* 1. Introduction

Football Database Management system deals with all kinds of game details, team details, player details, stats and other resource related details too. It tracks all the details of the game and the events and acknowledges the user about the particulars of the teams and the players in that team as well as their statistics.

Our design can facilitate us to explore all the activities happening in the game, we even get to know the teams involved in a match and the players involved in that match. The Football Database Management system can handle all details about the game. The details include Coach details, Team details, Player personnel details, Match Fixture details and the result of the game.

These categories of database management system helps the user to know the sequence of game series in one click and it will give them the information about the players and teams and their standings in the game. We have also included a feedback form that lets the users to review the developed system and the required features and refactoring can be done.

Our system has six modules, they are team, student, match fixture, stats, results, and coach. These modules and its attributes with entity relationship module presented in the ER diagram section.

Chapter 2

Project Review

2.1 Project management

Project management skills are put to good use for this project. Having gone through project management modules in Time Series Analysis, Optimization and with two interns Project Management for Business and IT respectively, they enhanced my knowledge on managing a project. Project management focuses on achieving the objectives by applying five processes presented in Figure below.

Monitoring

Initiating

Planning

Closing

Executing

Figure 2.1: Project Development Phases

2.2 Football database management system

There are many software development companies that offer football database management system in the market. There are records on the past years projects on football database management system is done by football enthusiasts. Through the researches, it is observed that there are features where this project can adopt and implement.

Chapter 3

Resources

3.1 System Development Life Cycle

Systems Development Life Cycle (SDLC) is the most common process adopted to develop a project and not surprisingly, this project is following this model too. To be precise, **agile model** is being applied. Agile SDLC model  is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations.

Requirements

Design and

Architecture

Test and Feedback

Development

Figure 3.1: SDLC Phases

There are five phases in this model and the first phase is the planning stage. The planning stage determines the objectives of the project and whether the project should be given the green light to proceed. This is where the proposal submission comes into picture. After obtaining the approval, the next phase is analysis. Gathering and analysing the system and user requirements is essential for entry to the design step.

With the user requirements gathering completed, there is a need to prepare the resources for the project. Be it software or hardware components, careful consideration and selection is to be taken care at this stage. The decision on the appropriate resources to be used is further elaborated under the subsections below. The next step is to design the system and database structure.

Results from the analysis and preparation that were concluded from the previous stage are put into action. With the user requirements in mind, the flow of the system is planned and the user interface is designed to suit their easy navigation needs. In addition, the number of tables, attributes, primary and unique keys of the database is listed.

After completing the design, actual coding begins. Database is created and codes are written. Some of the codes required amendments and improvement to it so these are being developed at this fourth stage of the waterfall model. With the development completed, testing will begin. The codes and database are tested to ensure the results obtained are as intended. More time is spent on both development and testing stages because it is inevitable to have errors and issues and buffer time is allocated for troubleshooting.

3.2 Scripting Language Selection

There are many scripting languages available in the market. VBScript, Perl, and PHP (Hypertext Pre-processor) are some of those commonly used. Yet for this project, Python is the language that is utilised for the coding piece because it is a general purpose scripting language, template-able HTML language. Being a most popular open source scripting language, it is free for everyone to use and is especially suited for web development, code machine learning models, web scrapping, creating search engines, and artificial intelligence. There are many advantages for using Python thus no need for the switch to another scripting language.

In addition to the standard libraries there are extensive collections of freely available add-on modules, libraries, frameworks, and tool-kits. These generally conform to similar standards and conventions; for example almost all of the database adapters (to talk to almost any client-server RDBMS engine such as MySQL, Postgres, SQLite3, Oracle, etc) conform to the Python DBAPI and thus can mostly be accessed using the same code. So it's usually easy to modify a Python program to support any database engine[1].

Python is famous for being the "batteries are included" language[2]. There are over 300 standard library modules which contain modules and classes for a wide variety of programming tasks. For example the standard library contains modules for safely creating temporary files (named or anonymous), mapping files into memory (including use of shared and anonymous memory mappings), spawning and controlling sub-processes, compressing and decompressing files (compatible with gzip or PK-zip) and archives files (such as Unix/Linux "tar"), accessing indexed "DBM" (database) files, interfacing to various graphical user interfaces (such as the TK toolkit and the popular WxWindows multi-platform windowing system), parsing and maintaining CSV (comma-separated values) and ".cfg" or ".ini" configuration files (similar in syntax to the venerable WIN.INI files from MS-DOS and MS-Windows), for sending e-mail, fetching and parsing web pages, etc. It's possible, for example, to create a custom web server in Python using less than a dozen lines of code, and one of the standard libraries, of course[1].

In conclusion, Python is the preferred selection due to the ease of usage and it can be uploaded and run on another platform with minimal changes needed to be done to the script. Beyond and above, the compiling time and speed for Python is faster and more efficient.

3.3 Database Selection

There are variety of databases that we can select from the market. The widely used databases are Microsoft Access, Microsoft SQL, Oracle and MySQL. Looking at Microsoft Access, it does not encourage concurrent usage and it may be inefficient, as the database needs to be saved into one file. It is also unable to process high speed and large size database as compared to SQLite.

In terms of costs, Oracle database requires a licensing fee but MySQL database is a freeware. SQLite is also a RDBMS but a lightweight one, it doesn't require it's own process, clustering or user management unlike MySQL or others RDBMS.

SQLite is a C library that provides a lightweight disk-based database that doesn’t require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language. Some applications can use SQLite for internal data storage. It’s also possible to prototype an application using SQLite and then port the code to a larger database such as PostgreSQL or Oracle[3].

To conclude, Python script is able to run faster with SQLite database as it is built-in and the processing time will definitely be shorter. This project does not require complex and costly software for its database management system hence SQLite is the ideal database.

3.4 Web Server Selection

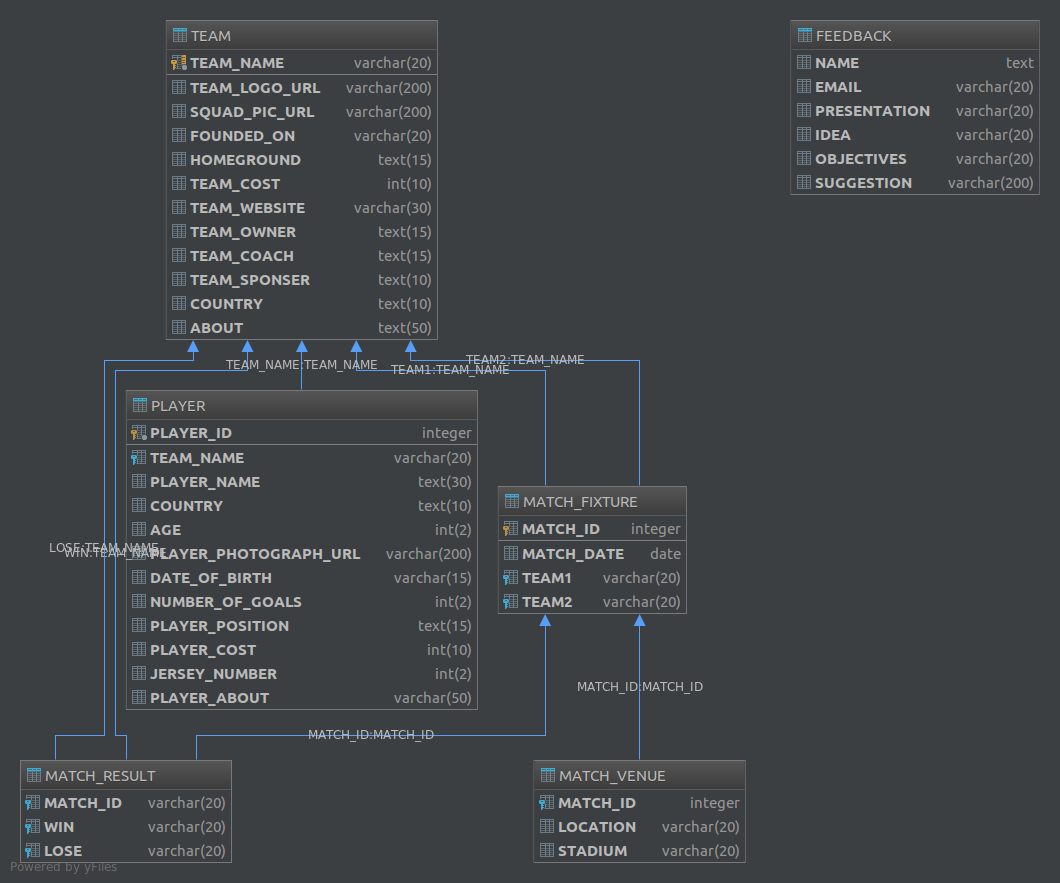
After deciding on the scripting language and database, next is to select the web server that can support them. Web server is necessary for the delivery of web content to the web browser. As such, **Flask** is a micro web framework written in Python and based on the Werkzeug toolkit and Jinja2 template engine[4].

Flask is called a micro framework because it does not require particular tools or libraries.[[6]](https://en.wikipedia.org/wiki/Flask_(web_framework)#cite_note-6) It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools[4].

Chapter 4

System Analysis and Design

4.1 Schema Diagram

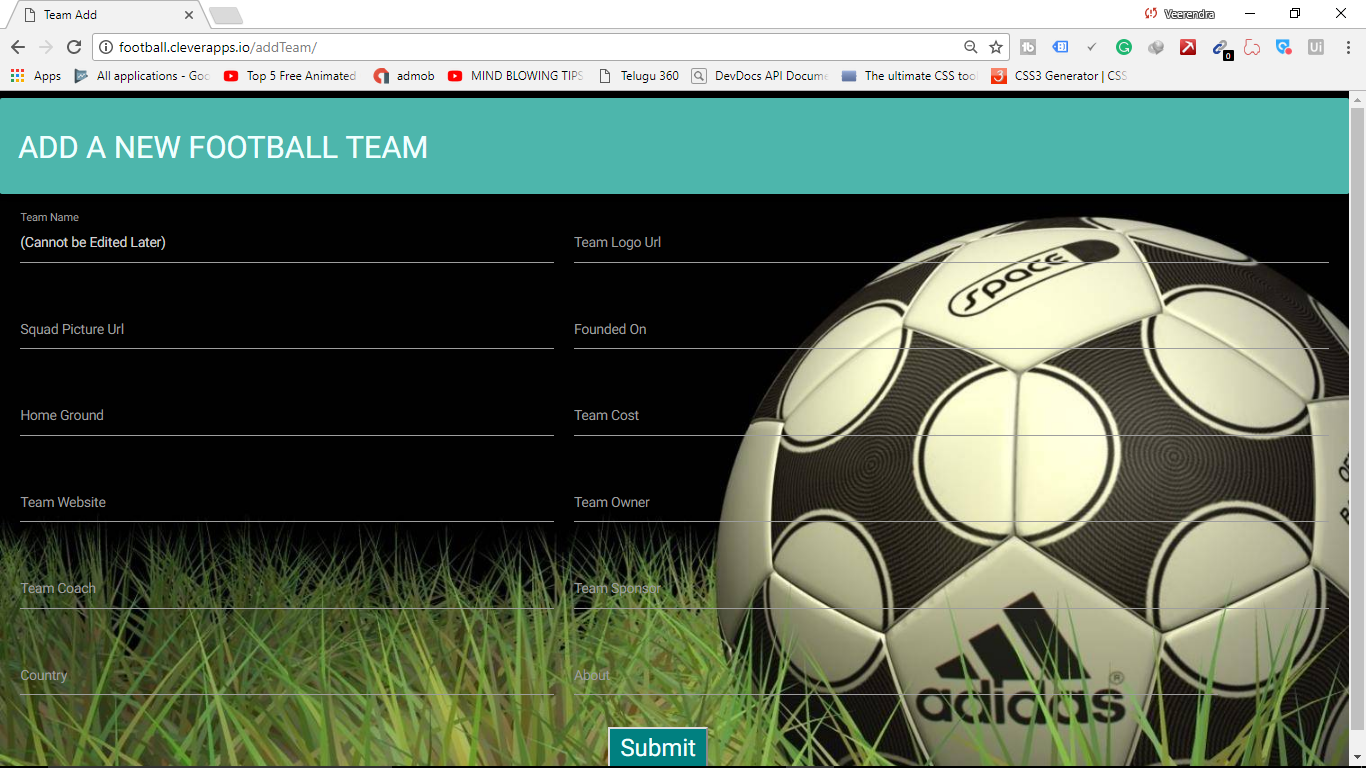


4.2 ER Diagram

4.3 System Design

Under the existing football database management module, the user has to first create a team. This includes the addition the team particulars like team name, logo, squad picture, founded on, home ground, team cost, website, team owner, coach, country, sponsor and about the team.

Figure below depicts the interface of team add form. After the user adds a team by providing the required information, the team gets added to the database.



Figure

References

[1]: “<https://en.wikiversity.org/wiki/Python_Concepts/Why_learn_Python#cite_note-3>”

[2]: “<https://en.wikipedia.org/wiki/Python_(programming_language)#Libraries>”

[3]: “<https://docs.python.org/2/library/sqlite3.html>”

[4]: “<https://en.wikipedia.org/wiki/Flask_(web_framework)>”

[5]: