**CHAPTER 1**

**Introduction**

The project is implemented using Python 3.7.4 and MySQL.

Python is an interpreted, high level programming language developed by Guido van Rossum in 1991, It has a comprehensive standard library, focuses on code readability and supports multiple programming paradigms such as procedural programming, object oriented programming and functional programming.

MySQL is an open-source relational database management system developed by MySQL AB, MySQL can be connected to Python and can be used in Python using the SQL connecter which can be installed using PIP.

This project focuses on implementing a Graphical User Interface application on python and connecting it to a backend database that allows for dynamic change of data on the Graphical User Interface.

* 1. **Course Objectives**
* Acquiring knowledge of python programming language.
* Acquiring knowledge of MySQL RDBMS.
* Using Tkinter to build a Graphical User Interface connected to SQL.
* Learn to develop ER models to make efficient and easily accessible databases.
* Develop problem solving ability and gain debugging skills.
  1. **Problem Definition**

To create a database consisting of player database belonging to different teams for a football league and creating a Graphical User Interface allowing users to create their own fantasy football team by allowing them to choose players from the various teams present in the league and earn points.

* 1. **Outcomes of the Project**

The outcomes of the project are: -

* Provide an interface for users to enter their details and have their own username and password for authentication.
* Provide user player data for them to make a fantasy football team of their choice and earn points to beat other players.
* A database of original football teams with players and their stats to allow users to create their fantasy football team.
* Maintain a database of registered members of the fantasy football league and rank them according to points earned.

**Chapter 2**

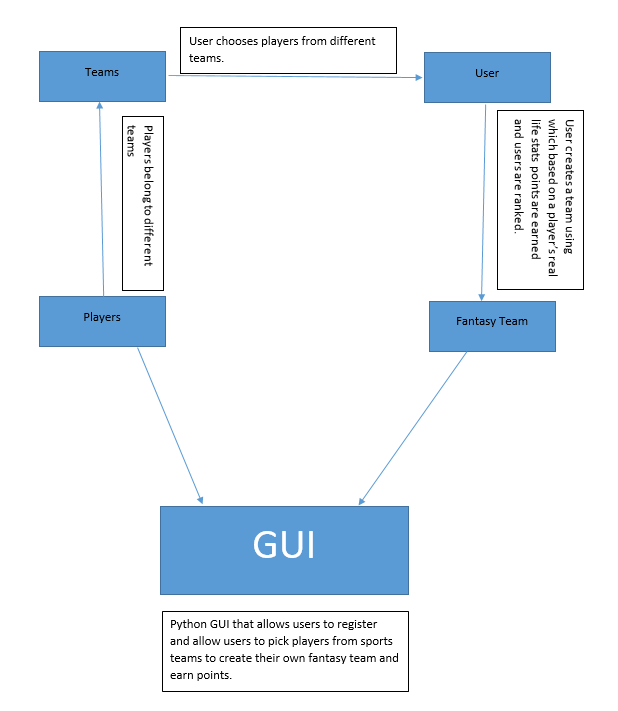
**Requirements and Design**

**2.1 Software Specification**

* Windows 10
* MySQL
* Python 3.7.4
* Tkinter
* Pillow

**2.2 Hardware Specification**

* Processor: Intel i3 or greater.
* RAM:1GB or more.
* Hard disk:256gb or more

**2.3 Flow Diagram**

2.3a Flow Diagram

**Chapter 3**

**Data Models and ER Models**

ER models are represented with entities, their attributes and relationships between the entities.

**3.1 Entity and Attributes**

**Entity –** An entity represents a real world object present in the mini world that is

described in the database.

In an ER diagram an entity is represented using a rectangle.



3.1a Entity

**Attributes** – Attributes are the properties that describe an entity, an entity

have several attributes and each can be of same type or different type as shown

below: -

* **Single Valued Attributes** – These are attributes that have only a single atomic value for each column. Example: First Name of a person.



3.1b Single Valued Attribute

* **Multi Valued Attributes** – These are attributes that have multiple values in each column, often separate tables are made for such values. Example: An office having multiple locations.



3.1c Multi Valued Attribute

* **Composite Attributes** – These are attributes that have multiple columns to address but belong to the same attribute type. Example: Name of a person split into First name, Middle name and Last name.



3.1d Composite Attribute

* **Key Attributes** – These are attributes that act as primary key attribute for an entity. Example: Unique ID of an employee in a company.



3.1e Key Attribute

* **Derived Attributes** – These are attributes that are related to another attribute or derived from another attribute. Example: Next of Kin of a patient in the hospital.



3.1f Derived Attribute

**3.2 Keys**

A key is a single or combination of fields in a table that allows to identify

records and also create relationship between different tables.

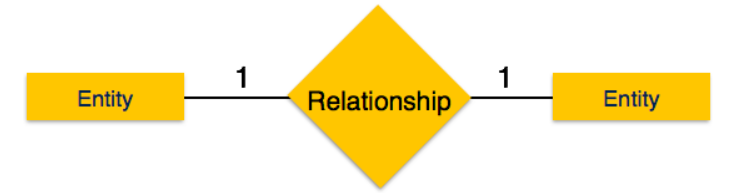
The various keys present in SQL are: -

* **Super Key** - A set of one or more keys that can used to identify a record uniquely in a table.
* **Primary Key** - A set of one or more fields that can identify a record in a database table. Only one primary key can exist per table and not more.
* **Unique Key** - A Unique Key specifies that a field can have only one null value and no duplicate values.
* **Alternate Key** - Alternate key works as a primary key if required.
* **Candidate Key** - Fields in a table that are capable of being a primary key are called candidate keys.
* **Foreign Key**- A key used to reference data in particular table and is the primary key of a different table.

**3.3 Relationship and Participation**

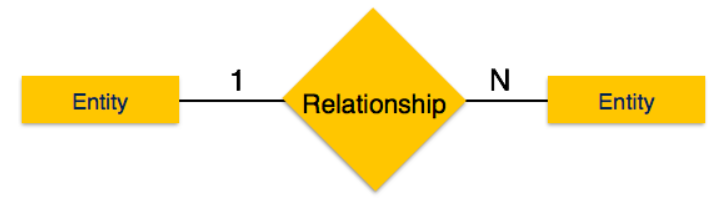
A relationship is the association between different entities in an ER model.

The various relationships are: -

* **1:1 Relationship**: In this project one fantasy team is owned by one user.

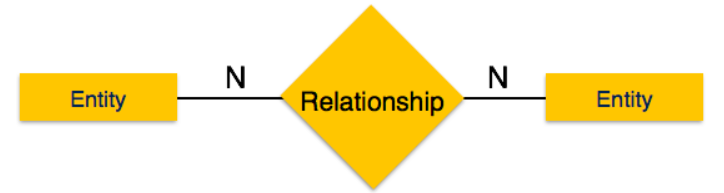
3.3a 1:1 Relationship

* **1:N Relationship**: In this project one football team contains many players.



3.3b 1:N Relationship

* **N:N Relationship**: In this project different fantasy team players are present in football teams.

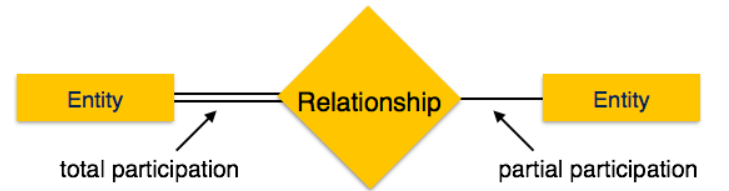


3.3c N:N Relationship

Participation refers to the number of participants from each entity are present in a relationship.

The various participation constraints are: -

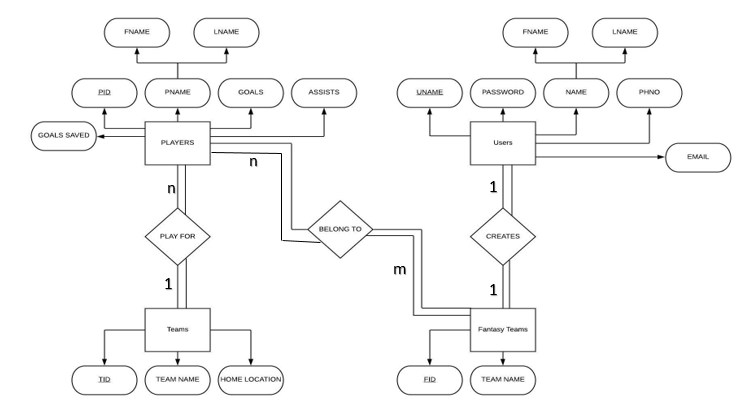
* **Total Participation**: Every User has a fantasy team and every fantasy team in the project is owned by a user.
* **Partial Participation**: All players in the fantasy teams have football players but not all football players are in a fantasy team.



3.3d Participation constraints

**3.4 Data Assumptions**

* There are football players with Player ID, Player name including first name and last name, Goals scored, Assists given to score goals, Goals Saved (Null for players who are not goal keepers).
* There are teams containing football players with Team ID, Team Name, Home Location.
* There are users in the fantasy league containing Username, Password, Name including first name and last name, Date of birth, Phone number, Email.
* There are fantasy teams created by users with players of their choice and the teams have a Team ID and Team name.
* All Teams have 11 players but each player is present in one and only one team.
* All Fantasy teams have 11 players and many players maybe present in many fantasy teams.
* Every Fantasy Team can be owned by only one user and every user owns only one fantasy team



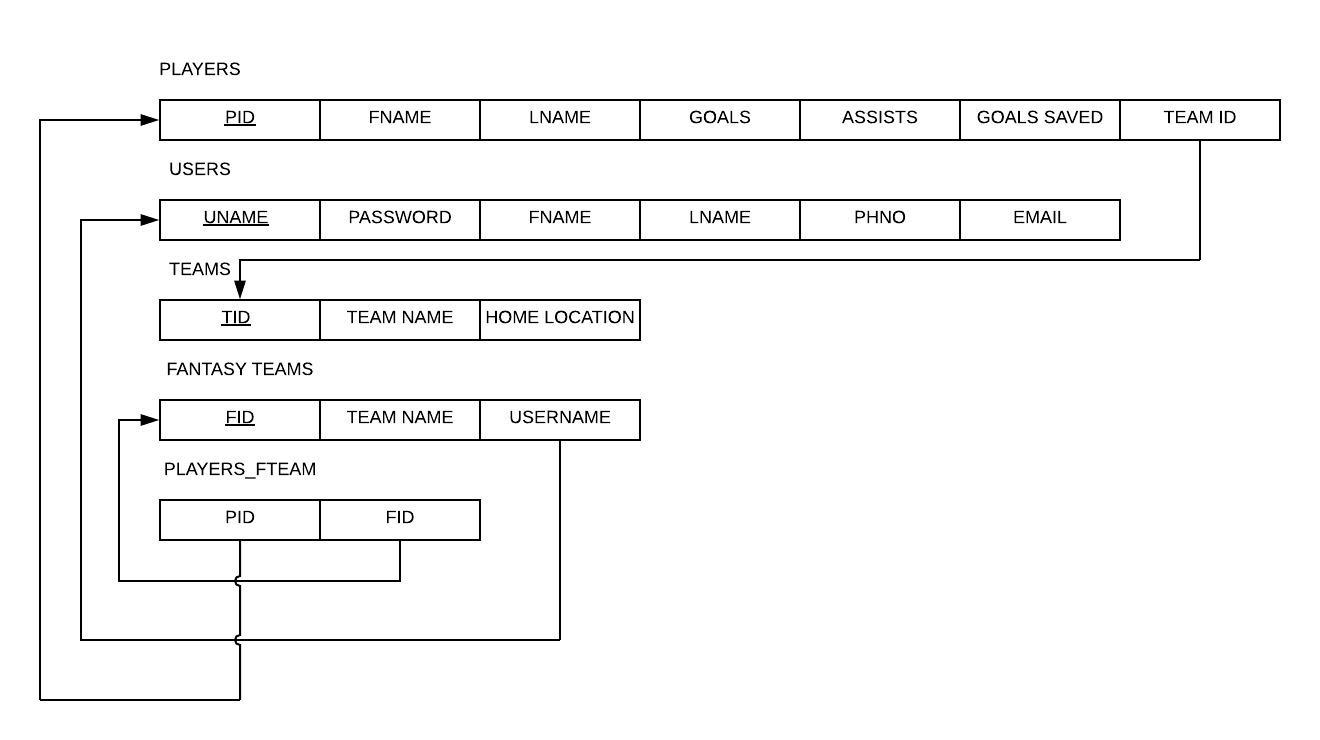
3.4a ER Diagram

**Chapter 4**

**Relational Model Concepts and Schema**

The various relational model concepts are: -

* **Attributes** – Every column present in a table and the properties which define a relation are called attributes.
* **Tables** – A combination of rows and columns that store records row wise corresponding to the attribute column.
* **Tuple** – Each record in a table is called a tuple.
* **Relation Schema** – It represents the name of the relation with attribute relations.
* **Degree** - Refers to the total number of attributes in a relation.
* **Cardinality** – Total number of rows present in a table.
* **Column** – The column represents the set of values for a specific attribute
* **Relation instance** – A finite set of tuples in the RDBMS system that are never duplicate.
* **Attribute domain** – A pre-define scope for an attribute is known as the attribute domain.

****

4.0a Relational Schema

**Chapter 5**

**SQL**

The Structured Query Language or SQL in show is a database language developed for the retrieval and management of a relational database.

SQL has five types of commands: -

* **Data Definition Language** - This refers to the commands used to define the type of data going to be used in the database.

The DDL commands are: -

* **CREATE** - Used to create tables or database.

Syntax: CREATE TABLE TABLE\_NAME (Column1 datatype, …….);

* **ALTER** - Used to modify columns in a table by changing constraints, data types or names.

Syntax: ALTER TABLE TABLE\_NAME ACTION\_TYPE ACTION;

* **RENAME** - Used to rename a table or database name.

Syntax: RENAME OLD\_TABLE TO NEW\_TABLE;

* **DROP** - Used to delete tables from databases.

Syntax: DROP TABLE TABLE\_NAME;

* **TRUNCATE** – Used to delete all the values of a table and retain the table structure.

Syntax: TRUNCATE TABLE TABLE\_NAME:

* **Data Manipulation Language** – This refers to the commands used to keep in check the data present in a database.

The DML commands are: -

* **INSERT** – Used to insert new rows into the table.

Syntax: INSERT INTO TABLE\_NAME VALUES(VALUE 1, …….);

* **DELETE** – Used to delete a row or an entire table.

Syntax: DELETE FROM TABLE\_NAME WHERE CONDITION;

* **UPDATE** - Used to update values of existing rows of tables.

Syntax: UPDATE TABLE\_NAME SET WHERE CONDITION;

* **Data Control Language** - This refers to the commands that allow database admins to give permissions to different users.

The DCL commands are: -

* **GRANT** – Grant is used to provide database access to users.

Syntax: GRANT USER USERNAME ALL ON DATABASE. \* IDENTIFIED BY ‘PASSWORD’;

* **REVOKE** – Used to take back access privileges from a user.

Syntax: REVOKE ALL PRIVILEGES, GRANT OPTION FROM USER;

* **Data Query Language** – There is only one DQL command that is used to retrieve data from a database.
* **SELECT**- Used to display data present in a table.

Syntax: SELECT \* FROM TABLE\_NAME;

* **Transaction Control Language** – Used for issues related to transactions in any database.

The TCL commands are: -

* **ROLLBACK**-Used to undo changes made in a database.

Syntax: ROLLBACK;

* **COMMIT**- Used to save and confirm all changes made in a database.

Syntax: COMMIT;

**Chapter 6**

**Python Features**

**6.1 Easy to Code and Read**

Python as a language is very simple to learn and is not very involved as compared to other popular languages like JAVA, C and C++ hence it is a programmer friendly language.

Python is easy to read due to the compulsory indentation present in the language and most of the syntax is like regular English.

**6.2 Expressive**

Python is an expressive language as it is very solution oriented rather than being

syntax oriented, the local transformations required for certain solutions are provided on python.

Other languages are more syntax oriented and solving problems is harder as there are not any functions present to help solve specific problems.

**6.3 Portable**

Python code is completely portable that is when code is written in python on a particular platform, it can be used across all platforms for example code written in Mac can be used in Windows as well.

However, using system-dependant software particular to a platform may not allow python code to run on any platform.

**6.4 High – Level**

Python is a high-level programming language that does not require the programmer to remember system architecture or do any memory management, all of that happens automatically.

This makes it the one of the most programmer friendly languages out in the world.

**6.5 Large Standard Library**

Python comes with a large standard library that you can use, this helps programmers avoiding to write large lines of code for single function.

There are libraries available on python for regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, image manipulation and email.

Some of the famous standard libraries present on python are Tkinter, PyQt5, PIL,

Mattplotlib, pandas and Numpy.

**Chapter 7**

**Tkinter Widgets**

**7.1 Button Widget**

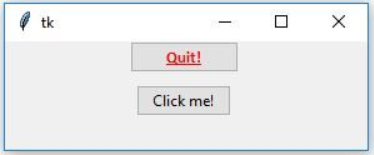
The button widget is used to add buttons to the tkinter window and on clicking the

button a certain operation is done.

Syntax: Button (window, option = value , .…);

Some of the formatting that can be done on button widget are: -

* **activebackground** – Background color when the button is under the cursor.
* **bd** – Border width in pixels and is 2 by default.
* **bg** – Used to set the background color of the button.
* **command**- Used to specify the function to be done on clicking the button.



7.1a Button Widget

**7.2 Label Widget**

The Label widget is used to display text on the tkinter window.

Syntax: Label (window, option, ….)

Some of the formatting that can be done to the label widget are: -

* **anchor** – This option controls where the text in positioned if the widge has more space than the text needs
* **bg** – Used to change the background color of a label.
* **bitmap** – Used to display a graphic in the label equivalent to a picture.



7.2a Label Widget

**7.3 Entry Widget**

The entry widget in tkinter is used to allow users to enter data into a box and then

retrieve the data using functions given.

Syntax: Entry (window, option, ….);

Some of the formatting that can be done in a entry widget are: -

* **bg** – Used to set the background color displayed behind the label.
* **font** – The font given to the text being entered.
* **fg**- The color used to render the font.



7.3a Entry Widget

**7.4 MessageBox Widget**

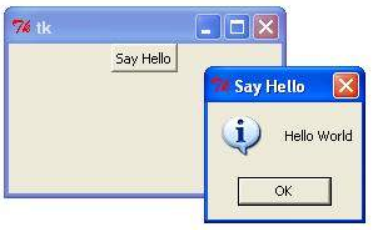
The MessageBox widget is used to display info, errors or warnings based on the

actions performed by a user.

Syntax: MessageBox.FunctionName(“Title”, ”Message”, options)

Some of the functions that can used with a MessageBox are: -

* **showinfo()** – Display some information.
* **showwarning()** – Display a warning.
* **showerror()** – Display an error.



7.4a MessageBox Widget

**7.5 Frame Widget**

The Frame widget creates an invisible space in the window used to hold other widgets in place.

Some of the formatting that can be done to a frame are: -

* **bg** – Used to set the background colour of a frame.
* **cursor** – Changes cursor type when hovering over the frame.
* **height** – The vertical dimension of the new frame is set. 

7.5a Entry Widget

**Chapter 8**

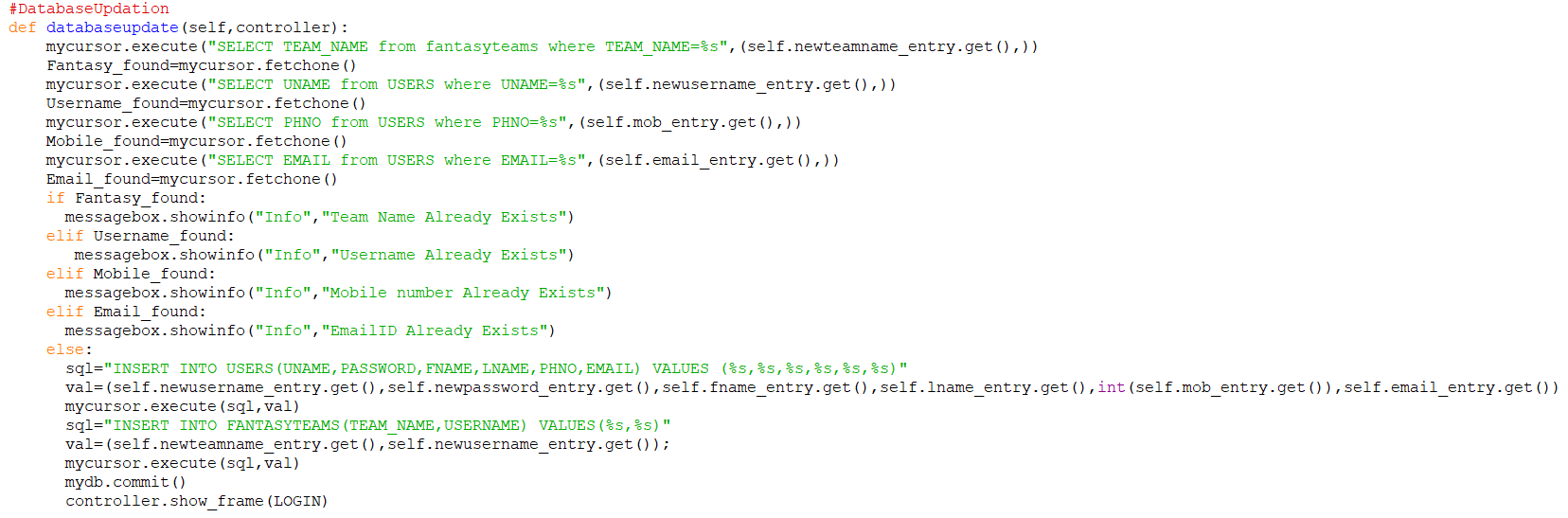
**Implementation**

* **Window Class** – Used as the parent class and window for all the frames or pages present in the application.



* **Signup Page** – Every user has to sign up to login into the application, the sign up page takes care of this and takes all the details required for the user profile.



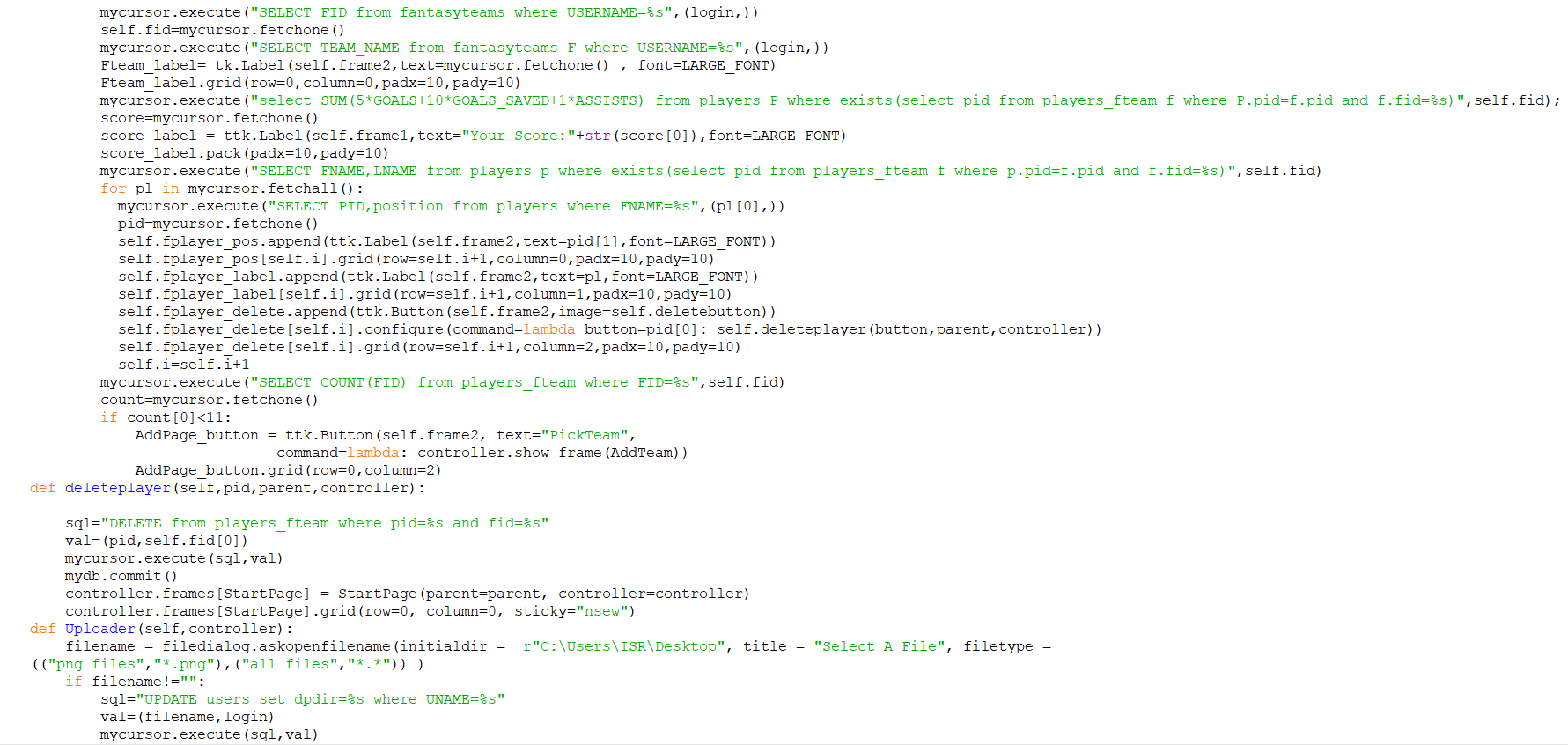


* **Login page** – This is the page where users can login to access the application.



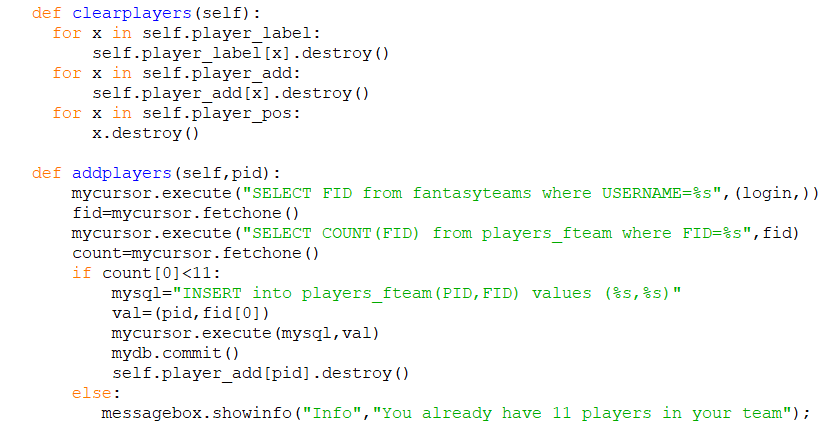
* **Homepage** – The part of the application where the user details along with the fantasy team is present.



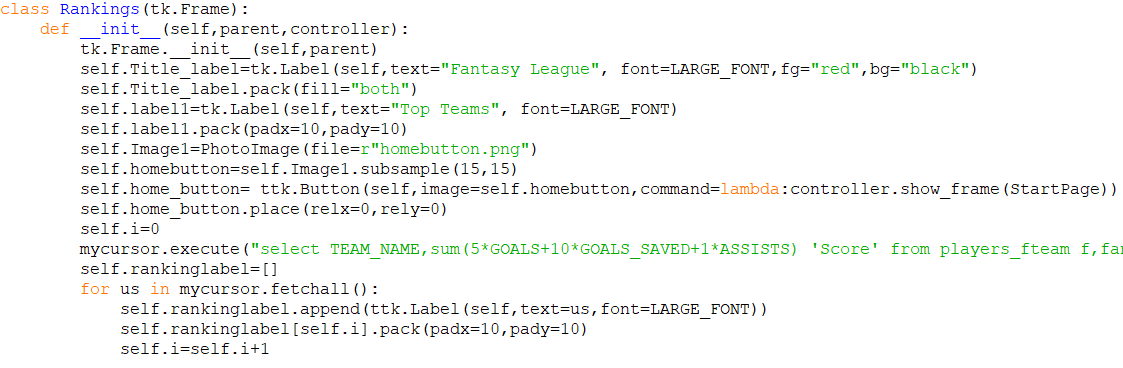


* **Team page** – Displays the teams from which users can pick football players for their team.





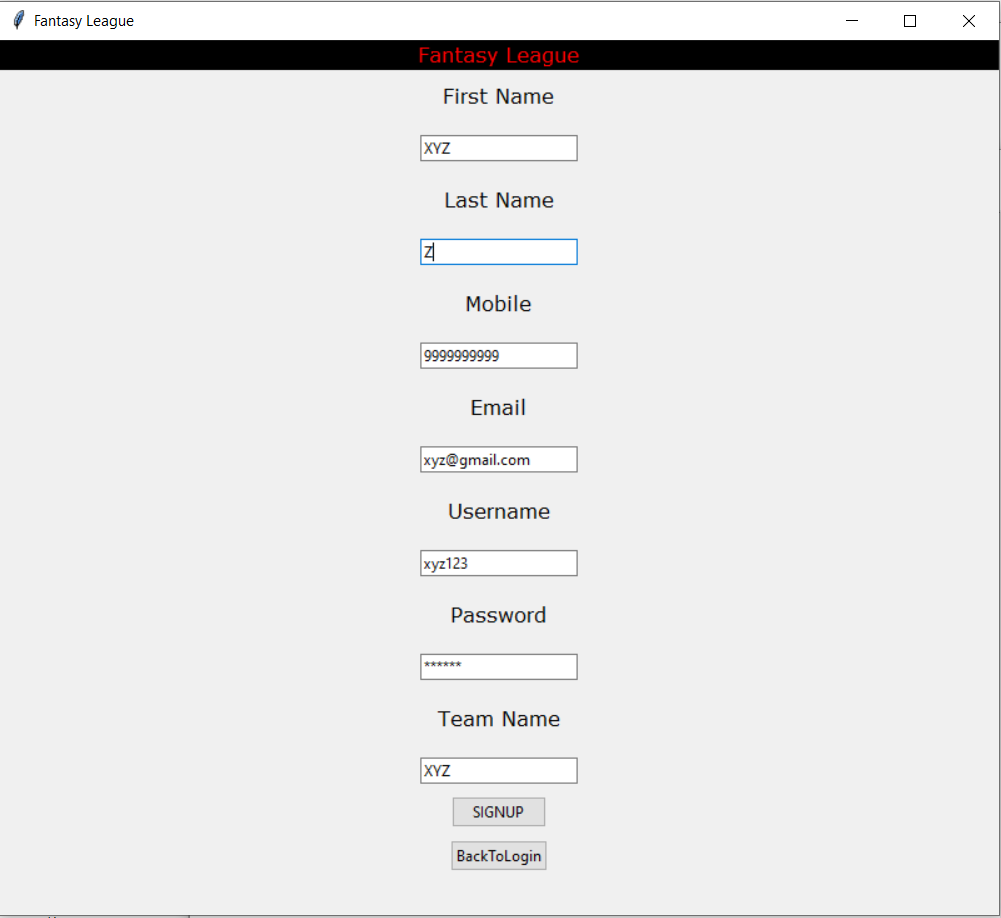
* **Rankings** – The page in the application that shows user rankings in order of the points owned by their respective fantasy teams.



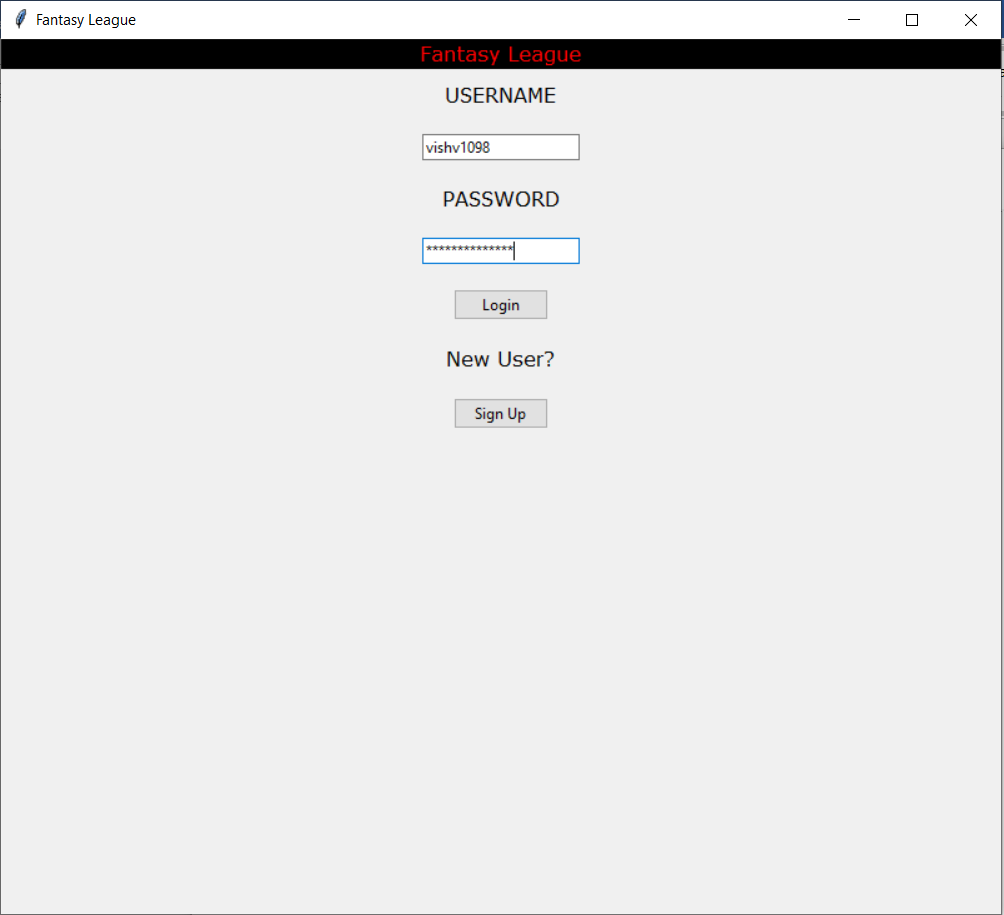
**Chapter 9**

**Output Snapshots**

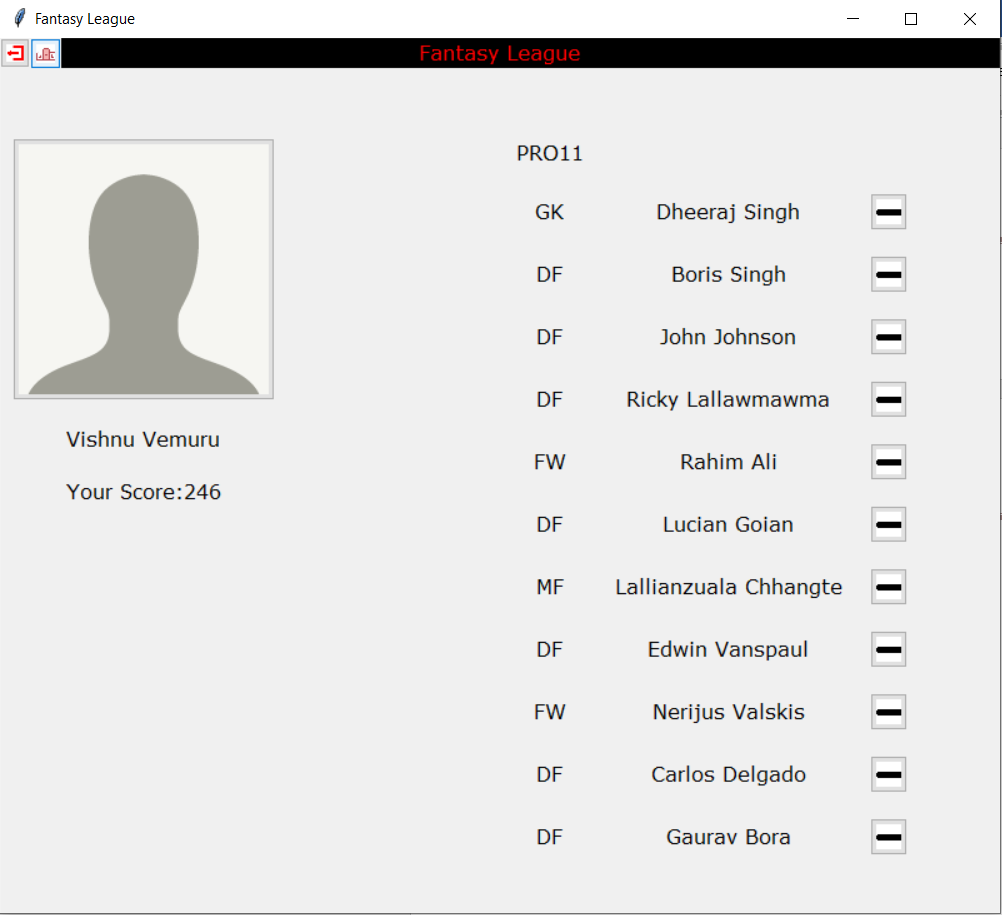
**Signup**



**Login**



**Homepage**



**Ranking**

