A

Report on

Fantasy Cricket Game

Submitted in partial fulfillment of the requirement for the award of the degree

Bachelor of Engineering
in
COMPUTER SCIENCE AND ENGINEERING
to
Chhattisgarh Swami Vivekananda Technical University, Bhilai

Session: 2020-2021

Submitted By:
Abdul Rahman Rathore
5th / CSE
Roll No.: 300302219002

Chhatrapati Shivaji Institute of Technology
Department of Computer Science and Engineering

.

DECLARATION

I hereby declare that I have completed my six weeks summer training at **INTERNSHALA** from **12/07/2021 to 30/08/2021** under the guidance of **INTERNSHALA**. I hereby undertake that the project undertaken by me is the genuine work of mine.

Date: 30/12/2021

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.

Last but not the least, we wish to thank our parents for financing our studies in this college as well as for constantly encouraging us to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged.

SUMMER TRAINING CERTIFICATE

INTERNSHALA TRAININGS

Certificate of Training

Abdul Rahman Rathore

from Chhatrapati Shivaji Institute Of Technology, Shivaji Nagar, Durg, Chhattisgarh, has successfully completed a six weeks online training on **Programming with Python**. The training consisted of Introduction to Python, Using Variables in Python, Basics of Programming in Python, Principles of Object-oriented Programming (OOP), Connecting to SQLite Database, Developing a GUI with PyQT, Application of Python in Various Disciplines and The Final Project modules.

In the final assessment, Abdul Rahman scored 80% marks.

We wish Abdul Rahman all the best for the future endeavours.

Sarvesh Agarwal

FOUNDER & CEO, INTERNSHALA

Date of certification: 2021-08-28 Certificate no. : 357E3343-38D5-374D-2F44-9FCECB6D814E

For certificate authentication, please visit https://trainings.internshala.com/verify_certificate

About Internshala

Internshala is an internship and online training platform, based in Gurgaon, India. Founded by Sarvesh Agrawal, an IIT Madras alumnus, in 2011, the website helps students find internships with organisations in India.

Its a website ample of opportunities well namely internships.You can select your preference or search it to find what you want.. The main thing about that website is lot of startups providing stipend and non stipend internship offers of all fields.. Even some tier 1 colleges internships will be offered.

TABLE OF CONTENTS

INTRODUCTION TO PYTHON

Python Language Introduction

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently. Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

PYTHON FEATURES

Python's features include −

• **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

• **Easy-to-read** − Python code is more clearly defined and visible to the eyes.

• **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.

• **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

• **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

• **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

• **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

• **Databases** − Python provides interfaces to all major commercial databases. • **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

• **Scalable** − Python provides a better structure and support for large programs than shell scripting. Apart from the above-mentioned features, Python has a big list of good features, few are listed below − • It supports functional and structured programming methods as well as OOP. • It can be used as a scripting language or can be compiled to byte-code for building large applications.

• It provides very high-level dynamic data types and supports dynamic type checking. • IT supports automatic garbage collection.

• It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

TRAINING CONTENTS

1. Introduction to Python

Learn how to install Python, distinguish between important data types and use basic features of the Python interpreter, IDLE.

2. Using Variables in Python

Learn about numeric, string, sequence and dictionary data types and relevant operations while practicing Python syntax.

3. Basics of Programming in Python

Learn how to write programs using conditionals, loops, iterators and generators, functions and modules and packages.

4. Principles of Object-oriented Programming (OOP)

Learn about the important features of Object-oriented Programming while using Classes and Objects, two main aspects of the OOP paradigm.

5. Connecting to SQLite Database

Learn about relational databases while learning how to store and retrieve data from an SQLite database through Python.

6. Developing a GUI with PyQT

Learn how to install PyQt5 toolkit, Qt Designer and create a graphical user interface using common widgets and menu systems.

7. Application of Python in Various Disciplines

Learn about various resources to extend your learning for the Python programming language.

Developing a GUI with PyQT

GUI and Event Driven Programming

GUI:- Graphical user interface (GUI), a computer program that enables a person to communicate with a computer through the use of symbols, visual metaphors, and pointing devices. Best known for its implementation in Apple Inc.'s Macintosh and Microsoft Corporation's Windows operating system, the GUI has replaced the arcane and difficult textual interfaces of earlier computing with a relatively intuitive system that has made computer operation not only easier to learn but more pleasant and natural. The GUI is now the standard computer interface, and its components have themselves become unmistakable cultural artifacts.

Event driven Programming:- An event-driven program is one that largely responds to user events or other similar input. The concept of event-driven programming is an important one in application development and other kinds of programming, and has spawned the emergence of event handlers and other resources. The idea in event-driven programming is that the program is designed to react. It reacts to specific kinds of input from users, whether it's a click on a command button, a choice from a drop-down list, an entry into a text box, or other kinds of user events.

PyQT :- 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 was developed by RiverBank Computing Ltd. The latest version of PyQt can be downloaded from its official website. PyQt is a blend of Python programming language and the Qt library. PyQT API is a collection of more then 400 classes.

Major classes in PyQT :-

QObject is at the top of the class hierarchy. It is the base class of all the Qt objects.

QApplication class manages the main settings and control flow.

QWidget is the base class for all user interface objects.

QDialogue and Qframe classes are derived from the QWidget class.

Using Common Widgets

Qt Designer

Qt Designer is the Qt tool for designing and building graphical user interfaces (GUIs) with Qt Widgets. You can compose and customize your windows or dialogs in a what-you-see-is -what

you-get (WYSIWYG) manner, and test them using different styles and resolutions.

Widgets and forms created with Qt Designer integrate seamlessly with programmed code, using

Qt's signals and slots mechanism, so that you can easily assign behavior to graphical elements.

All properties set in Qt Designer can be changed dynamically within the code. Furthermore,

features like widget promotion and custom plugins allow you to use your own components with

Qt Designer.

Qt Designer helps only in designing GUI file, then it needs to be converted into py file. By using pyuic 5 utility xml file can be converted into .py file and is installed along with PyQT5 package.

We use pyuic5-x myui.ui-o myui.py command to convert ui file to python file. For the sake of simplicity, we place myuic.ui file in the same folder where pyuic5.exe file is placed.

PROFILE OF THE PROBLEM

Create a Fantasy Cricket game in Python. The game should have all the features displayed in the mock-up screens in the scenario. To calculate the points for each player, we can use rules similar to the sample rules displayed below.

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

DATABASE DESIGN

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

Match1

	Scored		Four s		Bowled	Maide n			Catches	Stump	

***Run Out**

Team

Name	Players

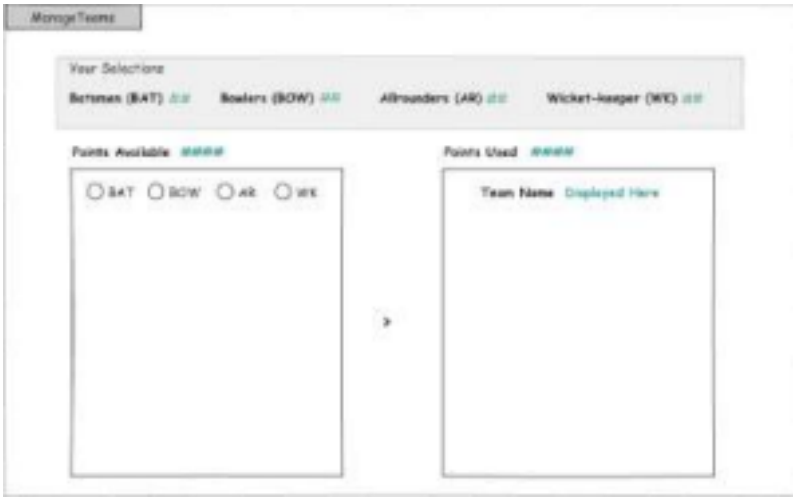
Stats

player	matches	runs	100s	50s	value	ctg

The data to enter in the remaining two tables is given below:

player	scored	faced	fours	sixes	bowled	maiden	given	wkts	catches	stumping	ro	value	matches	runs	100s	50s	ctg
Kohli	102	98	8	2	0	0	0	0	0	0	1	120	189	8257	28	43	BAT
Yuvraj	12	20	1	0	48	0	36	1	0	0	0	100	86	3589	10	21	BAT
Rahane	49	75	3	0	0	0	0	0	1	0	0	100	158	5435	11	31	BAT
Dhawan	32	35	4	0	0	0	0	0	0	0	0	85	25	565	2	1	AR
Dhoni	56	45	3	1	0	0	0	0	3	2	0	75	78	2573	3	19	BAT
Axar	8	4	2	0	48	2	35	1	0	0	0	100	67	208	0	0	BWL
Pandya	42	36	3	3	30	0	25	0	1	0	0	75	70	77	0	0	BWL
Jadeja	18	10	1	1	60	3	50	2	1	0	1	85	16	1	0	0	BWL
Kedar	65	60	7	0	24	0	24	0	0	0	0	90	111	675	0	1	BWL
Ashwin	23	42	3	0	60	2	45	6	0	0	0	100	136	1914	0	10	AR
Umesh	0	0	0	0	54	0	50	4	1	0	0	110	296	9496	10	64	WK
Bumrah	0	0	0	0	60	2	49	1	0	0	0	60	73	1365	0	8	WK
Bhuwaneshwar	15	12	2	0	60	1	46	2	0	0	0	75	17	289	0	2	AR
Rohit	46	65	5	1	0	0	0	0	1	0	0	85	304	8701	14	52	BAT
Kartick	29	42	3	0	0	0	0	0	2	0	1	75	11	111	0	0	AR

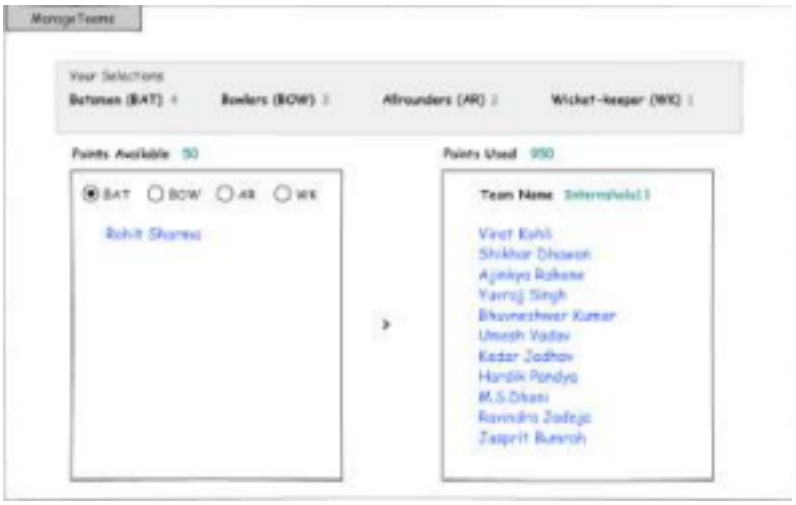
Testing /Deployment



- Opening screen of the application. You can see the players of each category by selecting the category. To begin with, the selection is disabled until a new team is created from the Manage Teams menu. A pop up asking the name of the team appears.

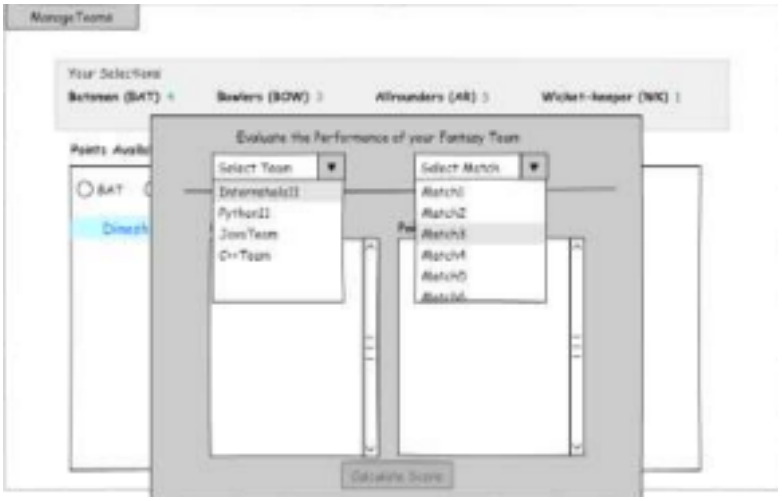
- The toolbar menu options which allow you to create a new team, open an existing team, save your team and finally evaluate the score of a saved team.

- After clicking New Team, the left box is populated with player names. As you select a different category, the corresponding list of players is displayed.



On double-clicking each player name, the right box gets populated. Points available and used are displayed accordingly.

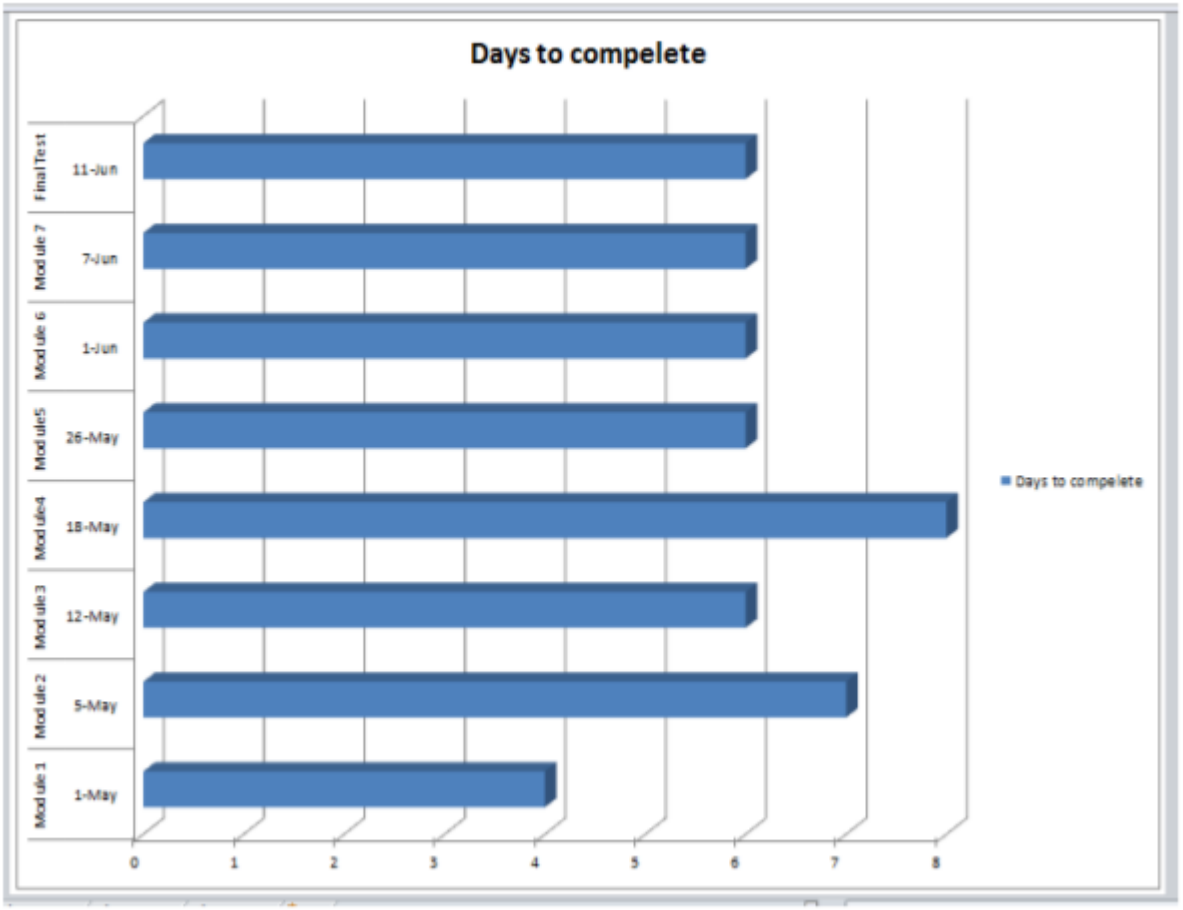
Message if the game logic is not followed



Upon opening the second file to evaluate the scores. You can select your team here and the match for which the players' performance is compared.

- The final score for your fantasy team based on the match selected.

Gantt chart


Days to compelete
Final Test
11-Jun
Module 7
7-Jun
Module 6
1-Jun
Module5
26-May
Module4
18-May
Module 3
12-May
Module 2
5-May
Module 1
1-May
0
1
2
3
4
5
6
7
8
Days to compelete

Problem Analysis

PRODUCT DEFINATION:-It is an game where you create a 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.

FEASIBILITY ANALYSIS:- I am building an software for gaming purposes using an specific technology named python. It is a game software where you can create virtual team according to your choice and score points to win an tournament.

This software is created for motivating street cricket and adding more fun and entertainment to cricket. The components that are used in this demo can be integrated to a high extent to provide statics to different components of cricket. This project helps in providing real time on field actions there by helping its user of the current actions happening on field.

Coding

File Edit Format Run Options Window Help

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'main_code.ui'
#'M O T CODE'
# Created by: PyQt5 UI code generator 5.14.2
#
# WARNING! All changes made in this file will be lost!

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 dialog

import sqlite3
fant=sqlite3.connect('cricket_db.db')  # connecting to database file(fandatabase
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)
```

Ln: 28 Col: 0

```
ile  Edit  Format  Run  Options  Window  Help
        self.BOWL.setText(str(self.bowlerscount))
        self.BAT.setText(str(self.batsmencount))
        self.ARL.setText(str(self.alrdscount))
        self.WK.setText(str(self.wicketerscount))
        self.list1.clear()
        self.load_names()

        self.sel_player.clear()


        #SAVE TEAM MENU
    def file_save(self):
        if not self.error():  #IF THERE IS AN ERROR
            msg = QMessageBox()
            msg.setIcon(QMessageBox.Critical)
            msg.setInformativeText(' Inufficient Players OR Points !!')
            msg.setWindowTitle("Fantasy Cricket")
            msg.exec_()
        elif self.error():  # IF NO ERROR
            try:
                fantcurs.execute("SELECT DISTINCT name FROM teams;")
                x = fantcurs.fetchall()
                for i in x:
                    if self.team_name.text() == i[0]:   # CHECKING IF THE TEAMNA
                        print('Updating already there')
                        fantcurs.execute("DELETE  FROM teams WHERE name='" + sel
            except:
                print('error')
            for i in range(self.sel_player.count()):
                # print('----addding--')
                # print('teamnane: ',self.team_name.text())
                # print('playername: ',self.list1[i])
                # print('points: ', player_points[self.list1[i]])
                try:
                    fantcurs.execute("INSERT INTO teams (name,players,value) VAL
                                     (self.team_name.text(), self.list1[i], play

                    # self.file_evaluate()
                except:
                    print('error in operation!')
```

File Edit Format Run Options Window Help

```
    def error(self):
        msg = QMessageBox()
        if self.wicketerscount > 1:
            msg.setIcon(QMessageBox.Critical)
            # msg.setText("Error")
            msg.setInformativeText('Only 1 wicketkeeper is allowed!')
            msg.setWindowTitle("Error")
            msg.exec_()
            return 0
        elif self.totalcount > 11:
            msg.setIcon(QMessageBox.Critical)
            msg.setInformativeText('No more than 11 players allowed!')
            msg.setWindowTitle("Selection Error")
            msg.exec_()
            return 0
        elif self.totalcount < 11 :
            return 0
        elif self.wicketerscount < 1:
            return 0
        elif self.avail_points <= -1:
            msg.setIcon(QMessageBox.Critical)
            msg.setInformativeText('Not enough points!')
            msg.setWindowTitle("Selection Cricket")
            msg.exec_()
            return 0

        return 1


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    MainWindow = QtWidgets.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

File Edit Format Run Options Window Help

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'newteam.ui'
#
# Created by: PyQt5 UI code generator 5.14.2
#
# WARNING! All changes made in this file will be lost!


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_Dialog(object):
    def setupUi(self, Dialog):
        Dialog.setObjectName("Dialog")
        Dialog.resize(407, 260)
        self.frame = QtWidgets.QFrame(Dialog)
        self.frame.setGeometry(QtCore.QRect(0, -20, 401, 231))
        self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)
        self.frame.setFrameShadow(QtWidgets.QFrame.Raised)
        self.frame.setObjectName("frame")
        self.label = QtWidgets.QLabel(self.frame)
        self.label.setGeometry(QtCore.QRect(110, 50, 210, 22))
        font = QtGui.QFont()
        font.setFamily("Segoe UI Semibold")
        font.setPointSize(11)
        font.setBold(True)
        font.setItalic(True)
        font.setWeight(75)
        self.label.setFont(font)
        self.label.setObjectName("label")
        self.team_name = QtWidgets.QLineEdit(self.frame)
        self.team_name.setGeometry(QtCore.QRect(90, 91, 221, 41))
        self.team_name.setObjectName("team_name")
        self.savename = QtWidgets.QPushButton(self.frame)
        self.savename.setGeometry(QtCore.QRect(140, 160, 93, 28))
        font = QtGui.QFont()
        font.setFamily("MS Sans Serif")
        font.setPointSize(8)
        font.setBold(False)
```

```
        font.setBold(True)
        font.setItalic(True)
        font.setWeight(75)
        self.label.setFont(font)
        self.label.setObjectName("label")
        self.team_name = QtWidgets.QLineEdit(self.frame)
        self.team_name.setGeometry(QtCore.QRect(90, 91, 221, 41))
        self.team_name.setObjectName("team_name")
        self.savename = QtWidgets.QPushButton(self.frame)
        self.savename.setGeometry(QtCore.QRect(140, 160, 93, 28))
        font = QtGui.QFont()
        font.setFamily("MS Sans Serif")
        font.setPointSize(8)
        font.setBold(False)
        font.setItalic(True)
        font.setWeight(50)
        self.savename.setFont(font)
        self.savename.setStyleSheet("font: italic 8pt \"MS Sans Serif\";")
        self.savename.setObjectName("savename")

        self.retranslateUi(Dialog)
        QtCore.QMetaObject.connectSlotsByName(Dialog)

    def retranslateUi(self, Dialog):
        _translate = QtCore.QCoreApplication.translate
        Dialog.setWindowTitle(_translate("Dialog", "new_team"))
        self.label.setText(_translate("Dialog", "Create New Team"))
        self.team_name.setPlaceholderText(_translate("Dialog", "enter team name"
        self.savename.setText(_translate("Dialog", "Save"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Dialog = QtWidgets.QDialog()
    ui = Ui_Dialog()
    ui.setupUi(Dialog)
    Dialog.show()
    sys.exit(app.exec_())
```

File Edit Format Run Options Window Help

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'open.ui'
#
# Created by: PyQt5 UI code generator 5.14.2
#
# WARNING! All changes made in this file will be lost!


from PyQt5 import QtCore, QtGui, QtWidgets
import sqlite3
match=sqlite3.connect('cricket_db.db')
matchcur=match.cursor()

class Ui_Dialog(object):
    def setupUi(self, Dialog):
        Dialog.setObjectName("Dialog")
        Dialog.resize(368, 274)
        self.label = QtWidgets.QLabel(Dialog)
        self.label.setGeometry(QtCore.QRect(80, 30, 299, 40))
        font = QtGui.QFont()
        font.setFamily("Segoe Print")
        font.setPointSize(14)
        font.setBold(True)
        font.setWeight(75)
        self.label.setFont(font)
        self.label.setObjectName("label")
        self.openbtn = QtWidgets.QPushButton(Dialog)
        self.openbtn.setGeometry(QtCore.QRect(120, 160, 93, 49))
        font = QtGui.QFont()
        font.setFamily("Microsoft Sans Serif")
        font.setPointSize(10)
        font.setBold(False)
        font.setItalic(False)
        font.setWeight(50)
        self.openbtn.setFont(font)
        self.openbtn.setStyleSheet("font: 10pt \"Microsoft Sans Serif\";")
        self.openbtn.setObjectName("openbtn")
```

File Edit Format Run Options Window Help

```
        font = QtGui.QFont()
        font.setFamily("Microsoft Sans Serif")
        font.setPointSize(10)
        font.setBold(False)
        font.setItalic(False)
        font.setWeight(50)
        self.openbtn.setFont(font)
        self.openbtn.setStyleSheet("font: 10pt \"Microsoft Sans Serif\";")
        self.openbtn.setObjectName("openbtn")
        self.open_cb = QtWidgets.QComboBox(Dialog)
        self.open_cb.setGeometry(QtCore.QRect(70, 100, 211, 31))
        self.open_cb.setObjectName("open_cb")

        self.retranslateUi(Dialog)
        QtCore.QMetaObject.connectSlotsByName(Dialog)

        teams= matchcur.execute("SELECT DISTINCT name FROM teams;")  # fetching
        y= teams.fetchall()
        for i in y:
            self.open_cb.addItem(i[0])



    def retranslateUi(self, Dialog):
        _translate = QtCore.QCoreApplication.translate
        Dialog.setWindowTitle(_translate("Dialog", "Dialog"))
        self.label.setText(_translate("Dialog", "select team  to open"))
        self.openbtn.setText(_translate("Dialog", "open"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Dialog = QtWidgets.QDialog()
    ui = Ui_Dialog()
    ui.setupUi(Dialog)
    Dialog.show()
    sys.exit(app.exec_())
```

File Edit Format Run Options Window Help

```
import sqlite3

# connecting to database file
mycurs = sqlite3.connect('cricket_db.db')
curs = mycurs.cursor()

#CREATING MATCH TABLE
curs.execute('''CREATE TABLE IF NOT EXISTS match (player TEXT NOT NULL,scored IN



#CREATING STATS TABLE
curs.execute('''CREATE TABLE IF NOT EXISTS stats (player PRIMARY KEY,matches INT


#CREATING TEAMS TABLE
curs.execute('''CREATE TABLE IF NOT EXISTS teams (name TEXT NOT NULL,players TEX



#DISPLAY DATA IF EXISTS IN DATABASE
sql="select * from match"
curs.execute(sql)
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) :")
#ADDING DATA FROM USER TO MATCH TABLE
while(opt=='y' or opt=='Y'):

    row=[input("Player name :")]
    row.append(int(input("Score:")))
```

File Edit Format Run Options Window Help

```
    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: ")))


    try:
        curs.execute("INSERT INTO match (player,scored, faced, fours,sixes,bowle
                     (row[0],row[1], row[2], row[3],row[4],row[5],row[6],ro
        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:       #try block to catch exceptions

        curs.execute("INSERT INTO stats (player,matches,runs, hundreds, fifties,
                     (row[0],row[12], row[13], row[14],row[15],row[16],row[
        mycurs.commit()
```

```
le  Edit  Format  Run  Options  Window  Help
mport sqlite3
connecting to cricket_db.db
b = sqlite3.connect("cricket_db.db")
ursor = db.cursor()
ursor.execute("SELECT * FROM match")
ow = cursor.fetchall()

ef calculate_points(row):
    points = 0.0
    score = row[1]
    try:
        strike_rate = float(row[1]) / float(row[2])  # strike rate =runs/balls f
    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])

    # 1 point for hitting a boundary and 2 runs ,2 points for over boundary,10 p
    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
```

Conclusion

I believe the trial has shown conclusively that it is both possible and desirable to use Python as the principal teaching language:

• It is Free (as in both cost and source code).

• It is trivial to install on a Windows PC allowing students to take their interest further. For many the hurdle of installing a Pascal or C compiler on a Windows machine is either too expensive or too complicated;

• It is a flexible tool that allows both the teaching of traditional procedural programming and modern OOP; It can be used to teach a large number of transferable skills;

• It appears to be quicker to learn and, in combination with its many libraries, this offers the possibility of more rapid student development allowing the course to be made more challenging and varied;

• Most importantly, its clean syntax offers increased understanding and enjoyment for students.

BIBLIOGRAPHY

• https://trainings.internshala.com/python-training
• https://www.w3schools.com/python/
• https://wiki.python.org/moin/PyQt/Tutorials
• https://www.tutorialspoint.com/pyqt/
• https://www.tutorialspoint.com/sqlite/sqlite_quick_guide.htm
• https://www.quora.com/

Weekly Progress Sheet

Progress report

- Getting Started
- Introduction to Python
  - Score: 85%
- Using Variables in Python
  - Score: 95%
  - Assignment submitted
- Basics of Programming in Python
  - Score: 89%
  - Assignment submitted
- Principles of Object-oriented Programming (OOP)
  - Score: 37%
  - Assignment submitted
- Connecting to SQLite Database
  - Score: 85%
  - Assignment submitted
- Developing a GUI with PyQT
  - Score: 67%
  - Assignment submitted
- Application of Python in Various Disciplines
  - Score: 100%
- The Final Project
  - Score: 50%
- Final test
  - Score: 80%