Report on Project

Introduction

This report includes the design, implementation, and difficulties encountered during the development of the project.

The website provides football fans with access to game results, competition information, player information, and chat rooms for team discussions.

It also includes advanced data analysis with Jupyter Notebook.

Solution

Design and its Motivations:
The main page connects to an Express server, which serves as the central server. This server interfaces with two additional servers:

1. **Spring Boot Server**: Connects to a PostgreSQL database storing slower-changing data (e.g., competitions, game lineups, players, clubs, player valuations).
2. **Express Server**: Connects to a MongoDB database for fast-changing data (e.g., appearances, game events, club games, games).

Jupyter Notebook allows users to select and visualize advanced statistics.

I implemented a web page that allows users to choose the desired stats, generating the query that can be executed in Jupyter Notebook.

Advantages/Disadvantages:

- **Advantages**: It's a modular system, which allows for easy adjustments and expansions. The Jupyter Notebook feature is particularly notable for providing advanced analytical capabilities.
- **Disadvantages**: Merging data from two different databases can become problematic when handling complex queries and can increase development time. Using a single database would have simplified writing queries and sped

up development. However, in the long term, it could have resulted in a poor user experience due to slow data retrieval for some types of information.

Challenges Faced:

- **Database Connectivity**: Establishing connections between servers and databases, particularly with Spring Boot, was challenging due to performance issues on my dated computer(Macbook pro 13’ 2017).
- **Complex Queries**: Some queries were difficult to implement, both for syntax and data merging.
- **CORS Policies**: I had some issues with CORS policies, which were preventing me from making calls to API’s endpoints
- **Assignment Misunderstandings**: Misinterpretations of the assignment led to multiple changes in the project, which significanlty slowed me down.

Requirements

The design meets all specified requirements listed in the assignment.

Limitations

Exceptional Situations and Extensibility:

- **Data Completeness**: Additional basic information in the dataset would have made for a better website.
- User **Login/Registration**: User data login (POST requests) are not implemented, which allows any user to enter the chat function.
- **Data Persistence**: Messages are not stored in a database, limiting chat history.
- **Data Insertion:** I purposely omitted creating POST request functions because I wrongly assumed that updated data would be inserted directly by an eventual admin through the DBMS or MongoDB interface.
- **Statistical Display**: Users must open separate pages(from the main website) to compare stats between multiple competitions, players, or clubs.

  I strongly beleive that this feature can be improved

Extensibility

The design can be easily be adapted for other requirements and is highly extensible. To add a new function, you need to follow simple steps:

1. **Data Availability**: Ensure the function uses data available in the database, if it requires data querying.
2. **Route Implementation**: Implement the route using the MVC pattern.
3. **PostgreSQL Database**: To introduce a new table, add its class in Java using JPA. Follow the pattern of creating a main class, a controller class, a service class, and a repository class.
4. **MongoDB Database**: For MongoDB, create the necessary files following the MVC pattern. Introduce the model, the controller class, and, if required, the view.

Future improvements could also include database integration for message persistence and user authentication.

Conclusions

Implementing this project from scratch displayed the importance of an organized development and attention to small details.

I beleive my understanding of server/database connections, coding practices, and project management, has greatly improved.

Division of Work

The entire project was completed by me.

Extra Information

Running the Code

1. **Database Setup**:

   Import the data into the databases.

   - Import PostgreSQL schema from `IUM-TWEB_Asuenimhen_Marvel/postgre_database_schema.sql`.

- Import MongoDB schema from `IUM-TWEB_Asuenimhen_Marvel/mongo_database_schema/Progetto_IUM_TWEB`.

2. **Server Configuration**:
   - Update PostgreSQL credentials in `application.properties` located at `IUM-TWEB_Asuenimhen_Marvel/solution/java_springboot/progetto_ium_tweb/src/main/resources`.
   - Update MongoDB credentials in the Express server configuration at `IUM-TWEB_Asuenimhen_Marvel/solution/express_second_server/databases`.
3. **Jupyter Notebook Configuration**:
   - Ensure correct credentials in `config.json` at `/Users/marvel/Documents/School/Uni/3°/IUM-TWEB-SERVIZI/Project/IUM-TWEB_Asuenimhen_Marvel/solution/jupyter_data_analysis/config`.
   - Run all cells before executing specific functions.

Bibliography

- OpenAI. (2024). ChatGPT [Large language model]. Retrieved from https://www.openai.com/chatgpt

  "ChatGPT was used to assist with the implementation of repetitive class structures and complex queries (OpenAI, 2024)."