

InCollege

Shahaddin Gafarov

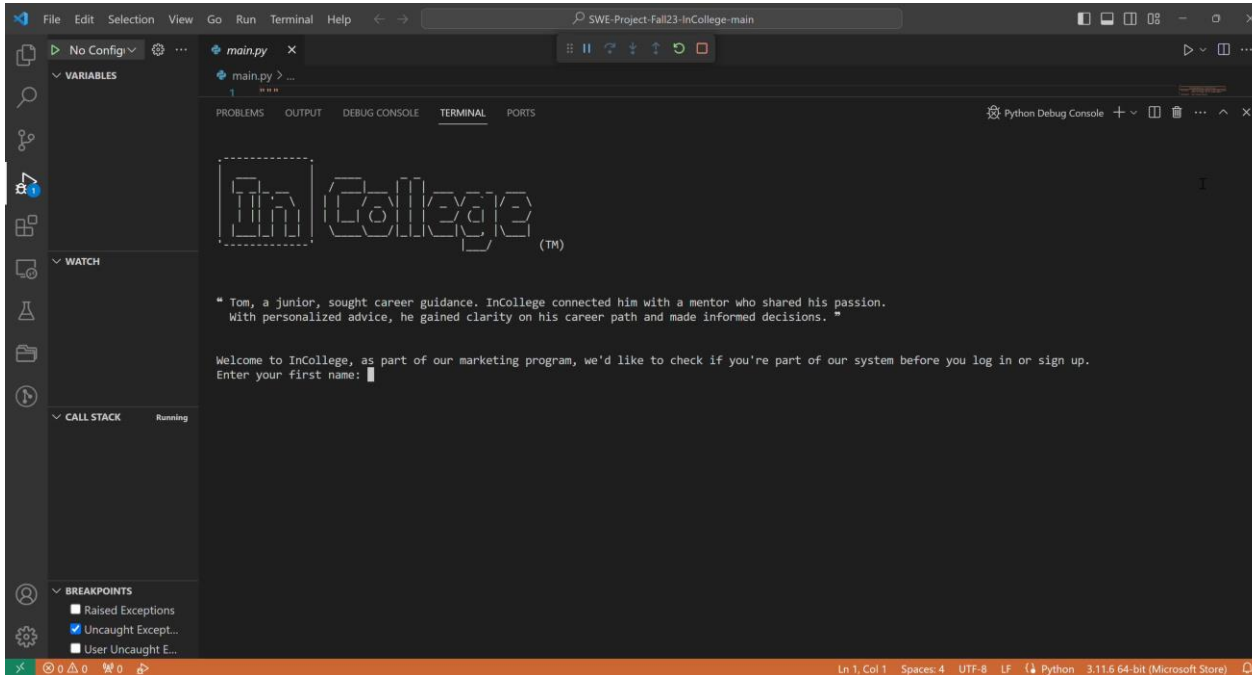
Daniel Escalona Gonzalez

Eloy Fernandes Ballesteros

Jason Greb

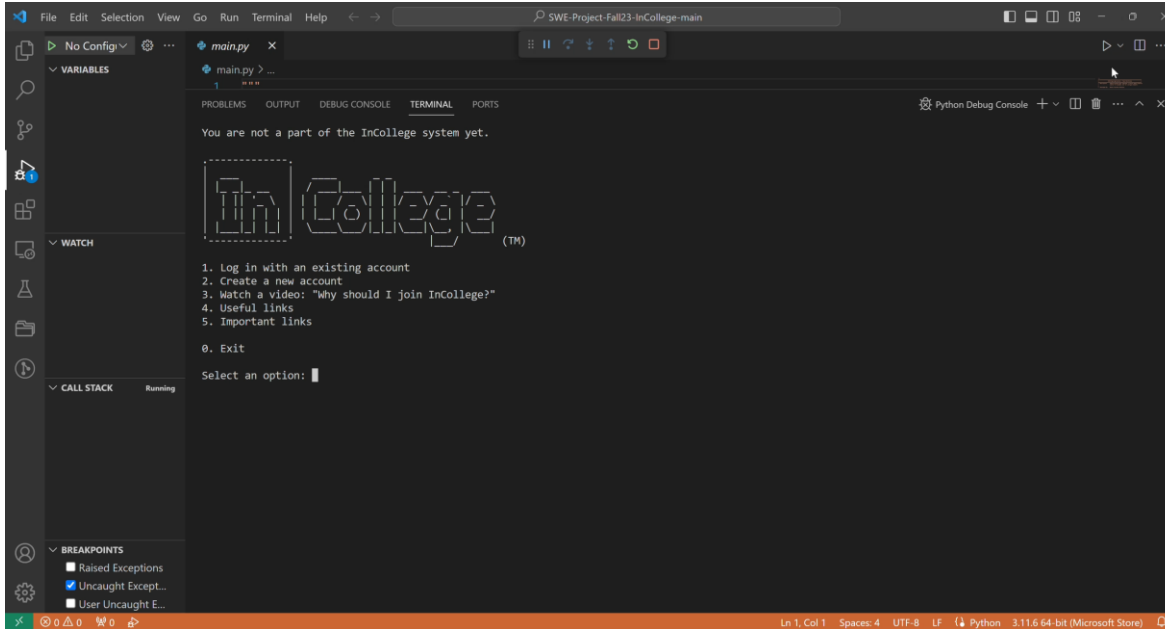
Massimo Giannini

Searching for a User Prior to Login



- On the start-up screen, the user is asked for their first then last name.
- It checks if their name is already stored in our database or not.
- Prints a message at the top of terminal notifying user if they are or are not already part of InCollege.

Creating an Account

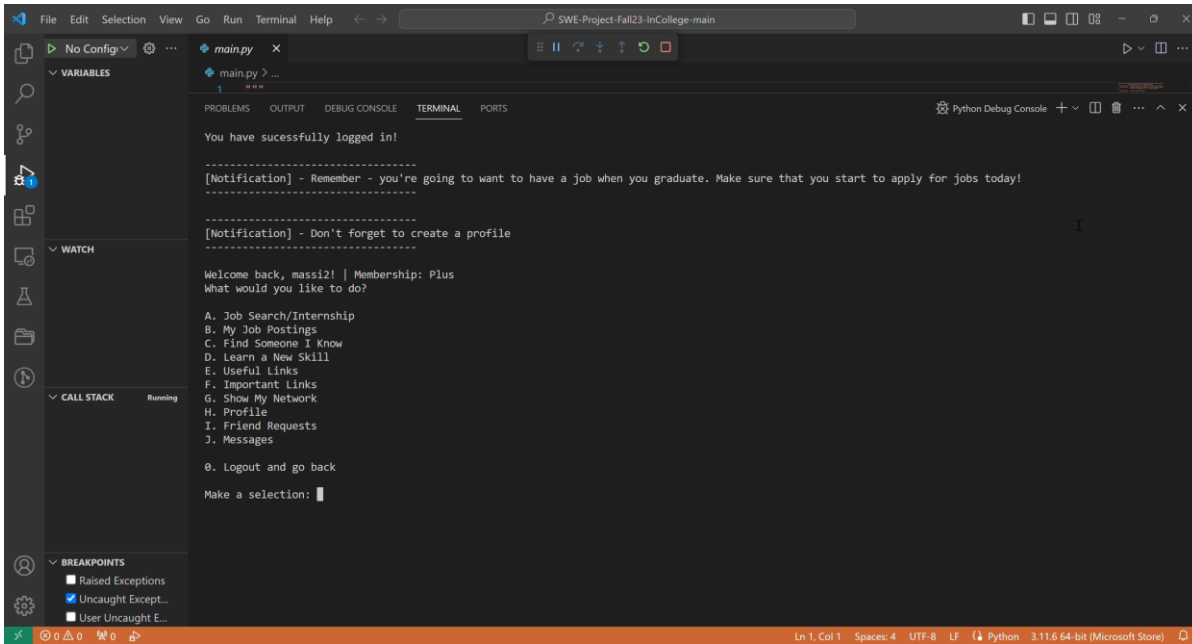


The screenshot shows a VS Code editor with a terminal window open. The terminal displays the InCollege logo and a menu of options. The menu includes: 1. Log in with an existing account, 2. Create a new account, 3. Watch a video: "Why should I join InCollege?", 4. Useful links, 5. Important links, 0. Exit. The prompt "Select an option:" is followed by a cursor.

```
main.py > ...  
1  
You are not a part of the InCollege system yet.  
  
InCollege (TM)  
  
1. Log in with an existing account  
2. Create a new account  
3. Watch a video: "Why should I join InCollege?"  
4. Useful links  
5. Important links  
  
0. Exit  
  
Select an option: |
```

- User selects option '2' to create an account.
- User enters their username, password, major, university.
- Offered to join the premium membership (y/n).
- Account is created if password is valid, account doesn't already exist or if there is space in our database.
- Returns to main menu and account created message prints at top of terminal

Posting a Job



The screenshot shows a Python IDE with a terminal window. The terminal output is as follows:

```
main.py > ...  
1  
-----  
You have successfully logged in!  
-----  
[Notification] - Remember - you're going to want to have a job when you graduate. Make sure that you start to apply for jobs today!  
-----  
[Notification] - Don't forget to create a profile  
-----  
Welcome back, massiz! | Membership: Plus  
What would you like to do?  
  
A. Job Search/Internship  
B. My Job Postings  
C. Find Someone I Know  
D. Learn a New Skill  
E. Useful Links  
F. Important Links  
G. Show My Network  
H. Profile  
I. Friend Requests  
J. Messages  
  
0. Logout and go back  
  
Make a selection: |
```

- Once logged in, the user can select option 'B': 'My job postings.'
- Then option '1': 'Post a new job.'
- User enters the title, description, employer, location and salary.
- Posts the job and displays it under 'job postings', displaying the job ID and title.

Creation of a user profile (1/3)

What would you like to do?

- A. Job Search/Internship
- B. My Job Postings
- C. Find Someone I Know
- D. Learn a New Skill
- E. Useful Links
- F. Important Links
- G. Show My Network
- H. Profile
- I. Friend Requests
- J. Messages

0. Logout and go back

Make a selection: H

Creation of a user profile (2/3)

```
1.Create/Edit Profile  
2.View Profile  
Press any key to return back  
Choose your option: 1█
```

Creation of a user profile (3/3)

```
1.Create/Edit Profile
2.View Profile
Press any key to return back
Choose your option: 1
Enter your title: Software Engineer
Enter your major: computer science
Enter about: Im cool
Enter your university: USF
Enter your degree: computer science
Enter your years attended (YYYY-YYYY): 2020-2024

Enter details for Experience 1 (leave the title empty to skip):
Title: Frontend Engineer
Employer: Issuance
Start: 20223
Invalid date format. Please follow the YYYY-MM-DD format.
Start: 2023-07-25
End: 2023-11-15
Location: Remote
Description: Frontend engineer for a start up

Enter details for Experience 2 (leave the title empty to skip):
Title:
Employer:
Start:
End:
Location:
Description: █
```

Display profile of a friend (1/3)

What would you like to do?

- A. Job Search/Internship
- B. My Job Postings
- C. Find Someone I Know
- D. Learn a New Skill
- E. Useful Links
- F. Important Links
- G. Show My Network
- H. Profile
- I. Friend Requests
- J. Messages

0. Logout and go back

Make a selection: g

Display profile of a friend (2/3)

Your Network:

1. jasangreb – View Profile (press p1) – Send Message (press m1)
2. mikegreb – View Profile (press p2) – Send Message (press m2)

Make your selection:

1. Disconnect from someone
0. Go back

Your choice: p1█

Display profile of a friend (3/3)

Your Network:

1. jasangreb – View Profile (press p1) – Send Message (press m1)
2. mikegreb – View Profile (press p2) – Send Message (press m2)

Make your selection:

1. Disconnect from someone
0. Go back

Your choice: p1

Profile Details:

Username: jasangreb
Title: Software Engineer
Major: Computer Science
About: Im cool
University: Usf
Degree: computer science
Years attended: 2020-2024

Experience 1:

Title: Frontend Engineer
Employer: Issuance
Start: 2023-07-25
End: 2023-11-15
Location: Remote
Description: Frontend engineer for a start up

Press Enter to return...■

Display a posted job (1/4)

Welcome back, mikegreb! | Membership: Plus
What would you like to do?

- A. Job Search/Internship
- B. My Job Postings
- C. Find Someone I Know
- D. Learn a New Skill
- E. Useful Links
- F. Important Links
- G. Show My Network
- H. Profile
- I. Friend Requests
- J. Messages

0. Logout and go back

Make a selection: A

Display a posted job (2/4)

1. List All Jobs/Internships
2. Applied Jobs
3. Saved Jobs

0. Go Back

Enter your selection: 1

Display a posted job (3/4)

List of all Jobs/Interns:

Job ID: 1, Job Title: test job, Status: You own this job post.

=====

To expand a job's details, enter: [job ID]

To apply for a job, enter: apply [job ID]

To add a job to your saved list, enter: save [job ID]

To go back, enter: 0

Enter input: 1

Display a posted job (4/4)

List of all Jobs/Interns:

```
-----  
Job ID: #1  
Title: test job  
Description: this is just a test job!  
Employer: jason  
Location: remote  
Salary: $100,000  
Status: You own this job post.  
-----
```

```
=====
```

To expand a job's details, enter: [job ID]
To apply for a job, enter: apply [job ID]
To add a job to your saved list, enter: save [job ID]
To go back, enter: 0

Enter input:

Apply for a job (1/2)

List of all Jobs/Interns:

Job ID: 1, Job Title: test job, Status: Available.

=====

To expand a job's details, enter: [job ID]

To apply for a job, enter: apply [job ID]

To add a job to your saved list, enter: save [job ID]

To go back, enter: 0

Enter input: apply 1

Apply for a job (2/2)

List of all Jobs/Interns:

Job ID: 1, Job Title: test job, Status: Available.

=====

To expand a job's details, enter: [job ID]

To apply for a job, enter: apply [job ID]

To add a job to your saved list, enter: save [job ID]

To go back, enter: 0

Enter input: apply 1

Enter your graduation date (mm/dd/yyyy): 06/25/2024

Enter your start date (mm/dd/yyyy): 07/25/2024

Explain why you are a good fit for this job: I'm a good software engineer

See a list of jobs that have been applied for (1/3)

What would you like to do?

- A. Job Search/Internship
- B. My Job Postings
- C. Find Someone I Know
- D. Learn a New Skill
- E. Useful Links
- F. Important Links
- G. Show My Network
- H. Profile
- I. Friend Requests
- J. Messages

0. Logout and go back

Make a selection: A

See a list of jobs that have been applied for (2/3)

```
-----  
[Notification] – You have applied for 1 job(s).  
-----
```

1. List All Jobs/Internships
2. Applied Jobs
3. Saved Jobs

0. Go Back

Enter your selection: 2█

See a list of jobs that have been applied for (3/3)

```
Jobs/Interns You Have Applied For:
```

```
Job ID: 1, Job Title: test job
```

```
=====
```

```
Enter any key to go back: █
```

See a list of jobs that have been saved (1/3)

Welcome back, mikegreb! | Membership: Plus
What would you like to do?

- A. Job Search/Internship
- B. My Job Postings
- C. Find Someone I Know
- D. Learn a New Skill
- E. Useful Links
- F. Important Links
- G. Show My Network
- H. Profile
- I. Friend Requests
- J. Messages

0. Logout and go back

Make a selection: A

See a list of jobs that have been saved (2/3)

1. List All Jobs/Internships
2. Applied Jobs
3. Saved Jobs

0. Go Back

Enter your selection: 3

See a list of jobs that have been saved (3/3)

Saved Jobs/Interns:

Job ID: 1, Job Title: test job

=====

To remove a job from your saved list, enter: remove [job_id]

To go back, enter: 0

Enter an input:

Notifications - Overview

You have successfully logged in!

[Notification] - New students in InCollege:
Student John Doe has joined InCollege!

[Notification] - Remember - you're going to want to have a job when you graduate. Make sure that you start to apply for jobs today!

[Notification] - Don't forget to create a profile

Welcome back, eloy2! | Membership: Plus
What would you like to do?

- A. Job Search/Internship
- B. My Job Postings
- C. Find Someone I Know
- D. Learn a New Skill
- E. Useful Links
- F. Important Links
- G. Show My Network
- H. Profile
- I. Friend Requests
- J. Messages

Ø. Logout and go back

Make a selection: █

When logging in, the user is presented with a variety of notifications with important updates, reminders and suggestions for using InCollege effectively.

Notifications include:

- Friend requests
- Reminders
- Newly joined users
- New messages
- New jobs posts
- Updates on the jobs you have applied for or saved.
- And more!

Some Notifications You Might Encounter

```
-----  
[Notification] - New students in InCollege:  
Student John Doe has joined InCollege!  
-----
```

```
-----  
[Notification] - You have 2 message(s).  
-----
```

```
-----  
[Notification] - Don't forget to create a profile  
-----
```

```
-----  
[Notification] - New job 'Software Engineer' has been posted  
-----
```

```
-----  
[Notification] - You have 1 pending friend request(s). Go to the 'Friend Requests' tab to accept/reject.  
-----
```

```
-----  
[Notification] - Remember - you're going to want to have a job when you graduate. Make sure that you start to apply for jobs today!  
-----
```


Plus Membership

InCollege also offers a Plus Membership Plan! For only \$10/month, users get access to exclusive features that will further improve the use of the platform.

- You can sign up for the Plus Membership when creating your account

```
* InCollege now offers a Plus Membership! Plus members get access to a variety of exclusive features for only $10/month. *  
Would you like to opt in to InCollege+? (y/n): █
```

- Membership Status will be shown at the top of the home page after you log in

```
Welcome back, john! | Membership: Plus  
What would you like to do?
```

```
A. Job Search/Internship  
B. My Job Postings  
C. Find Someone I Know  
D. Learn a New Skill  
E. Useful Links  
F. Important Links  
G. Show My Network
```

```
Welcome back, mary! | Membership: Standard  
What would you like to do?
```

```
A. Job Search/Internship  
B. My Job Postings  
C. Find Someone I Know  
D. Learn a New Skill  
E. Useful Links  
F. Important Links  
G. Show My Network  
H. Profile
```

Send a Message to Any User in the System!

Normally, to send a message to another user of InCollege one must first be in their Network. But with InCollege+ you are able to message any user in the system.

- As can be seen here, while Mary is only able to message her friend John, John is able to message not only Mary, but any other user thanks to his Plus Membership!

```
Inbox:

1. View Messages
2. Send a Message

0. Go back

Make your selection: 2
-----
Available Users:
1. john
-----
* Want to reach more people? Consider subscribing to InCollege+ *
Enter the username of the person you want to message (X to cancel): █
```

```
Inbox:


1. View Messages
2. Send a Message

0. Go back

Make your selection: 2
-----
Available Users:
1. john
2. mary
3. tyler
4. goku
5. marvin
6. luke
-----
Enter the username of the person you want to message (X to cancel): █
```


InCollege Modules

 ascii_art.py


 auth_test.py


 auth.py

 database.py


 driver_test.py


 history 2.txt


 home_test.py

 home.py


 incollege_database.db

 jobs_test.py

 jobs.py

 language_test.py


 main.py

 mock_incollege_datab


 notifications_test.py

 notifications.py


 pages_test.py

 pages.py


 README.md


 social_test.py

 social.py

 student_profile_test.py

 student_profile.py

 success_story.py

 util.py

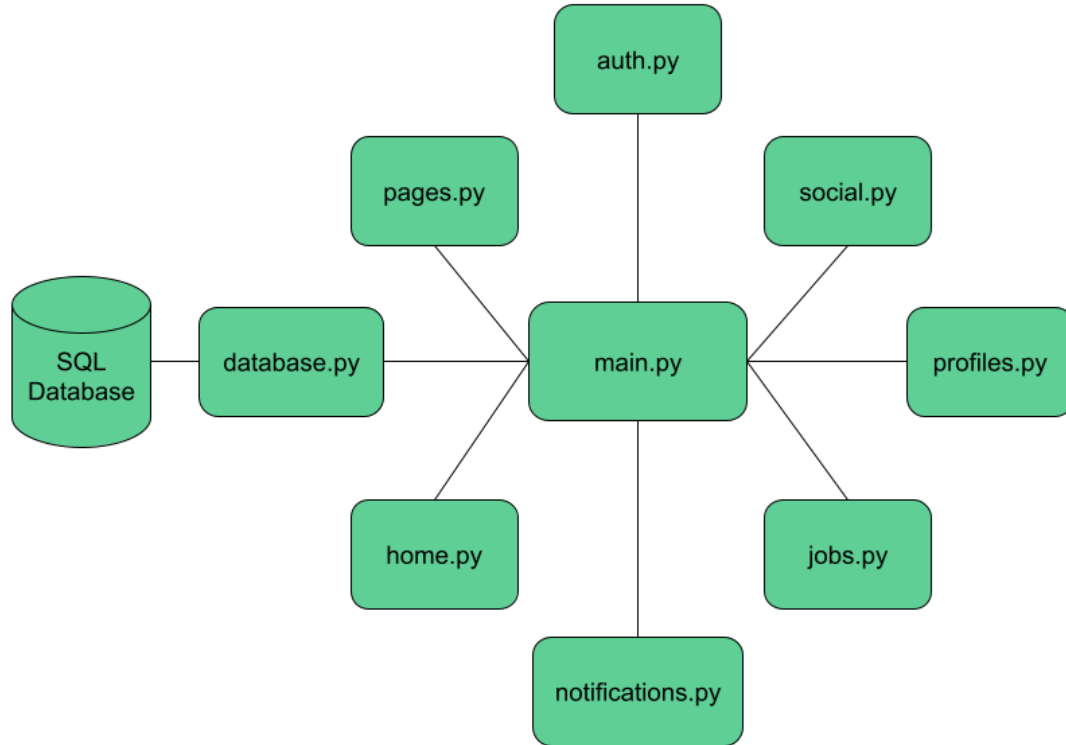
InCollege Modules

- `auth.py`
 - User sign up, log in, password validation, and authentication logic.
- `database.py`
 - A class to model our database schema, which includes methods for reading, writing, and updating the SQLite database.
- `home.py`
 - Homepage that displays the InCollege interface after a user logs in.
- `main.py`
 - Driver code that serves as the main program loop.
- `jobs.py`
 - All the job-related logic, including job listing, applying, saving, posting, and deleting jobs.

InCollege Modules

- **notifications.py**
 - A class that retrieves all the notifications and displays them after a user logs in.
- **pages.py**
 - Contains the code that organizes all the links/pages such as general links, guest controls, sign up page, app policies, etc. We built a data structure that connects all the links and the user can navigate between them.
- **social.py**
 - Contains the logic that allows students to connect with each other, send and view messages, and manage their connections.
- **student_profile.py**
 - Functions for the student profile feature, such as create, edit, save, and display profile.
- **util.py**
 - Utility module that handles clearing the terminal, date and time formatting and validation, and other helper functions used across the entire program.

InCollege Software Architecture



Data Storage and Retrieval

- All of our data is being stored in an SQLite database.
- We have a database class that encapsulates all the SQL queries.

```
self.cursor.execute(
    "CREATE TABLE IF NOT EXISTS college_students (id INTEGER PRIMARY KEY AUTOINCREMENT, username TEXT UNIQUE, firstname TEXT, lastname TEXT, pass TEXT, major TEXT, location TEXT)"
)
self.cursor.execute(
    "CREATE TABLE IF NOT EXISTS connections (user1 TEXT, user2 TEXT, PRIMARY KEY(user1, user2))"
)
self.cursor.execute(
    "CREATE TABLE IF NOT EXISTS pending_connections (requester TEXT, requestee TEXT, PRIMARY KEY(requester, requestee))"
)
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS job_posts (id INTEGER PRIMARY KEY AUTOINCREMENT, firstname TEXT, lastname TEXT, title TEXT, description TEXT, employer TEXT, location TEXT)"""
)
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS job_applications (id INTEGER PRIMARY KEY AUTOINCREMENT, job_id INTEGER, student_id TEXT, graduation_date TEXT, start_date TEXT, end_date TEXT)"""
)
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS job_saved (job_id INTEGER, student_id TEXT, FOREIGN KEY (job_id) REFERENCES job_posts(id) ON DELETE SET NULL)"""
)
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS student_profiles (username TEXT PRIMARY KEY, title TEXT, major TEXT, about TEXT, title1 TEXT, employer1 TEXT, start1 TEXT, end1 TEXT)"""
)
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS messages (id INTEGER PRIMARY KEY AUTOINCREMENT, sender INTEGER, receiver INTEGER, message TEXT, time DATETIME DEFAULT CURRENT_TIMESTAMP)"""
)
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS notifications (id INTEGER PRIMARY KEY AUTOINCREMENT, student_id TEXT, message TEXT, read INTEGER default 0)"""
)
```

Data Storage and Retrieval

- The database class has several methods that execute SQL queries in order to read, update, and insert information.
- Having this layer of abstraction allows us to separate the application logic from the database access. This contributes to code maintainability and efficiency.

```
def mark_all_notifications_as_read(self, student_id):  
    #mark all notifications as read where student id is student_id  
    self.cursor.execute(  
        "UPDATE notifications SET read = ? WHERE student_id = ?", (1, student_id)  
    )  
  
    self.connection.commit()
```

```
def get_unread_messages(self, receiver_id):  
    if not receiver_id:  
        return []  
    self.cursor.execute(  
        "SELECT * FROM messages WHERE read = ? AND receiver = ?", (0, receiver_id)  
    )  
    return self.cursor.fetchall()
```


What security features have been incorporated into your program?

We have taken several security measures on different levels:

1. During the login process, using a good password is highly advocated and user needs password to login into the account(auth.py file for more details)
2. A person needs to have a special username, using the UNIQUE command on SQLite ensures that no other person with the particular username can create a new account
3. Other database tables also have a similar security, where they are assigned to a special ID that differentiates one from the other
4. Taking care of special edge cases that might crash the software

Some snippet examples of the security measures

taken by our team:

```
self.cursor.execute(
    "CREATE TABLE IF NOT EXISTS college_students (id INTEGER PRIMARY KEY AUTOINCREMENT, username TEXT UNIQUE,
)
```

```
def get_job_by_id(self, job_id):
    self.cursor.execute("SELECT * FROM job_posts WHERE id = ?", (job_id,))
    return self.cursor.fetchone()
```

```
self.cursor.execute(
    """CREATE TABLE IF NOT EXISTS job_posts (id INTEGER PRIMARY KEY AUTOINCREMENT,
)
```

```
def display_applied_jobs(user_info):
    all_jobs = db.get_jobs()
    applied_jobs_ids = db.get_applications_of_student(user_info["id"])
```

```
print("Jobs/Interns You Have Applied For:\n")
for job in all_jobs:
    if job["id"] in applied_jobs_ids:
        print(f"Job ID: {job['id']}, Job Title: {job['title']}")
print("\n=====")
```

```
input("Enter any key to go back: ")
clear_terminal()
```

```
def validate_password(password):
    regex = r"^(?=.*[A-Z])(?=.*\d)(?=[_<_+:@#%&!]?^*)$.{8,12}$"
    return bool(re.match(regex, password))
```

```
def create_account(
    db, username, password, firstname, lastname, major, university, plus_tier
):
    if not firstname or not lastname:
        raise ValueError(
            "First name and Last name are required according to new InCollege rule"
        )
```

```
is_plus_tier = bool(re.match(r"^(y|yes)$", plus_tier, re.IGNORECASE))
```

```
if validate_password(password):
    number_accounts = db.get_number_of_accounts()
    if number_accounts < 10:
        try:
            db.add_new_student(
                username,
                firstname,
                lastname,
                password,
                major,
                university,
                is_plus_tier,
            )
            print("You have successfully created an account!\n")
        except sqlite3.IntegrityError:
            print("Error: User already exists. Please try another username.\n")
    else:
        print(
            "Error: All permitted accounts have been created. Please come back later.\n"
        )
    else:
        print(
            "Error: Invalid password.\n"
            "Your password must meet the following criteria:\n"
            "- Be between 8 and 12 characters long.\n"
            "- Contain at least one uppercase letter.\n"
            "- Contain at least one digit.\n"
            "- Contain at least one special character.\n"
        )
)
```

How many test cases have been developed for your program?

As of now, we have incorporated **106** test cases!

Scope of Testing:

- User Authentication
- Functionality and Performance
- Stability on all machines

Quality Assurance:

- Each test case aims quality and reliability.
- Regular updates and additions to the test suite to cover new features.

User Experience:

- Committed to delivering a seamless and bug-free user interface.
- Continuous testing to refine user interaction and system response.

```
123-InCollege> py -m pytest
===== test session starts =====
platform win32 -- Python 3.10.11, pytest-7.4.2, pluggy-1.3.0
rootdir: C:\Users\sahed\OneDrive\Desktop\New folder\SWE-Proje
ct-Fall123-InCollege
plugins: anyio-4.0.0, mock-3.11.1
collected 106 items

auth_test.py ..... [ 8%]
driver_test.py .. [ 10%]
home_test.py ..... [ 15%]
jobs_test.py ..... [ 29%]
language_test.py .. [ 31%]
notifications_test.py .... [ 34%]
pages_test.py ..... [ 62%]
social_test.py ..... [ 87%]
student_profile_test.py ..... [ 98%]
video_and_story_test.py .. [100%]

===== 106 passed in 1.02s =====
```

Thank you for taking your time!