

SWEAR ANALYSIS DOCUMENTATION

Initial Software Version

Swear Analysis is used to determine our Strengths, Weaknesses, Eligibility, Availability Resources. The student success is promoted by helping them through

Swear with:

- Mental health and life concerns
- Learning and academic skills challenges
- Career uncertainty
- Faculty/staff - student communication

This swear analysis allows students to provide good carrier guidance in their future path.

This also makes them better understand where they are lacking in the way to their career.

DESCRIPTION

This expert analysis software was created using Tkinter with python.

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 –

1. Import the Tkinter module.
2. Create the GUI application main window.
3. Add one or more of the above-mentioned widgets to the GUI application.

All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area. Tkinter exposes the following geometry manager classes: pack, grid, and place.

- The pack() Method – This geometry manager organizes widgets in blocks before placing them in the parent widget.
- The grid() Method – This geometry manager organizes widgets in a table-like structure in the parent widget.
- The place() Method – This geometry manager organizes widgets by placing them in a specific position in the parent widget.

Here we consider three modules .They are:

- Student Information Form
- Verbal
- Quantitative

In the Student Information Form we consider the details of the students ,whereas in verbal and Quantitative we will come to know their performance by providing some questions and by providing grades based on their answers for the particular questions.

Software Product

The final product of this software contains a log in for the student as well as a registration form for signing in. To Utilize this software product, the user must be aware of manipulating simple forms and some knowledge in typing.

For the developer, they need to have an understanding of python programming with good knowledge of the tkinter library. The developer should also have knowledge in databases with an understanding of how python interacts with the MySql database.

This final product of the software was built on python3, with the windows and User Interface is based on python's Tkinter library.

User Interfaces

The software is dependent on a couple of user interfaces to allow for interaction with the system.

The first form of interaction is the login form where the user can choose either to register or to log in depending on whether they have used the system before.

For first time users, the need for registration is inevitable in order to access and interact with the system.

In general, the system offers a simple UI for interaction which makes it easier for any user accessing the system.

Data collected from the forms in the UI is stored in the database. This information is used to provide analysis on the data collected .

The status for each questionnaire is displayed on the UI for the user to understand their progress.

Constraints, Assumptions and Dependencies

Dependencies

This software product is based on a number of third party libraries as well as few free libraries and software solutions . They include:

- Tkinter(Python GUI library.)
- Mysql(Database query performance language)
- Mysql connector for python

Assumptions

Since the software is an analysis-based tool, it is subject to a few assumptions:

- The users will answer truthfully as per the questionnaires are concerned.

- Additional functionality and features will certainly be added to the existing and proposed solution in the software.

Constraints

The software has a couple of constraints associated with it.

- The software use may be limited to use only on computers thus limited from mobile devices.
- Extra attention when using the software is required.

Operational Environment

The operational environment of the system is on and is dependent on the following:

- Mysql Database/Mariadb Database
- Python Environment
- Python Development
- Database Development

System Architecture

This part outlines the system architectural :

Hardware Architecture:

The swear analysis software has currently been deployed locally.

- Localhost Server
- MySql/Mariadb database to host the database
- Storage for various application configuration files.

System Descriptions

Programming Language	Python	<p>Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.</p> <p>Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. The language reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. https://www.python.org/doc/essays/blurb/</p>
Application Framework	Tkinter	<p>It is a python GUI framework built into the language as a standard library. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run.</p>
Database	Mysql/Mariadb	<p>As a database server, its primary function is to store data securely, and to allow for retrieval at the request of other software applications. It can handle workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users.</p>

With the variations in size of different deployments into the database, load testing is required to test the hardware requirements of the software service.

Software Architecture

The software embraces the three-tier architecture which is fairly popular for the development of user facing applications . The three tiers that comprise this architecture include:

- Presentation Tier
- Logic Tier
- Data Tier

Presentation tier

The tier is concerned with representing the components that users directly interact with. In this case , it comprises the desktop windows and forms used to present and collect data/information from the users interacting with the software.

The presentation tier in this software comprises the forms and windows in which the user inputs data and interacts with the system.

Logic Tier

Contains the code to translate the user actions at the presentation tier to the functionality that drives the application behavior .

Data Tier

This tier consists of storage media such as ,databases and file systems including caches, that hold the data relevant to the application software. In this software , it is made up of a MySql database which stores the user inputs.

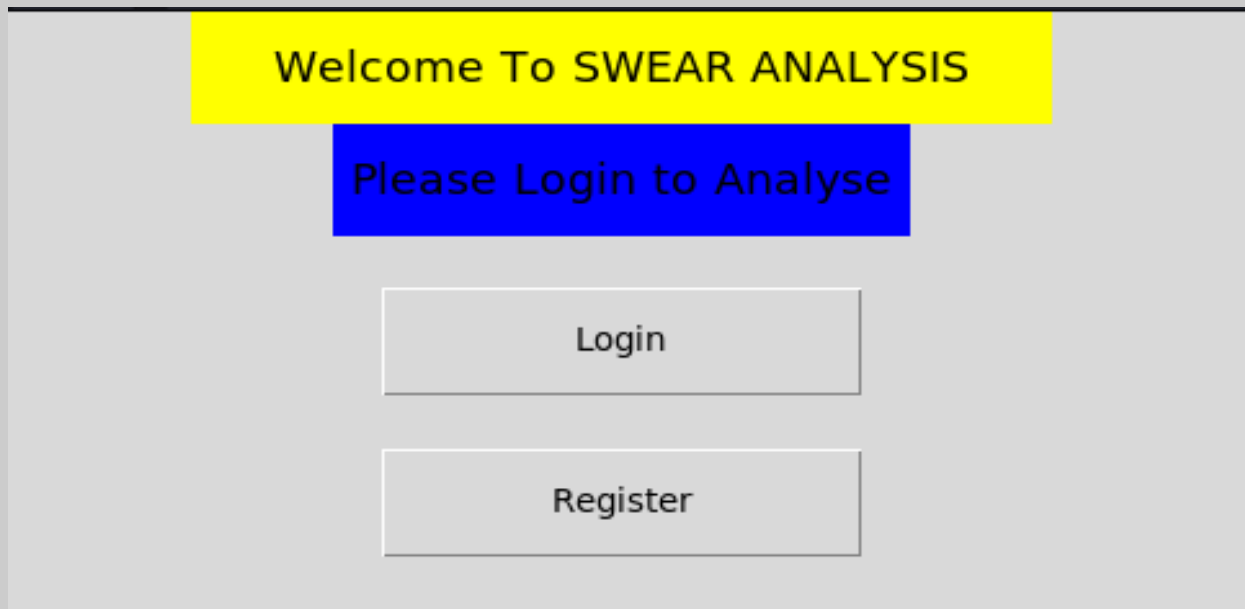
User Manual

On the directions of how to interact with the software product follow as below:

Login & Registration

After opening the software , you will be presented with a window with buttons:

- Login
- Registration



The screenshot shows a software window with a light gray background. At the top, there is a yellow rectangular banner with the text "Welcome To SWEAR ANALYSIS" in black. Below this banner is a blue rectangular button with the text "Please Login to Analyse" in black. Underneath the blue button are two white rectangular buttons with black borders. The top white button is labeled "Login" and the bottom white button is labeled "Register".

If you are a new user, click the register button .

The button directs you to a new window which requires some information.

Each field is mandatory.

Enter your information into the form and submit

Student ID :

First Name :

Last Name :

Email Id :

Contact Number :

Password :

Sign-Up

Click sign-up to submit your information

The information you have entered about yourself is then stored in the database.

After clicking sign -up, the login screen is presented to you . You can now log in.

If your details are already in the system simply click login.

You will be presented by the following screen .

Enter Student ID

Enter Password

Login

Enter your details and click login.

System Interaction

Welcome to Swear Analysis

Log-Out

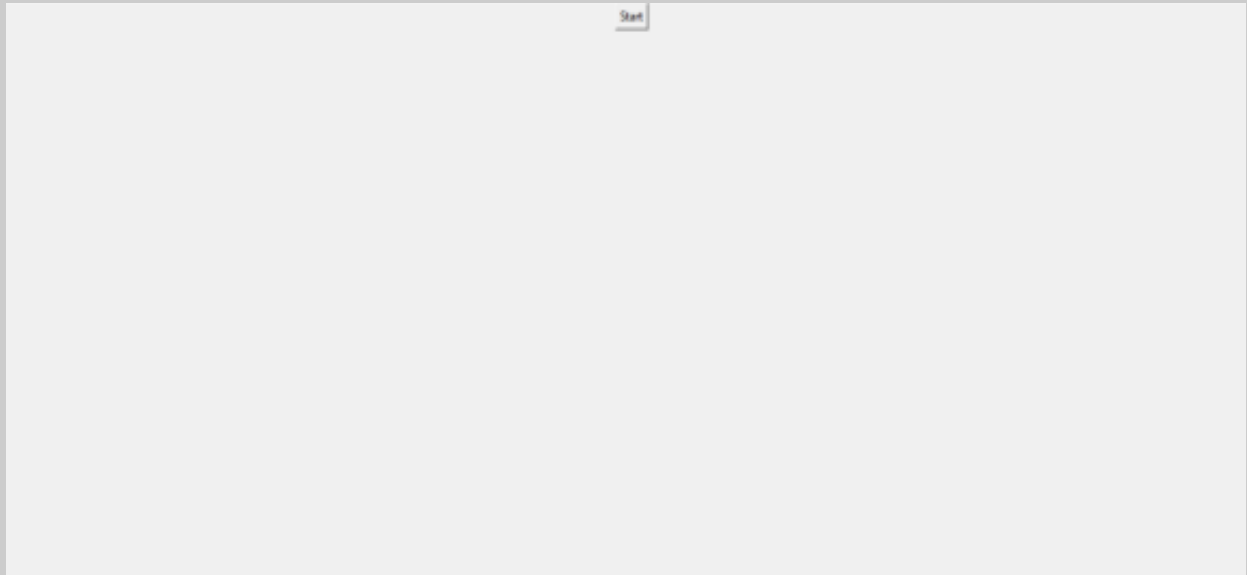
Your Analysis Status :

C Quiz :

Python Quiz :

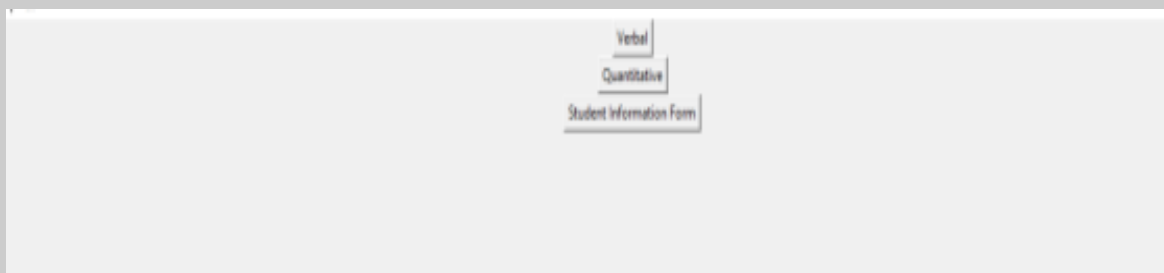
Quantitative Quiz :

Verbal Quiz :



Click start on the screen to proceed to the next module.

You will be presented with the following screen:



Select student information and click on it to update your information. There will be a window form with empty fields. Fill out the appropriate information.

First Name:

Last Name:

Father Name:

Mother Name:

Gender: ☐ MALE ☐ FEMALE

Date Of Birth:

State:

District:

Address:

Academic Details:

10th Class(CGPA):

Inter-Groups(%)

Inter-Language(%)

Interests:

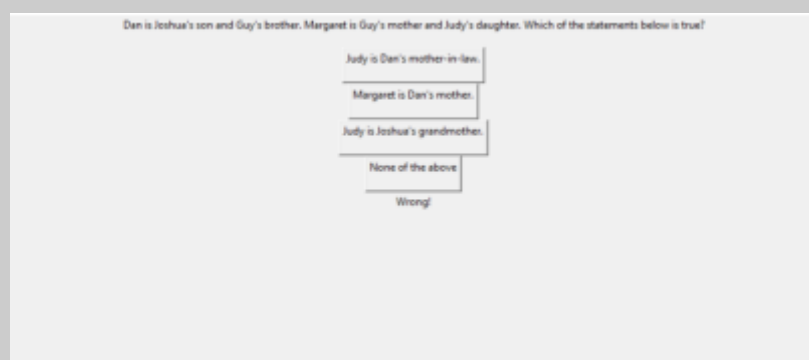
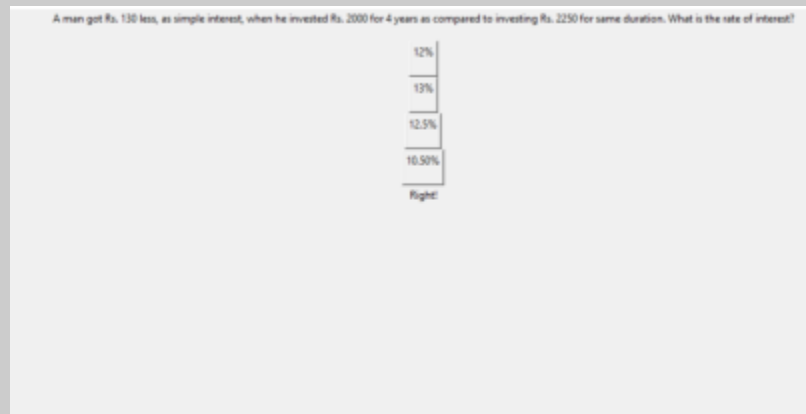
Strengths:

Languages Known:

Goals:

After filling out the information, click submit to save your details into the database.

You will be presented with the previous screen . Select any module and work your way through the various quizzes.



The above screenshots show some of the quiz interactions.

Developers Guide

This guide is for developers. It consists of ways of setting up the software on your local environment as well as a guide on how to handle the errors already Identified.

Setting up database:

- Make sure your MySql server is running and installed. To set up mysql follow the following link:https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwi93b--2bL7AhVri_0HHbqjAfMQFnoECAwQAQ&url=https%3A%2F%2Fdev.mysql.com%2Fdoc%2Fmysql-getting-started%2Fen%2F&usg=AOvVaw2YNC1233jxvEdiiHueVI8y

Setting up the environment

- Install python on your computer and make sure it's running. For more information on setting up python, follow the following link: <https://www.python.org>
- After this, set up and open the folder containing the project using an IDE for instance Pycharm or VS Code(preferably).
- Install the Tkinter library using pip. This library will help in managing the forms and windows in the software.
- Install the mysql connector to allow the software to interact with a local database and execute sql queries.
- Other libraries you will need to install using pip include:
 - ☐ Welcome
 - ☐ Fpdf
 - ☐ Reportlab
- After you are done setting up the environment.

Creating the database and Table

- In order to run the application successfully, you have to create a database using sql on your terminal then name it 'swear'.
- Create a table inside the database you just created . Name of the table should be 'student'.
- Add the following parameters as the columns for the table 'student':

```
emailid varchar(255) not null,
MariaDB [swear]> create table student(
-> id varchar(50),
-> fn varchar(255) not null,
-> ln varchar(255) not null,
-> emailid varchar(255) not null,
-> conno varchar(255) not null,
-> password varchar(255) not null
-> );
Query OK, 0 rows affected (0.172 sec)

MariaDB [swear]> show tables;
+-----+
| Tables_in_swear |
+-----+
| Students         |
| projects         |
| student          |
+-----+
```

Running the application:

- To run the application, open the swear_analysis python file and execute it. The result should be a pop up of the application.
- All stack trace will be on the terminal.

Screenshots for set up:

Starting mysql after setting it up:

```
$ sudo service mysql start
```

Check if mysql is running in order not to run into any error:

```
$ sudo service mysql status
```

Install the libraries using pip:

```
$ pip install fpdf || pip install welcome || pip install reportlab ||
```

Source Code:

Swear-Analysis

Swear-analysis.py

```
import welcome

welcome.welcome();
```

Portal.py

```
from tkinter import *
from tkinter import ttk
import mysql.connector
import cquiz
import pythonquiz
import quantquiz
import verbalquiz
import welcome
import report
global sid1,fn,ln,emailid,conno,passw>window;

def report1():
    global windowsid1;
    window.destroy();
    report.function(sid1)

def function(id):
    global sid1,fn,ln,emailid,conno,passw>window;
    cnx = mysql.connector.connect(user='root', password='root',
host='127.0.0.1',database='swear');
    cursor=cnx.cursor();
```

```

sql_c="select * from student where id='"+id+"'";
cursor.execute(sql_c);
rows=cursor.fetchall();
for row in rows:
    print(row)
    sid=sid1=row[0];
    fn=row[1];
    ln=row[2];
    print(sid,fn,ln);
    cstatus=row[6];
    pythonstatus=row[7];
    quantstatus=row[8];
    verbalstatus=row[9];
    quizstatus=[row[6],row[7],row[8],row[9]];
#print(sid,fn,ln);
def welco():
    window.destroy();
    welcome.welcome();
window = Tk();
window.title("Swear Analysis")
window.geometry('800x500')
#window.configure(background = "grey");
logout = Button(window ,text="Log-Out",command=welco).place(x = 600,y =
50)
a = Label(window ,text = "Welcome to Swear Analysis").place(x = 300,y =
50)
# b = Label(window ,text = "Your Name : "+fn+" "+ln).place(x = 200,y =
80)
# c = Label(window ,text = "Student Id : "+sid).place(x = 200,y = 100)
d = Label(window ,text = "Your Analysis Status : ").place(x = 200,y =
120)
e = Label(window ,text = "C Quiz :").place(x = 300,y = 150)
f = Label(window ,text = "Python Quiz :").place(x = 300,y = 170)
g = Label(window ,text = "Quantitative Quiz :").place(x = 300,y = 190)
h = Label(window ,text = "Verbal Quiz :").place(x = 300,y = 210)

for row in rows:
    quizstatus = [row[6],row[7],row[8],row[9]]

    xpos=420;ypos=150;

```



```

def cquizzes():
    window.destroy();
    cquiz.function(sid);
def pythonquizzes():
    window.destroy();
    pythonquiz.function(sid);
def quantquizzes():
    window.destroy();
    quantquiz.function(sid);
def verbalquizzes():
    window.destroy();
    verbalquiz.function(sid);
for i in range(0,4):
    if(quizstatus[i]=='0'):
        Label(window ,text = " Not Completed").place(x = xpos,y =
ypos);
    else:
        Label(window ,text = " Completed").place(x = xpos,y =
ypos);
    ypos=ypos+20;
    flag=1;
    if(quizstatus[0]=='0'):
        btn1=Button(window
, text="Click-Here",command=cquizzes).place(x=600,y=150);
        flag=0;
    if(quizstatus[1]=='0'):
        btn2=Button(window
, text="Click-Here",command=pythonquizzes).place(x=600,y=170);
        flag=0;
    if(quizstatus[2]=='0'):
        btn3=Button(window
, text="Click-Here",command=quantquizzes).place(x=600,y=190);
        flag=0;
    if(quizstatus[3]=='0'):
        btn4=Button(window
, text="Click-Here",command=verbalquizzes).place(x=600,y=210);
        flag=0;
    if flag==1:
        btn5=Button(window ,text="Click-Here for
Report",command=report1).place(x=500,y=300);

```

```
    window.mainloop()

#function("160031322")
```

Pythonquiz.py

```
import mysql.connector
import numpy
from tkinter import *
from tkinter import messagebox as mb
import json
import portal
global q_no1

def function(id):

    cnx = mysql.connector.connect(user='root', password='root',
host='127.0.0.1',database='swear');
    cursor=cnx.cursor();

    questions=[];
    flag=0;
    while(flag==0):
        num=numpy.random.randint(1,50);
        if str(num) not in questions:
            questions.append(str(num));
        if len(questions)==10:
            flag=1;
    questions.sort();
    #print(questions);
    questions=str(questions)
    questions='('+ questions[1:-1] +')'
    print(questions);
    sql_c="select * from pythonquestions where sno in "+questions+";"
    cursor.execute(sql_c);
    records = cursor.fetchall()
```

```

question=[];
options=[];
answer=[];
ans1={'A':1,'B':2,'C':3,'D':4}
for row in records:
    print(row);
    question.append(row[1]);
    op=[row[2],row[3],row[4],row[5]]
    options.append(op);
    print(row[6]);
    answer.append(ans1[row[6]])
print(question,answer);

class Quiz:

    def __init__(self):

        #self.display_question.delete(1.0,END)
        self.q_no=0

        self.display_title()
        self.display_question()

        self.opt_selected=IntVar()

        self.opts=self.radio_buttons()

        self.display_options()

        self.buttons()

        # no of questions
        self.data_size=len(question)

        # keep a counter of correct answers
        self.correct=0

    def display_result(self):

        wrong_count = self.data_size - self.correct

```

```

        correct = f"Correct: {self.correct}"
        wrong = f"Wrong: {wrong_count}"

        score = int(self.correct / self.data_size * 100)
        result = f"Score: {score}%"

        mb.showinfo("Result", f"{result}\n{correct}\n{wrong}")
        sql_c="update student set pythonstatus='"+str(self.correct)+"'
where id='"+id+"'";
        cursor.execute(sql_c);
        cnx.commit();
        print("sucess")
def check_ans(self, q_no):
    #print(self.opt_selected.get(),answer[q_no])

    if self.opt_selected.get() == answer[q_no]:
        return True

def next_btn(self):

    q_no1.config(text="")

    print(self.q_no)

    if self.check_ans(self.q_no):

        self.correct += 1

    self.q_no += 1

    if self.q_no==self.data_size:

        self.display_result()

        gui.destroy()
        portal.function(id)
    else:
        self.display_question()
        self.display_options()

```

```

def buttons(self):
    next_button = Button(gui, text="Next",command=self.next_btn,
        width=10,bg="blue",fg="white",font=("ariel",16,"bold"))

    # palcing the button  on the screen
    next_button.place(x=750,y=700)


    quit_button = Button(gui, text="Quit", command=gui.destroy,
        width=5,bg="black", fg="white",font=("ariel",16," bold"))

    quit_button.place(x=1200,y=50)

def display_options(self):
    val=0

    self.opt_selected.set(0)
    #print(self.opt_selected.get())

    for option in options[self.q_no]:
        self.opts[val]['text']=option
        val+=1
def display_question(self):
    global q_no1
    #self.delete(1.0,END)
    q_no1 = Label(gui, width=99,text=question[self.q_no],
        justify="left",font=( 'ariel' ,16, 'bold' ), anchor= 'w', wraplength=1000
    );

    #q_no.config(text=question[self.q_no])

    q_no1.place(x=70, y=100)
def display_title(self):

    title = Label(gui, text="Coding Section",
        width=99, bg="green",fg="white", font=("ariel", 20, "bold"))

```

```

        title.place(x=0, y=2)
    def radio_buttons(self):

        self.opt_selected.set(0);

        q_list = []

        y_pos = 300
        x_pos=100
        #opt=["A","B","C","D"]
        while len(q_list) < 4:

            radio_btn = Radiobutton(gui,text="
",variable=self.opt_selected,
            value = len(q_list)+1,font = ("ariel",14), wraplength=200)

            q_list.append(radio_btn)

            radio_btn.place(x = x_pos, y = y_pos)

            x_pos += 300

        return q_list

gui = Tk()
gui.geometry("1550x850")
gui.title("Coding Section")
question = (question)
options = (options)
answer = (answer)
quiz = Quiz()
gui.mainloop()

#function("160031322")

```

Registration.py

```
from tkinter import *
from tkinter import ttk
import mysql.connector
import welcome;
global sid1,fn,ln,emailid,conno,passw>window;
def function():
    global sid1,fn,ln,emailid,conno,passw>window;
    window = Tk()
    window.title("Registration Form")
    window.geometry('500x500')
    #window.configure(background = "grey");
    a = Label(window ,text = "First Name :").place(x = 100,y = 120)
    b = Label(window ,text = "Last Name :").place(x = 100,y = 140)
    c = Label(window ,text = "Email Id :").place(x = 100,y = 160)
    d = Label(window ,text = "Contact Number :").place(x = 100,y = 180)
    sid = Label(window ,text = "Student ID :").place(x = 100,y = 100)
    d = Label(window ,text = "Password :").place(x = 100,y = 200)
    sid1 = Entry(window)
    sid1.place(x = 250,y = 100)
    fn = Entry(window)
    fn.place(x = 250,y = 120)
    ln = Entry(window)
    ln.place(x = 250,y = 140)
    emailid = Entry(window)
    emailid.place(x = 250,y = 160)
    conno = Entry(window)
    conno.place(x = 250,y = 180)
    passw = Entry(window,show="*")
    passw.place(x = 250,y = 200)
    btn = ttk.Button(window ,text="Sign-Up",command=clicked)
    btn.place(x = 200,y = 280)
    window.mainloop()

def clicked():
    global sid1,fn,ln,emailid,conno,passw>window;
    cnx = mysql.connector.connect(user='root', password='root',
host='127.0.0.1',database='swear');
    #print(a1.get(),b1.get())
    print(cnx)
```

```

cursor=cnx.cursor();
id = sid1.get()
fname = fn.get()
lname = ln.get()
email = emailid.get()
c = conno.get()
password = passw.get()

try:
    sql_c='insert into studentest
values("'" +sid1.get()+"','"+fn.get()+"','"+ln.get()+"','"+emailid.get()+"',
'" +conno.get()+"','"+passw.get()+"","0","0","0","0");';
    query = "insert into student(id,fn,ln,emailid,conno,password)
values(%s,%s,%s,%s,%s,%s)"
    data = (id,fname,lname,email,c,password)
    cursor.execute(query,data);
    cnx.commit();
except Error as e:
    print (e)
msg = Label(window ,text = " You succesfully signed-up Please Close
this window and continue with Login.") .place(x = 50,y = 400)
window.destroy();
welcome.welcome();
cnx.close()

```

Report.py

```

import mysql.connector
import numpy
from fpdf import FPDF
from tkinter import *
from tkinter import messagebox as mb
import json
global q_no
from reportlab.lib.units import inch
from reportlab.lib import colors
from reportlab.lib.pagesizes import A6
from reportlab.platypus import SimpleDocTemplate, Table, TableStyle

```



```

from reportlab.graphics.shapes import Drawing
from reportlab.platypus import Paragraph
from datetime import datetime

import calendar
from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle

def function(id):

    cnx = mysql.connector.connect(user='root', password='root',
host='127.0.0.1', database='swear');
    cursor=cnx.cursor();
    sql_c="select * from student where id="+id+";"
    cursor.execute(sql_c);
    records = cursor.fetchall()
    for row in records:
        sid=row[0];
        fnam=row[1];
        lnam=row[2];
        email=row[3]
        contactno=row[4]
        #print(sid,fnam,lnam);
        #cstatus=row[6];
        #pythonstatus=row[7];
        #quantstatus=row[8];
        #verbalstatus=row[9];
        quizstatus=[row[6],row[7],row[8],row[9]];
        print(row)

    doc = SimpleDocTemplate(fnam+'_'+lnam+'_report.pdf', pagesize=A6)
    elements = []
    data= [['Swear-Analysis Report'], ['Report Generated
Time',datetime.now()], ['Name: ',fnam+' '+lnam],
        ["ID: ",sid], ['Email-ID: ',email], ['Contact No: ',contactno], ['Verbal
Section: ',quizstatus[3]+'/10'], ['Quantitative
Section',quizstatus[2]+'/10'], ['Coding
Section',str((int(quizstatus[0])+int(quizstatus[1]))/2)+'/10']]
    t=Table(data)
    t.setStyle(TableStyle([('BACKGROUND', (1,1), (-2,-2), colors.green),

```

```
    ('TEXTCOLOR', (0,0), (1,-1), colors.black]))  
    elements.append(t)  
    # write the document to disk  
    doc.build(elements)  
  
#function("160031322")
```