

## Unlocking Tennis Insights from SportRadar! 🚀

This project isn't just about tennis; it's about conquering data. We've harnessed the power of the Sport Radar API to deliver a comprehensive tennis data analytics solution.

From raw API responses to interactive dashboards, we've covered the entire court in 3 steps.

### 1. Data Acquisition & Transformation (Python's Precision):

- **API Web Scraping (Aggressive Baseline):** We didn't just get the data; we \*scraped\* it! The `"Tennis_data_web_scrapped.ipynb"` notebook showcases how we pulled data directly from the Sport Radar API using Python's `"requests"` library.
- **JSON to Python Mastery (Parsing Power):** The API serves data in JSON format, and we're not afraid of nested structures. We used Python's `"json"` library to parse these responses, transforming them into Python objects for efficient manipulation.
- **Pandas Powerhouse (Data Wrangling):** Pandas is our secret weapon! We used it extensively to structure the data into Data Frames, enabling tasks like Creating tables for competitors, complexes, and doubles data.
- **Merging DataFrames:** With Pandas library we combined competitor rankings with competitor details similar to joins in SQL using attribute `"merge()"`, creating a unified view of the data. Also, handled missing values (`"fillna()"`) to ensure data integrity.
- **Data Persistence (CSV Serve):** We saved the processed DataFrames into CSV files (`"competitor.csv"`, `"complexes.csv"`, `"doubles_data.csv"`) for easy access and use in our Streamlit app.

### 2. SQL (Query Control):

- **Targeted Queries:** Check out `"SQL_queries.sql"` We've crafted a set of SQL queries designed to extract specific insights from the tennis data. These queries demonstrate:
  - \* Basic data retrieval (`"select"`).
  - \* Filtering data based on conditions (`"where"`).
  - \* Grouping data and using aggregate functions (`group by, count(), sum() etc...`).
  - \* Ordering results (`order by`)

\* These queries are aligned with the analysis goals specified in the "Game Analytics" document.

### 3. Streamlit Showcase:

- **Dynamic Dashboard:** The “`Web_app_tennis_data.py`” script brings the data to life with a Streamlit web application.
- **Data Loading (Ready to Play):** The app reads the CSV files generated in the data processing step using “`pd.read_csv()`”.
- **Interactive Filtering:** Streamlit's widgets (“`st.selectbox()`”, “`st.slider()`”) allow users to dynamically filter the data:
  - \* Select different datasets (competitor, complexes, doubles).
  - \* Filter competitors by category, type, and gender.
  - \* Filter complexes by venue, country, timezone, and complex name.
  - \* Filter doubles data by rank, competitor name, country, competitions played, and points.
- **Data Display:** The filtered data is presented in an interactive DataFrame using “`st.dataframe()`”, along with row and column counts for quick analysis.

### Your Analytical Advantage: 🚀

This project empowers you to

- **Explore Competition Hierarchies:** Dive deep into tournament structures.
- **Analyze Event Trends:** Visualize data distributions and patterns.
- **Gain Player Insights:** Analyze competitor data and performance metrics.
- **Leverage Interactive Tools:** Use the Streamlit app to filter and explore the data on your own terms.