**TABLE OF CONTENT**

[**ONLINE EXAMINATION SYSTEM 2**](#_81dt7l9w7xbd)

[DESCRIPTION 2](#_ysuxrsaqamge)

[**LOGIN PAGE 3**](#_6e2n5ak840om)

[CODE DESCRIPTION 3](#_4214dpwq948p)

[**OPTIONS PAGE 4**](#_43x4z8x7h7x5)

[CODE DESCRIPTION 4](#_xmb57nzeopcz)

[**STUDENT DETAIL PANEL 5**](#_np2rmlh48bl)

[CODE DESCRIPTION 5](#_zf1144b1txgm)

[EDIT PROFILE PANEL 6](#_k3fn8ynotz9y)

[CODE DESCRIPTION 6](#_gy3wdny1tlja)

[**HOME PANEL 8**](#_bmk0db9ng2gw)

[CODE DESCRIPTION 8](#_h6sqd2ao9uwx)

[QUESTIONS PANEL 9](#_4dw7una27os8)

[DESCRIPTION 9](#_c3vaomohgiic)

[**RESULT PANEL 11**](#_jaounul46si5)

[CODE DESCRIPTION 11](#_qjhu3qjbogg)

[ANSWERS PANEL 12](#_ywdoimclv1b3)

[CODE DESCRIPTION 12](#_40zwgn6lhvzr)

[RESULT TABLE PANEL 13](#_8opow2180zfw)

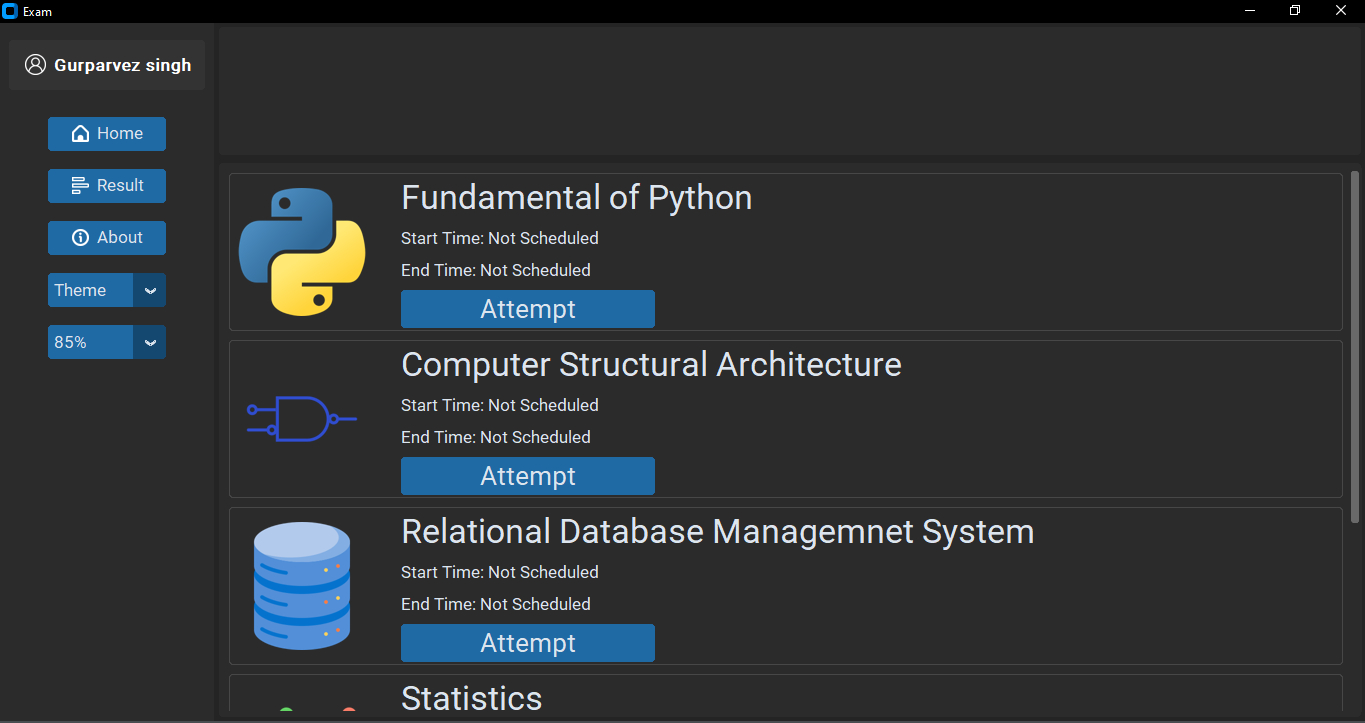
[CODE DESCRIPTION 13](#_86f8c29j7bqe)

[MAIL THE RESULT 14](#_xvb19tjlsyz)

[CODE DESCRIPTION 14](#_wg8xdwqjl5f2)

# 

# ONLINE EXAMINATION SYSTEM



## DESCRIPTION

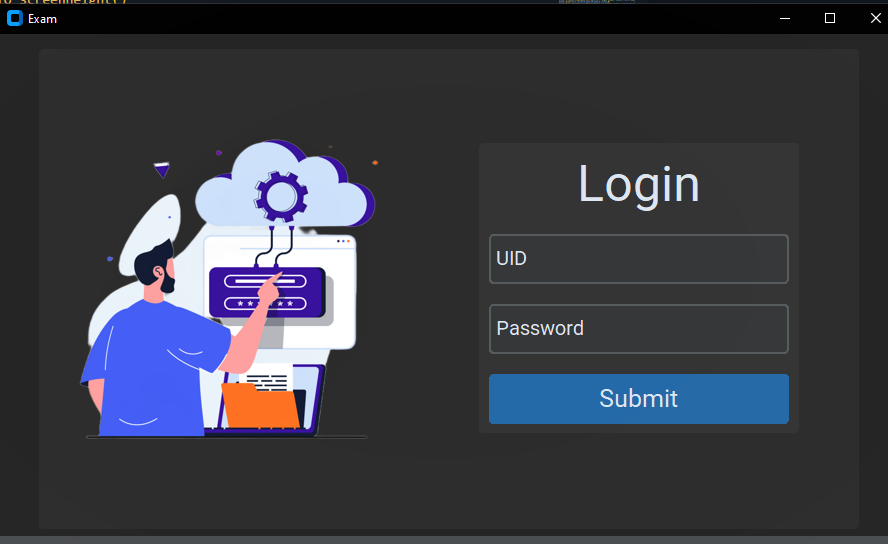
This is an online software created using customtkinter in Python and Postgre SQL database. It is a basic system designed for students in which they can view the subjects of the course in which they are enrolled, give the exam if available and view the result details.

The student’s name is displayed at the top left corner of the page which, when clicked on, opens a window allowing students to edit their information. The software has two main windows that provide the user with different information. The first window shows the subjects which the student is currently studying in the course, while the second window displays the result details of the student, which is subject-wise.

The software also includes additional features like switching to dark mode, which enables users to switch the software's interface from the default mode, light mode or a darker mode. The scaling feature is also available for users with different screen sizes, enabling them to adjust the software's interface according to their screen resolution.

Overall, this online software aims to provide students with an easy-to-use and accessible platform to view their course subjects and result details, with added features to enhance user experience.

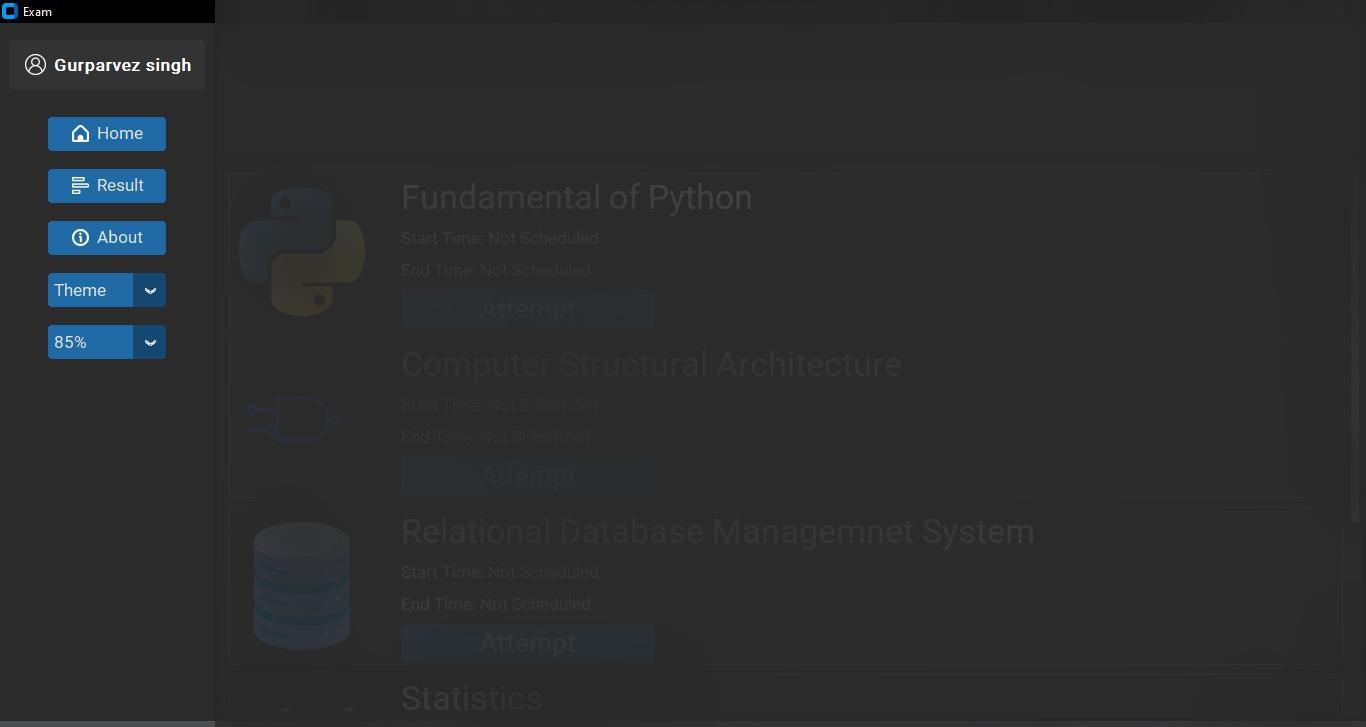
# LOGIN PAGE

The Student first has to login by using the UID provided by the university and the password (which is 12345 by default and students can change it in their profile editing section). Code for this page is something like this :

## CODE DESCRIPTION

The code for this page defines a function called loginGui() which creates a graphical user interface for user login. It contains several nested functions that handle various events such as mouse clicks and entry focus. The GUI consists of a logo image, a label with the text "Login", two input fields for UID and password, and a submit button. The function binds the enter key to the submit button and calls the submit() function when the submit button is clicked or enter key is pressed. The code uses the CTk library for creating the GUI components and also imports the PIL library to load the logo image.

# OPTIONS PAGE



## CODE DESCRIPTION

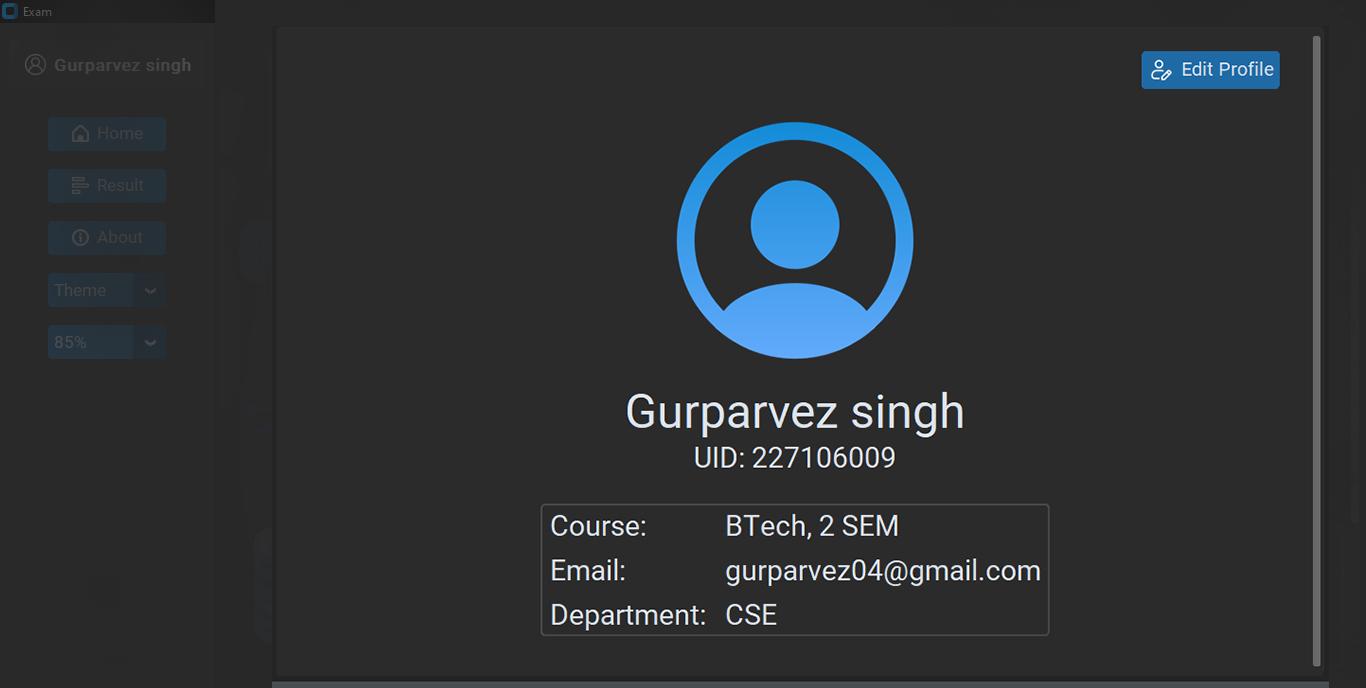
The panel includes various buttons and features that provide additional functionality to the user.

The panel displays the name of the student, obtained from the database, and includes a button that leads to the student's information page. It also includes buttons for navigating to the home page, result page, and about page. These buttons are represented by icons and labeled with text.

In addition, there are two special features provided by the panel. The first is a theme option that allows the user to switch between a light and dark mode. The second is a scaling feature that allows the user to adjust the size of the application.

All of the buttons and options in the panel are styled with appropriate font and size to ensure that they are easily readable and accessible to the user. Overall, the "options" panel provides an intuitive and user-friendly interface for navigating the online examination system.

# STUDENT DETAIL PANEL



## CODE DESCRIPTION

This code defines a function studentInformationGui(uid) that creates a GUI to display information about a student given their UID.

The function first hides any existing frames (subjectFrame, resultFrame, aboutFrame, and questionFrame) before creating a new frame studentInformationFrame to display the student's information.

It then queries the database using the uid to get the student's information, including their name, email, course name, semester, and department name.

Next, it creates a label with the student's profile picture, name, and UID. It also creates a frame subInformationFarme to display the student's course information, email, and department.

Finally, it packs all the created widgets and displays the studentInformationFrame.

## EDIT PROFILE PANEL

### CODE DESCRIPTION

The function starts by destroying all the children widgets of a frame called studentInformationFrame using a for loop and the destroy() method. It also clears the packFrame list.

The code then retrieves the first name, last name, email, and password of the user with the given ID from a database using a SELECT statement.

The function then creates a new frame called editFrame and adds some labels and entry fields to it to allow the user to edit their information. The labels and entry fields are labeled as "First Name," "Last Name," "Email," and "Password," respectively. The initial values of the entry fields are set to the corresponding values retrieved from the database.

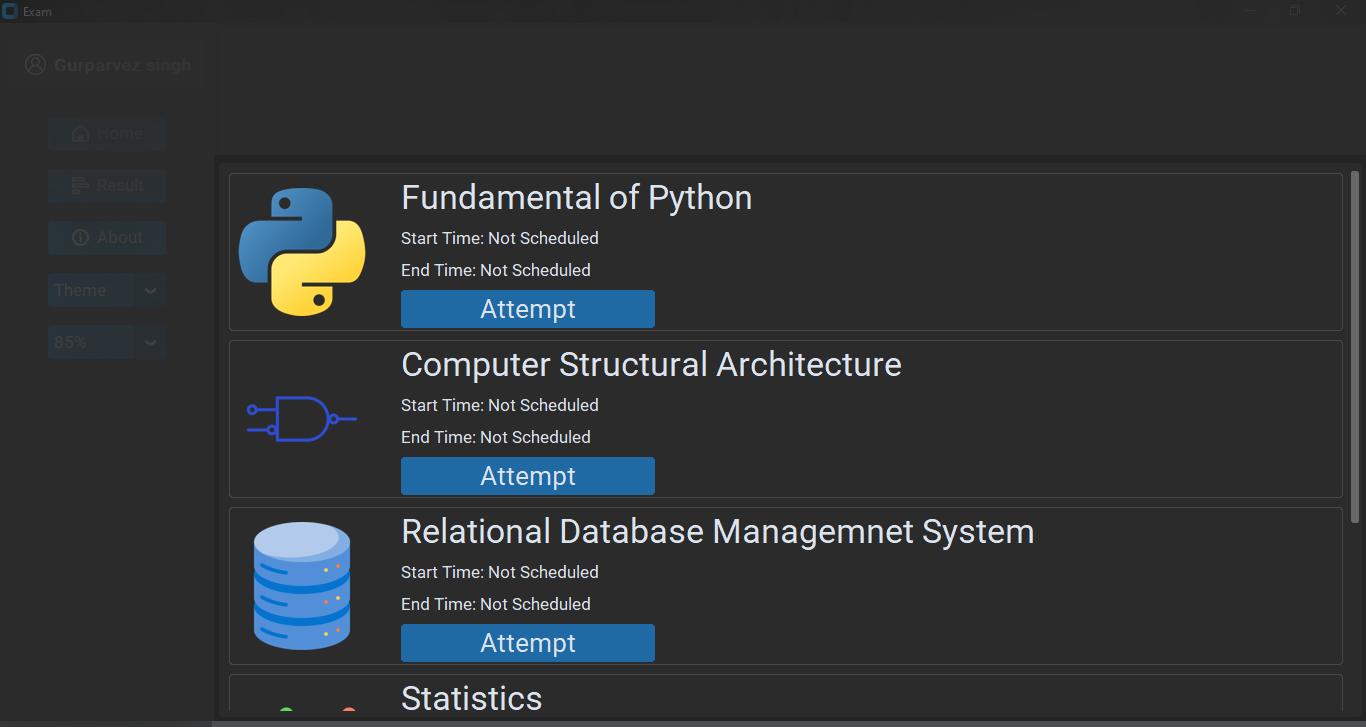
Finally, a submit button is added to the frame, which calls a function profileSubmit() with the updated values of the first name, last name, email, password, and user ID as input parameters when clicked. The editFrame is packed onto the studentInformationFrame.

There is another function which is defined as profileSubmit() that is called when the user submits their updated information in the editProfile() function. It takes four parameters - name1, name2, email, password, and uid.

The function first retrieves the values of name1, name2, email, and password using .get() method of each parameter, and then performs some validation checks on them. If any of the validation checks fail, it displays an appropriate message using the messagebox.showinfo() method and returns, meaning the profile is not updated.

If all the validation checks pass, the function executes an SQL query using the Database() function to update the student's information in the database. If the query is successful, the function displays a message saying the information has been updated, and calls the studentInformationGui() function to display the updated student information. It also updates the text on the nameButton widget to display the student's new full name.

# HOME PANEL



## CODE DESCRIPTION

This is a Python function called subjectFrameGui(uid) that creates a graphical user interface (GUI) for a student's subjects. The function takes a user ID (uid) as input to identify the student whose subjects will be displayed in the GUI.

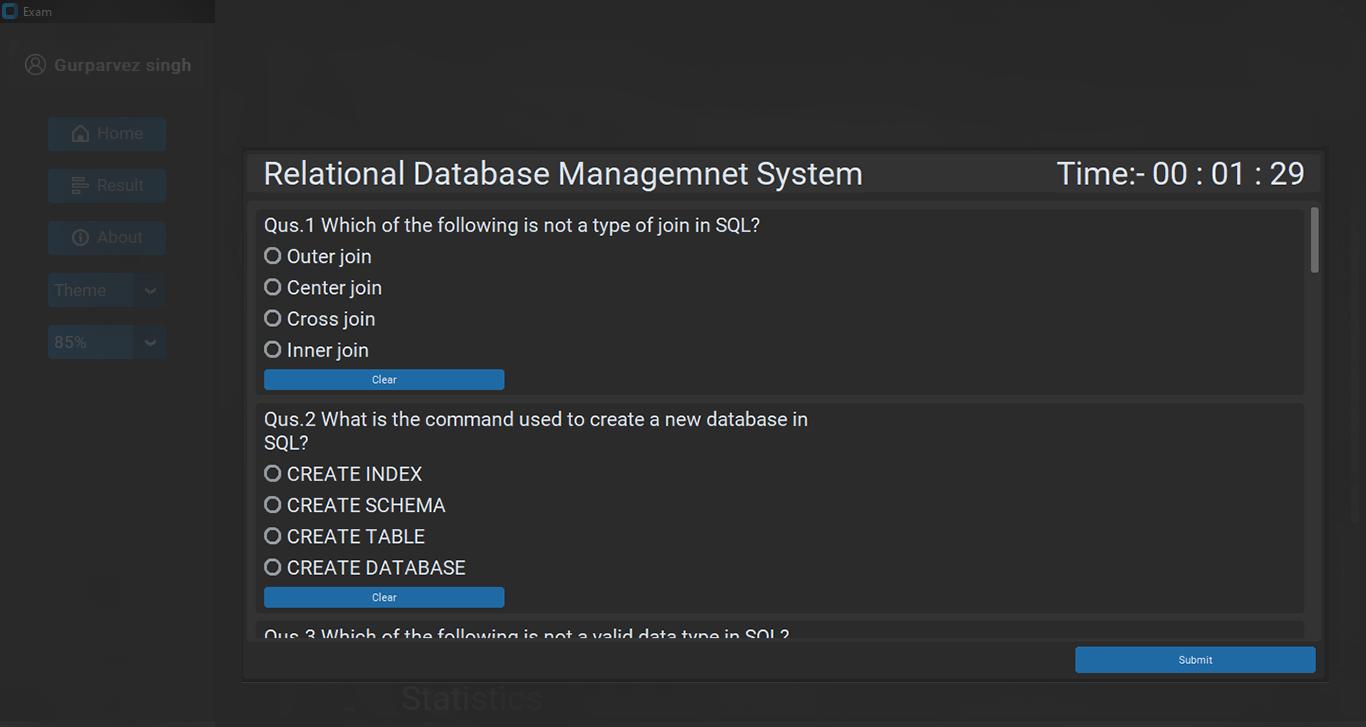
The function retrieves data about the student's subjects from a database using SQL queries, including the subject's title, ID, start date, end date, and image. It then loops through the retrieved data and creates a subject frame for each subject.

Each subject frame contains an image of a book (either a custom image or a default one), the subject's title, start date, end date (if available), and a button to attempt the subject's questions.

When the attempt button is clicked, the questionGui(uid,subject\_id) function is called with the student's user ID and the subject's ID as parameters.

The subject frames are displayed in a larger subject frame using the pack() method. The subjectFrameGui(uid) function returns nothing.

## QUESTIONS PANEL



### DESCRIPTION

This is a Python function called questionGui that creates a graphical user interface for a multiple-choice question exam. It takes two arguments: uid, which is an ID of a user, and subject\_id, which is an ID of a subject for which the exam is being taken.

First, the function checks whether the user has already submitted the exam by querying a database using the submitorNot variable. If the user has already submitted, the function displays a message box and returns. If not, it proceeds to create the GUI.

The function creates a questionFrame and packs it inside a subjectFrame. The questionFrame contains multiple singleQuestionFrames that each represent one question in the exam. Each singleQuestionFrame has a question label and four radio button options that represent the possible answers to the question. The options are randomised using the shuffle() function.

A timer is also displayed on the GUI, which is set to a predetermined duration of the exam. The timeFunction() is used to update the timer every second.

Finally, a "Submit" button is displayed at the bottom of the questionFrame. When the user clicks on the "Submit" button, the ansSubmit() function is called with the user's answers to the questions, uid, qId (which is a list of question IDs), and subject\_id.

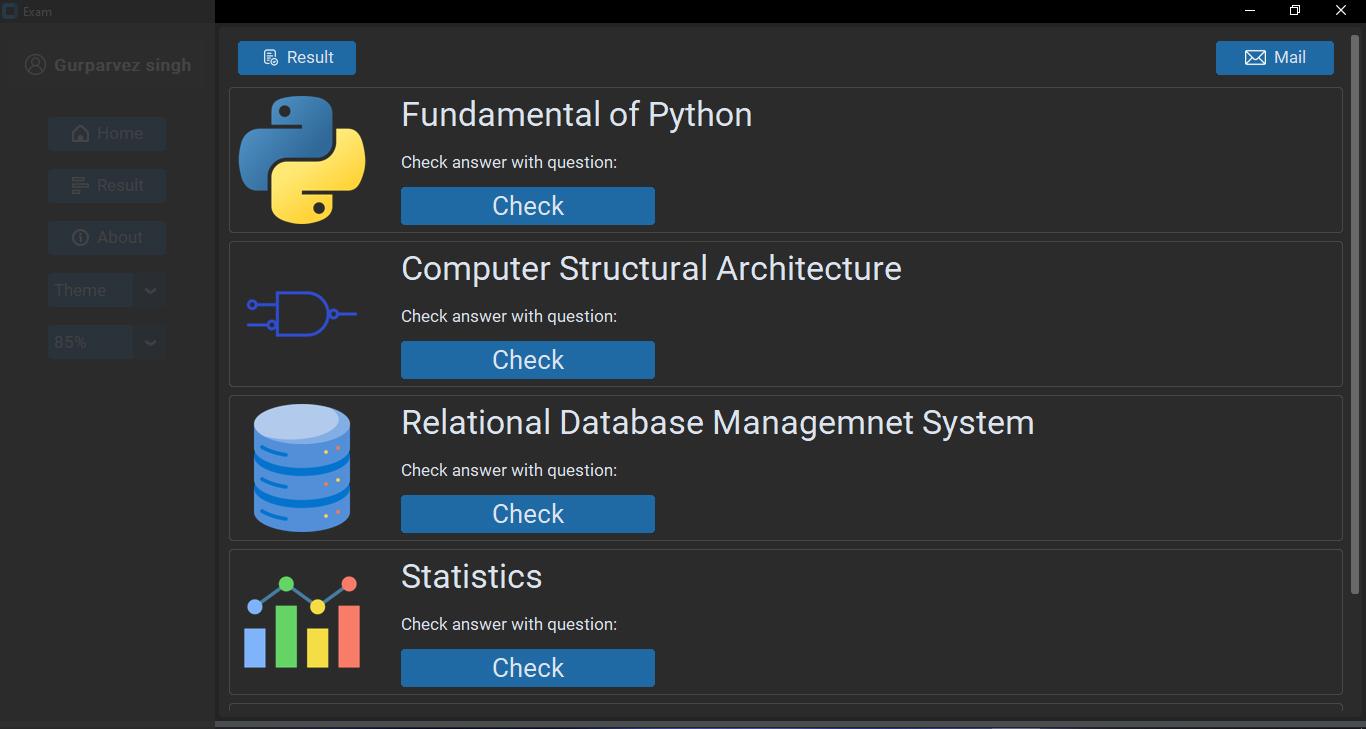
There is also a function named ansSubmit that takes four arguments: var, uid, qId, and sId, and an optional permission parameter with a default value of False.

The function is responsible for submitting the user's answers to a particular subject. If permission is not True, it will show a confirmation message box asking the user if they want to submit their answers. If the user confirms, the function will loop through all the answers stored in the var list and insert them into the database with the corresponding question and user IDs using the Database() function with send=True flag.

After inserting all the answers, it will also insert a record of the user's submission into the submit table.

Finally, the function will destroy all the widgets in the questionFrame, clear the packFrame, and call the home() function to take the user back to the home screen.

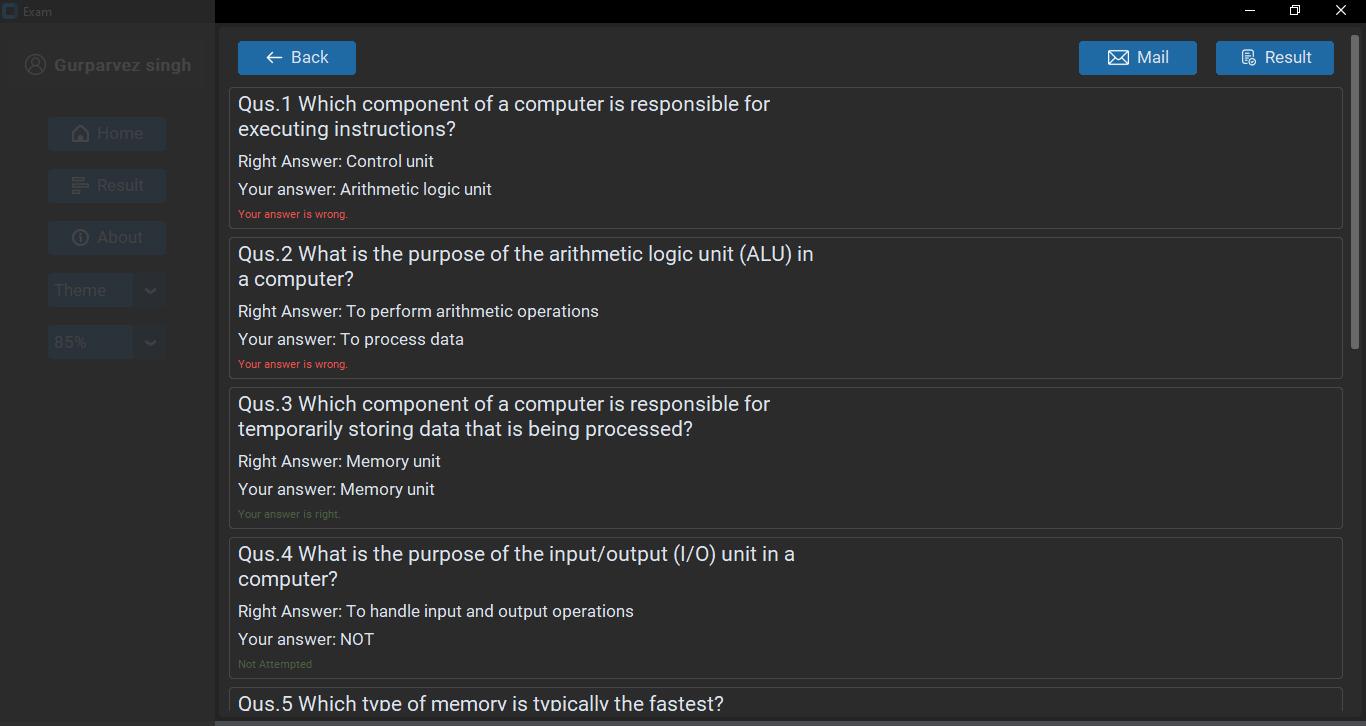
# RESULT PANEL



## CODE DESCRIPTION

This function creates the result GUI for the student, displaying a list of subjects with an image and a button to check answers for the questions of each subject. It starts by hiding all other frames and setting the current packFrame to "resultFrame". It then retrieves a list of subjects, along with their title, subject ID, and image (if available) from the database using the provided student UID. The function creates a frame for the result and mail buttons at the top of the window, along with two buttons for each subject: an image of the subject (or a default image if none is available), the subject title, a description, and a "Check" button to open the questions GUI for that subject. When a subject's "Check" button is clicked, the studentResult() and questionGui() functions are called with the provided UID and subject ID as parameters. Finally, the resultFrame is packed into the window.

## ANSWERS PANEL



### CODE DESCRIPTION

The function takes two parameters: uid (which I assume stands for User ID) and subject\_id. It queries the database to check if the student has submitted an answer for the given subject. If the student has not submitted an answer, it displays a message box saying that the result is not available and returns.

If the student has submitted an answer, the function queries the database to get the question, answer, and student answer for the given uid and subject\_id. It then creates a GUI to display the result for each question in the subject.

The GUI contains a frame with three labels: questionLabel, answer, and studentAnswer. It also has a rightwrong label, which displays if the student's answer is correct or incorrect. Finally, the GUI contains three buttons: Back, Result, and Mail.

Overall, the function seems to be well-structured and easy to understand. However, I cannot comment on its correctness without knowing the structure of the database and the larger application it is a part of.

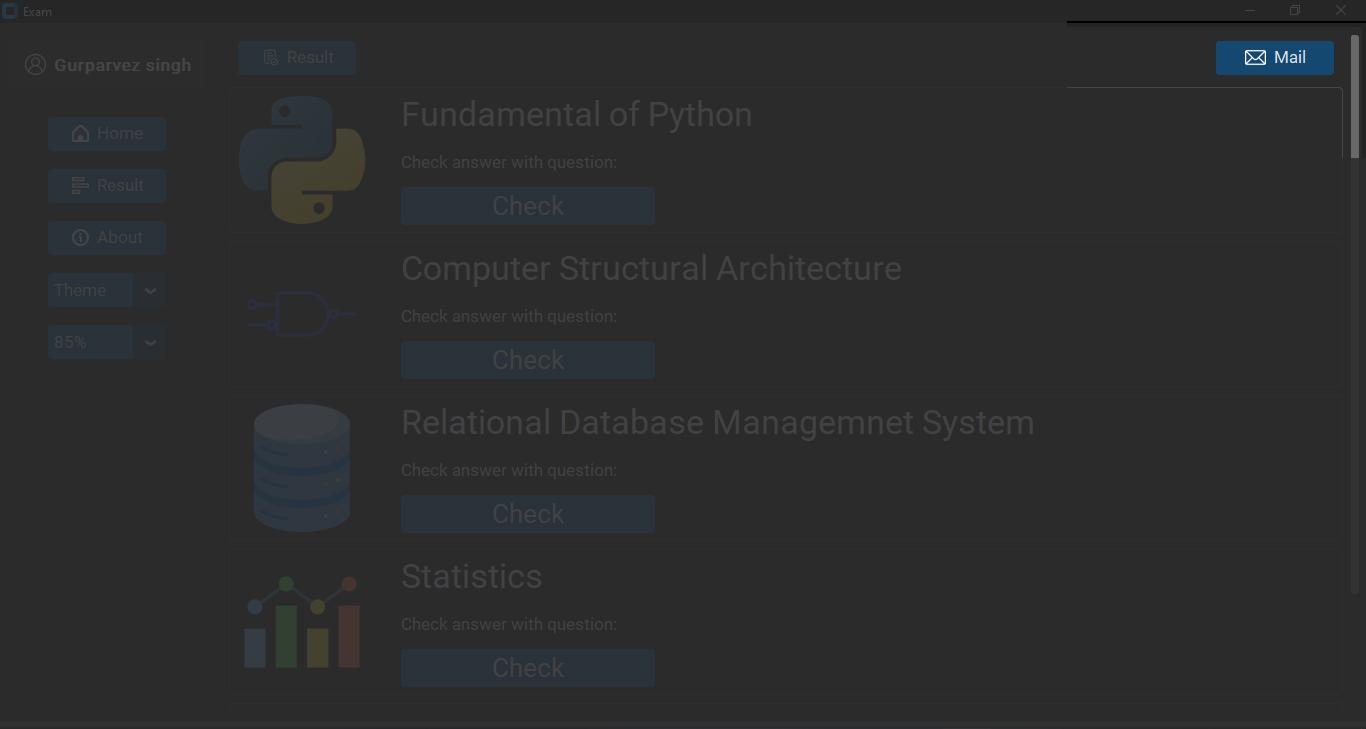
## RESULT TABLE PANEL

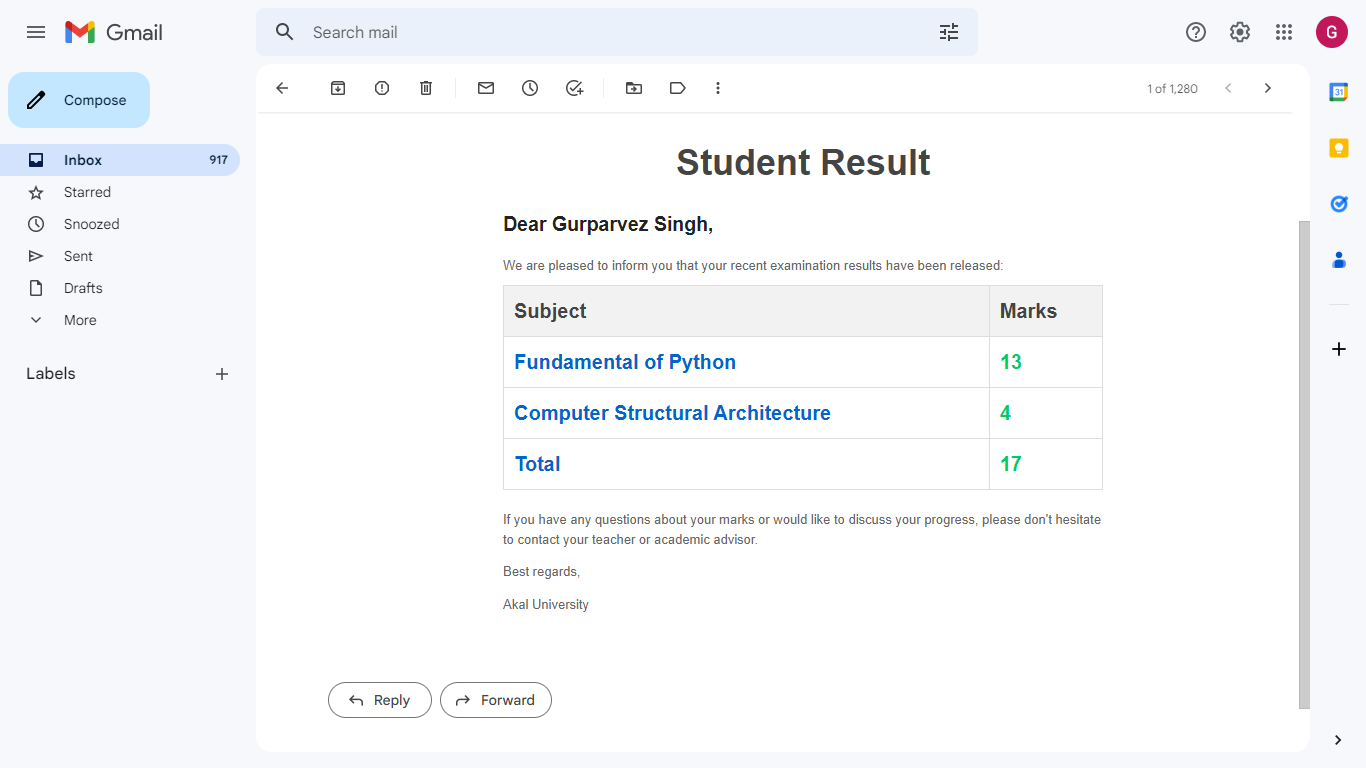
### CODE DESCRIPTION

It looks like the code defines two functions table and resultInApp that display a table of student result in a tkinter application. The table function takes three arguments: root (the parent widget), data (a 2D list containing the student result data), and totalNumber (the total marks obtained by the student). It creates a table with the given data and displays it in the parent widget.

The resultInApp function takes a single argument uid (the unique id of the student) and first clears the old widgets in the resultFrame. It then creates a label, a frame, and a 2D list containing the student result data by calling the resultQueries function. It then calculates the total marks obtained by the student and inserts a new row at the end of the result data list containing the total marks. Finally, it calls the table function to display the result data table in the resultFrame.

## MAIL THE RESULT





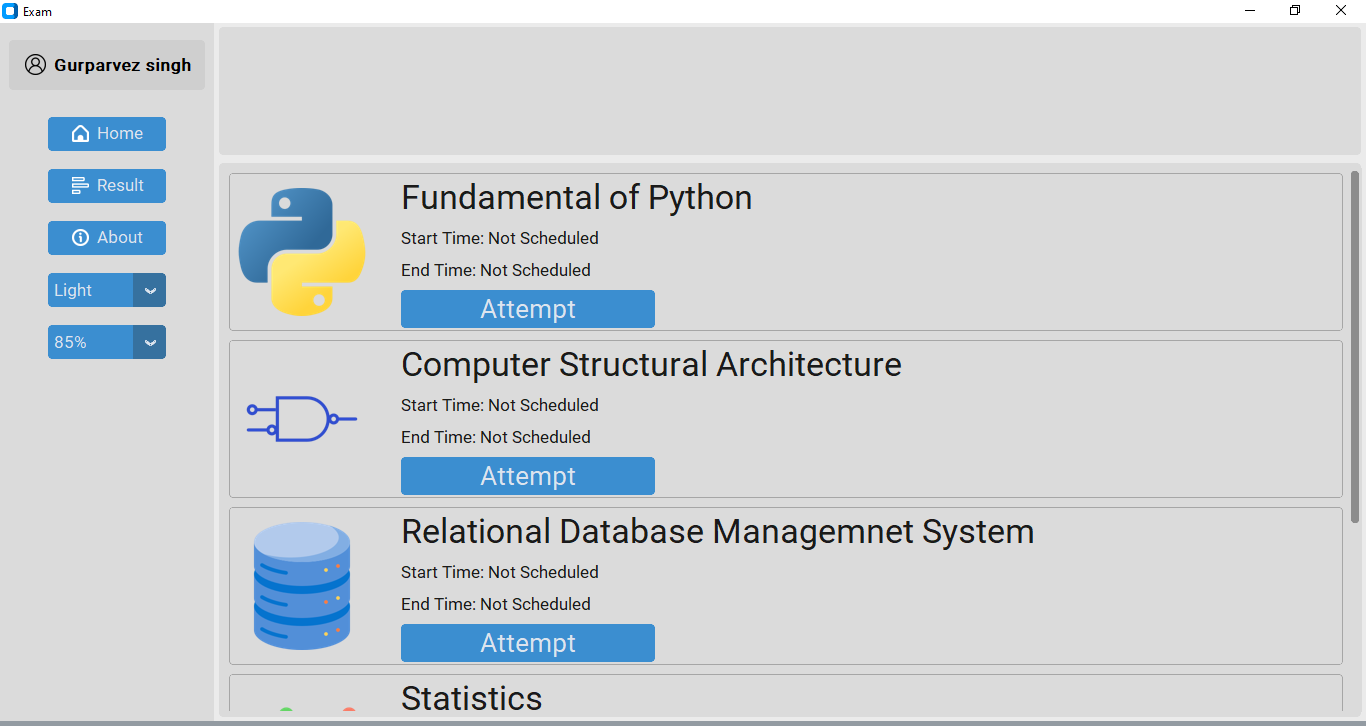
### CODE DESCRIPTION

This is a Python function that sends an email containing a student's exam result to their email address. The function takes in one argument, uid, which is the unique identifier of the student in the database.

The function first retrieves the email address of the student from the database using the uid. It then retrieves the student's name and exam results using various SQL queries. The exam results are formatted into an HTML table that is included in the email body.

The sendMail function is called to send the email. It uses the SMTP protocol to connect to a Gmail server and authenticate the sender's credentials. The email is then sent using an EmailMessage object that includes the email content and metadata such as the sender and receiver addresses and the email subject.

If there is an exception during the sending process, a message box is shown with the error message. If the email is sent successfully, a message box is shown confirming that the email has been sent.



THANK YOU

<https://drive.google.com/file/d/1FyZ9pk5WAMDheEcpsjCJUFpRJLm7v6DH/view?usp=sharing>

(link for the source codes)