

**Facultad de Ingeniería de Sistemas**

**y Electrónica**

**TRABAJO FINAL**

**"UTP FUTBOL"**

**CURSO:**

**Lenguaje de Programación**

**DOCENTE:**

**PEDRO HERNAN DE LA CRUZ VELAZCO**

**ALUMNO:**

**Inga Castillo Ronaldhiño Jefferson U21226022**

Lima, Perú  
**22 de Julio del 2025**

**1. Introducción**

**1.1. Resumen Ejecutivo**

El presente proyecto consiste en el desarrollo de una aplicación web dedicada a la gestión del equipo de fútbol de la Universidad Tecnológica del Perú (UTP). Utilizando el framework Flask con MySQL como sistema de base de datos, se ha implementado un sistema CRUD completo que permite administrar jugadores y visualizar información institucional del equipo. El sistema cuenta con una interfaz moderna, dinámica y responsive desarrollada con Bootstrap, que permite una experiencia de usuario llamativa.

**1.2. Descripción del Problema**

El equipo de fútbol de la UTP no cuenta con una plataforma digital para registrar, consultar o actualizar información sobre sus jugadores y su trayectoria. La falta de un sistema centralizado limita la difusión de logros deportivos y dificulta el acceso a la información por parte de estudiantes, entrenadores y personal administrativo. Este proyecto busca cubrir esta necesidad mediante una solución web accesible y funcional.

**1.3. Objetivos del Proyecto**

**1.3.1. Objetivo General**

Diseñar e implementar un sistema web con Flask y MySQL para la gestión del equipo de fútbol de la UTP, que permita administrar jugadores y mostrar contenido institucional e histórico del equipo.

**1.3.2. Objetivos Específicos**

* Desarrollar un CRUD funcional para registrar, editar y eliminar jugadores.
* Diseñar una interfaz atractiva que incluya secciones informativas (historia, trofeos, videos).
* Implementar la carga y visualización de imágenes de los jugadores y trofeos.
* Garantizar la escalabilidad y mantenibilidad del sistema.
* Asegurar que el sistema sea fácilmente replicable e instalable.

**1.4. Alcance del Proyecto**

**Incluye:**

* Módulo de administración de jugadores (CRUD).
* Visualización de historia, trofeos y multimedia del equipo.
* Diseño responsive con Bootstrap.
* Conexión a base de datos MySQL para persistencia de datos.
* Subida y almacenamiento de imágenes.

**No incluye:**

* Módulo de autenticación de usuarios (login).
* Control de permisos por roles.
* Estadísticas en tiempo real o generación de reportes PDF.

**2. Marco Teórico / Tecnológico**

**2.1. Fundamentos Teóricos**

Para el desarrollo del presente proyecto se utilizaron diversos conceptos teóricos fundamentales:

* **Programación orientada a objetos (POO):** Paradigma que permite estructurar el código en clases y objetos, facilitando la modularidad y el mantenimiento.
* **Arquitectura cliente-servidor:** Modelo en el que el cliente solicita recursos y el servidor los proporciona. En este caso, el navegador web actúa como cliente y Flask como servidor.
* **CRUD:** Conjunto de operaciones básicas que permiten Crear, Leer, Actualizar y Eliminar datos en una base de datos.
* **Bases de datos relacionales:** Modelo que organiza los datos en tablas relacionadas mediante claves primarias y foráneas.
* **MVC (Modelo-Vista-Controlador):** Patrón de diseño que separa la lógica del negocio, la interfaz y el flujo de datos.

**2.2. Tecnologías y Herramientas Utilizadas**

**2.2.1. Lenguaje de Programación**

* **Python 3.12**: Lenguaje de alto nivel utilizado para la lógica del sistema y conexión con la base de datos.

**2.2.2. Frameworks y Librerías**

* **Flask**: Microframework de Python que permite desarrollar aplicaciones web ligeras y escalables.
* **Flask-MySQLdb**: Librería que permite la conexión entre Flask y una base de datos MySQL.
* **Jinja2**: Motor de plantillas incluido en Flask que facilita la generación dinámica de HTML.

**2.2.3. Base de Datos**

* **MySQL**: Sistema de gestión de bases de datos relacional utilizado para almacenar información de los jugadores.

**2.2.4. Entorno de Desarrollo**

* **Visual Studio Code**: Editor de código con extensiones para desarrollo web y Python.

**2.2.5. Control de Versiones**

* **Git**: Sistema de control de versiones distribuido.
* **GitHub**: Plataforma para alojar y colaborar en proyectos con control de versiones.

**2.2.6. Otras Herramientas**

* **Bootstrap 5**: Framework CSS para el diseño responsive del frontend.
* **Google Fonts / Font Awesome**: Para tipografías e íconos.
* **XAMPP / MySQL Workbench**: Para la gestión y visualización de la base de datos local.

**2.3. Arquitectura del Sistema**

El sistema sigue una arquitectura **Cliente-Servidor** basada en el patrón **MVC**:

* **Modelo:** Encargado de la interacción con la base de datos (consultas SQL).
* **Vista:** Plantillas HTML con Bootstrap, renderizadas con Jinja2.
* **Controlador:** Funciones en Flask que gestionan rutas y lógica de negocio.

El flujo es el siguiente:

1. El usuario accede desde el navegador (cliente).
2. Flask procesa la solicitud y consulta/modifica la base de datos si es necesario.
3. Se renderiza una plantilla con los datos requeridos y se devuelve al cliente.

**3. Diseño del Sistema**

**3.1. Requisitos del Sistema**

**3.1.1. Requisitos Funcionales**

* El sistema debe permitir **registrar** un nuevo jugador con sus datos personales y foto.
* El sistema debe **mostrar una lista** de todos los jugadores registrados.
* El sistema debe permitir **editar** la información de un jugador existente.
* El sistema debe permitir **eliminar** jugadores de la base de datos.
* El sistema debe permitir **visualizar las imágenes** de los jugadores.
* El sistema debe mostrar **páginas informativas** como "Inicio" y "Nosotros".

**3.1.2. Requisitos No Funcionales**

* El sistema debe tener una **interfaz amigable y responsive** (adaptable a diferentes dispositivos).
* El sistema debe garantizar la **integridad de los datos** mediante validaciones básicas.
* El sistema debe permitir la **subida segura de archivos** (solo imágenes permitidas).
* El sistema debe ejecutarse en un entorno local con **mínimos requerimientos técnicos**.
* El código debe estar **documentado y organizado** para facilitar su mantenimiento.

**3.2. Diseño de la Base de Datos**

**3.2.1. Modelo Entidad-Relación (ERD)**

**Entidad principal:** Jugador

* Atributos:
  + id: entero, clave primaria, autoincrementable.
  + nombre: texto.
  + posición: texto.
  + número: entero.
  + edad: entero.
  + nacionalidad: texto.
  + foto: texto (ruta o nombre del archivo de imagen).

**3.2.2. Esquema de la Base de Datos**

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

**3.3. Diseño de la Interfaz de Usuario (UI/UX)**

**3.3.1. Bocetos/Wireframes (Descripción textual)**

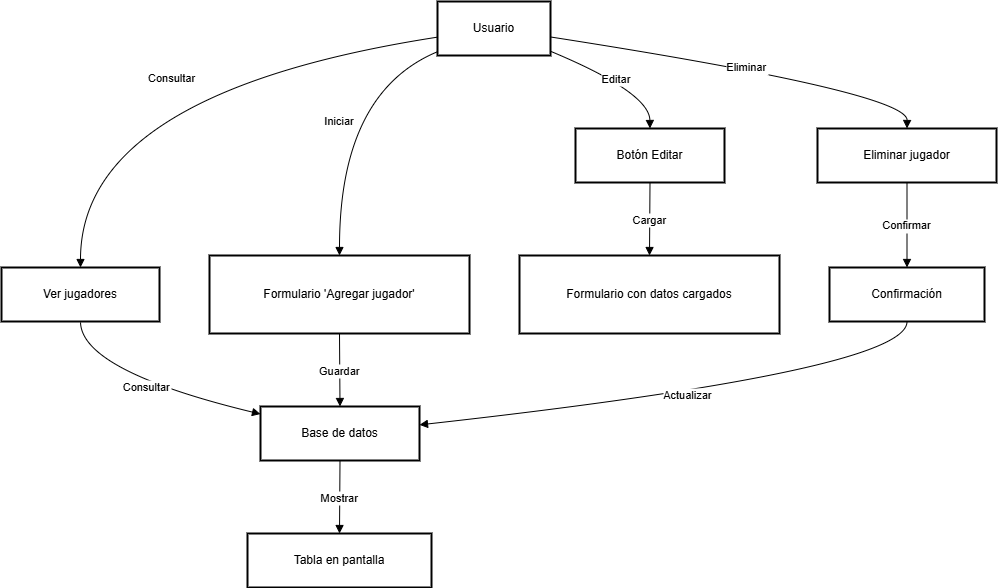
* **Página de Inicio (/inicio)**:
  + Banner del equipo.
  + Descripción del sitio.
  + Tarjetas de trofeos con imágenes generadas por IA.
* **Página de Jugadores (/)**:
  + Tabla con información de cada jugador.
  + Botón para agregar nuevo jugador.
  + Acciones: editar y eliminar.
* **Página de Nosotros (/nosotros)**:
  + Historia del equipo.
  + Misión, visión y valores.
  + Video incrustado de YouTube.

**3.3.2. Flujos de Usuario**

1. **Registrar jugador** → Formulario completo → Guardar → Redirige al listado.
2. **Editar jugador** → Carga formulario con datos existentes → Modificar → Guardar.
3. **Eliminar jugador** → Confirmación (en versión futura) → Eliminación directa.
4. **Visualizar sitio** → Navegación fluida entre Inicio, Jugadores y Nosotros.

**3.4. Diagramas de Diseño**

**3.4.1. Diagrama de Flujo - CRUD**

******

**3.4.2. Diagrama de Componentes**

Diagrama

El contenido generado por IA puede ser incorrecto.

**3.4.3. Diagrama ER (Entidad-Relación)**

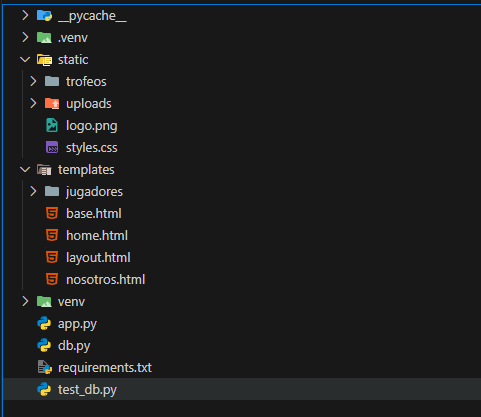
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

**4. Implementación**

**4.1. Estructura de Carpetas/Módulos**

El proyecto está organizado de la siguiente manera:



**4.2. Descripción de Componentes Clave**

**app.py:** Archivo principal que contiene:

* Conexión a la base de datos MySQL.
* Definición de rutas del CRUD.
* Funcionalidad para subir fotos.
* Rutas para páginas informativas (inicio, nosotros).

**Carpeta templates/jugadores/:**

* index.html: Lista todos los jugadores con opciones de editar/eliminar.
* nuevo.html: Formulario para registrar un nuevo jugador.
* editar.html: Formulario para editar un jugador existente.

**layout.html:**

* Plantilla base que contiene el encabezado con el logo de la UTP, el menú de navegación y los enlaces a Bootstrap.

**4.3. Consideraciones de Programación**

* **Patrón utilizado:** El proyecto sigue el patrón **MVC (Modelo - Vista - Controlador)** de forma simplificada.
  + Modelo: La base de datos MySQL.
  + Vista: Plantillas HTML con Bootstrap.
  + Controlador: Rutas definidas en app.py.
* **Buenas prácticas aplicadas:**
  + Separación entre lógica de negocio (Python) y vistas (HTML).
  + Validación básica de archivos al subir fotos (allowed\_file()).
  + Uso de secure\_filename() para evitar vulnerabilidades al guardar imágenes.
  + Uso de @app.route para definir rutas RESTful (/jugadores/nuevo, /editar/<id>, etc).

**4.4. Integración de Módulos**

* El **usuario** accede desde su navegador y envía solicitudes HTTP (GET/POST).
* **Flask** recibe las solicitudes, accede a **MySQL** para obtener o modificar datos, y devuelve vistas HTML al usuario.
* Las imágenes cargadas se almacenan en static/uploads y se muestran en las vistas con rutas relativas.
* Todo está integrado de forma que las operaciones CRUD se ven reflejadas inmediatamente en la interfaz, gracias al renderizado dinámico.

**5. Pruebas y Resultados**

**5.1. Plan de Pruebas**

**5.1.1. Tipos de Pruebas Realizadas**

* **Pruebas Funcionales:** Se verificó que cada funcionalidad del CRUD (crear, leer, actualizar, eliminar jugadores) funcione correctamente.
* **Pruebas de Integración:** Se comprobó la conexión entre Flask y MySQL, así como la carga de imágenes.
* **Pruebas de Interfaz (UI):** Se validó que la navegación entre páginas (Inicio, Jugadores, Nosotros) sea fluida y amigable.
* **Pruebas de Validación:** Se probó la validación del formato de imagen (.jpg, .jpeg, .png).

**5.1.2. Casos de Prueba**

| **Caso** | **Acción** | **Entrada Ejemplo** | **Resultado Esperado** |
| --- | --- | --- | --- |
| 1 | Crear jugador | Nombre: Luis,  Posición: Defensa, Edad: 22 | Jugador se guarda y aparece en la tabla |
| 2 | Editar jugador | Modificar edad a 23 | Edad actualizada correctamente |
| 3 | Eliminar jugador | Click en "Eliminar" en jugador ID 5 | Jugador desaparece de la base de datos |
| 4 | Subir archivo inválido | Imagen con extensión .pdf | Se bloquea subida, no se guarda el jugador |
| 5 | Acceder a página 'Nosotros' | URL: /nosotros | Carga correcta de información institucional |

**5.2. Resultados de las Pruebas**

A continuación, se presentan algunos resultados de las pruebas realizadas:

* Todos los formularios de registro y edición funcionaron sin errores.
* Las imágenes válidas se almacenaron correctamente en la carpeta static/uploads.
* Se validó que no se permitieran archivos no admitidos (como .exe o .pdf).
* Se evidenció que la eliminación y edición de jugadores se reflejan al instante en la tabla.
* Lista de jugadores después de agregar nuevos datos.
* Formulario de edición con campos precargados.
* Error al intentar subir imagen inválida.
* Página 'Inicio' y 'Nosotros' funcionando correctamente.

**5.3. Detección y Solución de Errores**

| **Error Detectado** | **Solución Aplicada** |
| --- | --- |
| Imágenes no se mostraban correctamente | Se corrigió la ruta de acceso agregando /static/uploads/ en la etiqueta <img> |
| Error al subir archivo sin imagen | Se agregó validación para aceptar solo extensiones permitidas |
| Datos no se actualizaban en edición | Se corrigió el método SQL UPDATE y el manejo del id en la ruta |
| Flask no reconocía cambios | Se activó debug=True para recargar automáticamente en desarrollo |

**6. Conclusiones y Recomendaciones**

**6.1. Conclusiones**

* Se logró desarrollar satisfactoriamente una aplicación web CRUD utilizando **Flask** como framework backend y **MySQL** como gestor de base de datos.
* La aplicación permite registrar, visualizar, editar y eliminar jugadores del equipo de fútbol UTP, incluyendo el manejo de imágenes.
* Se implementó un diseño visual atractivo y organizado utilizando **Bootstrap**, manteniendo una buena experiencia de usuario.
* El proyecto permitió aplicar conocimientos adquiridos en el curso de Lenguaje de Programación como: manejo de rutas, formularios, conexiones a bases de datos, plantillas HTML con Jinja2, y estructuras MVC.
* Se reforzó la importancia de una buena estructura del proyecto, el uso de funciones reutilizables y la validación de datos.
* Se desarrolló una funcionalidad adicional para subir imágenes, enriqueciendo la experiencia visual del sistema.

**6.2. Limitaciones del Proyecto**

* La aplicación está orientada a un único tipo de entidad: jugadores. No se incorporaron otras entidades relacionadas como entrenadores, partidos o estadísticas.
* No se implementó autenticación ni control de usuarios, lo cual limitaría la seguridad en un entorno real.
* El sistema no incluye validaciones de formulario avanzadas ni control de errores detallado al conectarse a la base de datos.
* El almacenamiento de imágenes se hace en el sistema de archivos local, lo cual podría no escalar adecuadamente en entornos de producción.

**6.3. Trabajos Futuros / Mejoras**

* Implementar **autenticación de usuarios** con roles (administrador, visitante, etc.) para controlar el acceso al CRUD.
* Agregar módulos adicionales como control de partidos, estadísticas de los jugadores, trofeos, calendarios, etc.
* Integrar **validaciones más robustas** en los formularios y manejo de errores en la conexión con la base de datos.
* Migrar el sistema a un entorno de despliegue profesional, incluyendo **almacenamiento de imágenes en la nube** (por ejemplo, AWS S3).
* Incluir una **API REST** para facilitar el consumo de datos desde otras plataformas (por ejemplo, apps móviles).
* Mejorar la interfaz gráfica incorporando animaciones o diseño responsive más avanzado.

**7. Referencias Bibliográficas**

* Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media.
* The Flask Project. (n.d.). *Flask Documentation*. Retrieved from https://flask.palletsprojects.com
* MySQL. (n.d.). *MySQL 8.0 Reference Manual*. Retrieved from <https://dev.mysql.com/doc/>
* Bootstrap. (n.d.). *Bootstrap v5 Documentation*. Retrieved from <https://getbootstrap.com/>
* Python Software Foundation. (n.d.). *Python Documentation (v3.11)*. Retrieved from <https://docs.python.org/3/>
* GitHub, Inc. (n.d.). *GitHub Docs*. Retrieved from <https://docs.github.com/>
* W3Schools. (n.d.). *Flask Tutorial*. Retrieved from https://www.w3schools.com/flask/
* FreeCodeCamp. (n.d.). *Create a Flask App with Python - Full Tutorial*. Retrieved from <https://www.freecodecamp.org/>

**8. Anexos**

**8.1. Código Fuente del Proyecto**

El código fuente completo del sistema web "Equipo de Fútbol UTP" se encuentra disponible en el siguiente repositorio de GitHub:

**Repositorio:** https://github.com/jeferson-ic/Crud\_FutbolUTP.git

**8.2. Manual de Usuario**

**Inicio de la aplicación:**

1. Ejecutar el servidor Flask (python app.py)
2. Ingresar en el navegador a http://localhost:5000

**Funcionalidades:**

* **Inicio:** Página de presentación del equipo.
* **Jugadores:** Acceso al CRUD donde se puede registrar, editar, ver y eliminar jugadores.
* **Nosotros:** Página informativa con historia, trofeos y ubicación del equipo.

**8.3. Capturas de Pantalla Adicionales**

**Captura 1 – Formulario de Registro de Jugador**

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

*Figura 1. Formulario para registrar un nuevo jugador con campos como nombre, posición, edad, nacionalidad, número y subida de imagen.*

**Captura 2 – Jugador Registrado Exitosamente en la Tabla**

Tabla

El contenido generado por IA puede ser incorrecto.

*Figura 2. Jugador agregado correctamente mostrado en la tabla de jugadores junto a su imagen.*

**Captura 3 – Edición de Jugador**

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

*Figura 3. Formulario precargado para editar los datos de un jugador específico.*

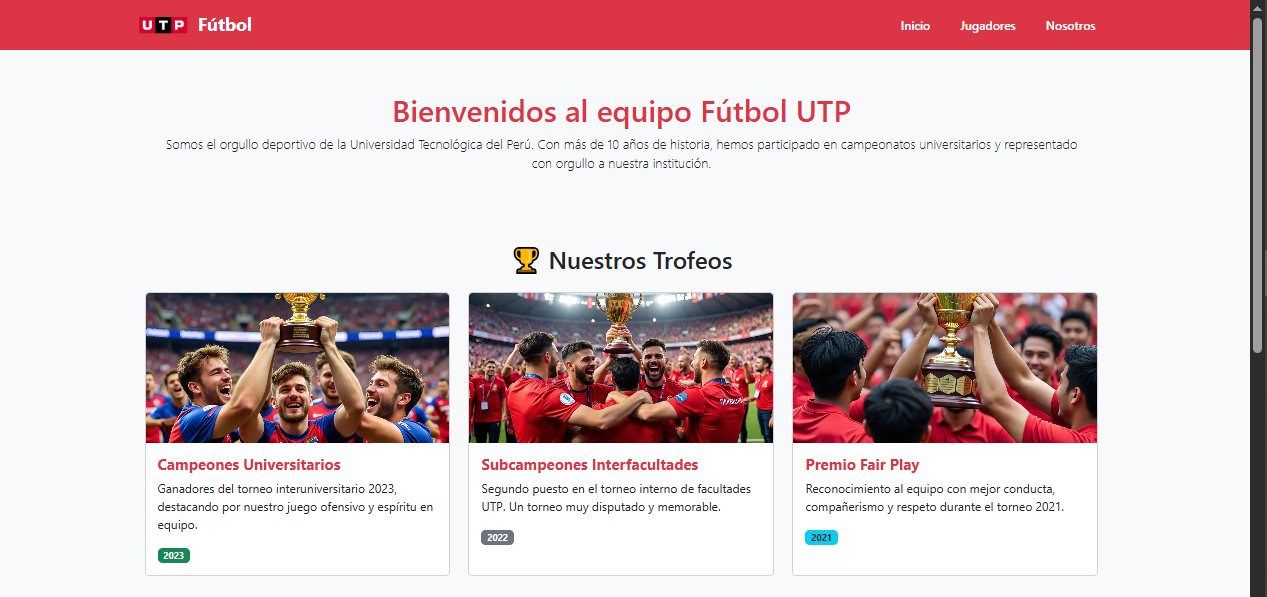
**Captura 4 – Eliminación de Jugador**

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

*Figura 4. Confirmación visual tras eliminar un jugador de la base de datos.*

**Captura 5 – Página de Inicio (Home)**



Una pantalla de un aeropuerto

El contenido generado por IA puede ser incorrecto.

*Figura 6. Página principal de la web del equipo Fútbol UTP con navbar, descripción y cartas de trofeos.*

**Captura 7 – Página Nosotros**

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

*Figura 7. Página institucional "Nosotros"*