

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgavi- 590 018



Mini-Project Synopsis

On

“RLJIT Student Registration Form”

Submitted in partial fulfilment of the requirement for the 6th semester

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

By

V. Pavan Kumar

1RL18EC118

T. Bhanu Prakash Reddy

1RL18EC113

S. Jagadeesh

1RL18EC102

**M. Durga Venkata Prasad
Reddy**

1RL18EC066

Under the guidance of

DR. SHIVAPRASAD.K.M

Professor,
Dept. of E&CE



DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

R.L. JALAPPA INSTITUTE OF TECHNOLOGY

Kodigehalli, Doddaballapur - 561 203

2020-21

R L JALAPPA INSTITUTE OF TECHNOLOGY

DODDABALLAPUR -561203(KARNATAKA)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING



CERTIFICATE

This is to certify that the Project work entitled "*RLJIT Student Registration Form*" is a bonafied work carried out by **V.PAVAN KUMAR (1RL18EC118)** ,**T.BHANU PRAKASH REDDY (1RL18EC113)**, **S.JAGADEESH (1RL18EC102)** and **M.DURGA VENKATA PRASAD REDDY (1RL18EC066)**, in partial fulfillment for the requirement of VI semester, Bachelor of Engineering in Electronics and Communication Engineering of **VISVESVARAYA TECHNOLOGICAL UNIVERSITY** , Belgavi, during the year **2020-2021**. It is certified that all corrections/suggestions indicated for the internal assessment have been incorporated in the report. This report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for said degree.

Signature of Guide

DR. SHIVA PRASAD KM

Dept. of E&CE, RLJIT

Signature of HOD

DR. ANIL KUMAR C

Dept. of, E&CE RLJIT

Signature of Principal

Dr.Sreenivasa Reddy M

RLJIT

Evaluators:

Name

Signature with Date

1. _____

2. _____

ABSTRACT

The aim of this project is to build a Student Registration System that will completely automate the process of new student registration in our Institute. The system will handle every semester mark storing and registration of new students.

The System will be GUI based, web based and will have two implementations i.e., user-side (student) and server-side (Institute).

The server-side implementation can only be accessed over the Institute while the client side can be accessed over anyone.

The process begins when prospective student wish to enroll in the institute. If they express interest in any course they will be required to visit the institute and can create seat in the institute. This is the only part of the system where every user has to have human interaction, this is intended as a security measure to prevent the creation of false IDs. After verification of any nationally approved user id will be created.

Student will be able to accept or decline the offers of admission and register using this portal

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance crowned my efforts with success.

We extend our deep sense of sincere gratitude to **Dr. Sreenivas Reddy M, Principal**, R L Jalappa Institute of Technology, Doddaballapur, for providing us an opportunity to continue our studies.

We express our heartfelt sincere gratitude to **Dr. Anil Kumar, Head, Department of Electronics and communication Engineering**, R L Jalappa Institute of Technology, Doddaballapur, for his valuable suggestions and support.

We express our special in-depth, heartfelt, sincere gratitude to **DR. SHIVAPRASAD K.M, Electronics and communication Engineering**, R L Jalappa Institute of Technology, Doddaballapur for his constant support.

Finally, we would like to thank all the faculty and supporting staff members of Department of Electronics and communication Engineering, R L Jalappa Institute of Technology, Doddaballapur, for their support.

V. Pavan Kumar

(1RL18EC118)

T. Bhanu Prakash Reddy

(1RL18EC113)

S. Jagadeesh

(1RL18EC102)

M. Durga Venkata Prasad

Reddy (1RL18EC066)

TABLE OF CONTENTS

Abstract		i
Acknowledgement		ii
Chapter NO	Chapter Name	Page NO
Chapter 1	Introduction	1-3
	1.1.Modules	
	1.2.Background	
Chapter 2	Block diagram & Algorithm	4-5
	2.1. activity diagram	
Chapter 3	System Environment	6-11
	3.1. Software components	
	3.2. hardware components	
	3.3. functional requirements	
	3.4. non-functional requirements	
Chapter 4	Applications & Limitations	12-13
	4.1. Applications	
	4.2. Advantages	
	4.3. Limitations	
Chapter 5	Implementation	14-46
	5.1. Creating Interface	
	5.2. PyMySQL	
	5.3. python code	
	5.4. Expected outcome	
	Snapshots	47
Chapter 6	Conclusion	48

CHAPTER-1

INTRODUCTION

Records management is the field of management responsible for efficient and systematic control of the creation, receipt, maintenance, use, and disposition of the registration records of students . Given the ever-increasing growth of litigation, investigations and regulatory actions coupled with the explosive growth of electronic records, automation of school's records management policies, processes and practices is no longer an option; it is now a necessity. Automated Student Registration Systems (ASRSs) have been available for nearly ten years. However, they have not been adopted or implemented at the anticipated level of success because the process of declaring and classifying records has been largely manual. To facilitate the adoption of needed new registration systems, all manual operating costs associated with the system's use and operation must be minimized.

Standard High School uses a file based system for keeping records such as admission book, school fees (ledger book), warden's book , also books and files for both teachers and students which are becoming expensive. These records include admission of students, discipline cases of students ,staff members and students performance among others however this is not efficient enough because the books wear out, it takes time looking for specific information in a lot of files and also get lost by malicious students or teachers retrenched hence loss of the school's records. This system of Student Registration was designed for improving the effectiveness and efficiency of managing student's registration electronic records.

The national focus on student outcomes has placed an additional burden on our nation's schools, school districts, and state education agencies according to The National Forum For Educational Statistics, for they must monitor the achievement of individual students, as well as groups of students, and show that all students are meeting high standards for learning. An education organization's ability to meet this challenge is affected by the organization's access to complete, accurate, and timely information about its students. This project has been developed to help education organizations plan and implement efficient systems for maintaining and using individual student records so that effective decisions can be made for the benefit of the students and administration including DOS, teachers etc.

Many schools, and state education agencies already collect and use data effectively. However, the proliferations of new reporting requirements and dramatic changes in technology have had a profound effect on the need for student data and the education community's ability to manage student records. Purchase of more powerful computer hardware and software and the reconfiguration of information systems have become essential components in efforts to meet the needs of all students.

1.1 MODULES

The system comprises of 4 major modules as follows:

- Tkinter:
 - It is used to create a new Interface
- PIL:
 - It is used to insert necessary images in interface
- pymysql:
 - It is used to store the data in Database.
- browser:
 - It is used to link the specific browser URL

1.2 Background

Standard High School is a privately owned, mixed boarding, O and A level secondary school offering a wide range of subjects for students to choose from.

The school was founded in 1994 by a group of experienced and committed professional teachers. It has since then grown and expanded enough to excel and compete with traditional schools in the country. The school is registered by the ministry of education and has a UNEB center U0833.

The school uses books to keep records concerning information about students admitted and staff employed.

Standard High School has been using a manual system to capture the department and students records that are obtained. Records that are used in the school by the department are kept in the files until the end of every Term or year for assessment of student's performance. A lot of time is needed to process the request for a particular department in case all the departments order at the same time for the students records.

Since the requests from other stuffs are also made through the head office, this manual system takes time and often time computational errors are in the system. Also due to this paper manual system, there isn't much security in the records and every one could get access to the store registry file even if not authorized to do so. Reference to a particular registration record in a particular year would imply manually opening all the old files until the record is found. This did not only consume time but also wasted resources and manual labor in shifting and moving the files.

1.2.3.PROBLEM STATEMENT

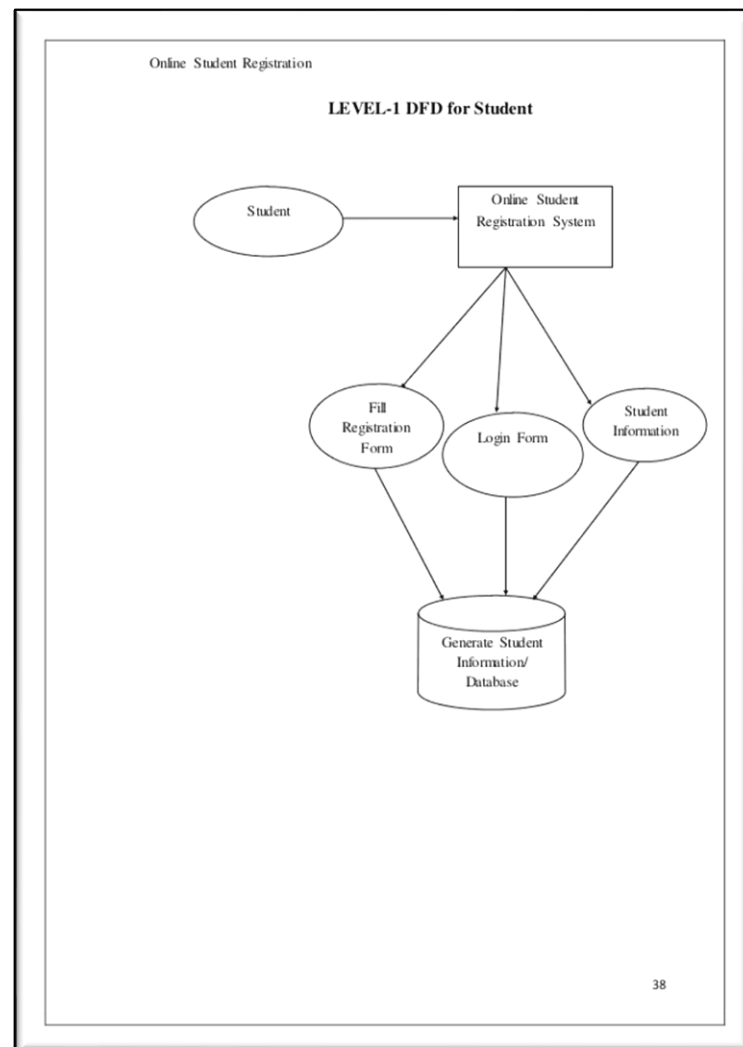
Standard High School uses a file based system for keeping records such as admission book, school fees (ledger book), warden's book , also books and files for both teachers and students which are becoming expensive. These records include admission of students, discipline cases of students ,staff members and students performance among others however this is not efficient enough because the books wear out, it takes time looking for specific information in a lot of files and also get lost by malicious students or teachers retrenched hence loss of the school's records. It also retains the schools records for the case where a teacher is dismissed from the school and a new teacher is supposed to begin commencement of teaching.

CHAPTER 2

BLOCK DIAGRAM & ALGORITHM

2.1. ACTIVITY DIAGRAM

Activity diagrams are a loosely defined diagram technique for showing work flows of stepwise activities and actions, with support for choice, iteration and concurrency.



2.1.1. Activity Diagram of RLJIT Student Registration Form

There are two different users who will use this application:

1. Administration
2. Student

Administration can view and edit the details of any students. Admin can add new users and he can edit and delete the students. Admin can also edit and delete the marks of the student.

Students can submit the details given in the form and register. He can also submit the marks like internal and external as per the semester. And we provide the “Info” button which gives all the details of our institute.

Features that are available for the students are:

Name, Contact, Email, Gender, Branch, Date of birth, Address 10&PUC percentage.

User can manage following tasks from this project:

- Student Registration
- Information about the College
- Semester wise selection
- Examination marks Entry

Modules of the Student Registration Form:

The main modules of the student as follows

Register Form:

This module help student to get registration. Using this system user can registered by filling up the registration form online. Note, after register student can't edit the details/information.

Marks:

With this module student can enter their marks as per their semester. After submitted student can't edit the marks only admin can edit it.

Features that are available to the admin are:

Admin can enable or disable student account. He can also edit student information in database. He can also search any student in the system by entering name or mail id.

CHAPTER 3

SYSTEM ENVIRONMENT

3.1. SOFTWARE COMPONENTS

3.1.1. Tkinter MODULE

“pip install tkinter”

Most of the time, tkinter is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named `_tkinter`. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

In addition to the Tk interface module, tkinter includes a number of Python modules, `tkinter.constants` being one of the most important. Importing tkinter will automatically import `tkinter.constants`, so, usually, to use Tkinter all you need is a simple import statement:

```
import tkinter
```

Or, more often:

```
from tkinter import *
```

```
class tkinter.Tk(screenName=None, baseName=None, className='Tk', useTk=1)
```

The Tk class is instantiated without arguments. This creates a top level widget of Tk which usually is the main window of an application. Each instance has its own associated Tcl interpreter.

```
tkinter.Tcl(screenName=None, baseName=None, className='Tk', useTk=0)
```

The Tcl() function is a factory function which creates an object much like that created by the Tk class, except that it does not initialize the Tk subsystem. This is most often useful when driving the Tcl interpreter in an environment where one doesn't want to create extraneous toplevel windows, or where one cannot (such as Unix/Linux systems without an X server). An object created by the Tcl() object can have a Toplevel window created (and the Tk subsystem initialized) by calling its `loadtk()` method.

3.1.2. PIL MODULE

“pip install pillow”

The Python Imaging Library adds image processing capabilities to your Python interpreter.

This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.

The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

3.1.3. PYMYSQL

This package contains a pure-Python MySQL client library, based on [PEP 249](#).

In the following example, we get the version of MySQL.

```
version.py
#!/usr/bin/python

import pymysql

con = pymysql.connect('localhost', 'user7',
    's$cret', 'testdb')

try:

    with con.cursor() as cur:

        cur.execute('SELECT VERSION()')

        version = cur.fetchone()

        print(f'Database version: {version[0]}')

finally:

    con.close()
```

In MySQL, we can use `SELECT VERSION()` to get the version of MySQL.

```
import pymysql
```

We import the pymysql module.

```
con = pymysql.connect('localhost', 'user7',
    's$cret', 'testdb')
```

We connect to the database with `connect`. We pass four parameters: the hostname, the MySQL user name, the password, and the database name.

```
with con.cursor() as cur:
```

Using the with keyword, the Python interpreter automatically releases the resources. It also provides error handling. We get a cursor object, which is used to traverse records from the result set.

```
cur.execute('SELECT VERSION()')
```

We call the execute function of the cursor and execute the SQL statement.

```
version = cur.fetchone()
```

The fetchone function fetches the next row of a query result set, returning a single sequence, or None when no more data is available.

```
print(f'Database version: {version[0]}')
```

We print the version of the database.

```
finally:
```

```
    con.close()
```

The pymysql module does not implement the automatic handling of the connection resource; we need to explicitly close the connection with close in the finally clause.

3.1.4. BROWSER

The web browser module provides a high-level interface to allow displaying Web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing.

Under Unix, graphical browsers are preferred under X11, but text-mode browsers will be used if graphical browsers are not available or an X11 display isn't available. If text-mode browsers are used, the calling process will block until the user exits the browser.

If the environment variable BROWSER exists, it is interpreted as the os.pathsep-separated list of browsers to try ahead of the platform defaults. When the value of a list part contains the string %s, then it is interpreted as a literal browser command line to be used with the argument URL substituted for %s; if the part does not contain %s, it is simply interpreted as the name of the browser to launch. 1

For non-Unix platforms, or when a remote browser is available on Unix, the controlling process will not wait for the user to finish with the browser, but allow the remote browser to maintain its own windows on the display. If remote browsers are not available on Unix, the controlling process will launch a new browser and wait.

The script `web browser` can be used as a command-line interface for the module. It accepts a URL as the argument. It accepts the following optional parameters: `-n` opens the URL in a new browser window, if possible; `-t` opens the URL in a new browser page ("tab"). The options are, naturally, mutually exclusive.

3.1.5. PYTHON

Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language construct and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

3.1.6. PYCHARM

PyCharm is a cross platform integrated development environment (IDE) for Python programmers. It embodies all the tooling a Python programmer needs to be productive including an excellent programming text editor, syntax highlighting, code completion, project navigation, database tooling, and project options for web development. Pycharm is an IDE tool kit basically meant for developing programs and/or building software's in python. Developed by JetBrains it provides assistive toolkit and build-in features like most of the IDE's like eclipse. It also supports development for JavaScript, Coffee Script, Typescript, CSS etc.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition released under a proprietary license - this has extra features.

3.2. HARDWARE COMPONENTS

- INTEL i3 or higher \ AMD Ryzen 3 or higher
- PC \ Laptop
- Hard-drive : minimum 10 GB

- PyCharm and Xampp
- Internet Connection

3.3. FUNCTIONAL & NONFUNCTIONAL REQUIREMENTS

3.3.1. FUNCTIONAL REQUIREMENTS

The following are the functional requirements of our project, they are:

- Device must be enabled or disabled by user.
- Xampp application should be enabled
- Device should be able to display form application.

3.3.2. NONFUNCTIONAL REQUIREMENTS

Performance

The system must be interactive and the delays involved must be less .So in every action-response of the system, there are no immediate delays. In case of opening windows forms, of popping error messages and saving the settings or sessions there is delay much below 2 seconds, In case of opening databases, sorting questions and evaluation there are no delays and the operation is performed in less than 2 seconds for opening ,sorting, computing, posting > 95% of the files. Also when connecting to the server the delay is based editing on the distance of the 2 systems and the configuration between them so there is high probability that there will be or not a successful connection in less than 20 seconds for sake of good communication.

Safety

Information transmission should be securely transmitted to server without any changes in information

Reliability

As the system provide the right tools for discussion, problem solving it must be made sure that the system is reliable in its operations and for securing the sensitive details.

Availability

If the internet service gets disrupted while sending information to the server, the information can be send again for verification.

Security

The main security concern is for users account hence proper login mechanism should be used to avoid hacking. The tablet id registration is way to spam check for increasing the security. Hence, security is provided from unwanted use of recognition software.

Usability

As the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.

CHAPTER 4

APPLICATIONS AND LIMITATIONS

4.1. APPLICATIONS

- Save time
- Free Online Registration
- Centralized data management
- Improve event efficiency
- Customized online registration
- Detailed reporting
- Improve marketing efforts
- Set up online surveys

4.2. ADVANTAGES

- Convenience and Speed
- Immediate Confirmation
- Online registration systems and secure
- Realtime update about the statistics
- Online registrations are eco-friendly

4.3. LIMITATIONS

- Computer Literacy and Internet Access
- Low Computer Literacy
- Security Concerns
- Authenticity
- Infrastructural Requirements

CHAPTER 5

IMPLEMENTATION

5.1. CREATING INTERFACE

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

5.2. PYMYSQL

You might already know that to use any database we need the database driver. And PyMySQL is a pure python driver for operating MySQL in Python. Now, remember, this PyMySQL is not available by default. So, first, we will learn how do we download and install this driver.

Installing PyMySQL

Here, I am assuming you have python installed already on your machine. And also you have done with the setting environment variables stuff.

“pip install pymysql”

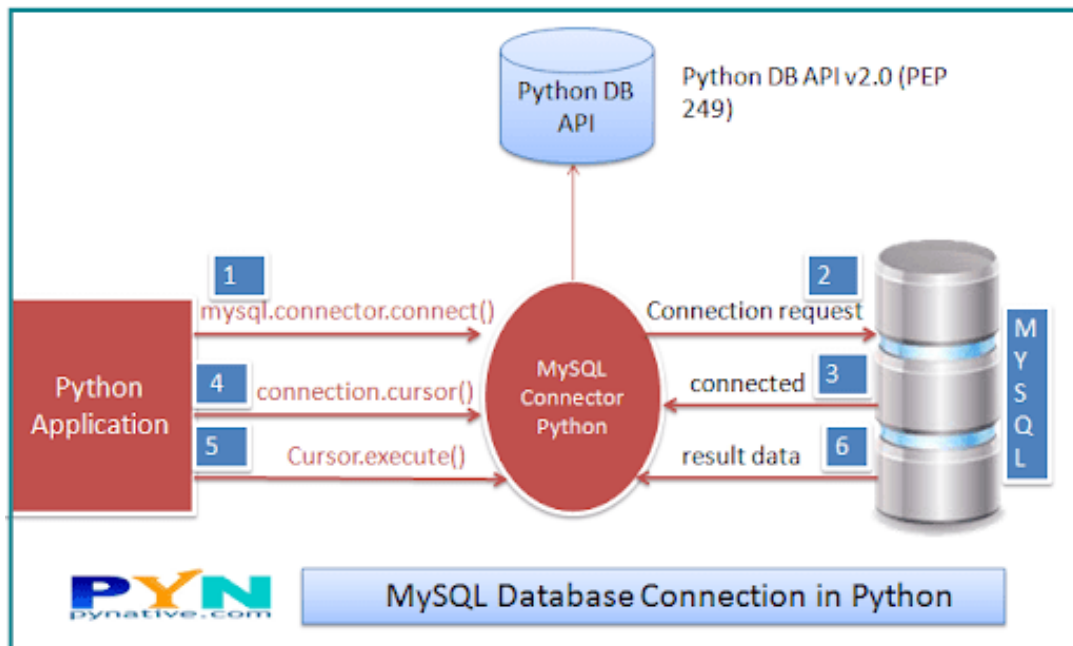


FIG 5.2.1: Python to Database Connection

5.4. python code for RLJIT Student Registration Form

```
from tkinter import *
from tkinter import ttk, messagebox
from PIL import Image, ImageTk
import pymysql
import webbrowser as wb

class Register:
    def __init__(self, root):
        self.root = root
        self.root.title("Registration Form")
        self.root.geometry("1550x800")
        self.root.config(bg="white")

        # -----BackGround Image-----
        self.bg = ImageTk.PhotoImage(file="images/overview-bg.jpg")
        bg = Label(self.root, image=self.bg).place(x=200, y=0, relwidth=1,
        relheight=1)

        # -----Left Image-----
        self.left = ImageTk.PhotoImage(file="images/left_pic2.png")
        left = Label(self.root, image=self.left).place(x=80, y=100,
        width=400, height=600)

        # -----RLJIT Logo-----
        self.logo = ImageTk.PhotoImage(file="images/rljit2.jpg")
        logo = Label(bg, image=self.logo).place(x=150, y=110) # ,
        height=200, width=100)
```

```

# -----Registration Frame-----
frame1 = Frame(self.root, bg="white")
frame1.place(x=480, y=100, width=850, height=600)

# -----Title-----
title = Label(frame1, text="REGISTER HERE", font=("times new
roman", 20, "bold"), bg="white", fg="green").place(
    x=50, y=10)

# ----- First name-----
# self.var_frame = StringVar()
f_name = Label(frame1, text="First name", font=("times new roman",
15, "bold"), bg="white", fg="black").place(
    x=50, y=70)
self.txt_first = Entry(frame1, font=("times new roman", 15,
"bold"), bg="white", fg="blue")
self.txt_first.place(x=50, y=100, width=250)
# textvariable=self.var_frame)

# -----Last_name-----
l_name = Label(frame1, text="Last name", font=("times new roman",
15, "bold"), bg="white", fg="black").place(
    x=370, y=70)
self.txt_last = Entry(frame1, font=("times new roman", 15, "bold"),
bg="white", fg="blue")
self.txt_last.place(x=370, y=100, width=250)

# -----Contact Number-----
contact = Label(frame1, text="Contact No", font=("times new roman",
15, "bold"), bg="white", fg="black").place(
    x=50, y=140)
self.txt_contact = Entry(frame1, font=("times new roman", 15,
"bold"), bg="white", fg="blue")
self.txt_contact.place(x=50, y=180, width=250)

# -----Email ID-----
mail = Label(frame1, text="Email ID", font=("times new roman", 15,
"bold"), bg="white", fg="black").place(
    x=370, y=140)
self.txt_email = Entry(frame1, font=("times new roman", 15,
"bold"), bg="white", fg="blue")
self.txt_email.place(x=370, y=180, width=250)

# -----Date of birth ID-----
dob = Label(frame1, text="Date Of Birth", font=("times new roman",
15, "bold"), bg="white", fg="black").place(
    x=50, y=230)
self.txt_dob = Entry(frame1, font=("times new roman", 15, "bold"),
bg="white", fg="blue")
self.txt_dob.place(x=50, y=270, width=250)

# -----Gender-----
gender = Label(frame1, text="Gender", font=("times new roman", 15,
"bold"), bg="white", fg="black").place(
    x=370, y=230)
self.cmb_gender = ttk.Combobox(frame1, font=("times new roman", 15,
"bold"), justify=CENTER)
self.cmb_gender['values'] = ("Select", "Male", "Female")
self.cmb_gender.place(x=370, y=270, width=250)
# self.cmb_gender.current(0)

# ----- 10 per -----

```

```

        ten_per = Label(frame1, text="10% / SGPA", font=("times new roman",
15, "bold"), bg="white", fg="black").place(
            x=50, y=320)
        self.txt_ten = Entry(frame1, font=("times new roman", 15, "bold"),
bg="white", fg="blue")
        self.txt_ten.place(x=50, y=350, width=250)

        # ----- PUC per -----
        puc_per = Label(frame1, text="PUC %", font=("times new roman", 15,
"bold"), bg="white", fg="black").place(x=370,
y=320)
        self.txt_puc = Entry(frame1, font=("times new roman", 15, "bold"),
bg="white", fg="blue")
        self.txt_puc.place(x=370, y=350, width=250)

        # -----Address-----
        address = Label(frame1, text="Address", font=("times new roman",
15, "bold"), bg="white", fg="black").place(
            x=50, y=400)
        self.txt_address = Entry(frame1, font=("times new roman", 15,
"bold"), bg="white", fg="blue")
        self.txt_address.place(x=50, y=430, width=250)

        # -----Branch-----
        branch = Label(frame1, text="Branch", font=("times new roman", 15,
"bold"), bg="white", fg="black").place(
            x=370, y=400)
        self.cmb_branch = ttk.Combobox(frame1, font=("times new roman", 15,
"bold"), justify=CENTER)
        self.cmb_branch['values'] = ("Select", "Computer Science and
Engineering",
                                   "Electronics and Communication
Engineering",
                                   "Mechanical Engineering")
        self.cmb_branch.place(x=370, y=430, width=250)
        # self.cmb_branch.current(0)

        # -----terms and conditions-----
        self.var_chk = IntVar()
        self.chk = Checkbutton(frame1, text="I agree the terms and
conditions", variable=self.var_chk, offvalue=0,
                                onvalue=1, bg="white", font=("times new
roman", 10), justify=CENTER).place(x=50, y=480)

        # -----Register-----
        self.btn_register = Button(frame1, text="Register", font=("times
new roman", 15, "bold"), bg="green",
                                fg="white",
                                justify=CENTER,
command=self.register_data).place(x=50, y=530, width=200)

        # -----Clear -----
        self.btn_clear = Button(frame1, text="Clear", font=("times new
roman", 15, "bold"), bg="blue", fg="white",
                                command=self.clear_data).place(x=370,
y=530, width=200)

        # ----- Info button -----
        info = Label(self.root, text="Click to check details of this
college", font=("times new roman", 10)
                    , bg="pink").place(x=170, y=510)
        info_btn = Button(self.root, text="Info", font=("times new roman",

```

```

15, "bold"), bg="white", fg="black",
                    command=self.info).place(x=200, y=550, width=150)

# ----- Marks button -----
marks = Label(self.root, text="Click to enter the marks",
font=("times new roman", 15, "bold"),
                    bg="pink").place(x=150, y=610)
marks_btn = Button(self.root, text="Marks", font=("times new
roman", 15, "bold"), bg="white", fg="black",
                    command=self.login).place(x=200, y=650,
width=150)

# ----- Sem GUI -----

def login(self):
    frame2 = Tk()
    frame2.geometry("1550x800")
    frame2.title("ECE")

    # -----title-----
    tit = Label(frame2, text="Select the Semester", font=("times new
roman", 30, "bold"), bg="white"
                ).place(x=500, y=10)

    # ----- p cycle -----
    p_cycle = Label(frame2, text="P-Cycle :", font=("times new roman",
20, "bold")).place(x=500, y=100)
    btn_p = Button(frame2, text="Click here", font=("times new roman",
15, "bold"), bg="white",
                    command=self.p_cycle) \
                .place(
                    x=700, y=100)

    # ----- c cycle -----
    c_cycle = Label(frame2, text="C-Cycle :", font=("times new roman",
20, "bold")).place(x=500, y=170)
    btn_c = Button(frame2, text="Click here", font=("times new roman",
15, "bold"), bg="white", command=self.c_cycle
                ).place(x=700, y=170)

    # ----- 3th Sem -----
    three = Label(frame2, text="3th Sem :", font=("times new roman",
20, "bold")).place(x=500, y=240)
    btn_three = Button(frame2, text="Click here", font=("times new
roman", 15, "bold"), bg="white",
                       command=self.three_sem).place(x=700, y=240)

    # ----- 4th Sem -----
    four = Label(frame2, text="4th Sem :", font=("times new roman", 20,
"bold")).place(x=500, y=310)
    btn_four = Button(frame2, text="Click here", font=("times new
roman", 15, "bold"), bg="white",
                       command=self.four_sem).place(x=700, y=310)

    # ----- 5th Sem -----
    five = Label(frame2, text="5th Sem :", font=("times new roman", 20,
"bold")).place(x=500, y=380)
    btn_five = Button(frame2, text="Click here", font=("times new
roman", 15, "bold"), bg="white",
                       command=self.five_sem).place(x=700, y=380)

    # ----- 6th Sem -----
    six = Label(frame2, text="6th Sem :", font=("times new roman", 20,

```

```

"bold")).place(x=500, y=450)
    btn_six = Button(frame2, text="Click here", font=("times new
roman", 15, "bold"), bg="white",
                    command=self.six_sem).place(x=700, y=450)

    # ----- 7th Sem -----
    seven = Label(frame2, text="7th Sem :", font=("times new roman",
20, "bold")).place(x=500, y=520)
    btn_seven = Button(frame2, text="Click here", font=("times new
roman", 15, "bold"), bg="white",
                    command=self.seven_sem).place(x=700, y=520)

    # ----- 8th Sem -----
    eight = Label(frame2, text="8th Sem :", font=("times new roman",
20, "bold")).place(x=500, y=590)
    btn_eigth = Button(frame2, text="Click here", font=("times new
roman", 15, "bold"), bg="white",
                    command=self.eight_sem).place(x=700, y=590)

    frame2.mainloop()
    # -----#
-----#

# ----- P Cycle -----#

def p_cycle(self):
    frame3 = Tk()
    frame3.geometry("1550x800")
    frame3.title("sem-1")

    # ----- title -----
    title = Label(frame3, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame3, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
    self.text_name = Entry(frame3, font=("times new roman", 15,
"bold"), fg="blue")
    self.text_name.place(x=150, y=100, width=250)

    # ----- USN -----
    usn = Label(frame3, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
    self.text_usn = Entry(frame3, font=("times new roman", 15, "bold"),
fg="blue")
    self.text_usn.place(x=150, y=150, width=250)

    sem = Label(frame3, text="1 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=
CENTER).place(x=600, y=200)

    # -----sub code -----
    code = Label(frame3, text="Code", font=("times new roamn", 15,
"bold"), fg="black").place(x=50, y=250)
    code1 = Label(frame3, text="18MAT11", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
    code2 = Label(frame3, text="18PHY12", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
    code3 = Label(frame3, text="18ELE13", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
    code4 = Label(frame3, text="18CIV14", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
    code5 = Label(frame3, text="18EGDL15", font=("times new roman", 15,

```



```

"bold"), fg="black").place(x=50, y=500)
    code6 = Label(frame3, text="18PHYL16", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
    code7 = Label(frame3, text="18LELE17", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
    code8 = Label(frame3, text="18EGH18", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)

    # ----- Subject -----
    subject = Label(frame3, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)
    sub1 = Label(frame3, text="Calculus and linear algebra",
font=("times new roman", 15, "bold"), fg="black"
    ).place(x=200, y=300)
    sub2 = Label(frame3, text="Engineering Physics", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=350)
    sub3 = Label(frame3, text="Basic Electrical Engineering",
font=("times new roman", 15, "bold"), fg="black"
    ).place(x=200, y=400)
    sub4 = Label(frame3, text="Element of Civil Engineering and
Machanics", font=("times new roman", 15, "bold"),
    fg="black").place(x=200, y=450)
    sub5 = Label(frame3, text="Engineering Graphics", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=500)
    sub6 = Label(frame3, text="Engineering Physics Laboratory",
font=("times new roman", 15, "bold"), fg="black"
    ).place(x=200, y=550)
    sub7 = Label(frame3, text="Basic Electrical Engineering
Laboratory", font=("times new roman", 15, "bold"),
    fg="black").place(x=200, y=600)
    sub8 = Label(frame3, text="Technical English-1", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=650)

    # ----- Internal -----
    internal = Label(frame3, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
    self.text_int1 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int1.place(x=600, y=300, width=100)
    self.text_int2 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int2.place(x=600, y=350, width=100)
    self.text_int3 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int3.place(x=600, y=400, width=100)
    self.text_int4 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int4.place(x=600, y=450, width=100)
    self.text_int5 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int5.place(x=600, y=500, width=100)
    self.text_int6 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int6.place(x=600, y=550, width=100)
    self.text_int7 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int7.place(x=600, y=600, width=100)
    self.text_int8 = Entry(frame3, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int8.place(x=600, y=650, width=100)

```

```

# ----- External -----
external = Label(frame3, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
self.txt_ext1 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext1.place(x=800, y=300, width=100)
self.txt_ext2 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext2.place(x=800, y=350, width=100)
self.txt_ext3 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext3.place(x=800, y=400, width=100)
self.txt_ext4 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext4.place(x=800, y=450, width=100)
self.txt_ext5 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext5.place(x=800, y=500, width=100)
self.txt_ext6 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext6.place(x=800, y=550, width=100)
self.txt_ext7 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext7.place(x=800, y=600, width=100)
self.txt_ext8 = Entry(frame3, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext8.place(x=800, y=650, width=100)

# ----- Submit -----
btn_submit = Button(frame3, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
command=lambda:[self.sem1(),
self.ext_sem1()]).place(x=700, y=700, width=150)

frame3.mainloop()

# ----- C Cycle -----#

def c_cycle(self):
    frame4 = Tk()
    frame4.title("sem-2")
    frame4.geometry("1550x800")
    # ----- title -----
    title = Label(frame4, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame4, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
    self.text_name = Entry(frame4, font=("times new roman", 15,
"bold"), fg="blue")
    self.text_name.place(x=150, y=100, width=250)

    # ----- USN -----
    usn = Label(frame4, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
    self.text_usn = Entry(frame4, font=("times new roman", 15, "bold"),
fg="blue")
    self.text_usn.place(x=150, y=150, width=250)

    sem = Label(frame4, text="2 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=

```

```

        CENTER).place(x=600, y=200)
        # -----sub code -----
        code = Label(frame4, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
        code1 = Label(frame4, text="18MAT21", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
        code2 = Label(frame4, text="18CHE22", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
        code3 = Label(frame4, text="18CPS23", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
        code4 = Label(frame4, text="18ELN24", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
        code5 = Label(frame4, text="18ME25", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)
        code6 = Label(frame4, text="18CHEL26", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
        code7 = Label(frame4, text="18CPL27", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
        code8 = Label(frame4, text="18EGH28", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)

        # ----- Subject -----
        subject = Label(frame4, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)
        sub1 = Label(frame4, text="Advanced Calculus and linear algebra",
font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=300)
        sub2 = Label(frame4, text="Engineering Chemistry", font=("times new
roman", 15, "bold"), fg="black"
).place(x=200, y=350)
        sub3 = Label(frame4, text="C Programming For Solving Problems",
font=("times new roman", 15, "bold"), fg="black"
).place(x=200, y=400)
        sub4 = Label(frame4, text="Basic Electronics", font=("times new
roman", 15, "bold"),
fg="black").place(x=200, y=450)
        sub5 = Label(frame4, text="Elements Of Mechanical Engineering",
font=("times new roman", 15, "bold"), fg="black"
).place(x=200, y=500)
        sub6 = Label(frame4, text="Engineering Chemistry Laboratory",
font=("times new roman", 15, "bold"), fg="black"
).place(x=200, y=550)
        sub7 = Label(frame4, text="C Programming Laboratory", font=("times
new roman", 15, "bold"),
fg="black").place(x=200, y=600)
        sub8 = Label(frame4, text="Technical English-2", font=("times new
roman", 15, "bold"), fg="black"
).place(x=200, y=650)

        # ----- Internal -----
        internal = Label(frame4, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
        self.text_int1 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int1.place(x=600, y=300, width=100)
        self.text_int2 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int2.place(x=600, y=350, width=100)
        self.text_int3 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int3.place(x=600, y=400, width=100)
        self.text_int4 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")

```

```

        self.text_int4.place(x=600, y=450, width=100)
        self.text_int5 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int5.place(x=600, y=500, width=100)
        self.text_int6 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int6.place(x=600, y=550, width=100)
        self.text_int7 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int7.place(x=600, y=600, width=100)
        self.text_int8 = Entry(frame4, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int8.place(x=600, y=650, width=100)
        # ----- External -----
        external = Label(frame4, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
        self.txt_ext1 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext1.place(x=800, y=300, width=100)
        self.txt_ext2 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext2.place(x=800, y=350, width=100)
        self.txt_ext3 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext3.place(x=800, y=400, width=100)
        self.txt_ext4 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext4.place(x=800, y=450, width=100)
        self.txt_ext5 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext5.place(x=800, y=500, width=100)
        self.txt_ext6 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext6.place(x=800, y=550, width=100)
        self.txt_ext7 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext7.place(x=800, y=600, width=100)
        self.txt_ext8 = Entry(frame4, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext8.place(x=800, y=650, width=100)

        # ----- Submit -----
        btn_submit = Button(frame4, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
                           command=lambda:[self.sem2(),
self.ext_sem2()]).place(x=700, y=700, width=150)

        frame4.mainloop()

# ----- 3rd sem -----#
def three_sem(self):
    frame5 = Tk()
    frame5.title("sem-3")
    frame5.geometry("1550x800")

    # ----- title -----
    title = Label(frame5, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame5, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)

```

```

self.text_name = Entry(frame5, font=("times new roman", 15,
"bold"), fg="blue")
self.text_name.place(x=150, y=100, width=250)

# ----- USN -----
usn = Label(frame5, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
self.text_usn = Entry(frame5, font=("times new roman", 15, "bold"),
fg="blue")
self.text_usn.place(x=150, y=150, width=250)

sem = Label(frame5, text="3 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=
CENTER).place(x=600, y=200)

# -----sub code -----
code = Label(frame5, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
code1 = Label(frame5, text="18MAT31", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
code2 = Label(frame5, text="18EC32", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
code3 = Label(frame5, text="18EC33", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
code4 = Label(frame5, text="18EC34", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
code5 = Label(frame5, text="18EC35", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)
code6 = Label(frame5, text="18EC36", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
code7 = Label(frame5, text="18ECL37", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
code8 = Label(frame5, text="18ECL38", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)
code9 = Label(frame5, text="18CPC8", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=700)

# ----- Subject -----
subject = Label(frame5, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)
sub1 = Label(frame5, text="Transform Calculus,Fourier Series &\n
Numerical Techniques", font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=300)
sub2 = Label(frame5, text="Network Theory", font=("times new
roman", 15, "bold"), fg="black"
).place(x=200, y=350)
sub3 = Label(frame5, text="Electronic Devices", font=("times new
roman", 15, "bold"), fg="black"
).place(x=200, y=400)
sub4 = Label(frame5, text="Digital System Design", font=("times new
roman", 15, "bold"),
fg="black").place(x=200, y=450)
sub5 = Label(frame5, text="Computer Organization & Architecture",
font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=500)
sub6 = Label(frame5, text="Power Electronics & Instrumentation",
font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=550)
sub7 = Label(frame5, text="Electronics Devices & Instrumentation
Lab", font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=600)
sub8 = Label(frame5, text="Digital System Design Lab", font=("times
new roman", 15, "bold"), fg="black"
).place(x=200, y=650)

```

```

sub9 = Label(frame5, text="Const. Of India Prof. Ethics & Cyber
Law", font=("times new roamn", 15, "bold"),
fg="black").place(x=200, y=700)

# ----- Internal -----
internal = Label(frame5, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
self.text_int1 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int1.place(x=600, y=300, width=100)
self.text_int2 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int2.place(x=600, y=350, width=100)
self.text_int3 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int3.place(x=600, y=400, width=100)
self.text_int4 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int4.place(x=600, y=450, width=100)
self.text_int5 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int5.place(x=600, y=500, width=100)
self.text_int6 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int6.place(x=600, y=550, width=100)
self.text_int7 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int7.place(x=600, y=600, width=100)
self.text_int8 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int8.place(x=600, y=650, width=100)
self.text_int9 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
self.text_int9.place(x=600, y=700, width=100)

# ----- External -----
external = Label(frame5, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
self.txt_ext1 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext1.place(x=800, y=300, width=100)
self.txt_ext2 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext2.place(x=800, y=350, width=100)
self.txt_ext3 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext3.place(x=800, y=400, width=100)
self.txt_ext4 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext4.place(x=800, y=450, width=100)
self.txt_ext5 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext5.place(x=800, y=500, width=100)
self.txt_ext6 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext6.place(x=800, y=550, width=100)
self.txt_ext7 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext7.place(x=800, y=600, width=100)
self.txt_ext8 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext8.place(x=800, y=650, width=100)
self.txt_ext9 = Entry(frame5, font=("times new roman", 15, "bold"),

```

```

fg="black")
    self.txt_ext9.place(x=800, y=700, width=100)

    # ----- Submit -----
    btn_submit = Button(frame5, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
                        command=lambda:[self.sem3(),
self.ext_sem3()]).place(x=700, y=750, width=100)

    frame5.mainloop()

# ----- 4rd sem -----#
def four_sem(self):
    frame5 = Tk()
    frame5.title("sem-4")
    frame5.geometry("1550x800")

    # ----- title -----
    title = Label(frame5, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame5, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
    self.text_name = Entry(frame5, font=("times new roman", 15,
"bold"), fg="blue")
    self.text_name.place(x=150, y=100, width=250)

    # ----- USN -----
    usn = Label(frame5, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
    self.text_usn = Entry(frame5, font=("times new roman", 15, "bold"),
fg="blue")
    self.text_usn.place(x=150, y=150, width=250)

    sem = Label(frame5, text="4 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=
        CENTER).place(x=600, y=200)

    # ----- sub code -----
    code = Label(frame5, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
    code1 = Label(frame5, text="18MAT41", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
    code2 = Label(frame5, text="18EC42", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
    code3 = Label(frame5, text="18EC43", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
    code4 = Label(frame5, text="18EC44", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
    code5 = Label(frame5, text="18EC45", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)
    code6 = Label(frame5, text="18EC46", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
    code7 = Label(frame5, text="18ECL47", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
    code8 = Label(frame5, text="18ECL48", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)
    code9 = Label(frame5, text="18CPC49", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=700)

    # ----- Subject -----
    subject = Label(frame5, text="Subjects", font=("times new roman",

```

```

15, "bold"), fg="black").place(x=200, y=250)
    sub1 = Label(frame5, text="Complex Analysis, Probability and\n
Statistical Methods", font=("times new roman", 15, "bold"),
    fg="black").place(x=200, y=300)
    sub2 = Label(frame5, text="Analog Circuits", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=350)
    sub3 = Label(frame5, text="Control Systems", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=400)
    sub4 = Label(frame5, text="Engineering Statistics & Linear
Algebra", font=("times new roman", 15, "bold"),
    fg="black").place(x=200, y=450)
    sub5 = Label(frame5, text="Signals and Systems", font=("times new
roman", 15, "bold"),
    fg="black").place(x=200, y=500)
    sub6 = Label(frame5, text="MicroController", font=("times new
roman", 15, "bold"),
    fg="black").place(x=200, y=550)
    sub7 = Label(frame5, text="MicroController Lab", font=("times new
roman", 15, "bold"),
    fg="black").place(x=200, y=600)
    sub8 = Label(frame5, text="Analog Circuits Lab", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=650)
    sub9 = Label(frame5, text="Kannada", font=("times new roamn", 15,
"bold"),
    fg="black").place(x=200, y=700)

    # ----- Internal -----
    internal = Label(frame5, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
    self.text_int1 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int1.place(x=600, y=300, width=100)
    self.text_int2 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int2.place(x=600, y=350, width=100)
    self.text_int3 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int3.place(x=600, y=400, width=100)
    self.text_int4 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int4.place(x=600, y=450, width=100)
    self.text_int5 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int5.place(x=600, y=500, width=100)
    self.text_int6 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int6.place(x=600, y=550, width=100)
    self.text_int7 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int7.place(x=600, y=600, width=100)
    self.text_int8 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int8.place(x=600, y=650, width=100)
    self.text_int9 = Entry(frame5, font=("times new roman", 15,
"bold"), fg="black")
    self.text_int9.place(x=600, y=700, width=100)

    # ----- External -----
    external = Label(frame5, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)

```



```

        self.txt_ext1 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext1.place(x=800, y=300, width=100)
        self.txt_ext2 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext2.place(x=800, y=350, width=100)
        self.txt_ext3 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext3.place(x=800, y=400, width=100)
        self.txt_ext4 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext4.place(x=800, y=450, width=100)
        self.txt_ext5 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext5.place(x=800, y=500, width=100)
        self.txt_ext6 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext6.place(x=800, y=550, width=100)
        self.txt_ext7 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext7.place(x=800, y=600, width=100)
        self.txt_ext8 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext8.place(x=800, y=650, width=100)
        self.txt_ext9 = Entry(frame5, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext9.place(x=800, y=700, width=100)

        # ----- Submit -----
        btn_submit = Button(frame5, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
                           command=lambda:[self.sem4(),
self.ext_sem4()]).place(x=700, y=750, width=100)

        frame5.mainloop()

# ----- 5rd sem -----#
def five_sem(self):
    frame6 = Tk()
    frame6.title("sem-5")
    frame6.geometry("1550x800")

    # ----- title -----
    title = Label(frame6, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame6, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
    self.text_name = Entry(frame6, font=("times new roman", 15,
"bold"), fg="blue")
    self.text_name.place(x=150, y=100, width=250)

    # ----- USN -----
    usn = Label(frame6, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
    self.text_usn = Entry(frame6, font=("times new roman", 15, "bold"),
fg="blue")
    self.text_usn.place(x=150, y=150, width=250)

    sem = Label(frame6, text="5 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=

```

```

        CENTER).place(x=600, y=200)
        # -----sub code -----
        code = Label(frame6, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
        code1 = Label(frame6, text="18ES51", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
        code2 = Label(frame6, text="18EC52", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
        code3 = Label(frame6, text="18EC53", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
        code4 = Label(frame6, text="18EC54", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
        code5 = Label(frame6, text="18EC55", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)
        code6 = Label(frame6, text="18EC56", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
        code7 = Label(frame6, text="18ECL57", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
        code8 = Label(frame6, text="18ECL58", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)
        code9 = Label(frame6, text="18CIV59", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=700)

        # ----- Subject -----
        subject = Label(frame6, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)
        sub1 = Label(frame6, text="Technological Innovation Management\n&
Entrepreneurship", font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=300)
        sub2 = Label(frame6, text="Digital Signal Processing", font=("times
new roman", 15, "bold"), fg="black"
        ).place(x=200, y=350)
        sub3 = Label(frame6, text="Principles Of Communication Systems",
font=("times new roman", 15, "bold"), fg="black"
        ).place(x=200, y=400)
        sub4 = Label(frame6, text="Information Theory & Coding",
font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=450)
        sub5 = Label(frame6, text="Electronic Waves", font=("times new
roman", 15, "bold"),
fg="black").place(x=200, y=500)
        sub6 = Label(frame6, text="Verilog HDL", font=("times new roman",
15, "bold"),
fg="black").place(x=200, y=550)
        sub7 = Label(frame6, text="Digital Signal Processing Lab",
font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=600)
        sub8 = Label(frame6, text="HDL Lab", font=("times new roman", 15,
"bold"), fg="black"
        ).place(x=200, y=650)
        sub9 = Label(frame6, text="Environmental Studies", font=("times new
roamn", 15, "bold"),
fg="black").place(x=200, y=700)

        # ----- Internal -----
        internal = Label(frame6, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
        self.text_int1 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int1.place(x=600, y=300, width=100)
        self.text_int2 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int2.place(x=600, y=350, width=100)

```

```

        self.text_int3 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int3.place(x=600, y=400, width=100)
        self.text_int4 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int4.place(x=600, y=450, width=100)
        self.text_int5 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int5.place(x=600, y=500, width=100)
        self.text_int6 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int6.place(x=600, y=550, width=100)
        self.text_int7 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int7.place(x=600, y=600, width=100)
        self.text_int8 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int8.place(x=600, y=650, width=100)
        self.text_int9 = Entry(frame6, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int9.place(x=600, y=700, width=100)

        # ----- External -----
        external = Label(frame6, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
        self.txt_ext1 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext1.place(x=800, y=300, width=100)
        self.txt_ext2 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext2.place(x=800, y=350, width=100)
        self.txt_ext3 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext3.place(x=800, y=400, width=100)
        self.txt_ext4 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext4.place(x=800, y=450, width=100)
        self.txt_ext5 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext5.place(x=800, y=500, width=100)
        self.txt_ext6 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext6.place(x=800, y=550, width=100)
        self.txt_ext7 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext7.place(x=800, y=600, width=100)
        self.txt_ext8 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext8.place(x=800, y=650, width=100)
        self.txt_ext9 = Entry(frame6, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext9.place(x=800, y=700, width=100)

        # ----- Submit -----
        btn_submit = Button(frame6, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
                           command=lambda:[self.sem5(),
self.ext_sem5()]).place(x=700, y=750, width=100)

        frame6.mainloop()

        # ----- 6rd sem -----#
        def six_sem(self):

```

```

frame7 = Tk()
frame7.title("sem-6")
frame7.geometry("1550x800")

# ----- title -----
title = Label(frame7, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
    x=250, y=20)

# ----- Name -----
name = Label(frame7, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
self.text_name = Entry(frame7, font=("times new roman", 15,
"bold"), fg="blue")
self.text_name.place(x=150, y=100, width=250)

# ----- USN -----
usn = Label(frame7, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
self.text_usn = Entry(frame7, font=("times new roman", 15, "bold"),
fg="blue")
self.text_usn.place(x=150, y=150, width=250)

sem = Label(frame7, text="6 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=
    CENTER).place(x=600, y=200)

# -----sub code -----
code = Label(frame7, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
code1 = Label(frame7, text="18EC61", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
code2 = Label(frame7, text="18EC62", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
code3 = Label(frame7, text="18EC63", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
code4 = Label(frame7, text="18EC646", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
code5 = Label(frame7, text="18ME653", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)
code6 = Label(frame7, text="18ECL66", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
code7 = Label(frame7, text="18ECL67", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
code8 = Label(frame7, text="18ECMP68", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)

# ----- Subject -----
subject = Label(frame7, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)
sub1 = Label(frame7, text="Digital Communication", font=("times new
roman", 15, "bold"),
    fg="black").place(x=200, y=300)
sub2 = Label(frame7, text="Embedded Systems", font=("times new
roman", 15, "bold"), fg="black"
    ).place(x=200, y=350)
sub3 = Label(frame7, text="MicroWaves And Antenna", font=("times
new roman", 15, "bold"), fg="black"
    ).place(x=200, y=400)
sub4 = Label(frame7, text="Professional Elective-1", font=("times
new roman", 15, "bold"),
    fg="black").place(x=200, y=450)
sub5 = Label(frame7, text="Open Elective-1", font=("times new
roman", 15, "bold"),

```

```

        fg="black").place(x=200, y=500)
        sub6 = Label(frame7, text="Embedded Systems Lab", font=("times new
roman", 15, "bold"),
        fg="black").place(x=200, y=550)
        sub7 = Label(frame7, text="Communication Lab", font=("times new
roman", 15, "bold"),
        fg="black").place(x=200, y=600)
        sub8 = Label(frame7, text="Mini-Project", font=("times new roman",
15, "bold"), fg="black"
        ).place(x=200, y=650)

        # ----- Internal -----
        internal = Label(frame7, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
        self.text_int1 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int1.place(x=600, y=300, width=100)
        self.text_int2 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int2.place(x=600, y=350, width=100)
        self.text_int3 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int3.place(x=600, y=400, width=100)
        self.text_int4 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int4.place(x=600, y=450, width=100)
        self.text_int5 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int5.place(x=600, y=500, width=100)
        self.text_int6 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int6.place(x=600, y=550, width=100)
        self.text_int7 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int7.place(x=600, y=600, width=100)
        self.text_int8 = Entry(frame7, font=("times new roman", 15,
"bold"), fg="black")
        self.text_int8.place(x=600, y=650, width=100)

        # ----- External -----
        external = Label(frame7, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
        self.txt_ext1 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext1.place(x=800, y=300, width=100)
        self.txt_ext2 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext2.place(x=800, y=350, width=100)
        self.txt_ext3 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext3.place(x=800, y=400, width=100)
        self.txt_ext4 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext4.place(x=800, y=450, width=100)
        self.txt_ext5 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext5.place(x=800, y=500, width=100)
        self.txt_ext6 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext6.place(x=800, y=550, width=100)
        self.txt_ext7 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext7.place(x=800, y=600, width=100)

```

```

        self.txt_ext8 = Entry(frame7, font=("times new roman", 15, "bold"),
fg="black")
        self.txt_ext8.place(x=800, y=650, width=100)

        # ----- Submit -----
        btn_submit = Button(frame7, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
            command=lambda:[self.sem6(),
self.ext_sem6()]).place(x=700, y=700, width=150)

        frame7.mainloop()

# ----- 7rd sem -----#
def seven_sem(self):
    frame8 = Tk()
    frame8.title("sem-7")
    frame8.geometry("1550x800")

    # ----- title -----
    title = Label(frame8, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame8, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
    self.text_name = Entry(frame8, font=("times new roman", 15,
"bold"), fg="blue")
    self.text_name.place(x=150, y=100, width=250)

    # ----- USN -----
    usn = Label(frame8, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
    self.text_usn = Entry(frame8, font=("times new roman", 15, "bold"),
fg="blue")
    self.text_usn.place(x=150, y=150, width=250)

    sem = Label(frame8, text="7 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=
        CENTER).place(x=600, y=200)
    # ----- sub code -----
    code = Label(frame8, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
    code1 = Label(frame8, text="18EC71", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
    code2 = Label(frame8, text="18EC72", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
    code3 = Label(frame8, text="18EC73X", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)
    code4 = Label(frame8, text="18EC74X", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
    code5 = Label(frame8, text="18XX75X", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)
    code6 = Label(frame8, text="18ECL76", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=550)
    code7 = Label(frame8, text="18ECL77", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=600)
    code8 = Label(frame8, text="18ECP78", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=650)

    # ----- Subject -----
    subject = Label(frame8, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)

```

```

        sub1 = Label(frame8, text="Computer Networks", font=("times new
roman", 15, "bold"),
                    fg="black").place(x=200, y=300)
        sub2 = Label(frame8, text="VLSI Design", font=("times new roman",
15, "bold"), fg="black"
                    ).place(x=200, y=350)
        sub3 = Label(frame8, text="Professional Elective-2", font=("times
new roman", 15, "bold"), fg="black"
                    ).place(x=200, y=400)
        sub4 = Label(frame8, text="Professional Elective-3", font=("times
new roman", 15, "bold"),
                    fg="black").place(x=200, y=450)
        sub5 = Label(frame8, text="Open Elective-B", font=("times new
roman", 15, "bold"),
                    fg="black").place(x=200, y=500)
        sub6 = Label(frame8, text="Computer Networks Lab", font=("times new
roman", 15, "bold"),
                    fg="black").place(x=200, y=550)
        sub7 = Label(frame8, text="VLSI Lab", font=("times new roman", 15,
"bold"),
                    fg="black").place(x=200, y=600)
        sub8 = Label(frame8, text="Project Work Phase-1", font=("times new
roman", 15, "bold"), fg="black"
                    ).place(x=200, y=650)

# ----- Internal -----
internal = Label(frame8, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
self.text_int1 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int1.place(x=600, y=300, width=100)
self.text_int2 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int2.place(x=600, y=350, width=100)
self.text_int3 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int3.place(x=600, y=400, width=100)
self.text_int4 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int4.place(x=600, y=450, width=100)
self.text_int5 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int5.place(x=600, y=500, width=100)
self.text_int6 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int6.place(x=600, y=550, width=100)
self.text_int7 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int7.place(x=600, y=600, width=100)
self.text_int8 = Entry(frame8, font=("times new roman", 15,
"bold"), fg="black")
self.text_int8.place(x=600, y=650, width=100)

# ----- External -----
external = Label(frame8, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
self.txt_ext1 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext1.place(x=800, y=300, width=100)
self.txt_ext2 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext2.place(x=800, y=350, width=100)
self.txt_ext3 = Entry(frame8, font=("times new roman", 15, "bold"),

```

```

fg="black")
    self.txt_ext3.place(x=800, y=400, width=100)
    self.txt_ext4 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
    self.txt_ext4.place(x=800, y=450, width=100)
    self.txt_ext5 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
    self.txt_ext5.place(x=800, y=500, width=100)
    self.txt_ext6 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
    self.txt_ext6.place(x=800, y=550, width=100)
    self.txt_ext7 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
    self.txt_ext7.place(x=800, y=600, width=100)
    self.txt_ext8 = Entry(frame8, font=("times new roman", 15, "bold"),
fg="black")
    self.txt_ext8.place(x=800, y=650, width=100)

    # ----- Submit -----
    btn_submit = Button(frame8, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
                        command=lambda:[self.sem7(),
self.ext_sem7()] ).place(x=700, y=700, width=150)

    frame8.mainloop()

# ----- 8rd sem -----#
def eight_sem(self):
    frame9 = Tk()
    frame9.title("sem-8")
    frame9.geometry("1550x800")

    # ----- title -----
    title = Label(frame9, text="VISVESVARAYA TECHNOLOGICAL UNIVERSITY",
font=("times new roman", 30, "bold")).place(
        x=250, y=20)

    # ----- Name -----
    name = Label(frame9, text="Name :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=100)
    self.text_name = Entry(frame9, font=("times new roman", 15,
"bold"), fg="blue")
    self.text_name.place(x=150, y=100, width=250)

    # ----- USN -----
    usn = Label(frame9, text="USN :", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=150)
    self.text_usn = Entry(frame9, font=("times new roman", 15, "bold"),
fg="blue")
    self.text_usn.place(x=150, y=150, width=250)

    sem = Label(frame9, text="8 Semester", font=("times new roman", 15,
"bold"), fg="white", bg="black", justify=
        CENTER).place(x=600, y=200)

    # ----- sub code -----
    code = Label(frame9, text="Code", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=250)
    code1 = Label(frame9, text="18EC81", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=300)
    code2 = Label(frame9, text="18XX82X", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=350)
    code3 = Label(frame9, text="18ECP83", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=400)

```



```

code4 = Label(frame9, text="18ECS84", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=450)
code5 = Label(frame9, text="18EC185", font=("times new roman", 15,
"bold"), fg="black").place(x=50, y=500)

# ----- Subject -----
subject = Label(frame9, text="Subjects", font=("times new roman",
15, "bold"), fg="black").place(x=200, y=250)
sub1 = Label(frame9, text="Wireless and Cellular Communication",
font=("times new roman", 15, "bold"),
fg="black").place(x=200, y=300)
sub2 = Label(frame9, text="Professional Elective-4", font=("times
new roman", 15, "bold"), fg="black"
).place(x=200, y=350)
sub3 = Label(frame9, text="Project Work-2", font=("times new
roman", 15, "bold"), fg="black"
).place(x=200, y=400)
sub4 = Label(frame9, text="Technical Seminar", font=("times new
roman", 15, "bold"),
fg="black").place(x=200, y=450)
sub5 = Label(frame9, text="Internship-3", font=("times new roman",
15, "bold"),
fg="black").place(x=200, y=500)

# ----- Internal -----
internal = Label(frame9, text="Internal", font=("time new roman",
15, "bold"), fg="black").place(x=600, y=250)
self.text_int1 = Entry(frame9, font=("times new roman", 15,
"bold"), fg="black")
self.text_int1.place(x=600, y=300, width=100)
self.text_int2 = Entry(frame9, font=("times new roman", 15,
"bold"), fg="black")
self.text_int2.place(x=600, y=350, width=100)
self.text_int3 = Entry(frame9, font=("times new roman", 15,
"bold"), fg="black")
self.text_int3.place(x=600, y=400, width=100)
self.text_int4 = Entry(frame9, font=("times new roman", 15,
"bold"), fg="black")
self.text_int4.place(x=600, y=450, width=100)
self.text_int5 = Entry(frame9, font=("times new roman", 15,
"bold"), fg="black")
self.text_int5.place(x=600, y=500, width=100)

# ----- External -----
external = Label(frame9, text="External", font=("times new roman",
15, "bold"), fg="black").place(x=800, y=250)
self.txt_ext1 = Entry(frame9, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext1.place(x=800, y=300, width=100)
self.txt_ext2 = Entry(frame9, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext2.place(x=800, y=350, width=100)
self.txt_ext3 = Entry(frame9, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext3.place(x=800, y=400, width=100)
self.txt_ext4 = Entry(frame9, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext4.place(x=800, y=450, width=100)
self.txt_ext5 = Entry(frame9, font=("times new roman", 15, "bold"),
fg="black")
self.txt_ext5.place(x=800, y=500, width=100)

# ----- Submit -----

```

```

        btn_submit = Button(frame9, text="Submit", font=("times new roman",
15, "bold"), fg="white", bg="blue",
                                command=lambda:[self.sem8(),
self.ext_sem8()]).place(x=700, y=700, width=150)

        frame9.mainloop()

# ----- Connecting to database -----

def register_data(self):
    if self.txt_first.get() == "" or self.txt_last.get() == "" or
self.txt_contact.get() == "" or \
        self.txt_email.get() == "" or self.cmb_gender.get() == ""
or self.txt_dob.get() == "" or \
        self.txt_ten.get() == "" or self.txt_puc.get() == "" or \
        self.cmb_branch.get() == "" or self.txt_address.get() ==
"":
        messagebox.showerror("Error", "All details should be fill",
parent=self.root)
    elif self.var_chk.get() == 0:
        messagebox.showerror("Error", "Please agree terms and
conditions", parent=self.root)
    else:
        try:
            con = pymysql.connect(host="localhost", user="root",
password="", database="details")
            cur = con.cursor()
            cur.execute("select * from details where email=%s",
self.txt_email.get())
            row = cur.fetchone()
            # print(row)
            if row != None:
                messagebox.showerror("Error", "User already exist.
Please try with other", parent=self.root)
            else:
                cur.execute(
                    "insert into details (first_name, last_name,
contact, email, dob, gender, ten_per, puc_per, address, branch) values(%s,
%s, %s, %s, %s, %s, %s, %s, %s, %s)",
                    (self.txt_first.get(),
                    self.txt_last.get(),
                    self.txt_contact.get(),
                    self.txt_email.get(),
                    self.txt_dob.get(),
                    self.cmb_gender.get(),
                    self.txt_ten.get(),
                    self.txt_puc.get(),
                    self.txt_address.get(),
                    self.cmb_branch.get()
                    ))

                con.commit()
                con.close()
                messagebox.showinfo("Success", "You are successfully
registered")

        except Exception as es:
            messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

# -----Sem-1-----
def sem1(self):

```

```

        if self.text_name.get()==" " or self.text_usn.get()==" " or
self.text_int1.get()==" " or self.text_int2.get()==" " or \
        self.text_int3.get()==" " or self.text_int4.get()==" " or
self.text_int5.get()==" " or self.text_int6.get()==" " or \
        self.text_int7.get()==" " or self.text_int8.get()==" ":
            messagebox.showerror("Error", "Must give all internal and
external marks")
        else:
            try:
                con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-1")
                cur1 = con1.cursor()
                cur1.execute("select * from sem1 where USN=%s",
self.text_usn.get())
                row = cur1.fetchone()
                cur1.execute(
                    "insert into sem1(Name, USN, Maths1,
Engineering_Physics, Basic_Electrical_Engineering,
Elements_of_Civil_Engg_and_Mech, Engineering_Graphics,
Engineering_Physics_Lab, Basic_Electrical_Engg_Lab,Technical_English1)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
                    (self.text_name.get(),
                    self.text_usn.get(),
                    self.text_int1.get(),
                    self.text_int2.get(),
                    self.text_int3.get(),
                    self.text_int4.get(),
                    self.text_int5.get(),
                    self.text_int6.get(),
                    self.text_int7.get(),
                    self.text_int8.get()
                    ))

                con1.commit()
                con1.close()
                messagebox.showinfo("Success", "You are successfully
registered")
            except Exception as es:
                messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

        def ext_sem1(self):
            con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-1")
            cur1 = con1.cursor()
            cur1.execute(
                "insert into ext_sem1(Name, USN, Maths1, Engineering_Physics,
Basic_Electrical_Engineering, Elements_of_Civil_Engg_and_Mech,
Engineering_Graphics, Engineering_Physics_Lab,
Basic_Electrical_Engg_Lab,Technical_English1) values(%s, %s, %s, %s, %s,
%s, %s, %s, %s, %s)",
                (self.text_name.get(),
                self.text_usn.get(),
                self.txt_ext1.get(),
                self.txt_ext2.get(),
                self.txt_ext3.get(),
                self.txt_ext4.get(),
                self.txt_ext5.get(),
                self.txt_ext6.get(),
                self.txt_ext7.get(),
                self.txt_ext8.get()
                ))

```

```

        con1.commit()
        con1.close()

# ----- sem-2 -----
def sem2(self):
    if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
        self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "" or self.text_int6.get() == "" or \
        self.text_int7.get() == "" or self.text_int8.get() == "":
        messagebox.showerror("Error", "Must give all internal and
external marks")
    else:
        try:
            con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-2")
            cur1 = con1.cursor()
            cur1.execute("select * from int_sem2 where USN=%s",
self.text_usn.get())
            row = cur1.fetchone()
            cur1.execute(
                "insert into int_sem2(Name, USN, Maths2,
Engineering_Chemistry, C_Programming_for_Solving_Problems,
Basic_Electronics, Elements_of_Mechanical_Engineering,
Engineering_Chemistry_Lab, C_Programming_Lab, Technical_English2)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
                (self.text_name.get(),
                self.text_usn.get(),
                self.text_int1.get(),
                self.text_int2.get(),
                self.text_int3.get(),
                self.text_int4.get(),
                self.text_int5.get(),
                self.text_int6.get(),
                self.text_int7.get(),
                self.text_int8.get()
                ))

            con1.commit()
            con1.close()
            messagebox.showinfo("Success", "You are successfully
registered")
        except Exception as es:
            messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

def ext_sem2(self):
    con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-2")
    cur1 = con1.cursor()
    cur1.execute(
        "insert into ext_sem2(Name, USN, Maths2, Engineering_Chemistry,
C_Programming_for_Solving_Problems, Basic_Electronics,
Elements_of_Mechanical_Engineering, Engineering_Chemistry_Lab,
C_Programming_Lab, Technical_English2) values(%s, %s, %s, %s, %s, %s, %s,
%s, %s, %s)",
        (self.text_name.get(),
        self.text_usn.get(),
        self.txt_ext1.get(),
        self.txt_ext2.get(),
        self.txt_ext3.get(),
        self.txt_ext4.get(),
        self.txt_ext5.get(),

```

```

        self.txt_ext6.get(),
        self.txt_ext7.get(),
        self.txt_ext8.get()
    ))

    con1.commit()
    con1.close()

    # ----- sem-3 -----
    def sem3(self):
        if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
        self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "" or self.text_int6.get() == "" or \
        self.text_int7.get() == "" or self.text_int8.get() == "" or
self.text_int9.get() == "":
            messagebox.showerror("Error", "Must give all internal and
external marks")
        else:
            try:
                con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-3")
                cur1 = con1.cursor()
                cur1.execute("select * from int_sem3 where USN=%s",
self.text_usn.get())
                row = cur1.fetchone()
                cur1.execute(
                    "insert into int_sem3(Name, USN, Maths3,
Network_Theory, Electronic_Devices, Digital_System_Design,
Computer_Organization_and_Architecture,
Power_Electronic_and_Instrumentation,
Electronic_Devices_and_Instrumentation_Lab, Digital_System_Design_Lab,
Cont_of_Inia_Prof_Ethics_and_Cyber_law) values(%s, %s, %s, %s, %s, %s, %s,
%s, %s, %s, %s)",
                    (self.text_name.get(),
                    self.text_usn.get(),
                    self.text_int1.get(),
                    self.text_int2.get(),
                    self.text_int3.get(),
                    self.text_int4.get(),
                    self.text_int5.get(),
                    self.text_int6.get(),
                    self.text_int7.get(),
                    self.text_int8.get(),
                    self.text_int9.get()
                    ))

                con1.commit()
                con1.close()
                messagebox.showinfo("Success", "You are successfully
registered")
            except Exception as es:
                messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

    def ext_sem3(self):
        con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-3")
        cur1 = con1.cursor()
        cur1.execute(
            "insert into ext_sem3(Name, USN, Maths3, Network_Theory,
Electronic_Devices, Digital_System_Design,
Computer_Organization_and_Architecture,

```

```

Power_Electronic_and_Instrumentation,
Electronic_Devices_and_Instrumentation_Lab, Digital_System_Design_Lab,
Cont_of_Inia_Prof_Ethics_and_Cyber_law) values(%s, %s, %s, %s, %s, %s, %s,
%s, %s, %s, %s)",
        (self.text_name.get(),
        self.text_usn.get(),
        self.txt_ext1.get(),
        self.txt_ext2.get(),
        self.txt_ext3.get(),
        self.txt_ext4.get(),
        self.txt_ext5.get(),
        self.txt_ext6.get(),
        self.txt_ext7.get(),
        self.txt_ext8.get(),
        self.txt_ext9.get()
        ))

        con1.commit()
        con1.close()

        # -----4th sem -----
        -
        def sem4(self):
            if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
                self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "" or self.text_int6.get() == "" or \
                self.text_int7.get() == "" or self.text_int8.get() == "" or
self.text_int9.get() == "":
                messagebox.showerror("Error", "Must give all internal and
external marks")
            else:
                try:
                    con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-4")
                    cur1 = con1.cursor()
                    cur1.execute("select * from int_sem4 where USN=%s",
self.text_usn.get())
                    row = cur1.fetchone()
                    cur1.execute(
                        "insert into int_sem4 (Name, USN, Maths4,
Analog_Circuits, Control_Systems,
Engineering_Statistics_and_Linear_Algebra, Signals_and_Systems,
MicroControllers, MicroControllers_Lab, Analog_Circuits_Lab, Kannada)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
                        (self.text_name.get(),
                        self.text_usn.get(),
                        self.text_int1.get(),
                        self.text_int2.get(),
                        self.text_int3.get(),
                        self.text_int4.get(),
                        self.text_int5.get(),
                        self.text_int6.get(),
                        self.text_int7.get(),
                        self.text_int8.get(),
                        self.text_int9.get()
                        ))

                    con1.commit()
                    con1.close()
                    messagebox.showinfo("Success", "You are successfully
registered")
                except Exception as es:

```

```

        messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

    def ext_sem4(self):
        con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-4")
        cur1 = con1.cursor()
        cur1.execute(
            "insert into ext_sem4(Name, USN, Maths4, Analog_Circuits,
Control_Systems, Engineering_Statistics_and_Linear_Algebra,
Signals_and_Systems, MicroControllers, MicroControllers_Lab,
Analog_Circuits_Lab, Kannada) values(%s, %s, %s, %s, %s, %s, %s, %s, %s,
%s, %s)",
            (self.text_name.get(),
            self.text_usn.get(),
            self.txt_ext1.get(),
            self.txt_ext2.get(),
            self.txt_ext3.get(),
            self.txt_ext4.get(),
            self.txt_ext5.get(),
            self.txt_ext6.get(),
            self.txt_ext7.get(),
            self.txt_ext8.get(),
            self.txt_ext9.get()
            ))

        con1.commit()
        con1.close()

# ----- sem-5 -----
def sem5(self):
    if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
        self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "" or self.text_int6.get() == "" or \
        self.text_int7.get() == "" or self.text_int8.get() == "" or
self.text_int9.get() == "":
        messagebox.showerror("Error", "Must give all internal and
external marks")
    else:
        try:
            con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-5")
            cur1 = con1.cursor()
            cur1.execute("select * from int_sem5 where USN=%s",
self.text_usn.get())
            row = cur1.fetchone()
            cur1.execute(
                "insert into int_sem5(Name, USN, TIME,
Digital_Signal_Processing, Principles_of_Communication_Systems,
Information_Theory_and_encoding, Electro_Maganetic_Waves, Verilog_HDL,
Digital_Signal_Processing_Lab, HDL_Lab, Environmental_Studies) values(%s,
%s, %s, %s, %s, %s, %s, %s, %s, %s)",
                (self.text_name.get(),
                self.text_usn.get(),
                self.text_int1.get(),
                self.text_int2.get(),
                self.text_int3.get(),
                self.text_int4.get(),
                self.text_int5.get(),
                self.text_int6.get(),
                self.text_int7.get(),
                self.text_int8.get(),
                self.text_int9.get(),
                self.text_ext1.get(),
                self.text_ext2.get(),
                self.text_ext3.get(),
                self.text_ext4.get(),
                self.text_ext5.get(),
                self.text_ext6.get(),
                self.text_ext7.get(),
                self.text_ext8.get(),
                self.text_ext9.get()
            ))
            con1.commit()
            con1.close()

```

```

        self.text_int8.get(),
        self.text_int9.get()
    ))

    con1.commit()
    con1.close()
    messagebox.showinfo("Success", "You are successfully
registered")
    except Exception as es:
        messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

    def ext_sem5(self):
        con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-5")
        cur1 = con1.cursor()
        cur1.execute(
            "insert into ext_sem5(Name, USN, TIME,
Digital_Signal_Processing, Principle_of_Communication_Systems,
Information_Theory_and_coding, EletroMagnetic_Waves, Verilog_HDL,
Digital_Signal_Processing_Lab, HDL_Lab, Environmental_Studies) values(%s,
%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
            (self.text_name.get(),
            self.text_usn.get(),
            self.txt_ext1.get(),
            self.txt_ext2.get(),
            self.txt_ext3.get(),
            self.txt_ext4.get(),
            self.txt_ext5.get(),
            self.txt_ext6.get(),
            self.txt_ext7.get(),
            self.txt_ext8.get(),
            self.txt_ext9.get()
            ))

        con1.commit()
        con1.close()

# ----- sem-6 -----
def sem6(self):
    if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
        self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "" or self.text_int6.get() == "" or \
        self.text_int7.get() == "" or self.text_int8.get() == "":
        messagebox.showerror("Error", "Must give all internal and
external marks")
    else:
        try:
            con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-6")
            cur1 = con1.cursor()
            cur1.execute("select * from int_sem6 where USN=%s",
self.text_usn.get())
            row = cur1.fetchone()
            cur1.execute(
                "insert into int_sem6(Name, USN, Digital_Communication,
Embedded_Systems, Microwaves_and_Antenna, Professional_Elective_1,
Open_Elective_1, Embedded_Systems_Lab, Communication_Lab, Mini_Project)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
                (self.text_name.get(),
                self.text_usn.get(),

```



```

        self.text_int1.get(),
        self.text_int2.get(),
        self.text_int3.get(),
        self.text_int4.get(),
        self.text_int5.get(),
        self.text_int6.get(),
        self.text_int7.get(),
        self.text_int8.get()
    ))

    con1.commit()
    con1.close()
    messagebox.showinfo("Success", "You are successfully
registered")
    except Exception as es:
        messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

    def ext_sem6(self):
        con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-6")
        cur1 = con1.cursor()
        cur1.execute(
            "insert into ext_sem6(Name, USN, Digital_Communication,
Embedded_Systems, Microwaves_and_Antenna, Professional_Elective_1,
Open_Elective_1, Embedded_Systems_Lab, Communication_Lab, Mini_Project)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
            (self.text_name.get(),
            self.text_usn.get(),
            self.txt_ext1.get(),
            self.txt_ext2.get(),
            self.txt_ext3.get(),
            self.txt_ext4.get(),
            self.txt_ext5.get(),
            self.txt_ext6.get(),
            self.txt_ext7.get(),
            self.txt_ext8.get()
            ))

        con1.commit()
        con1.close()

# ----- sem-7 -----
def sem7(self):
    if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
        self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "" or self.text_int6.get() == "" or \
        self.text_int7.get() == "" or self.text_int8.get() == "":
        messagebox.showerror("Error", "Must give all internal and
external marks")
    else:
        try:
            con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-7")
            cur1 = con1.cursor()
            cur1.execute("select * from int_sem7 where USN=%s",
self.text_usn.get())
            row = cur1.fetchone()
            cur1.execute(
                "insert into int_sem7(Name, USN, Computer_Networks,
VLSI_Design, Professional_Elective_2, Professional_Elective_3,

```

```

Open_Elective_B, Computer_Networks_Lab, VLSI_Lab, Project_Work_Phase1)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
        (self.text_name.get(),
         self.text_usn.get(),
         self.text_int1.get(),
         self.text_int2.get(),
         self.text_int3.get(),
         self.text_int4.get(),
         self.text_int5.get(),
         self.text_int6.get(),
         self.text_int7.get(),
         self.text_int8.get()
        ))

        con1.commit()
        con1.close()
        messagebox.showinfo("Success", "You are successfully
registered")
    except Exception as es:
        messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

    def ext_sem7(self):
        con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-7")
        cur1 = con1.cursor()
        cur1.execute(
            "insert into ext_sem7(Name, USN, Computer_Networks,
VLSI_Design, Professional_Elective_2, Professional_Elective_3,
Open_Elective_B, Computer_Networks_Lab, VLSI_Lab, Project_Work_Phase1)
values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
            (self.text_name.get(),
             self.text_usn.get(),
             self.txt_ext1.get(),
             self.txt_ext2.get(),
             self.txt_ext3.get(),
             self.txt_ext4.get(),
             self.txt_ext5.get(),
             self.txt_ext6.get(),
             self.txt_ext7.get(),
             self.txt_ext8.get()
            ))

        con1.commit()
        con1.close()

# ----- sem-8 -----
--
    def sem8(self):
        if self.text_name.get() == "" or self.text_usn.get() == "" or
self.text_int1.get() == "" or self.text_int2.get() == "" or \
        self.text_int3.get() == "" or self.text_int4.get() == "" or
self.text_int5.get() == "":
            messagebox.showerror("Error", "Must give all internal and
external marks")
        else:
            try:
                con1 = pymysql.connect(host="localhost", user="root",
password="", database="sem-8")
                cur1 = con1.cursor()
                cur1.execute("select * from int_sem8 where USN=%s",
self.text_usn.get())
                row = cur1.fetchone()

```

```

        cur1.execute(
            "insert into int_sem8 (Name, USN,
Wireless_and_Cellular_Communication, Professional_Elective_4,
Project_Work_2, Technical_Seminar, Internship_3) values(%s, %s, %s, %s, %s,
%s, %s)",
            (self.text_name.get(),
            self.text_usn.get(),
            self.text_int1.get(),
            self.text_int2.get(),
            self.text_int3.get(),
            self.text_int4.get(),
            self.text_int5.get()
            ))

        con1.commit()
        con1.close()
        messagebox.showinfo("Success", "You are successfully
registered")
    except Exception as es:
        messagebox.showinfo("Error", f"Error due to {str(es)}",
parent=self.root)

    def ext_sem8(self):
        con1 = pymysql.connect(host="localhost", user="root", password="",
database="sem-8")
        cur1 = con1.cursor()
        cur1.execute(
            "insert into ext_sem8 (Name, USN,
Wireless_and_Cellular_Communication, Professional_Elective_4,
Project_Work_2, Technical_Seminar, Internship_3) values(%s, %s, %s, %s, %s,
%s, %s)",
            (self.text_name.get(),
            self.text_usn.get(),
            self.txt_ext1.get(),
            self.txt_ext2.get(),
            self.txt_ext3.get(),
            self.txt_ext4.get(),
            self.txt_ext5.get()
            ))

        con1.commit()
        con1.close()

    def info(self):
        wb.open("https://www.careers360.com/colleges/rl-jalappa-institute-
of-technology-bangalore")

    def clear_data(self):
        self.txt_first.delete(0, END)
        self.txt_last.delete(0, END)
        self.txt_contact.delete(0, END)
        self.txt_email.delete(0, END)
        self.txt_dob.delete(0, END)
        self.cmb_gender.delete(0, END)
        self.txt_address.delete(0, END)
        self.cmb_branch.delete(0, END)
        self.txt_ten.delete(0, END)
        self.txt_puc.delete(0, END)
        messagebox.showinfo("Information", "Data cleared successfully")

root = Tk()

```

```
obj = Register(root)
root.mainloop()
```

5.5. EXPECTED OUTCOME

SNAPSHOTS

The screenshot shows a web browser window displaying the 'Registration Form' page. The page has a dark blue header and a light blue sidebar on the left. The main content area is white and contains a registration form titled 'REGISTER HERE' in green. The form includes fields for First name, Last name, Contact No, Email ID, Date Of Birth, Gender, 10% / SGPA, PUC %, Address, and Branch. There are also checkboxes for 'I agree the terms and conditions' and buttons for 'Register' (green) and 'Clear' (blue). The sidebar on the left features the RLJIT logo, the text 'Estd:2001', and links for 'Info', 'Click to check details of this college', 'Click to check the previous marks', and 'Marks Check'.

Registration Form

StudentRegistrationFo... Synopsis Presentation... Batch12 - Word block diagram 1.mp4 -... StudentRegistrationFo... Registration Form

ENG 07:47

REGISTER HERE

First name

Last name

Contact No

Email ID

Date Of Birth

Gender


10% / SGPA

PUC %

Address

Branch

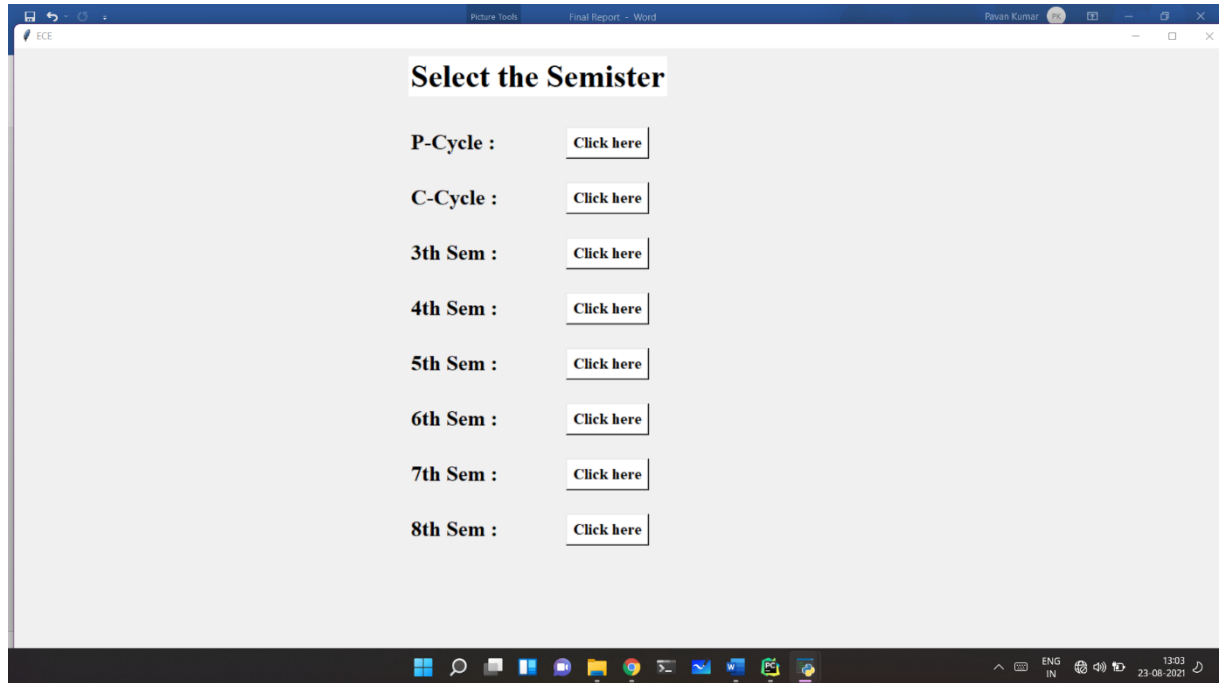
☐ I agree the terms and conditions


Estd:2001

RLJIT
Student Registration
Form

[Click to check details of this college](#)

[Click to check the previous marks](#)



The screenshot shows a web browser window with a single tab titled 'ECE'. The browser's address bar is empty. The page content is a registration form with the title 'Select the Semester' in bold black text. Below the title, there are seven rows, each containing a semester label and a 'Click here' button. The labels are 'P-Cycle', 'C-Cycle', '3th Sem', '4th Sem', '5th Sem', '6th Sem', '7th Sem', and '8th Sem'. The buttons are rectangular with a thin black border and the text 'Click here' in black. The browser's taskbar is visible at the bottom, showing various application icons and the system clock displaying 13:03 on 23-08-2021.

Select the Semester

P-Cycle : [Click here](#)

C-Cycle : [Click here](#)

3th Sem : [Click here](#)

4th Sem : [Click here](#)

5th Sem : [Click here](#)

6th Sem : [Click here](#)

7th Sem : [Click here](#)

8th Sem : [Click here](#)

CHAPTER 6

CONCLUSION

Online application of the whole system helps easy access to the system anywhere. Physical appearance is not required. The time taken for process completion is now largely reduced. After the registration the database is automatically updated at the end of process completion. Data is managed through MySQL, data duplication is eliminated and thereby reducing chances of error. Also, data can be now easily edited and printed whenever required. Also, database access is authorized and cannot be viewed or edited by other persons expect Admin.