

GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY

Summer Training Report



Fantasy Cricket League Using Python

SUBMITTED TO:

Dr. Sartaj Singh Sodhi
USICT, GGSIPU

SUBMITTED BY:

Vikrant Singh Jamwal
B. Tech IT (7th Sem)
Roll No: 03116401517

Acknowledgement

It is my proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this report. It would not have been possible to prepare this report in this form without their valuable help, cooperation and guidance.

First and foremost, I wish to record our sincere gratitude to Internshala Coordinators for their constant support and encouragement in preparation of this report and for making available videos and interface facilities needed to prepare this report. The seminar on “Python” was very helpful to us in giving the necessary background information and inspiration in choosing this topic for the seminar. Their contributions and technical support in preparing this report are greatly acknowledged.

I hereby declare that I have completed my six weeks’ summer training at INTERNSHALA from 12/06/2020 to 24/07/2020 under the guidance of INTERNSHALA. I hereby undertake that the project undertaken by me is the genuine work of mine.

(Signature of student)

Name: Vikrant Singh Jamwal

Roll number: 03116401517

Certificate of Training



Contents of the Report

- All about the Training
- Introduction to Python
- GUI with python
- Database connection
- Project: A fantasy cricket league
- Approach
- Creating Database
- Creating GUI
- Creating Python Files
- Codes and Screenshots
- Bibliography

All about the Training

The Training is divided into Four Parts:

1. Programming with python

Here, I learned about the features of the python language. Similarities and differences between Python and other Languages like Java and C++. Data variables, types, loops, conditions, features of OOPS and syntax of the python language were also taught.

2. GUI with Python language

Learned to connect the user interface platform with the Python codes for creating applications.

PyQt5 was used to create the UI for the applications. With easy Drag and Drop options and widgets it helps the developer to create the front end more efficiently.

3. Database

Connected Database with Python codes. First creating the database, adding the elements using SQLite3 and then connecting with the Python codes so that data can be easily stored and read.

4. Project work: Fantasy Cricket League

To create a fantasy cricket league where database is used to access the data and team was made. Then according to the given criteria Points are calculated using Python platform and the frontend of the application is created using PyQt5 designer.

Introduction to Python

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming (OOP) as well as procedural oriented programming (POP).

There are many features in Python, some of which are:

1. Easy to code:

Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, Java, etc. It is also a developer-friendly language.

2. Free and Open Source:

Since it is open-source, this means that source code is also available to the public. So you can download it as, use it as well as share it.

3. Object-Oriented Language:

One of the key features of python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, objects encapsulation, etc.

4. Interpreted Language:

Python is an Interpreted Language because Python code is executed line by line at a time. like other languages C, C++, Java, etc. there is no need to compile python code this makes it easier to debug our code. The source code of python is converted into an immediate form called **bytecode**.

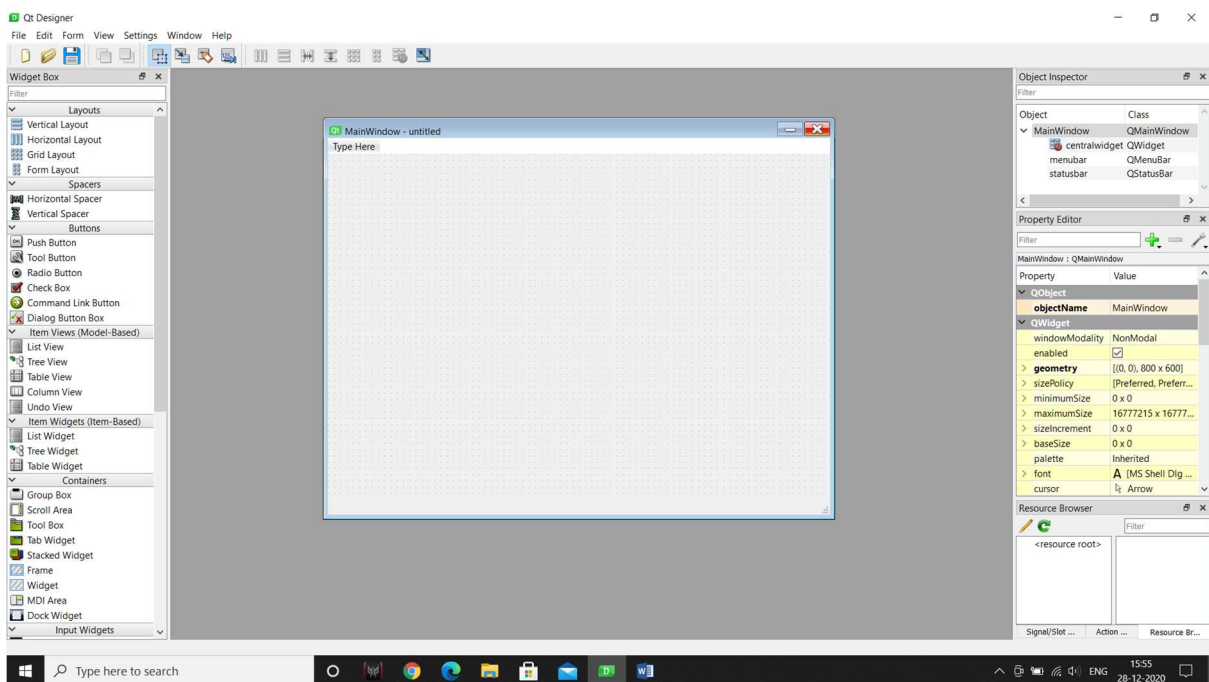
5. Large Standard Library

Python has a large standard library which provides a rich set of module and functions so you do not have to write your own code for every single thing. There are many libraries present in python for such as regular expressions, unit-testing, web browsers, etc.

GUI with Python

PyQt is a GUI widgets toolkit. It is a Python interface for **Qt**, one of the most powerful, and popular cross-platform GUI library. PyQt is a blend of Python programming language and the Qt library. This introductory tutorial will assist you in creating graphical applications with the help of PyQt.

We have used the 5th version of the PyQt to create our project. Some classes like **QDialog** and **QFrame** are derived from QWidget class and are used in the project. They have their own sub-class system.



PyQt5 designer for User interface

Database Connection

SQLite is an in-process library that implements a self-contained, server less, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.

SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

SQLite3 can be integrated with Python using sqlite3 module, which was written by Gerhard Haring. It provides an SQL interface compliant with the DB-API 2.0 specification. You do not need to install this module separately because it is shipped by default along with Python version 2.5.x onwards.

To use sqlite3 module, you must first create a connection object that represents the database and then optionally you can create a cursor object, which will help you in executing all the SQL statements. Extension is (.db).

Connect- Used to connect the database to the python file. Importing the libraries of SQLite3 and data in the Database.

Curser- Used as an object of the SQLite to execute its methods in the python file using simple SQL statements like SELECT, DELETE, WHERE, etc.

Project Work: Fantasy Cricket League

Problem Statement

It is an online game where you create a virtual team of real cricket players and score points depending on how your chosen players perform in real life matches. To win a tournament, you must try and get the maximum points and the No. 1 rank amongst other participants. Here's how a Fantasy Cricket game may look like.

Sample of Rules Batting

- 1 point for 2 runs scored
- Additional 5 points for half century
- Additional 10 points for century
- 2 points for strike rate (runs/balls faced) of 80-100
- Additional 4 points for strike rate > 100
- 1 point for hitting a boundary (four) and 2 points for over boundary (six)
- Bowling
- 10 points for each wicket
- Additional 5 points for three wickets per innings
- Additional 10 points for 5 wickets or more in innings
- 4 points for economy rate (runs given per over) between 3.5 and 4.5

- 7 points for economy rate between 2 and 3.5
- 10 points for economy rate less than 2 Fielding
- 10 points each for catch/stumping/run out

For the database, you are required to use three tables – match, stats and teams.

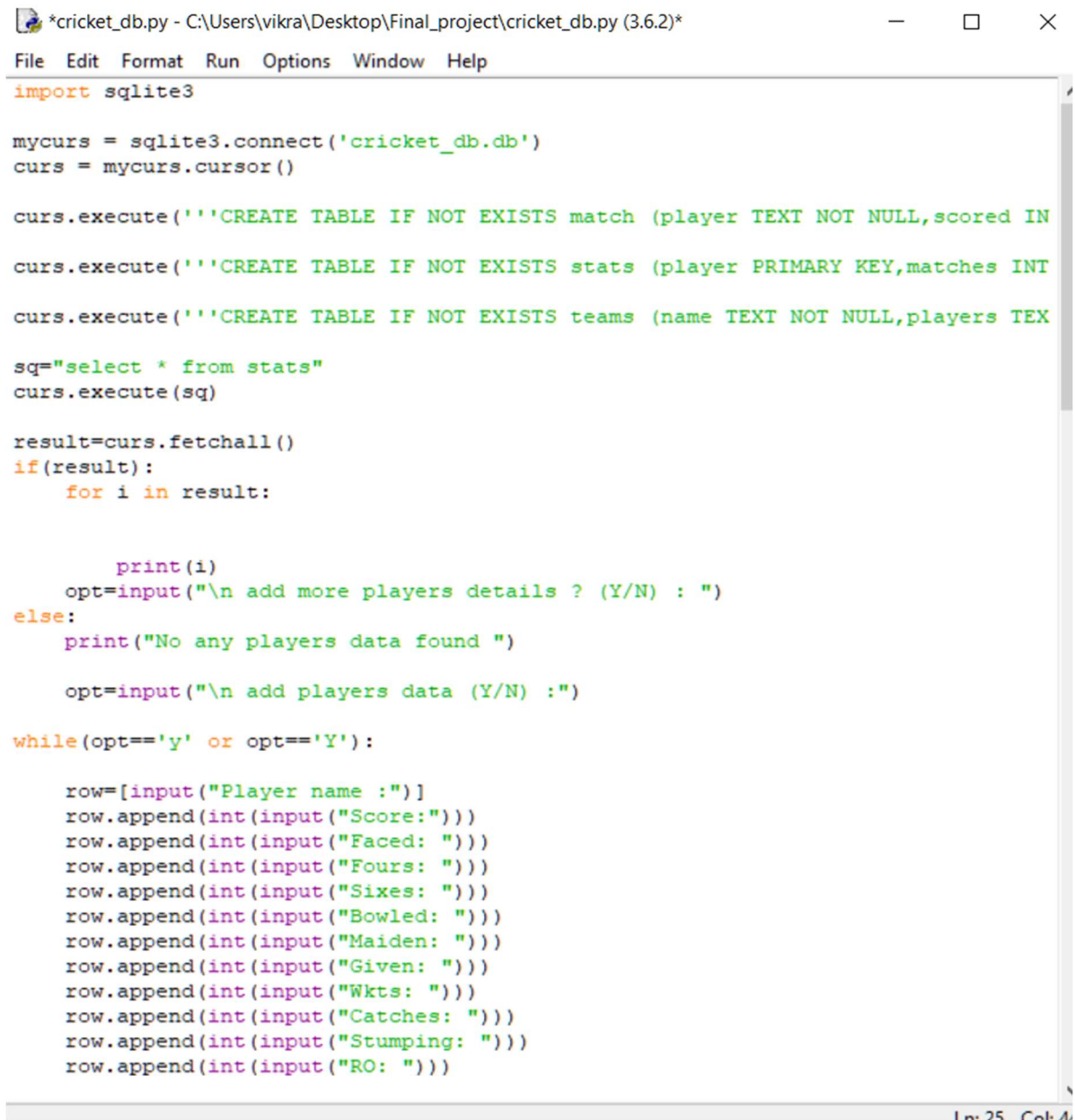
At the end, application should provide an option for evaluating the total points according to the rules and displaying the total sum of all the points acquired by each player.

Approach

1. First creating the database and the three tables required.
2. Then creating the User interface using PyQt5 designer and then converting those files to the Python format.
3. Then writing some methods to connect different windows together to make a working multipage application.

Progress

1. Creating a database



```
*cricket_db.py - C:\Users\vikra\Desktop\Final_project\cricket_db.py (3.6.2)*
File Edit Format Run Options Window Help
import sqlite3

mycurs = sqlite3.connect('cricket_db.db')
curs = mycurs.cursor()

curs.execute('''CREATE TABLE IF NOT EXISTS match (player TEXT NOT NULL,scored IN
curs.execute('''CREATE TABLE IF NOT EXISTS stats (player PRIMARY KEY,matches INT
curs.execute('''CREATE TABLE IF NOT EXISTS teams (name TEXT NOT NULL,players TEX

sq="select * from stats"
curs.execute(sq)

result=curs.fetchall()
if(result):
    for i in result:

        print(i)
    opt=input("\n add more players details ? (Y/N) : ")
else:
    print("No any players data found ")

    opt=input("\n add players data (Y/N) :")

while(opt=='y' or opt=='Y'):

    row=[input("Player name :")]
    row.append(int(input("Score:")))
    row.append(int(input("Faced: ")))
    row.append(int(input("Fours: ")))
    row.append(int(input("Sixes: ")))
    row.append(int(input("Bowled: ")))
    row.append(int(input("Maiden: ")))
    row.append(int(input("Given: ")))
    row.append(int(input("Wkts: ")))
    row.append(int(input("Catches: ")))
    row.append(int(input("Stumping: ")))
    row.append(int(input("RO: ")))
```

Connected the python file with the SQLite, then match table is created.

```
*cricket_db.py - C:\Users\vikra\Desktop\Final_project\cricket_db.py (3.6.2)*
File Edit Format Run Options Window Help

row.append(int(input("Bowled: ")))
row.append(int(input("Maiden: ")))
row.append(int(input("Given: ")))
row.append(int(input("Wkts: ")))
row.append(int(input("Catches: ")))
row.append(int(input("Stumping: ")))
row.append(int(input("RO: ")))

try:
    curs.execute("INSERT INTO match (player,scored, faced, fours,sixes,bowled,maiden,given,wkts,catches,stumping,ro) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)",
        (row[0],row[1], row[2], row[3],row[4],row[5],row[6],row[7],row[8],row[9],row[10],row[11]))
    mycurs.commit()

    print("records added successfully match table.")
except: # except block to handle exceptions
    print("Error in operation.")
    mycurs.rollback()

#ADDING DATA TO STATS TABLE FROM USER
print("player information for State table ")
row.append(int(input("Total matches: ")))
row.append(int(input("Total runs: ")))
row.append(int(input("100s: ")))
row.append(int(input("50s: ")))
row.append(int(input("Value: ")))
row.append(input("Category as (BAT,BWL,AR,WK): "))

try:
    curs.execute("INSERT INTO stats (player,matches,runs, hundreds, fifties,value,ctg) VALUES (?, ?, ?, ?, ?, ?)", #adding data to database
        (row[0],row[12], row[13], row[14],row[15],row[16],row[17]))
    mycurs.commit()

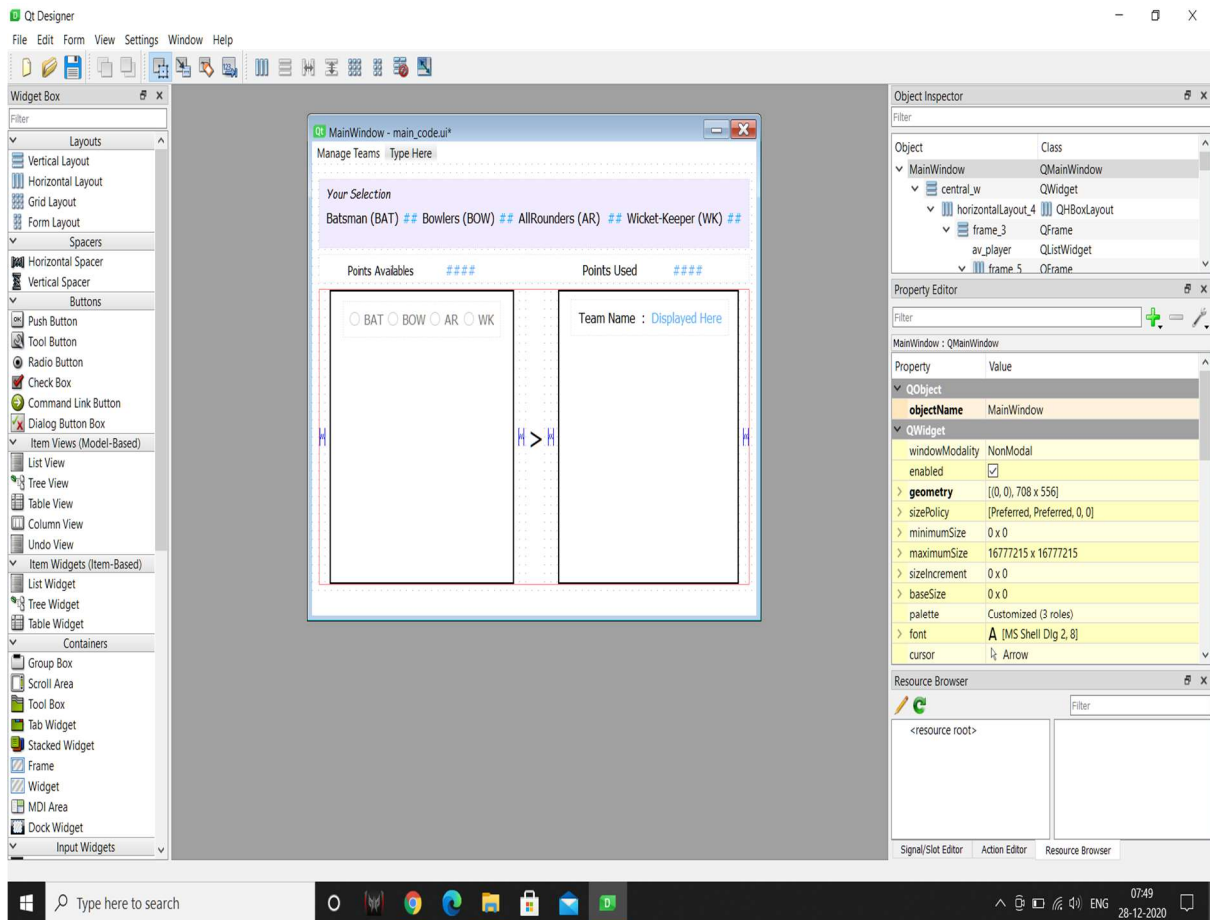
    print("records added successfully for stats table.")
except:
    print("Error in operation.")
    mycurs.rollback()
```

Ln 25 Col 4

Here, Stat table is connected and then the updating feature for the database is provided by asking for the values of a new player to be added by the user in near future.

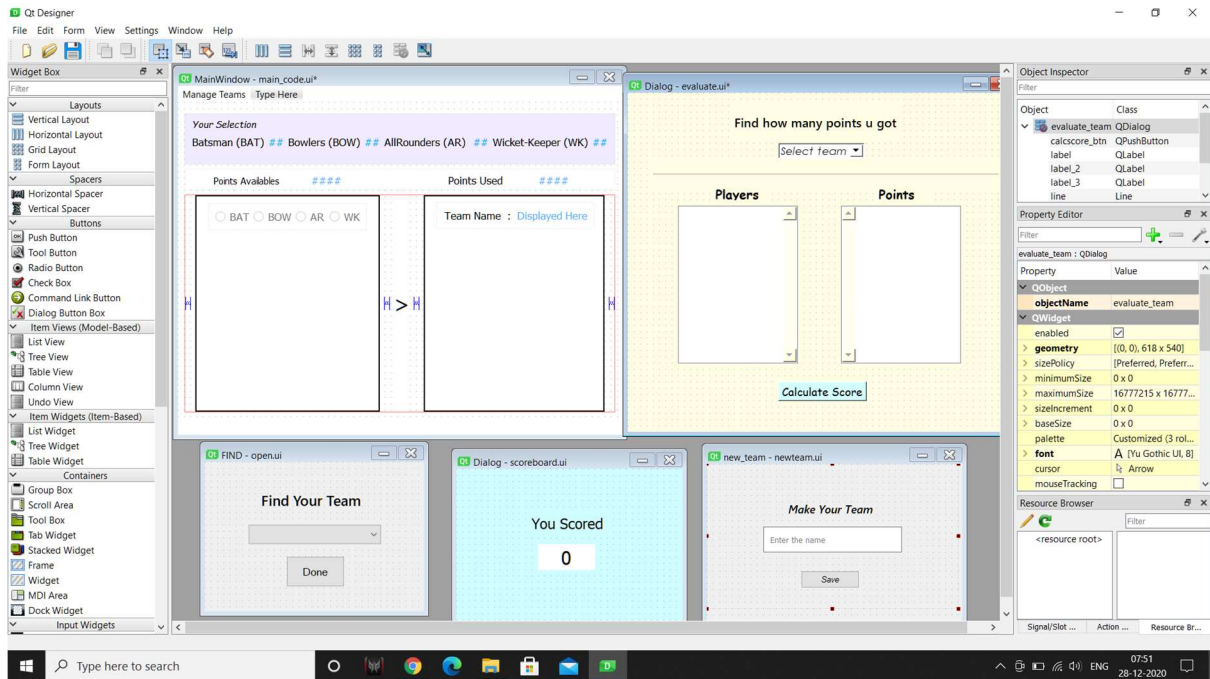
This is the completion of the database creation.

2. Creating GUI for the application using PyQt5.



Using Text widgets, radio buttons, Frames, List widgets and some other widgets Projects main window is ready.

For Making New Team, Opening Previously made teams, evaluating Score and displaying Score, four more windows are made.



These are the complete 5 windows which is required to create the application.

First, main window of the application- presents the Menu bar for the user to perform an action.

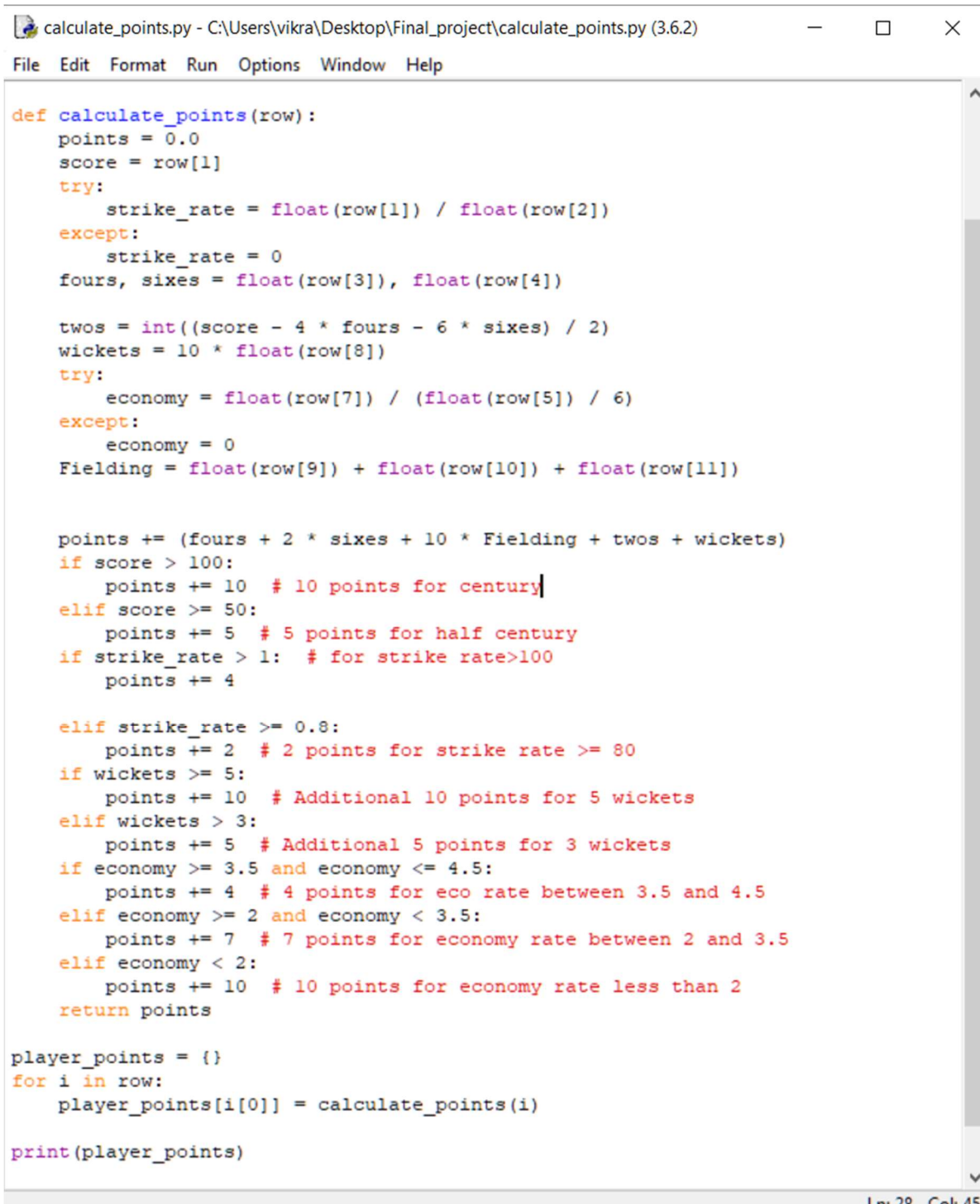
Second, New team window- for making new team by giving the name of the team and selecting the players to be kept in that team considering the given conditions.

Third, Open window- to open the Already made team for updating or reviewing.

Fourth, Evaluation window- provides with 2 frames to display the team players and the points scored by them, containing a button for scorecard.

Fifth, Scorecard window- for displaying the total points acquired by the Team.

3. Generating Python codes and files.



```
def calculate_points(row):
    points = 0.0
    score = row[1]
    try:
        strike_rate = float(row[1]) / float(row[2])
    except:
        strike_rate = 0
    fours, sixes = float(row[3]), float(row[4])

    twos = int((score - 4 * fours - 6 * sixes) / 2)
    wickets = 10 * float(row[8])
    try:
        economy = float(row[7]) / (float(row[5]) / 6)
    except:
        economy = 0
    Fielding = float(row[9]) + float(row[10]) + float(row[11])

    points += (fours + 2 * sixes + 10 * Fielding + twos + wickets)
    if score > 100:
        points += 10 # 10 points for century
    elif score >= 50:
        points += 5 # 5 points for half century
    if strike_rate > 1: # for strike rate>100
        points += 4

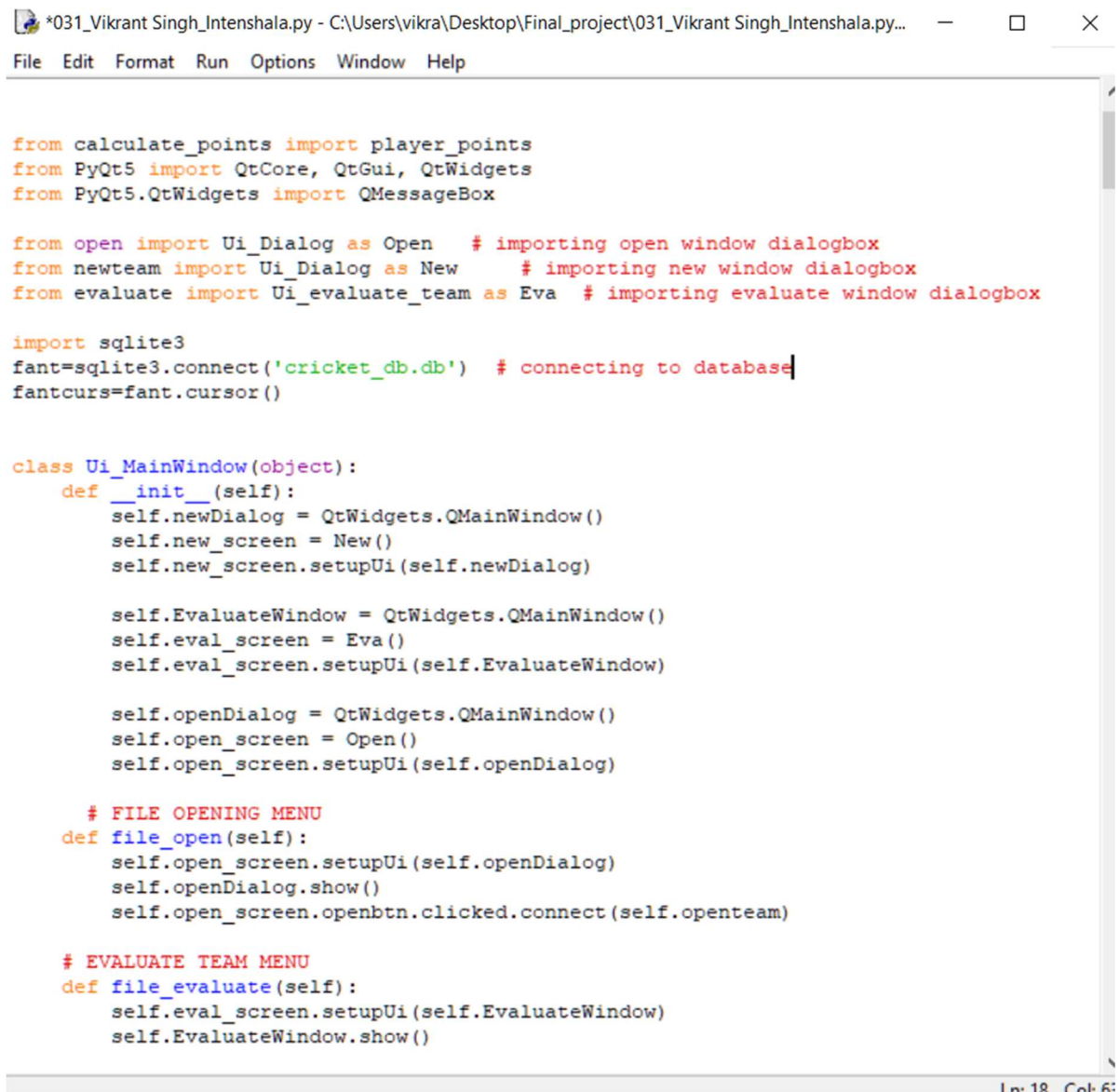
    elif strike_rate >= 0.8:
        points += 2 # 2 points for strike rate >= 80
    if wickets >= 5:
        points += 10 # Additional 10 points for 5 wickets
    elif wickets > 3:
        points += 5 # Additional 5 points for 3 wickets
    if economy >= 3.5 and economy <= 4.5:
        points += 4 # 4 points for eco rate between 3.5 and 4.5
    elif economy >= 2 and economy < 3.5:
        points += 7 # 7 points for economy rate between 2 and 3.5
    elif economy < 2:
        points += 10 # 10 points for economy rate less than 2
    return points

player_points = {}
for i in row:
    player_points[i[0]] = calculate_points(i)

print(player_points)
```

Ln 28 Col 45

Python File created to calculate the Points for each individual player according to the rules given in the Problem Statement.



```
*031_Vikrant Singh_Intenshala.py - C:\Users\vikra\Desktop\Final_project\031_Vikrant Singh_Intenshala.py...
File Edit Format Run Options Window Help

from calculate_points import player_points
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import QMessageBox

from open import Ui_Dialog as Open # importing open window dialogbox
from newteam import Ui_Dialog as New # importing new window dialogbox
from evaluate import Ui_evaluate_team as Eva # importing evaluate window dialogbox

import sqlite3
fant=sqlite3.connect('cricket_db.db') # connecting to database
fantcurs=fant.cursor()

class Ui_MainWindow(object):
    def __init__(self):
        self.newDialog = QtWidgets.QMainWindow()
        self.new_screen = New()
        self.new_screen.setupUi(self.newDialog)

        self.EvaluateWindow = QtWidgets.QMainWindow()
        self.eval_screen = Eva()
        self.eval_screen.setupUi(self.EvaluateWindow)

        self.openDialog = QtWidgets.QMainWindow()
        self.open_screen = Open()
        self.open_screen.setupUi(self.openDialog)

        # FILE OPENING MENU
    def file_open(self):
        self.open_screen.setupUi(self.openDialog)
        self.openDialog.show()
        self.open_screen.openbtn.clicked.connect(self.openteam)

        # EVALUATE TEAM MENU
    def file_evaluate(self):
        self.eval_screen.setupUi(self.EvaluateWindow)
        self.EvaluateWindow.show()
```

Line 18, Col 65

This Python file the final project file where all the other windows python files are imported to connect each other using buttons clicks. Dialog boxes, error messages and each window is defined individually to create a working application.

Project: FANTASY CRICKET LEAGUE is completed.

Bibliography

1. https://www.tutorialspoint.com/sqlite/sqlite_quick_guide.htm
2. <https://www.geeksforgeeks.org/python-features>
3. <https://wiki.python.org/moin/PyQt>
4. <https://trainings.internshala.com/python-training>
5. <https://www.python.org>
6. <https://www.youtube.com>