**Final Project Synopsis**

On

Fantasy Cricket Game

**Submitted by**

Himanshu Rajput

College : GH Patel College of Engineering and Technology

Email Id: [rajputhimanshu126@gmail.com](mailto:rajputhimanshu126@gmail.com)

Contact No: 7016398281

Note: - For Running the Standalone app, main.exe application should be run.

**Content of Synopsis**

1. **Title**

Fantasy Cricket Game is a Python Graphical User Interface (GUI) application. This application lets the user to create their own team and Evaluate their performance. For Creating Fantasy Cricket Game GUI, I am using PyQt5 library. For Database, I am using SQlite3.

1. **Introduction**

**PyQt** is a Python binding of the cross-platform GUI toolkit Qt, implemented as a Python plug-in. PyQt is free software developed by the British firm Riverbank Computing. It is available under similar terms to Qt versions older than 4.5; this means a variety of licenses including GNU General Public License (GPL) and commercial license, but not the GNU Lesser General Public License (LGPL).[]](https://en.wikipedia.org/wiki/PyQt#cite_note-3) PyQt supports Microsoft Windows as well as various flavours of UNIX, including Linux and MacOS (or Darwin).

PyQt implements around 440 classes and over 6,000 functions and methods including:

* a substantial set of GUI widget
* classes for accessing SQL databases ODBC, MySQL, PostgreSQL, Oracle, SQLite
* QScintilla, Scintilla-based rich text editor widget
* data aware widgets that are automatically populated from a database
* an XML parser
* SVG support
* classes for embedding ActiveX controls on Windows (only in commercial version)

For Database, I am using SQLite.SQLite is a relational database management system (RDBMS) contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

1. **Objectives:**

Fantasy Cricket Game consists of a UI with radio, list and label. On toggling of radio, the left side input list is inflated with player name same as the category of radio. On toggling of radio button name of the player is inflated into list from the local database. End user have 4 different option

* + New Team
  + Open Team
  + Save Team
  + Evaluate Team

When the end user clicks on the new team action from the menu, user can now toggle the radio button previously if the user tries to toggle the radio button user will get a Warning Message. In the Main UI there are top label which shows how many Batsman, how many Bowler, how many All Rounder and how many Wickets Keeper are selected.

There is label for Point available and point used for displaying how many points are available for the user for making a new team selection and point used for displaying the how much point is consumed.

If the user double clicks on the input list the player will be moved to output list.

During Selection point to remember is you can select only one Wicket Keeper and for creating a perfect team you have to select 11 players. You can not take more than 11 players or less than 11 players in a team. Also, you can select the player until you have point Available to select the player.

After Creating team user can save the team by Save Team Action from the Menu

You can also open a created team from the database by clicking on Open Team Action from the menu.

You can evaluate the score of the team by clicking the evaluate Action from the menu. On clicking Evaluate Team a new window is opened where you can select the team you want to evaluate and the match on which you are evaluating. After selecting team and match you need to click on Evaluate button and their point will be displayed on the output list.

1. **Implementation:**

For Implementation of the Main UI, I had created

* IntershallaCricket is the Main package containing the
  + *Design* package containing the design.py and \_\_\_init\_\_.py
  + Database package containing the databaseFile.py and \_\_init\_\_.py
  + Evaluate package containing the evaluate.py and \_\_init\_\_.py
  + EvaluateWindow package containing the evaluateWindow.py and \_\_init\_\_.py
  + EvaluationPackage contains the EvaluationLogic.py and \_\_init\_\_.py
  + Image Package containing the image for the main UI and the \_\_init\_\_.py
  + OpenTeam package contains the openWidnow.py and the \_\_init\_\_.py
  + PreProcessing package contains the Handling.py and the \_\_init\_\_.py
  + \_\_init\_\_.py file
  + main.py file\
* setup.py

This the Tree Hierarchy of my project.

**design.py** contains the code for the main Window UI inflation.

**databaseFile**.py contains the code for the database file creation.

**evaluate.py** contains the code for how the evaluation action should work.

**evaluateWindow.py** contains the code for how the evaluate window should look like.

**evaluationLogic.py** contains the code for how the player evaluation should be done (point for every stats)

**openWindow.py** contains the code for how the new window opened by clicking on the open Team action clicked should look like.

**Handling.py** contains the code for how the UI should work like

* + radio button toggled
  + button clicked
  + list widget double clicked
  + action triggering

Every event is handled by the Handled.py. GUI is an event driven programming paradigm.

For this many packages to work collaboratively, I had created **\_\_init\_\_.py** inside every package which tells the python interpreter that the current folder is a package which contains modules. Parent folder of the package, I had created **setup.py** for configuring how the packages should work.

After the package is created, I had created a **exe file** from the main script. By doing so I make sure that all package combines and make a single executable file.

This executable file can be run on any environment independent of

* + whether the device has python installed or not

For Creating the exe file, I had used **pyinstaller.** Output of pyinstaller is the *dist folder* which contains the exe file.

1. **Software Development Technique:**

For Development of software, I had used **Incremental Model**. Starts from subpart of the project and the developing different subpart as I progress.

Incremental Model is a process of software development where requirements are broken down into multiple standalone modules of software development cycle. Incremental development is done in steps from analysis design, implementation, testing/verification, maintenance

For Testing of the software, I had used **UNIT TESTING** is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software

Unit Testing Tasks

* Unit Test Plan
  + Prepare
  + Review
  + Rework
  + Baseline
* Unit Test Cases/Scripts
  + Prepare
  + Review
  + Rework
  + Baseline
* Unit Test
  + Perform

**INTEGRATION TESTING** is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

References;

-Wikipedia

- Intershalla

-Python Documentation

-PyQt5 documentation

-Sqlite3 documentation

-zetcode