OTH REGENSBURG

Applied Data Science with Python: Task 3 – Project 2: Football Players

Manuel Martínez Cobos

Description:

This project analyzes football player data sourced from Transfermarkt (transfermarkt.de), ranking players by market value due to the lack of an objective performance metric. The application allows users to select from predefined clubs or leagues, choose 1–10 options, and specify the number of players to display. After loading data, it presents a ranked player list, with profile pages showing photos, achievements, and statistics visualized using matplotlib and pandas. Key challenges included managing loading times, handling API data dynamically without downloading, and resolving issues with data formats, such as interpreting players' minutes played accurately in charts.

Module, data structures and tools

Modules:

The project utilizes the following modules and libraries to manage data, enhance functionality, and support web interactions:

- **Flask**:
  - A lightweight web framework used for creating routes, handling HTTP requests, and rendering HTML templates.
  - Enables the core functionality of the web application.
- **Requests**:
  - For sending HTTP requests to external APIs to retrieve data dynamically, such as player statistics, market values, and achievements.
- **Pandas**:
  - Utilized for processing and manipulating structured data, specifically for organizing player statistics and generating visualizations.
- **Matplotlib**:
  - For creating and rendering bar charts that display player performance metrics.
- **Base64**:
  - For encoding generated graphs into a format suitable for embedding in HTML templates.
- **Requests:**
  - Connecting the backend with the frontend (HTML files)

Data Structures:

The project employs several key data structures to efficiently store and manage data:

- **Dictionaries:**
    - Usage:
        - Represent leagues and clubs with their names, codes, and logos.
        - Organize player data, such as statistics, achievements, and market values.
- **Lists:**
    - Usage:
        - Store all players globally (*todos_los_jugadores*).
        - Process and filter selected leagues or clubs.
        - Store stats and achievements from players
        - Store all players in a club
        - Store all clubs in a league
        - Store all leagues/clubs selected

- **Custom Classes:**
    - Usage:
        - Store all data from players, clubs and leagues

Tools:

The project integrates various tools and libraries to streamline development and visualization:

- ThreadPoolExecutor:
    - For parallelizing API requests (e.g., fetching player statistics and achievements concurrently) and to accelerate a lot of the requests of the project.
- Flask Templates:
    - Enable dynamic rendering of web pages with data passed from Python scripts.
- Base64 Encoding:
    - Convert charts into an embeddable format for seamless HTML integration.
- External API:
    - Provide real-time data for leagues, clubs, players, and their performance metrics.
- Requests:
    - Connects the backend (python language) to the frontend (HTML language)

High-Level design of the project

The project is organized into modular components that interact through clearly defined responsibilities. The communication between modules ensures data flow and functionality across the application.

Module Overview:

1. Flask Application:
    - Acts as the core of the project, handling HTTP requests and routing.
    - Communicates with the frontend (HTML templates) and backend modules.
2. API Interaction (via requests):
    - Fetches data dynamically from external APIs (e.g., player statistics, achievements).
    - Modules like funcionesJugadores, funcionesLiga, and funciones_club handle the logic for API calls and processing responses.
3. Data Processing Modules:
    - funciones_club:
        - Retrieves players for specific clubs and processes their details.
    - funcionesJugadores:
        - Handles operations like converting market values and generating visualizations.
    - funcionesLiga:
        - Manages leagues and their associated teams.
4. Frontend Integration:
    - Flask templates render the processed data (e.g., player statistics, league details) dynamically for user interaction.

Module Communication:

1. User sends a request (selecting one or more leagues or clubs).
2. Flask routes process the request and call the appropriate backend functions.
3. Backend functions fetch or process data:
    - Communicate with external APIs for dynamic data retrieval.
    - Use helper functions for calculations and data formatting.
4. Processed data is returned to Flask for rendering via templates and shows as many players as the user selects.
5. Now the user can select a player and see his complete profile.

Flowchart

1. Fetching Top Players in Selected Leagues

1. User selects leagues and the number of players.
2. Flask route receives request.
3. Backend fetches all teams and players for selected leagues.
4. Players are sorted by market value.
5. Top x players are sent to the frontend for display.

    2. Displaying Player Profiles

1. User selects a player.
2. Flask route fetches the player details.
3. Additional player data (statistics, achievements) is retrieved via API.
4. Statistics are processed into graphs.
5. Profile data is displayed to the user

3. Selecting and Displaying Teams in a League

1. User selects a league.
2. Flask route retrieves all teams in the league.
3. Teams are displayed to the user for further interaction.

Key Functions

Problem 1: Fetching Teams in a Selected League

- Function: *obtener_equipos_por_liga(liga_id)*
    - Parameters:
        - liga_id (string): The unique identifier for the league.
    - Description: Fetches all teams in a given league using an API call.

Problem 2 : Fetching Players for a Club

- Function: obtener_jugadores_por_club_simple(club_id)
    - Parameters:
        - club_id (string): The unique identifier for the club.
    - Description: Retrieves a list of players belonging to a specific club using the club ID.

Problem 3: Fetching Top X Players from Selected Leagues

- Function: X_jugadores_ligas(ligas, x)
  - Parameters:
    - ligas (list): A list of league objects containing their teams and players.
    - x (int): The number of top players to fetch based on market value.
  - Description: Sorts all players across selected leagues by market value and retrieves the top x players.

Problem 4: Displaying Player Profiles

- Function: completar_jugador(jugador)
  - Parameters:
    - jugador (Jugador): An object representing the player to be completed.
  - Description: Fetches additional data for a player, including achievements, statistics, nationality, and profile picture, and updates the Jugador object.

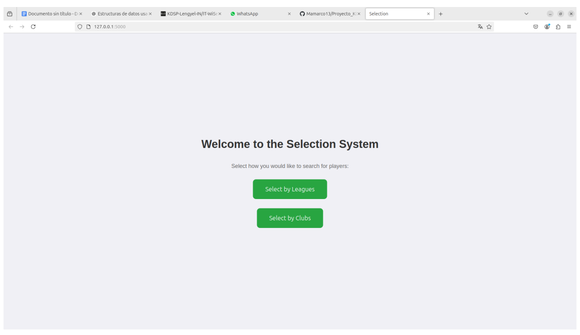
Problem 5: Converting Market Values

- Function: convertir_valor_mercado(valor_texto)
  - Parameters:
    - valor_texto (string): The textual representation of a market value (e.g., "€80M").
  - Description: Converts textual market values into numeric values for sorting and calculations.

Problem 6: Displaying Player Statistics in Graphs

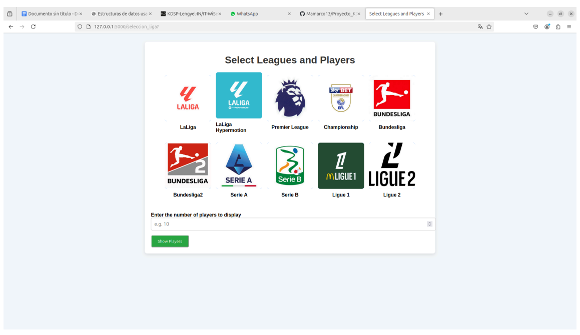
- Function: mostrar_estadisticas_grafico(estadisticas)
  - Parameters:
    - estadisticas (list): A list of dictionaries containing player statistics (e.g., goals, assists, appearances).
  - Description: Generates a bar chart for player statistics, encodes it in Base64, and returns the string representation.

Screenshots


Welcome to the Selection System

Select how you would like to search for players:

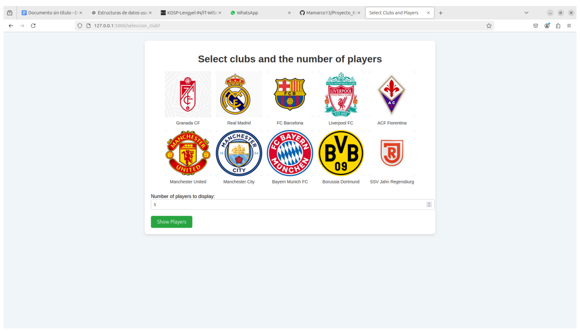
Main Screen


Select Leagues and Players

Enter the number of players to display

e.g. 10

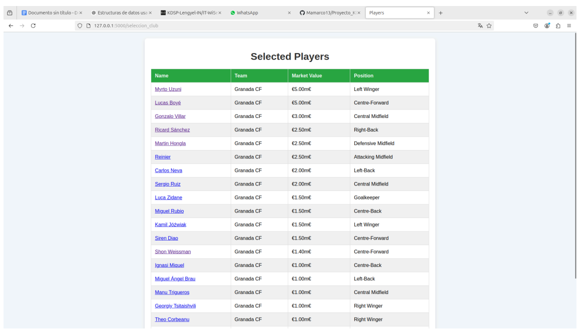
Selecting Leagues


Select clubs and the number of players

Number of players to display:

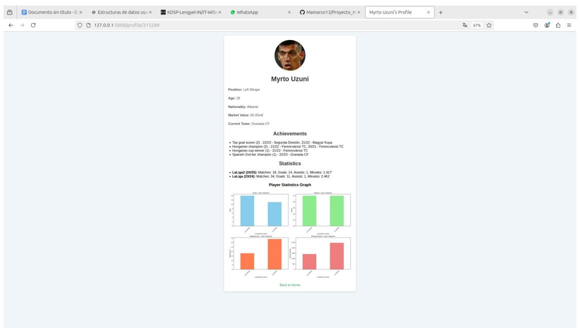
5

Selecting Clubs


Selected Players

Name	Team	Market Value	Position
Myrto Uzuni	Granada CF	€5.00m€	Left Winger
Lucas Boyé	Granada CF	€5.00m€	Centre-Forward
Gonzalo Villar	Granada CF	€3.00m€	Central Midfield
Ricard Sánchez	Granada CF	€2.50m€	Right-Back
Martin Hongla	Granada CF	€2.50m€	Defensive Midfield
Reinier	Granada CF	€2.50m€	Attacking Midfield
Carlos Neva	Granada CF	€2.00m€	Left-Back
Sergio Ruiz	Granada CF	€2.00m€	Central Midfield
Luca Zidane	Granada CF	€1.50m€	Goalkeeper
Miguel Rubio	Granada CF	€1.50m€	Centre-Back
Kamil Jóźwiak	Granada CF	€1.50m€	Left Winger
Siren Diao	Granada CF	€1.50m€	Centre-Forward
Shon Weissman	Granada CF	€1.40m€	Centre-Forward
Ignasi Miquel	Granada CF	€1.00m€	Centre-Back
Miguel Ángel Brau	Granada CF	€1.00m€	Left-Back
Manu Trigueros	Granada CF	€1.00m€	Central Midfield
Georgiy Tsitaishvili	Granada CF	€1.00m€	Right Winger
Theo Corbeanu	Granada CF	€1.00m€	Right Winger

Players selected


Myrto Uzuni

Position: Left Winger

Age: 29

Nationality: Albania

Market Value: €5.00m€

Current Team: Granada CF

Achievements

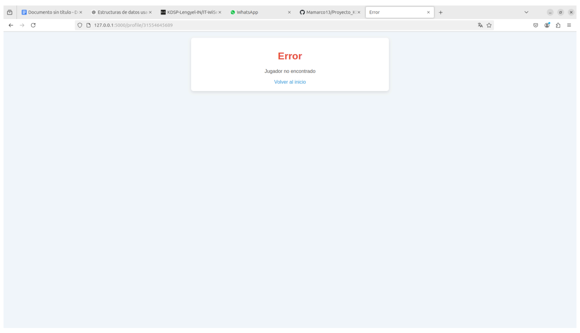
- Top goal scorer (2) - 22/23 - Segunda División, 21/22 - Magyar Kupa
- Hungarian champion (2) - 21/22 - Ferencvárosi TC, 20/21 - Ferencvárosi TC
- Hungarian cup winner (1) - 21/22 - Ferencvárosi TC
- Spanish 2nd tier champion (1) - 22/23 - Granada CF

Statistics

- **LaLiga2 (24/25):** Matches: 18, Goals: 14, Assists: 1, Minutes: 1.417'
- **LaLiga (23/24):** Matches: 34, Goals: 11, Assists: 1, Minutes: 2.461'

Player Statistics Graph

Player Profile


Error

Jugador no encontrado

Error Page