Team Project:
Deliverable 3 – Project Phase 1
CSCE 5430 (Spring 2023)

Project Title:
Online Library Management System

Group Name:
Penguins

Group Members:

No.	Name	Student ID	Email
1	Ashraf Syed	11618668	AshrafSyed@my.unt.edu
2	Padmini Kuchukulla	11605145	KuchukullaPadminiLNU@my.unt.edu
3	Abhijith Reddy Mandagiri	11603247	AbhijithReddyMandagiri@my.unt.edu
4	Deepna Thalanki	11644830	deepnathalanki@my.unt.edu
5	Raghu Vamsi Kondapalli	11663360	raghuvamsikondapalli@my.unt.edu
6	Vamsi Venkat Manepalli	11656561	Vamsivenkatmanepalli@my.unt.edu
7	Sai Vishwak Jadala	11673439	saivishwakjadala@my.unt.edu

3.Report

a.Requirements

FUNCTIONAL REQUIREMENTS:

Landing Page:

Description of feature:
Gives the user a choice, whether to join as an admin or a student as per credentials.
Functional Requirements :
On choosing the respective options, each of the buttons redirects the user to the login page required for accessing the library system for utilization.

New User Registration:

Description of feature:
All users i.e students in this context have the option to use this feature to provide their contact information when creating accounts. This information includes name, email address, and contact number etc as part of the registration process.
Functional requirements
Possible feature for users to protect their account privacy is the ability to create strong passwords. This requirement allows for safe browsing and secure use of the system for students and protects their privacy.

The system must be able to validate data and establish whether all of the data was provided by the user, among other functional requirements. This mode will be more beneficial with the backend part of the website being embedded with the front end. Which will happen in the deliverables to come.

User Login Page:

Description of feature:
The user logs into the system using this function. Before logging on, they must input their user ID and password for authentication. The user cannot access the system if they are invalid.
If a user forgets their password, the system should offer a mechanism to reset it. This is accomplished by sending a reset code to their registered cellphone number. For this deliverable we will be preparing only the front end pages that will be used for all the

above mentioned functionalities. The implementation will be possible in the upcoming phases of the project.

Functional requirements
When a user registers, they supply a user id(similar to that of student id). To access the system, the user must identify themselves with a valid user id and password. The system then executes an authorization procedure to determine what user level has access.

The user must have the capacity to log out after they are done using the system. The system should restrict access to specific features or data based on the user's role in the library system, such as a librarian or a user.

Search Book Page:

Description of feature
Both admins and students have access to this function. We may search for books using the book's ID, name, category, or author. But each type of user has different use cases with the search feature of the website.
Functional requirements
The database must be searchable by the system using a chosen search type.

Admin User:

Register New Book:

Description of feature
With this feature, the administrator may include new books in the library. Based on the necessities of the students, the admin has this feature to add new books to the system for expanding the horizon of the options for the students.

Functional requirements
Information verification capabilities for systems are required. Only authenticated users are allowed to access this page and use the feature for the required purposes.

View Registered Students:
Description of feature:
The admin user has a page to view a report of the students who have registered with the system. This allows the admin to keep track of all the interactions a given student is having with the system and maintain stability of the library inventory.

Functional requirements
The page will have a search feature that enables the admin to track a particular student using their name or student ID number. This allows the admin to check for deadlines and potential dues that the student owes to the library

Student User:

View Books Issued to the user :
Description of feature
The system should make it possible for the student to examine their personal data, including their name, student ID, contact details, and borrowing history. Students can view the books that have been issued to them using this page.

Functional requirements
If the item is not overdue and has not been requested by another user, the system should enable the student to renew it. This would allow for the student to retain the book without causing any conflicts with other requests for that particular book.

Performance Requirements
As of this phase of the project, there cannot be a performance requirement as the front end alone cannot determine any quantifiable measure of the performance of the system. Hence, these requirements will be introduced in the later phases of the project.

Design Constraints

Web support
Web requirement is a major support which is needed as the entire application will be loaded on to the cloud. Internet access and server maintenance play a significant role in the functionality of the project. As of this phase, the front end constraints would be version mismatch between browsers and the script code used for the front end development.

Security
This being only phase 1 of the project, the project will not be having any security measures. Since it's only a front end phase, there will not be a need for security since it's only on a local system.

NON-FUNCTIONAL REQUIREMENTS:

Requirements Of Product :

Requirement Efficiency :
Staff members of Library and patrons will have quick access to books due to the computerization of library operations like issuing, tracking and fine calculation etc.

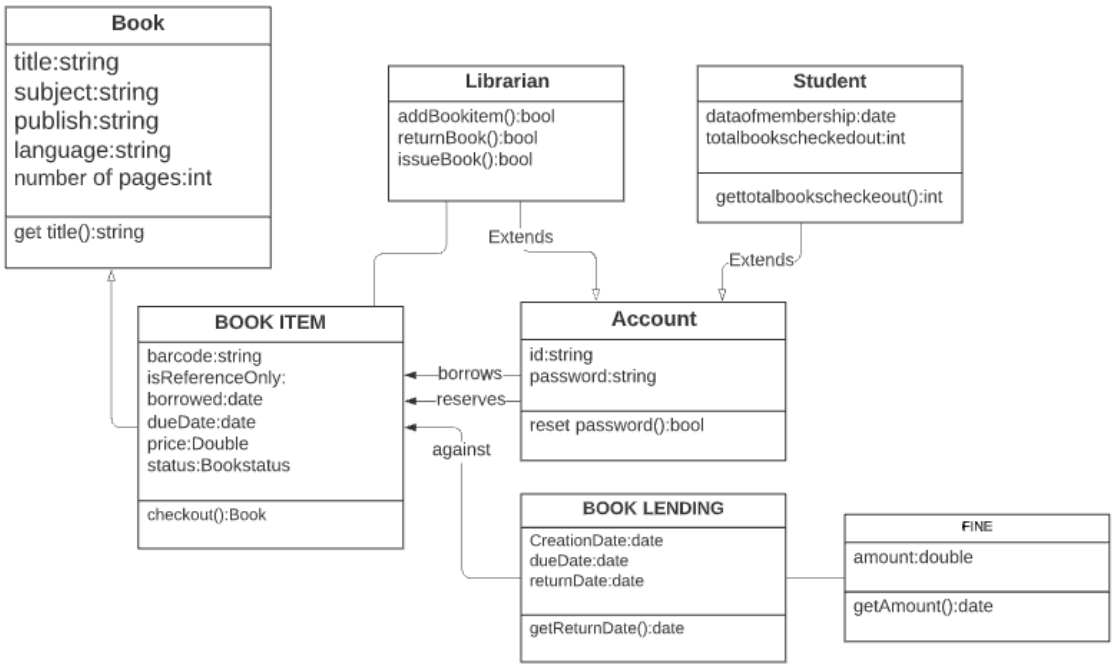
Reliability Requirement :
The system will accurately be able to handle member registration, member validation, book classification, transaction and search, and fine calculation, if necessary and any other tasks of the project. The project being sophisticated and reliable becomes our major priority

Demand for Usability :
The web has been designed in a simple, sophisticated and user friendly manner. Any user, be it admin or student will find it quite easy and convenient to navigate through the site to complete any of their tasks or jobs for which they are using the site.,

Scalability :
The system will be designed keeping in mind future potential for growth and changes to the functionalities. The desire would be to keep it scalable to software growth and also to the users threshold for the website.

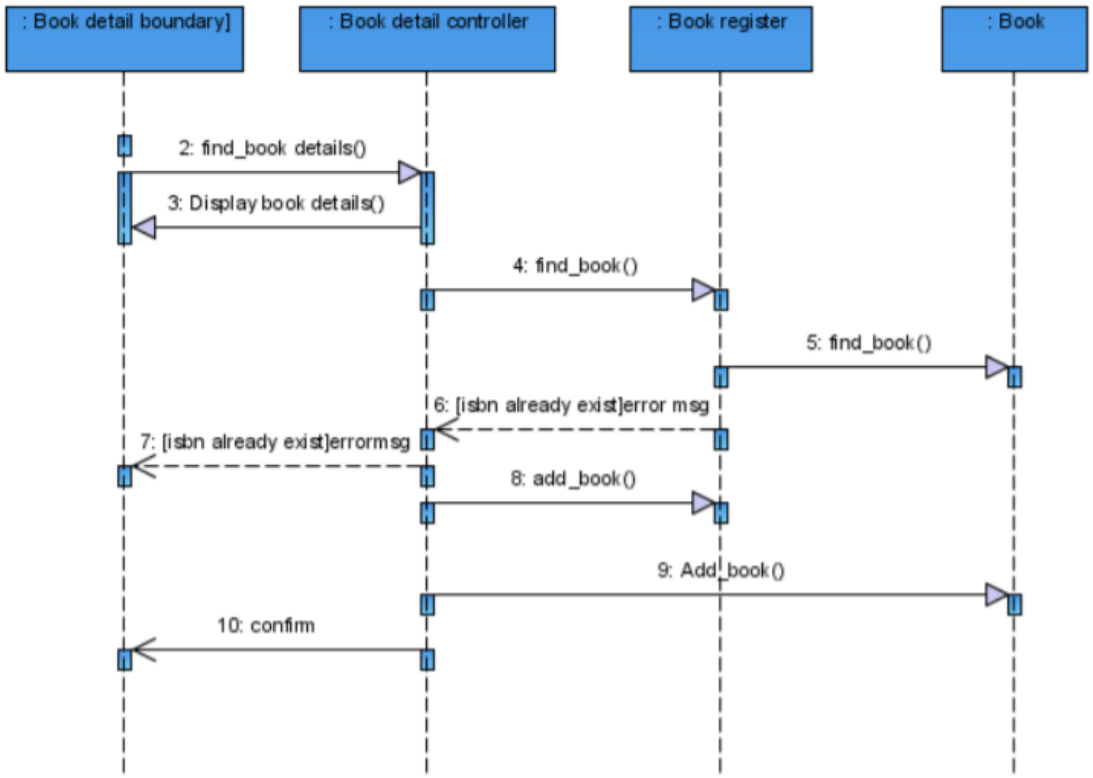
b.UML Design

Class diagram - It shows the static view of the system. Here we have multiple components like book, librarian, student, book item, account, book lending and fine. Each component has multiple entities associated with it.

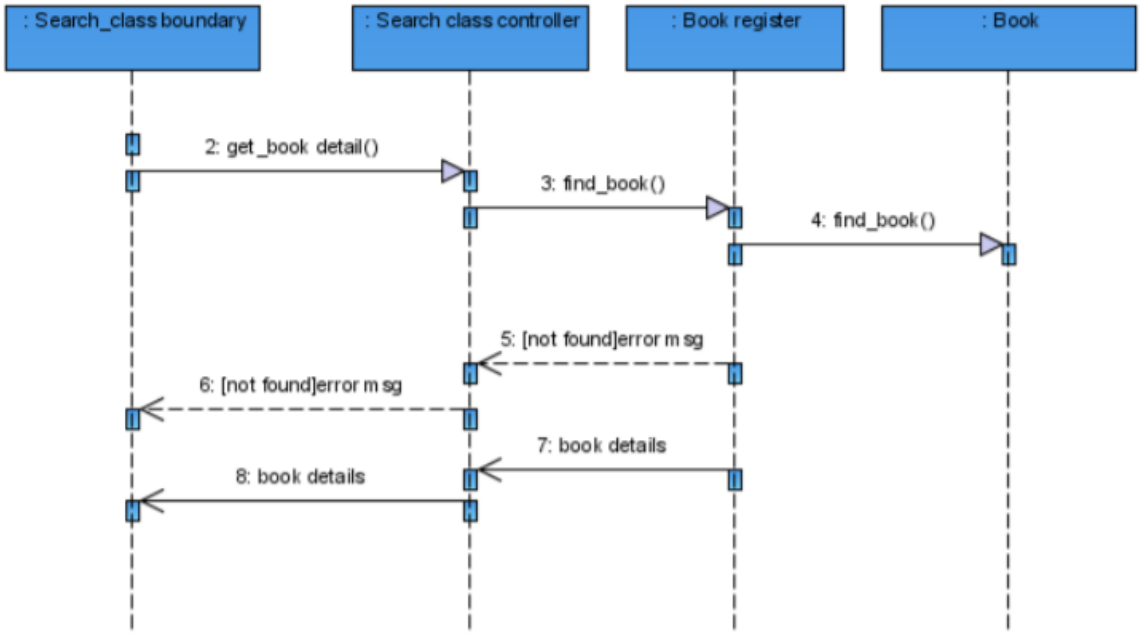


Sequence diagram - It shows the time sequence associated with the object interactions.Here we have multiple use cases like handling book details, search books and issue books. Below are the sequence diagrams.

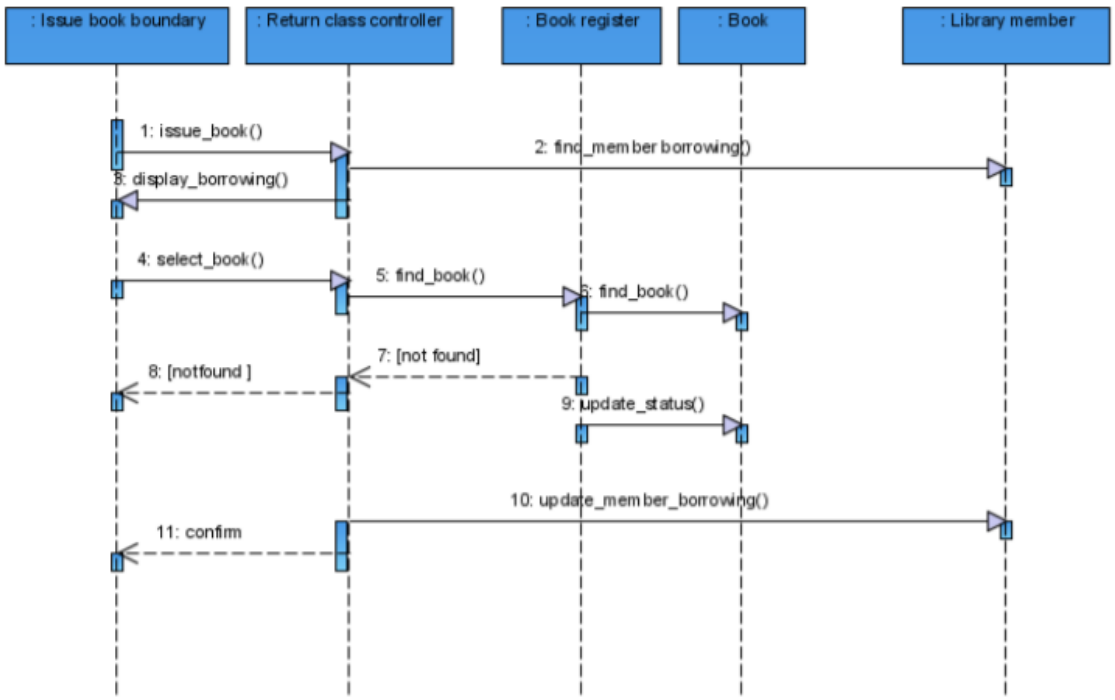
Handle Book Details



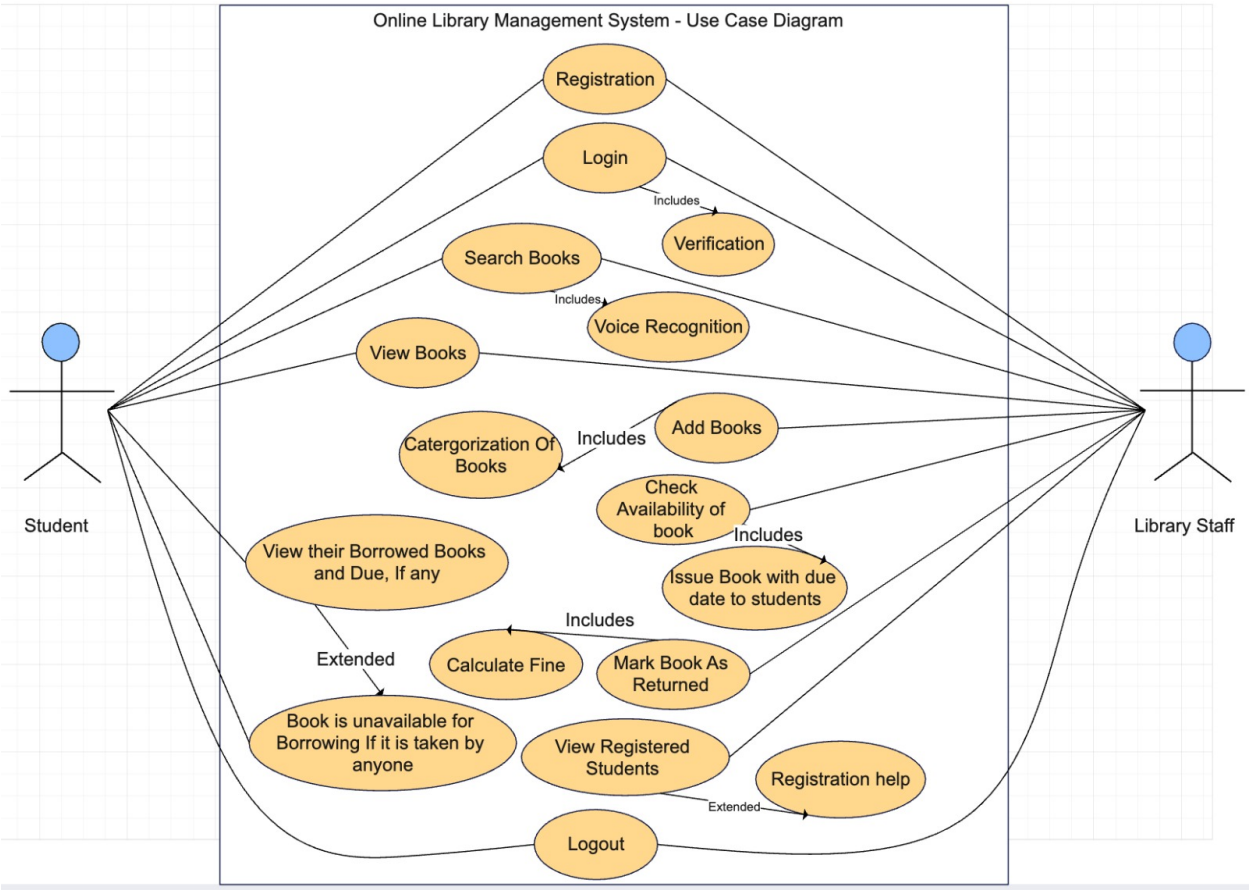
Search Books



Issue Book



Use case diagram - This shows the relations between use cases and actors. Below is the diagram that depicts how a student can do multiple actions in the Online Library Management System



c. Test Cases

1. Student Login Page Test Case:

The Test case script for the Student login page is

```
<script>

     const loginForm = document.getElementById("login-form");

     const loginButton =
document.getElementById("login-form-submit");

     loginButton.addEventListener("click", (e) => {

     e.preventDefault();

     const username = loginForm.username.value;

     const password = loginForm.password.value;

     if (username === "penguins" && password === "penguins") {

          alert("You have successfully logged in " + username);

          location.reload();

     } else {

          alert("Invalid Credentials");

     }

     })

</script>
```

This is an HTML code for a student login page. It uses Bootstrap to style the page and includes a form for users to input their username and password. If the user enters incorrect login credentials, an alert message will appear. The page also includes a link for users to sign up if they do not have an account. The code includes JavaScript that listens for a click event on the login button, preventing the default form submission and obtaining the values entered in the username and password fields.

The Above Script is to validate the Login page for the student and to check whether the page is working as intended. As it is a Static page the username and password are set to “**penguins**” and “**penguins**” to test. Further, in the dynamic page, the username and password are retrieved from the backend server and validated to login.

The Below screenshot shows the page response when an invalid username and password are entered. Thus this shows the functionality of the test script.

When the correct inputs that is “**penguins**” and “**penguins**” as username and password are given to the page the page response is as below.

> This shows the successful validation of the user and sends an alert message of the successful login.

2. Admin Login Page Test Case:

The Test Case script for the Admin login page is

```
<script>

   const loginForm = document.getElementById("login-form");

     const loginButton =
     document.getElementById("login-form-submit");

        loginButton.addEventListener("click", (e) => {

            e.preventDefault();

            const username = loginForm.username.value;

            const password = loginForm.password.value;

        if (username === "admin" && password === "admin") {

                alert("You have successfully logged in.");

                window.location.href = "afterlogin.html";

            } else {

                alert("Invalid username or password. Please
try again.");

            }
```

```
        });

    </script>
```

This code is a JavaScript function that handles a login form submission. It checks if the entered username and password match the expected values, and if so, it redirects the user to a page called "afterlogin.html". Otherwise, it displays an error message indicating that the username or password is invalid. As a test case, one could try entering a valid username and password and verify that the function correctly logs in the user and redirects them to the desired page. Additionally, one could try entering an invalid username or password and verify that the function correctly displays an error message and does not redirect the user to the afterlogin.html page. One could also test the behavior of the function when the login form is submitted without any input or with incomplete input, and verify that it behaves correctly in these cases as well.

The Admin Login page response when invalid input is given as shown below. This shows that the validation is working as intended.

The Admin login page works as intended with valid authentication when “**admin**” and “**admin**” as input for the username and password in their respective fields. The page response is shown in the snippet below.

On successful authentication the page redirects to the afterlogin.html User interface.

d. User Manual

Since our project is designing a web page for Online Library Management System, the end user is provided with the following link
http://127.0.0.1:8000/, which navigates to our web page.

The home page of our website for the end user looks like

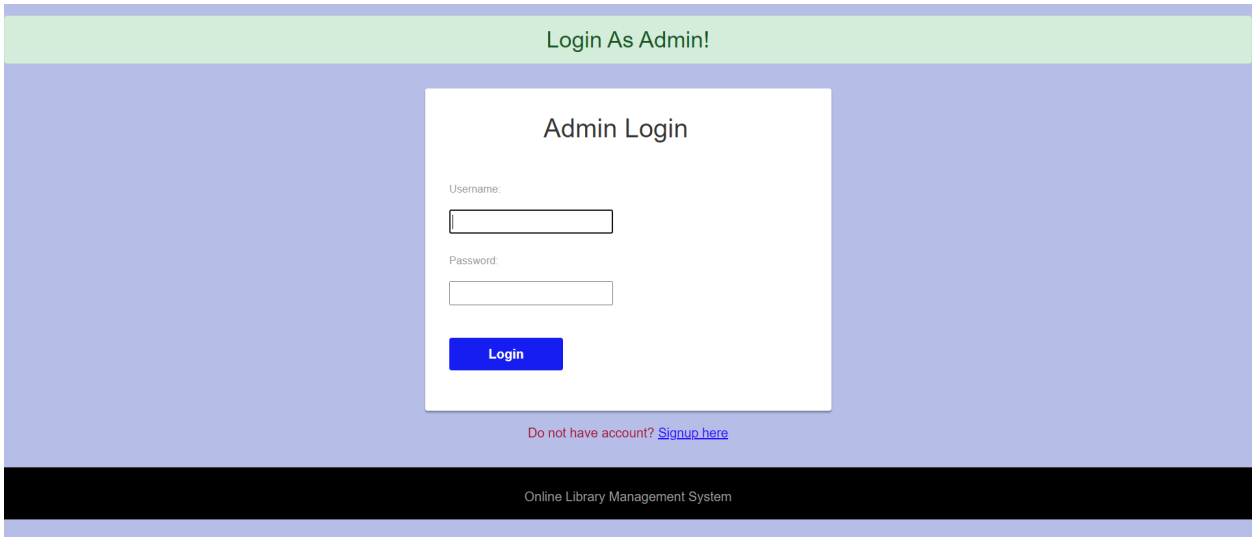
As he/she scrolls down, will be able to see the following, where there are two different portals i.e, Admin and Student.

Now, if he/she wants to enter into the Admin portal, then he/she can either click the Admin button which is in blue color or directly the Admin button which is on the top left corner of the screen.Then, he/she can view the following

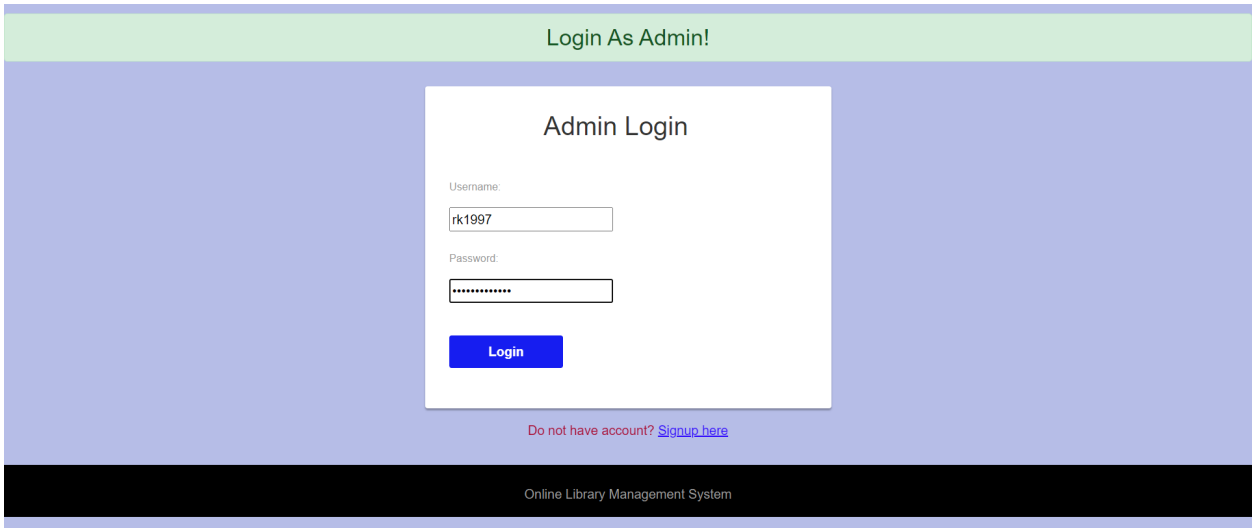
If the user is using our website for the first time, he/she can click on the signup. This redirects the user to the admin signup page, which looks like the following

Provide the required details for signing up as shown below.

After giving details, one can click on the Sign Up button. Then, it redirects to the login page as below.



Give the details as given in the previous page to login as admin.



After successful login, one can view what all the functionalities can be performed by admin as below.

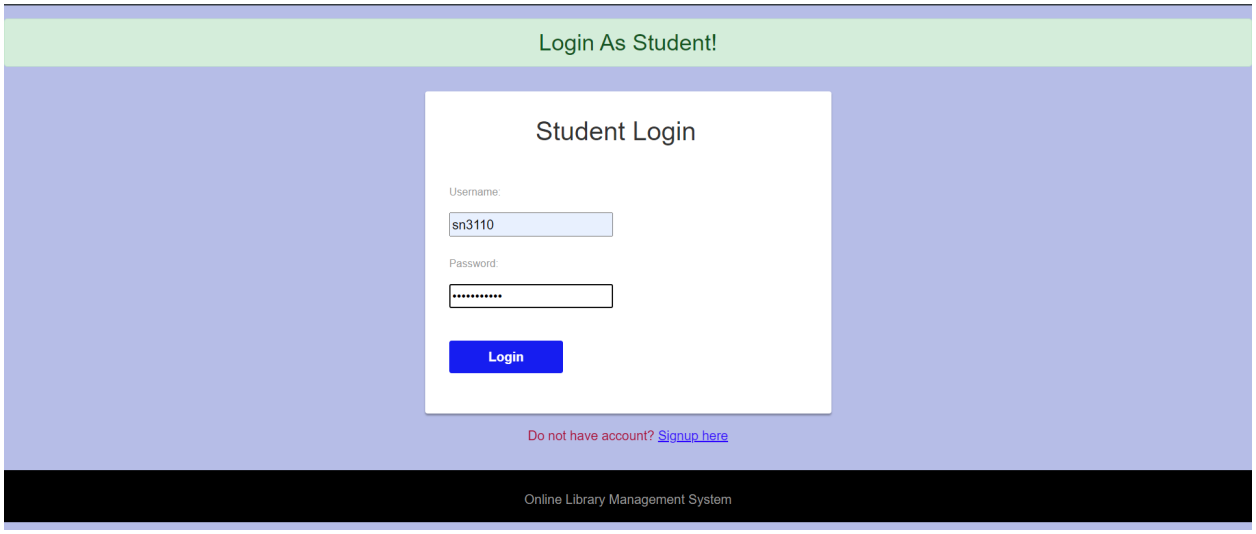
Now, to get into the student portal click on the logout button on the top right corner of the screen, then one redirects to the home page as below.

Now the user can either click the Student button which is in blue color or directly the Student button which is on the top left corner of the screen.Then, he/she can view the following screen.

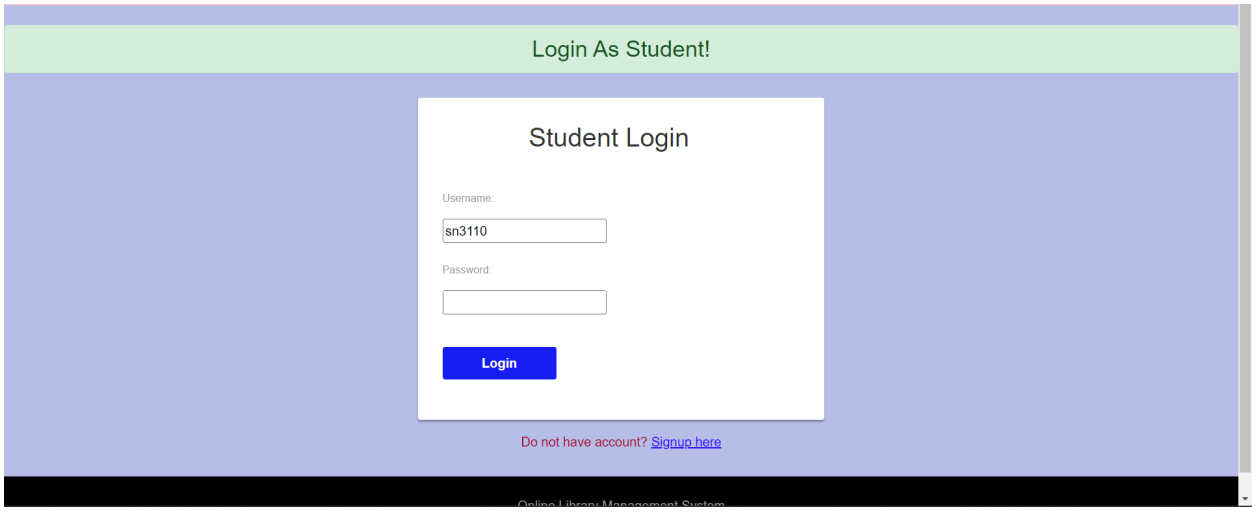
If the student is using our website for the first time, he/she can click on the signup. This redirects the user to the student signup page, which looks like the following.

Provide the required details for signing up as shown below.

After giving details, one can click on the Sign Up button. Then, it redirects to the login page as below.



If credentials don't match the previous page, then there comes a pop up as shown below.



After successful login, one can view what all the functionalities can be performed by the student as below.

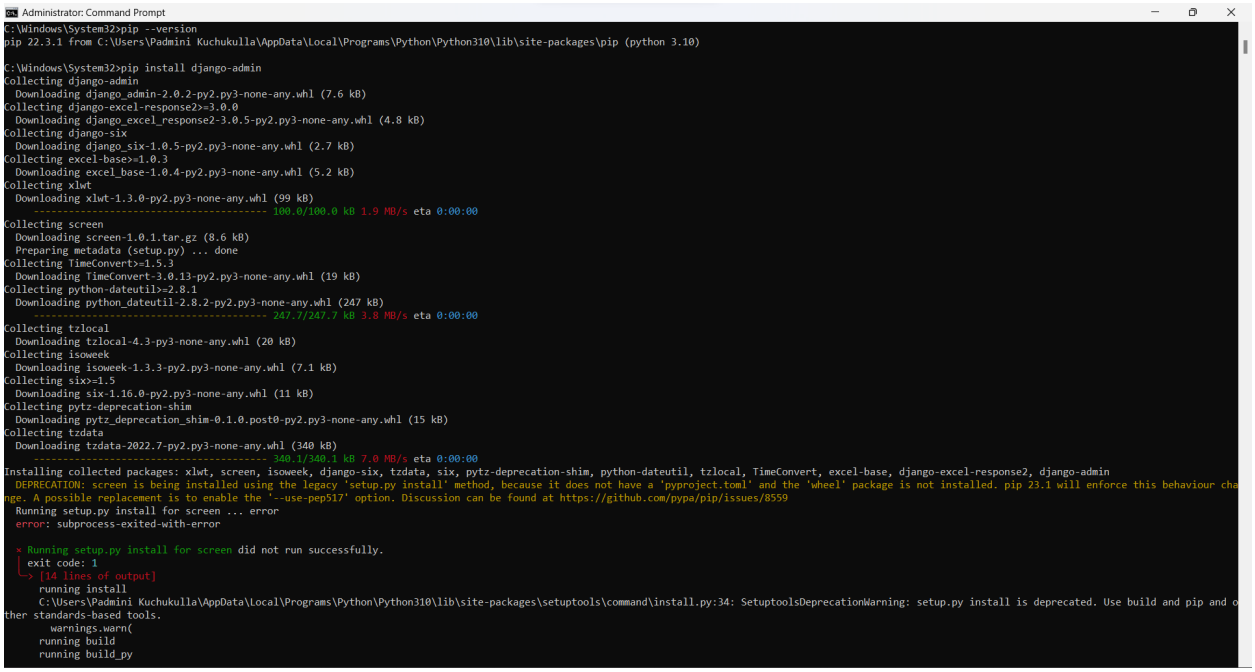
e. Compilation Instructions

For the ease of connecting backend and frontend, we are using the django framework to build our project.

To compile/execute our project, the basic requirement is the system you are working on should have python.

If there is no Python, then download and install the latest version, here we are using Python 3.10 using the following link.
https://www.python.org/downloads/

After successful installation of Python, installed django-admin as shown below using following command
`pip install django-admin`

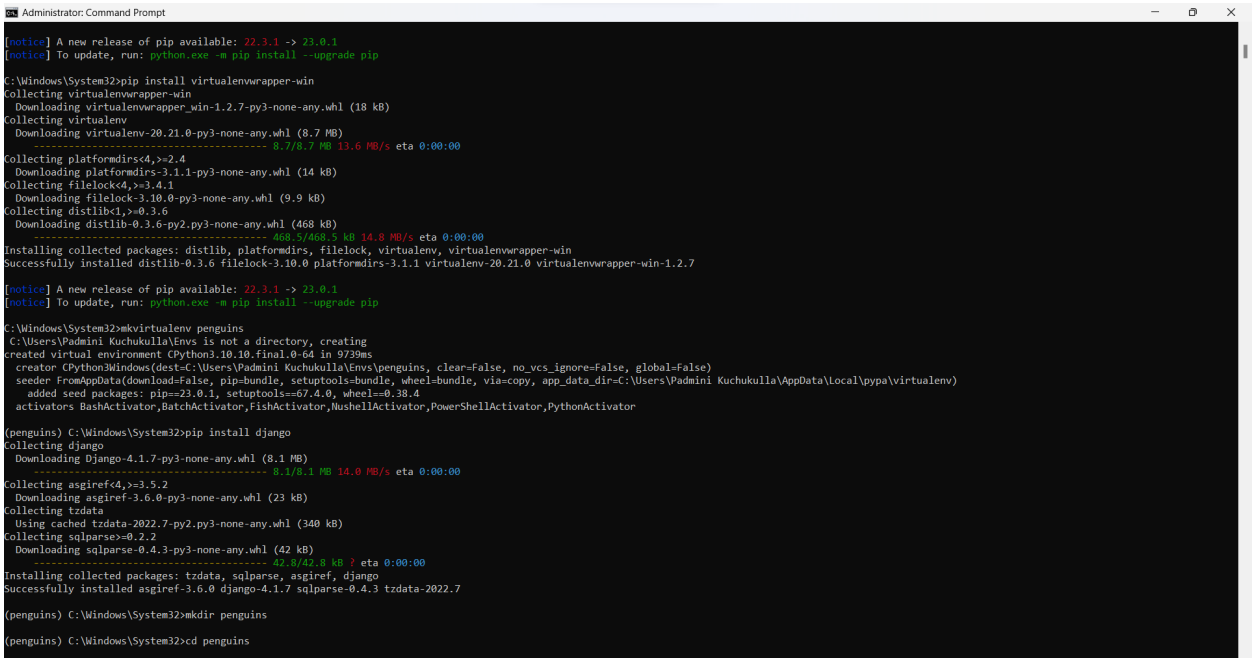


After this installed the virtualwrapper, this is because we are creating our whole project in separate virtual environment using the following command
`pip install virtualenvwrapper-win`

After installing the wrapper for windows, now created an environment for our project using the following command
`mkvirtualenv penguins`

As soon as we create the virtual environment named “penguins”, we will start working on the environment.
Now, in the environment we have installed django using the following command.

```
pip install django
```



Now, after successful installation of django, we created the directory named “penguins” in the virtual environment using the following command

```
mkdir penguins
```

Now, change the path to the newly created directory using the following command

```
cd penguins
```

Now, in this directory we start our project using the django framework command which is as follows

```
django-admin startproject OnlineLibraryMangementSystem
```

After successful creation of the project directory, change the working path to this directory using the following command

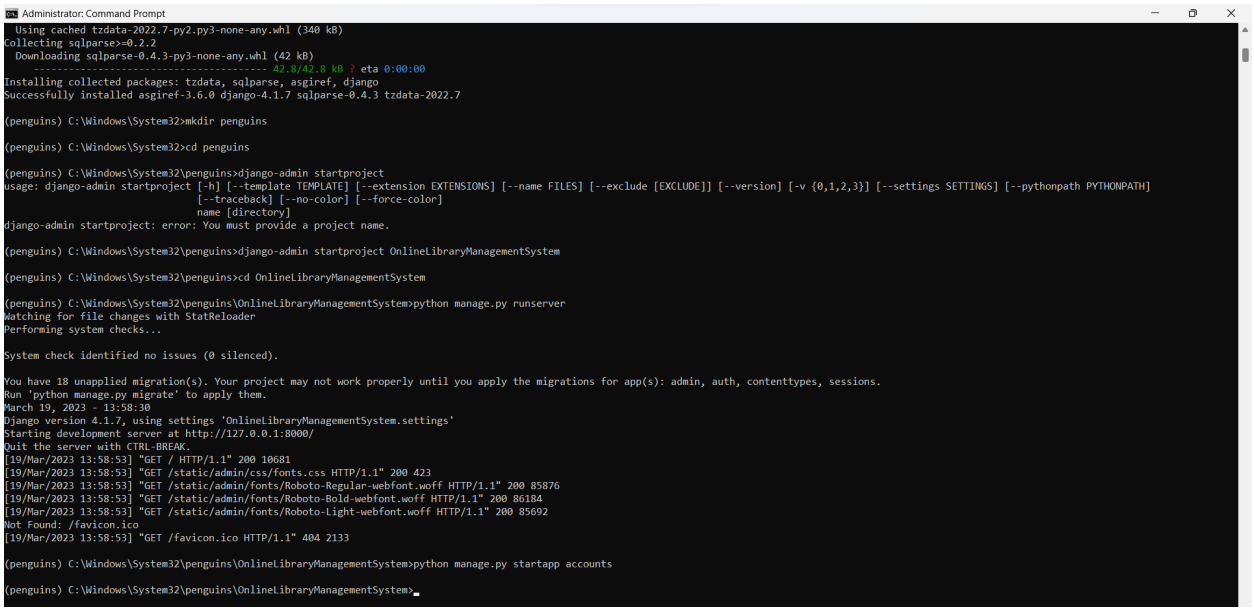
```
cd OnlineLibraryMangementSystem
```

Now, to get the webpage link, we used python command which is as follows

```
python manage.py runserver
```

The manage python file is created by django in default when we start the project.

In the output we can see our website link as http://127.0.0.1:8000/



Now, when we open the link, we can just see that installation is successful.
Then, download and install postgresql, here we downloaded the 14.7 version using the following link
https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

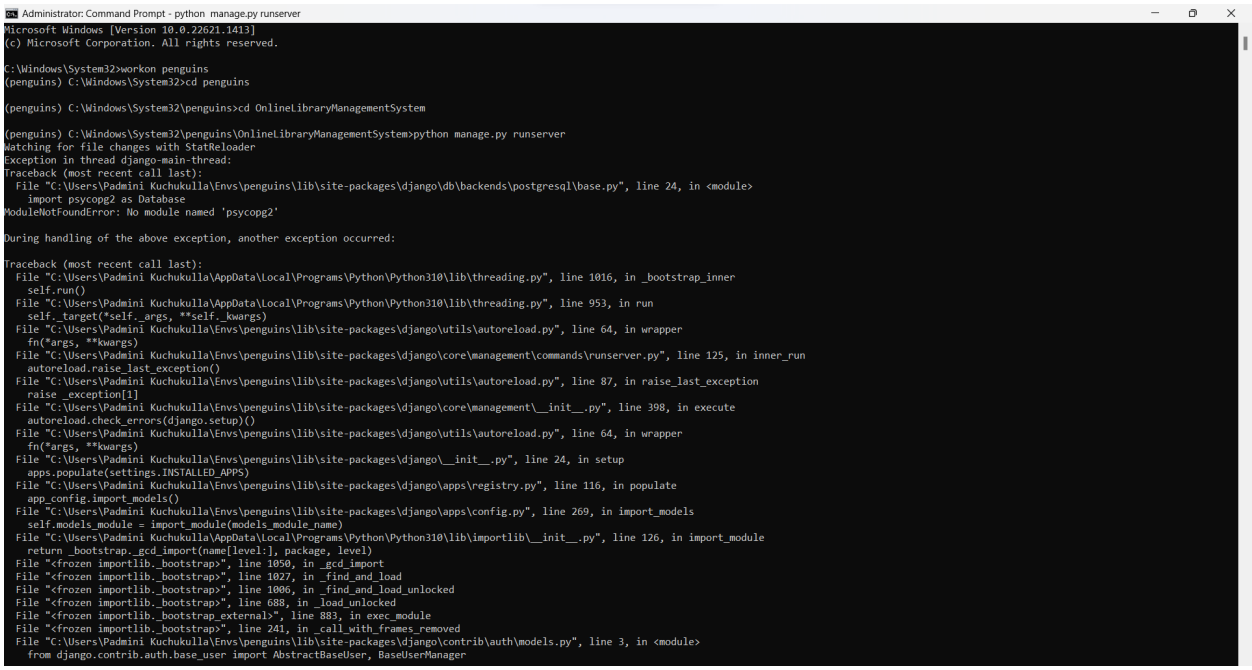
After successful installation of postgresql, now download and install pgadmin4, here we downloaded the 6.19 version using the following link
https://www.postgresql.org/ftp/pgadmin/pgadmin4/v6.19/windows/

Now, we created a folder named “templates”, which contains all the html files required for our project, which are coded separately by all of our teammates.

Then, we also created a folder named “static”, which contains another folder inside it named “images”, which contains all the images we used in developing the frontend.

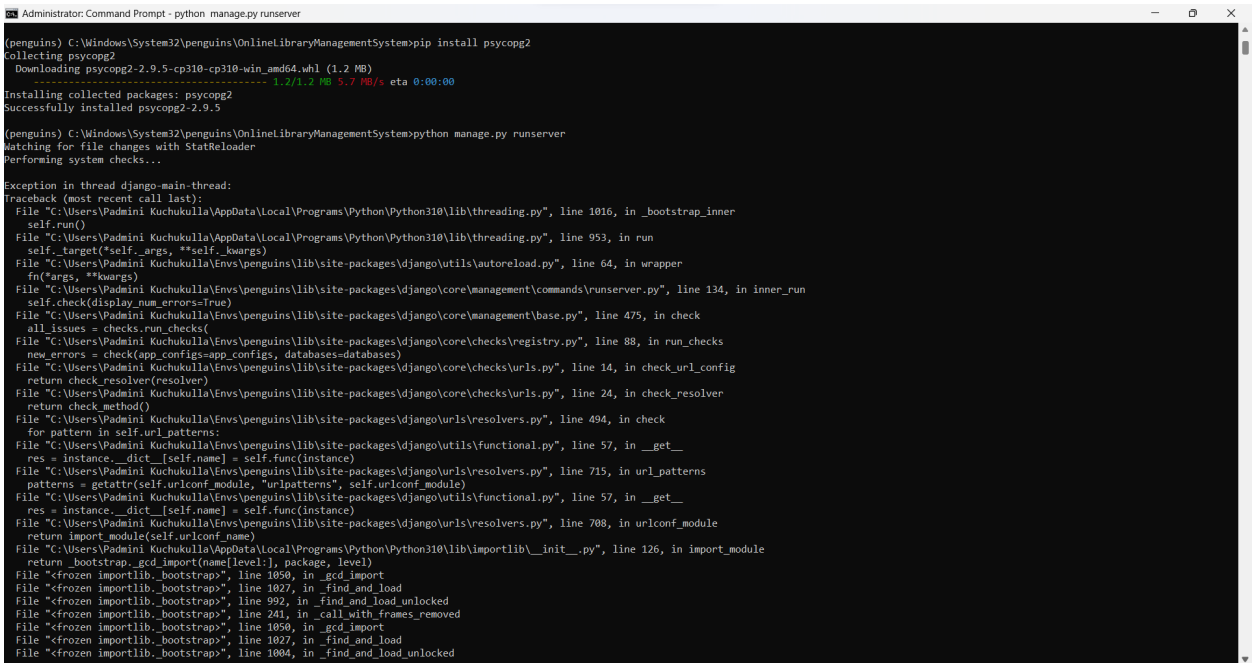
While we are setting up these, it asks for a password, which we gave it as “Penguins”. This should be updated in the settings.py file, which is created by django in default.

After this step, we will try to run the server using the following command
`python manage.py runserver`



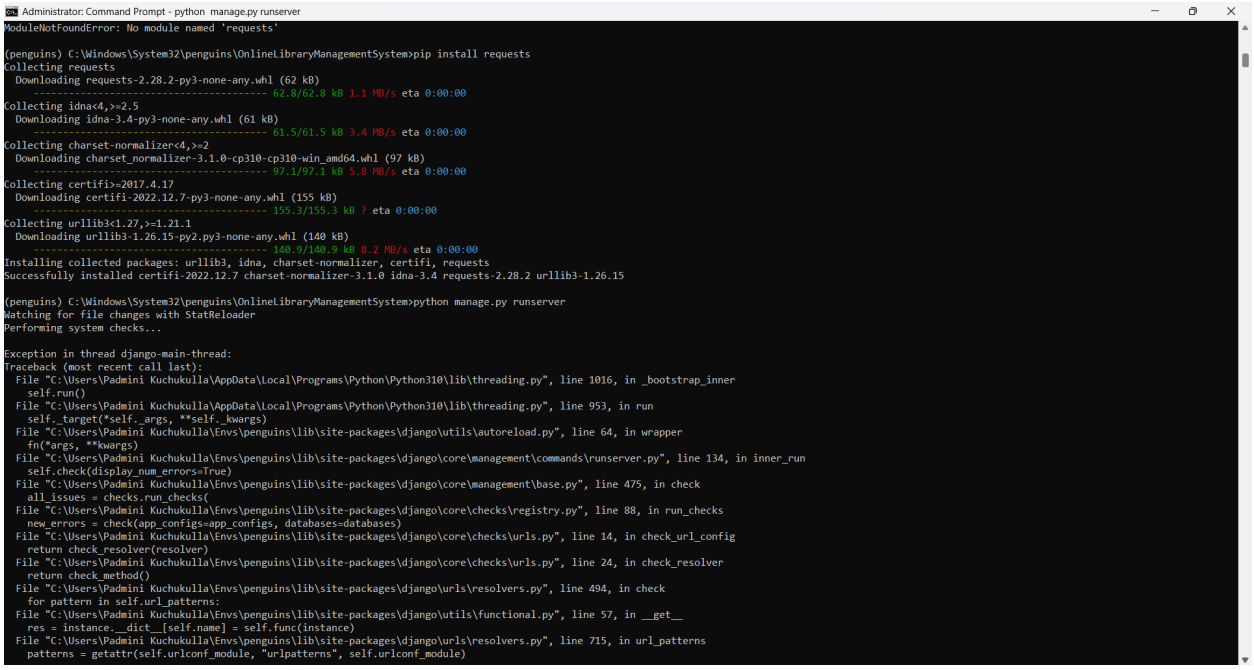
Now if all the required modules were installed, we would be able to see the frontend working fine on our webpage. But we can see the ModuleNotFoundError, which says there is no “psycopg2” module. This module is used as the database connecting adapter to postgresql. So, now we install this module using the following command.

```
pip install psycopg2
```



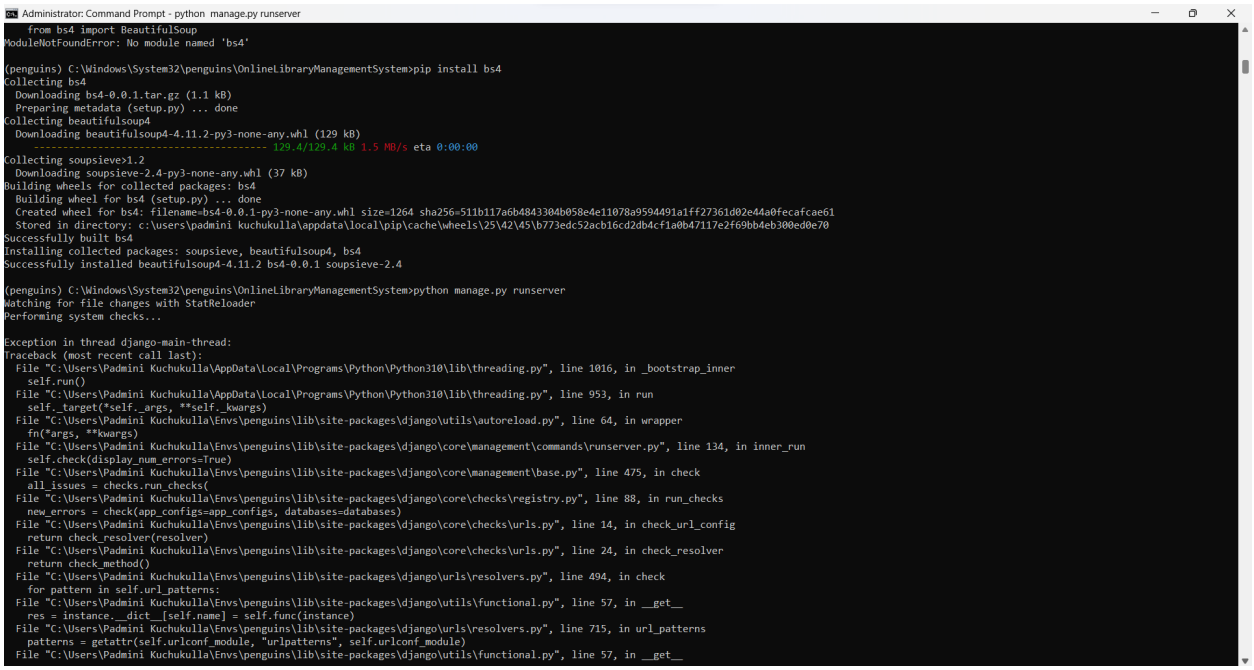
After this step, again we will try to run the server using the following command
`python manage.py runserver`

We can see the ModuleNotFoundError, which says there is no “requests” module. This module is used for handling all the http requests. So, now we install this module using the following command.
`pip install requests`



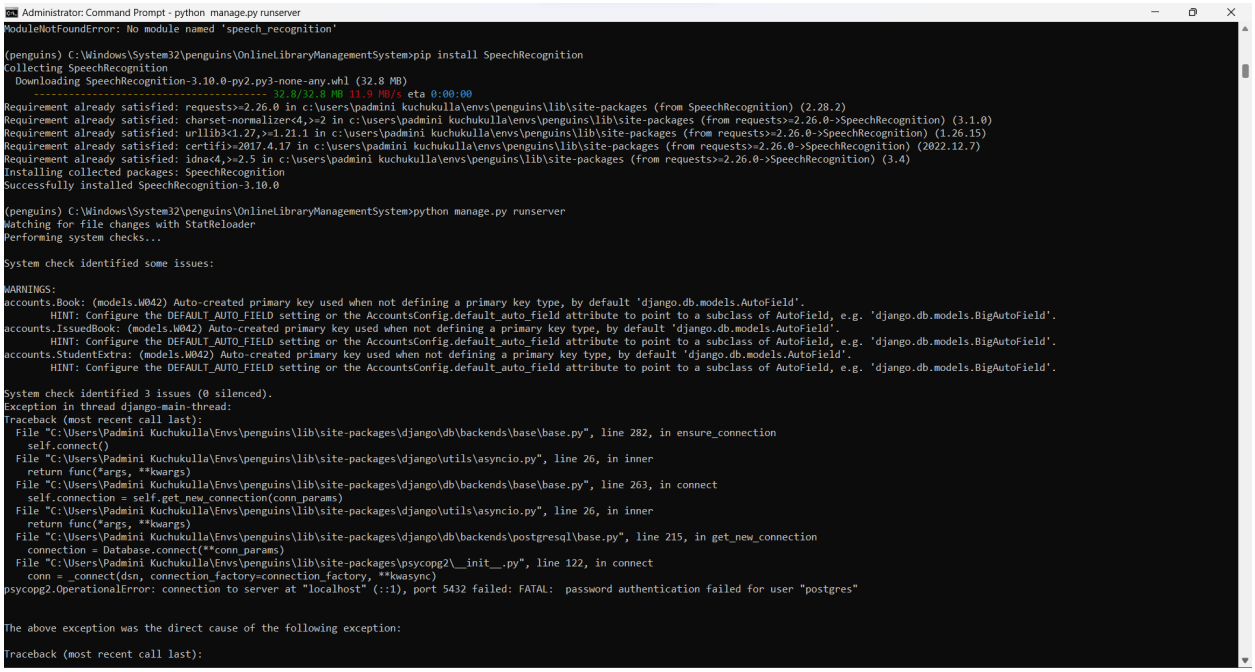
After this step, again we will try to run the server using the following command
`python manage.py runserver`

We can see the ModuleNotFoundError, which says there is no “bs4” module. This module is beautifulsoup4, which is used for web scraping. So, now we install this module using the following command.
`pip install bs4`



After this step, again we will try to run the server using the following command `python manage.py runserver`

We can see the ModuleNotFoundError, which says there is no “speech_recognition” module. This module is used for recognizing speech, which we will be using for voice-based search. So, now we install this module using the following command. `pip install SpeechRecognition`



After this step, again we will try to run the server using the following command `python manage.py runserver`

f. Peer review feedback

During the productive peer review session, the team engaged in an open and collaborative discussion, sharing their thoughts and concerns about the project and identifying ways to overcome hurdles. The team focused on the front-end parts of the project, specifically the user interface, to improve its user-friendliness and efficiency.

In the review session, the team received feedback to improve the login process by creating separate pages for admin and user. To implement this suggestion, the team decided to include an interactive page with various options for admin and user features, instead of just a navigation bar.

As part of the improvement process, the team also planned to incorporate alerts to keep the users informed about the status of the pages which was pointed out in the peer review session. This would help them stay updated and aware of any changes or updates to the system. Moreover, the team made the page more responsive to user actions, ensuring that the users could navigate the pages smoothly and easily. As per the suggestions regarding the backend implementation that were raised will be used in the phase 2 implementation of the project.

In summary, the team received valuable feedback during the review session, which they implemented with add-ons like an interactive page and alerts to improve the login process. The changes were aimed at making the system more user-friendly and efficient. The team also made the page responsive to user actions to enhance the overall user experience.

g. Reflection

In Phase 1, we have accomplished the majority of the tasks that we had initially planned for. The project required extensive analysis and design, which took up a considerable amount of time. Once that was completed, we distributed the tasks among our team members and began coding.In this phase, we focused solely on the front end, using HTML and other front-end technologies for the web pages. Fortunately, the spring break allowed us to invest more time and effort into the project, allowing us to complete all of our objectives.

We are particularly proud of our team's collaboration and communication. We regularly connected and delegated tasks among ourselves, which enabled us to reach our targets efficiently. The testing for the front end pages was difficult but we faced this situation by using static web pages to test the pages. Moreover, constant involvement from the entire team significantly reduced code redundancy, which was a crucial improvement for the project. Looking forward, we believe that our successful completion of Phase 1 will set us up well for the next phases of the project.

Member's Contribution:

Member name	Contribution description	Overall Contribution (%)	Note (if applicable)
Ashraf Syed	Requirements, Meeting minutes updation, involved in building the code	14.3%	
Padmini Kuchukulla	Contributed in pushing code to github, user manual, code compilation,involved in building the code	14.3%	
Abhijith Reddy Mandagiri	Contributed in Test cases, Reflection, Review feedback	14.3%	

	from peer Evaluation, involved in building the code		
Deepna Thalanki	Contributed in UML diagrams, involved in building the code	14.3%	
Raghu Vamsi Kondapalli	Contributed in UML diagrams, involved in building the code, Meeting minutes updation	14.3%	
Vamsi Venkat Manepalli	Involved in core functionalities document, building the code	14.3%	
Sai Vishwak Jadala	Involved in core functionalities document, building the code	14.3%	