (API/UI)</b>	ApiServer, ConsolaUI	Manejar la interacción (REST o Consola) y la entrada/salida de datos.	Arquitectura en Capas
<b>Servicio / Negocio</b>	AtletaService, GestorFinanciero	Implementar la lógica de negocio (reglas de pago, coordinación de CRUD, reportes).	Principio de Responsabilidad Única (SRP)
<b>Persistencia</b>	AtletaDAO, ConexionDB	Abstraer el acceso a la base de datos (MySQL).	<b>Data Access Object (DAO)</b>
<b>Dominio</b>	Atleta, Deporte	Representar los datos y el estado del sistema.	Modelos POJO (Plain Old Java Objects)

#### 2. Especificación de la API REST

El módulo ApiServer utiliza el *framework* **Spark** para levantar un servidor REST en el puerto por defecto 4567.

Método	Endpoint	Descripción	Cuerpo Requerido
GET	/api/v1/atletas	Obtener la lista completa de atletas.	Ninguno
GET	/api/v1/atletas/{id}	Obtener un atleta específico por su ID.	Ninguno
POST	/api/v1/atletas	<b>Crear</b> un nuevo atleta.	Objeto JSON con datos de Atleta y Deporte.
PUT	/api/v1/atletas/{id}	<b>Actualizar</b> un atleta existente.	Objeto JSON completo del Atleta actualizado.
DELETE	/api/v1/atletas/{id}	<b>Eliminar</b> un atleta por su ID.	Ninguno

### 3. Persistencia de Datos (Base de Datos)

- **Motor:** MySQL.
- **Conexión:** Manejada por la clase ConexionDB que utiliza el conector JDBC.
- **Tabla Principal:** atletas.

Columna	Tipo de Dato	Notas
id	INT	Clave primaria.
nombre, apellido	VARCHAR	Requerido (NOT NULL).
nombre_deporte, especialidad_deporte	VARCHAR	Datos del objeto Deporte.
es_extranjero	BOOLEAN	Utilizado para cálculos en GestorFinanciero.

## II. Manual de Usuario (Modo Consola)

El sistema se puede iniciar en **Modo Consola** si no se proporciona el argumento api al ejecutar Main.java.

## 1. Requisitos y Ejecución

- **Requisitos:** Tener instalado el JDK (Java Development Kit) y los archivos JAR de las librerías (lib/\*) disponibles.
- **Ejecución:** En la línea de comandos, navegue a la carpeta raíz del proyecto y ejecute:

Bash

```
java -cp "lib/*;bin" com.proyecto.Main
```

## 2. Menú Principal de Opciones

Al iniciar, se presentará el menú principal:

1. **Registrar Atleta y Entrenamiento:** Permite ingresar los datos de un nuevo atleta, incluyendo su nombre, país, edad, y deporte principal.
2. **Consultar Estadísticas:** Muestra un conteo de la cantidad de atletas agrupados por deporte.
3. **Procesar Pago de Planilla (Gestión Financiera):** Permite ingresar el ID de un atleta para calcular el pago mensual basado en la lógica de GestorFinanciero.
4. **Gestión de Persistencia (Guardar/Cargar):** (Opcional, si está implementado) Accede a funcionalidades de importación/exportación.
5. **Generar Reporte CSV:** Crea un archivo reporte\_atletas.csv con todos los datos de los atletas.
6. **Salir:** Finaliza la aplicación.

## 3. Uso de la Opción 1: Registrar Atleta

Cuando seleccione la opción 1, el sistema le pedirá los siguientes datos de forma interactiva:

Campo	Ejemplo de Entrada	Notas
Nombre del Atleta	Sofía	Requerido.
País	Guatemala	
Edad	26	Debe ser un número entero.
Deporte principal	Gimnasia	
¿Es extranjero?	true o false	Afecta el cálculo de pago.

Si el registro es exitoso, recibirá un mensaje de **"Atleta [Nombre] registrado exitosamente"**. Si falla, indicará el error de la base de datos (ej: Unknown column o cannot be null).