

Virtual Learning & Session Management Environment Web Application (CSBox)

*A Project Submitted in Partial Fulfillment of the Requirements for the
Degree of
Bachelor of Science in Computer Science and Engineering*

by
Tanvir Mahmud
ID: CSE 06607742

Supervised by: **Md Samiul Islam**
Lecturer



Department of Computer Science and Engineering
STAMFORD UNIVERSITY BANGLADESH

September 2023

Abstract

CSBox introduces an innovative online educational platform aimed at enhancing virtual learning experiences. This project responds to the changing nature of education by offering educators and learners a dynamic virtual environment. At its core, CSBox features an advanced session management system, enabling the creation, organization, and administration of virtual sessions. These sessions act as central hubs for interactive learning, promoting real-time communication, content sharing, and collaborative engagement among educators and learners.

CSBox empowers users through its user-friendly interface and comprehensive CRUD operations, providing a seamless experience for post creation, file uploads, and discussion facilitation. The platform goes beyond traditional educational approaches, allowing educators to lead virtual sessions while enabling learners to actively participate through comments and interactions. CSBox also streamlines content management, allowing educators to share resources effortlessly and learners to access materials from a centralized repository. With its responsive design, CSBox ensures a consistent user experience across various devices, accommodating both educators' and learners' preferences. Through its interactive and adaptable framework, CSBox redefines online education, creating a collaborative and engaging virtual learning environment.

Approval

The project report “Virtual Session Management Environment Web Application (CSBox)” submitted by Tanvir Mahmud ID:CSE06607742, to the Department of Computer Science & Engineering, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science (B.Sc.) in Computer Science & Engineering and as to its style and contents.

Board of Examiners Name, Signature and Date:

.....

(Board Member 1)

Date:

(Board Member 2)

Date:

(Board Member 3)

Date:

Supervisor's Signature and Date:

.....

Supervisor Name

Date:

Declaration

We, hereby, declare that the work presented in this Thesis Project is the outcome of the investigation performed by us under the supervision of Md Samiul Islam, Lecturer, Department of Computer Science & Engineering, Stamford University Bangladesh. We also declare that no part of this Project and thereof has been or is being submitted elsewhere for the award of any degree or Diploma.

Signature and Date:

.....
Student Name: Tanvir Mahmud

Date: 28.08.2023

Dedicated to

To my beloved mother and father.

Acknowledgement

First and foremost, we are grateful to the Almighty creator without whose blessing, this project would not have been successful. We would like to express our sense of gratitude to Md. Samiul Islam, Lecturer of the Department of Computer Science & Engineering, Stamford University Bangladesh, for his unconditional guidance, insights, immense patience, unlimited encouragement and inspirations throughout this whole project. He is the one who supported us every step of the way in every possible way. Without him, this project would not have come to fruition. He provided valuable technical advice and pointed us in the right directions which were much required for a project like this. We are truly grateful for having a teacher like him as our supervisor. We specially owe thanks to all the teachers of the faculty of Computer Science & Engineering, Stamford University Bangladesh for their help, valuable suggestions and discussions. Moreover, we would like to thank all our friends who have supported and helped us throughout this project. Last of all we would also like to thank our parents who have walked along with us in every step of our life and always encouraged and inspired us to do greater things in our life.

Table of Contents

List of Figures.....	vii
1. Introduction.....	1
1.1 Motivation.....	1
1.2 Objective.....	1
1.3 Problems and Definition.....	1
1.4 Proposed System.....	2
1.5 Market & Scope.....	2
2. Planning & Analysis.....	3
2.1 Planning.....	3
2.2 Requirement Analysis.....	4
2.3 System Analysis.....	5
2.4 Feasibility Study.....	7
2.5 Tools & Technology.....	8
3. System Design.....	9
3.1 Use Case Diagram.....	9
3.2 Entity Relationship Diagram.....	15
3.3 Data Flow Diagram.....	22
4. Implementation.....	23
4.1 Configuration.....	23
4.2 Interfaces.....	24
5. Conclusion.....	52
5.1 Future Work.....	52
References.....	55

List of Figures

2.1.1	Seven phases of system development life cycle	3
2.3.1	Model View Template (MVT) Architecture	6
3.1.1	Account & User System Use Cases	9
3.1.2	Session Management System Use Cases.	10
3.1.3	Real-time Chatting System Use Cases.	11
3.1.4	Post and File Management Use Case Diagram in Individual Session.	12
3.1.5	Search and Filtering System Use Case	13
3.1.6	Notification System Use Case	14
3.2.1	Entity Relationship Diagram	17
3.3.1	Data Flows Diagram Level - 0	18
3.3.2	Data Flow Diagram Level - 1	19
3.3.3	Data Flow Diagram Level - 2	20
4.2.1	Interface for user as landing page	24
4.2.2	Interface for user registration form	25
4.2.3	Interface for confirmation of verified users.	25
4.2.4	Interface for user landing home page	26
4.2.5	Interface for creating Session	26
4.2.6	Interface for created and joined session	27
4.2.7	Interface for settings (created session)	27
4.2.8	Interface for Edit Created Session	28
4.2.9	Interface for Delete Created Session	28
4.2.10	Interface for settings (joined session)	28
4.2.11	Interface for Leave Joined Session	29
4.2.12	Interface for Individual Session Landing page	29
4.2.13	Interface for Creating post in Session	30
4.2.14	Interface for Session post	30
4.2.15	Interface for Post's file access	30
4.2.16	Interface for post CRUD accessibility	31
4.2.17		

1 Introduction

1.1 Motivation

In our fast-changing world, CSBox springs forth as a project that strives to make learning exciting and accessible. Imagine a world where learning isn't just limited to classrooms, but reaches everyone wherever they are. CSBox aims to create this world, where session hosts lead the way in interactive sessions, and session members actively join in. Our project's goal is to bring education closer to people, no matter where they live or what their situations are.

Think of CSBox as a hub where session hosts orchestrate engaging learning experiences, and session members contribute and participate. We believe that education should be interesting, engaging, and open to all. With CSBox, we want to make things easy for session hosts so they can focus on sharing knowledge, while session members can take part in learning, even from afar.

As the world transforms, CSBox wants to be at the forefront of this transformation, making education more accessible and enjoyable. We aim to create a simple platform where session hosts can effortlessly manage virtual sessions, and session members can learn and engage, irrespective of their physical location. By building this platform, CSBox aims to break down barriers and offer an inclusive educational experience for everyone.

1.2 Objectives

The CSBox project seeks to achieve several key objectives in the realm of online education. Its primary goal is to create a user-friendly platform that enables session hosts to efficiently manage virtual sessions and empowers session members to actively participate and engage. CSBox aims to simplify the process of content sharing, discussions, and collaborative learning, fostering an inclusive and dynamic online learning environment. Additionally, the project aims to provide a responsive design that ensures a consistent and seamless experience across various devices, catering to the preferences of both session hosts and session members. By fulfilling these objectives, CSBox aims to contribute to the transformation of traditional education, making it more interactive, accessible, and engaging for all.

1.3 Problems & Definitions

The CSBox project addresses several challenges in modern education. One of the key problems is the lack of flexible and interactive platforms for conducting virtual learning sessions. Many existing solutions fail to provide an intuitive interface for session hosts to manage sessions effectively or for session members to actively engage in discussions. Additionally, the absence of a centralized system for content sharing often results in fragmented resources and difficulty in accessing learning materials. The term "session hosts" refers to educators or instructors responsible for leading and managing virtual learning sessions, while "session members" denotes students or participants who actively take part in these sessions. "Content sharing" refers to the exchange and distribution of educational materials among participants. Through the CSBox project, these problems are being addressed by creating a comprehensive platform that empowers both session hosts and session members, while ensuring seamless content sharing and interactive engagement within the virtual learning environment.

1.4 Proposed Systems

- A Reliable Web-based Learning and Session Management
- Basic Accounts & Profile Management
- Manual User Verification with Email
- Easy Session Creating with Automated Unique Token
- Easy Session Joining with Token
- CRUD Operation of Created Session and Joined Session
- Creating Post with File Upload in Individual Session (Real Time Mechanism)
- File Management System with Download
- Individual User Profile Management
- Automatic Notifying System
- Search and Filtering System for Post, Session, User
- Real Time Conversation System

2 Planning & Analysis

2.1 Planning (change)

A plan to make an application which will cover all the requirements. We decided to make some micro applications to cover all the functionalities.

- Account & User Management Application
- Session Management Application
- Post and File Management Application
- Settings Application
- Search Application
- Notification Application
- Real Time Chat Application

As our project strategy we decided to choose Agile Model to get successful, with a great product.

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software products. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks.

Agile will be a great model for this project. We will follow the extreme programming approach to solve all problems. Requirement gathering and fast development is the main goal in extreme programming. A big project comes with an easy solution with agile. And that is a one man army project, and I think that will be a great model for this.

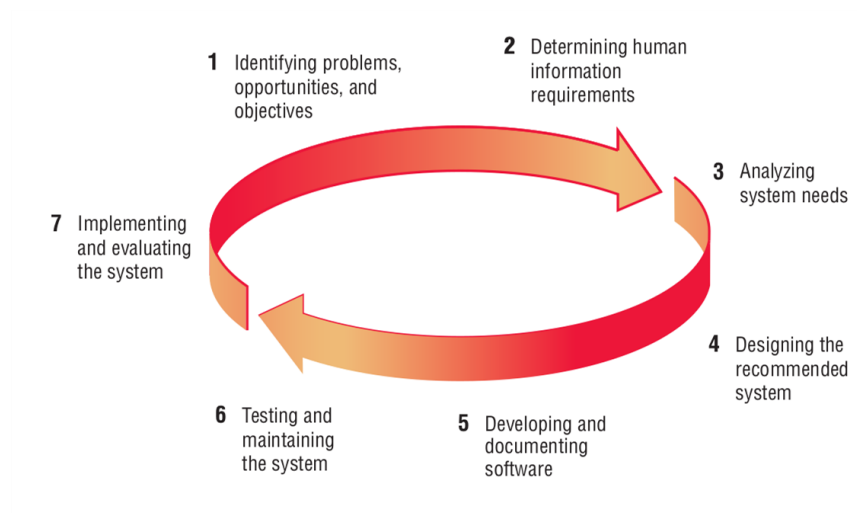


Figure 2.1.1 Seven phases of system development life cycle.

2.2 Requirements

Analysis Requirements for

all User:

- In this application users can Sign-up & Sign-in.
- Need email verification before 1st Sign-in.
- Have to apply for profile verification with a valid email.
- After profile verification, users can interact with this system.
- All user can create or join session
- A user can be a host of his own session and a session member of other's session

Requirements for Session Host:

1. Can create posts with or without uploading files.
2. Can operate all CRUD operations of own posts and files.
3. Can operate only Delete operation of session member's posts.
4. Can Open, Download and Remove the uploaded files of its own posts.
5. Can Open and Download the uploaded files of session Member's posts.
6. Can comment in every posts in the session
7. Can operate CRUD operation of all comments in its own post except Edit of other session member's comment.
8. Can remove a session member.
9. Can block a session member in an individual session .
10. Can visit a session member's profile.
11. Can Edit and Delete the entire session.

Requirements for Session Member:

1. Can create posts with or without uploading files.
2. Can operate all CRUD operations of own posts and files.
3. Can Open, Download and Remove the uploaded files of its own posts.
4. Can Open and Download the uploaded files that are posted by session Host.
5. Can comment on every post in the session.
6. Can operate CRUD operation of all comments in its own post except Edit of other session member's comment.
7. Can visit the profile of session host and other member in the session.
8. Can leave the session.

2.3 System Analysis

In our system there will be two actors, User and Admin.

Admin have the superpower in this system, he can access the admin panel, where every data model and data will show and he can interact with all kinds of CRUD action.

We decided to use Python as a core programming backend language, in the backend we will use Django as a web backend framework, Django Rest framework as Restful APIs. And React frontend framework for front-End interaction. Bootstrap as UI framework . PostgreSQL for database.

There are a lot of back-end frameworks but we choose Django.

Why is Django best from others? [4]

With Django, you can take web applications from concept to launch in a matter of hours. Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

- **Ridiculously fast** - Django was designed to help developers take applications from concept to completion as quickly as possible.
- **Fully loaded** - Django includes dozens of extras you can use to handle common web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds, and many more tasks — right out of the box.
- **Reassuringly secure** - Django takes security seriously and helps developers avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery and clickjacking. Its user authentication system provides a secure way to manage user accounts and passwords.
- **Exceedingly scalable** - Some of the busiest sites on the planet use Django's ability to quickly and flexibly scale to meet the heaviest traffic demands.
- **Incredibly versatile** - Companies, organizations and governments have used Django to build all sorts of things — from content management systems to social networks to scientific computing platforms.

What does Django Structure Look Like? [5]

In a traditional data-driven website, a web application waits for HTTP requests from the web browser (or other client). When a request is received the application works out what is needed based on the URL and possibly information in POST data or GET data. Depending on what is required it may

then read or write information from a database or perform other tasks required to satisfy the request. The application will then return a response to the web browser, often dynamically creating an HTML page for the browser to display by inserting the retrieved data into placeholders in an HTML template.

Django web applications typically group the code that handles each of these steps into separate files:

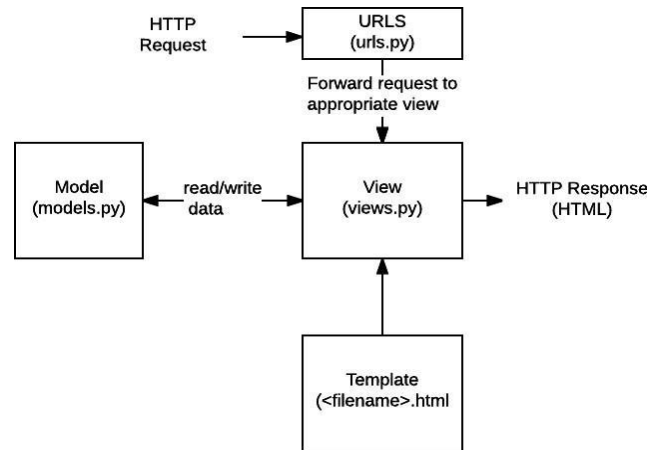


Figure 2.3.1 – Model View Template (MVT) Architecture. [3]

- **URLs:** While it is possible to process requests from every single URL via a single function, it is much more maintainable to write a separate view function to handle each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data.
- **View:** A view is a request handler function, which receives HTTP requests and returns HTTP responses. Views access the data needed to satisfy requests via *models*, and delegate the formatting of the response to *templates*.
- **Models:** Models are Python objects that define the structure of an application's data, and provide mechanisms to manage (add, modify, delete) and query records in the database.
- **Templates:** A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A *view* can dynamically create an HTML page using an HTML template, populating it with data from a *model*. A template can be used to define the structure of any type of file; it doesn't have to be HTML

In the chat module we used ASGI server for asynchronous programming.

ASGI (*Asynchronous Server Gateway Interface*) is a spiritual successor to WSGI, intended to provide a standard interface between async-capable Python web servers, frameworks, and applications. Where WSGI provided a standard for synchronous Python apps, ASGI provides one for both asynchronous and synchronous apps, with a WSGI backwards-compatibility implementation and multiple servers and application framework. [6]

And this is the actual architecture of our system.

2.4 Feasibility Study

The very first phase in developing any life cycle is preliminary investigation. In the preliminary study we examine the project feasibility. This project has been tested in the following areas of feasibility:

- **Operational Feasibility**
- **Technical Feasibility**
- **Market Feasibility**
- **Organizational Feasibility**

Operational Feasibility:

- ✓ Microservice based architecture makes the system independent.
- ✓ The system is designed in such a way that it is easy to operate.
- ✓ Easy business model makes the system more reliable.

Technical Feasibility:

- ✓ The system has a very simple structure and easy to understand body.
- ✓ Impressive User Interface and Good User Experience
- ✓ The system is platform independent and browser independent.
- ✓ The system can be expanded if so decided.
- ✓ Latest technologies make this system faster and more secure.

Market Feasibility:

- ✓ There is no successful virtual learning and session management system in our local industry.
- ✓ Our latest technologies and services will attract users.

Organizational Feasibility:

- ✓ And we can operate all kinds of action from our home.
- ✓ We can grow our system remotely by developers.

2.5 Tools & Technologies

Programming Languages-

- Python 3.10
- Javascript

Backend Framework -

- Django version 4.2.3
- Django Rest Framework 3.14.0
- Json Web Token 5.2.2

Frontend Framework -

- React version 18.2.0
- React-dom 18.2.0
- React-Router-dom 6.14.2

Channels -

- ASGI/Channels version 3.0.4

Database -

- PostgreSQL

UI Framework -

- Bootstrap 5.3.1
- React-bootstrap 2.8.0

React plugins -

- JQuery
- jwt-decode
- Popper.js
- react-modal

Font Script -

- Font Awesome
- Google Font API

Version Controlling System -

- Github

Operating System -

- MacOS - Ventura 13.4.1

3 System Design

3.1 Use Case Diagram

There are some major interactions with the system as a user. We showed booth users and admin as actors. This use case shows that the user can sign up and then he needs to verify his email by a verification link from his given email. Without verifying email users can't sign-in. After a successful step, the user can access the application. Without verification, users can't interact with other features.

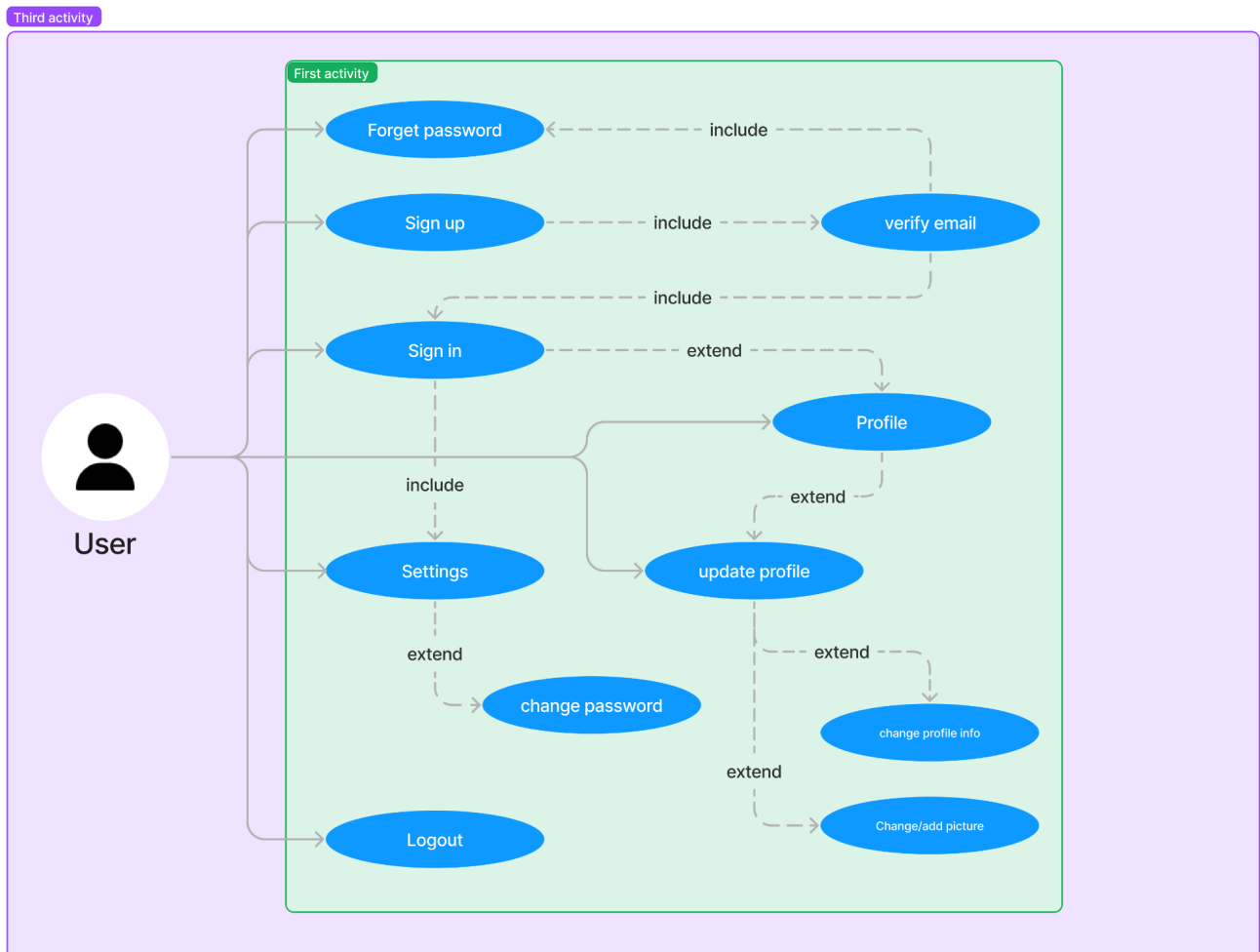


Figure 3.1.1 – Account & User Management System Use Cases.

This is a Session Management App which will allow users to create sessions and join sessions with valid tokens. Which core system follows the Session Management System. In this system users can create sessions with automated token generation and users also can join other sessions with valid token provided by session host. Users can perform all CRUD operations in a created session like edit and delete the entire session and also leave sessions that are hosted by other users.

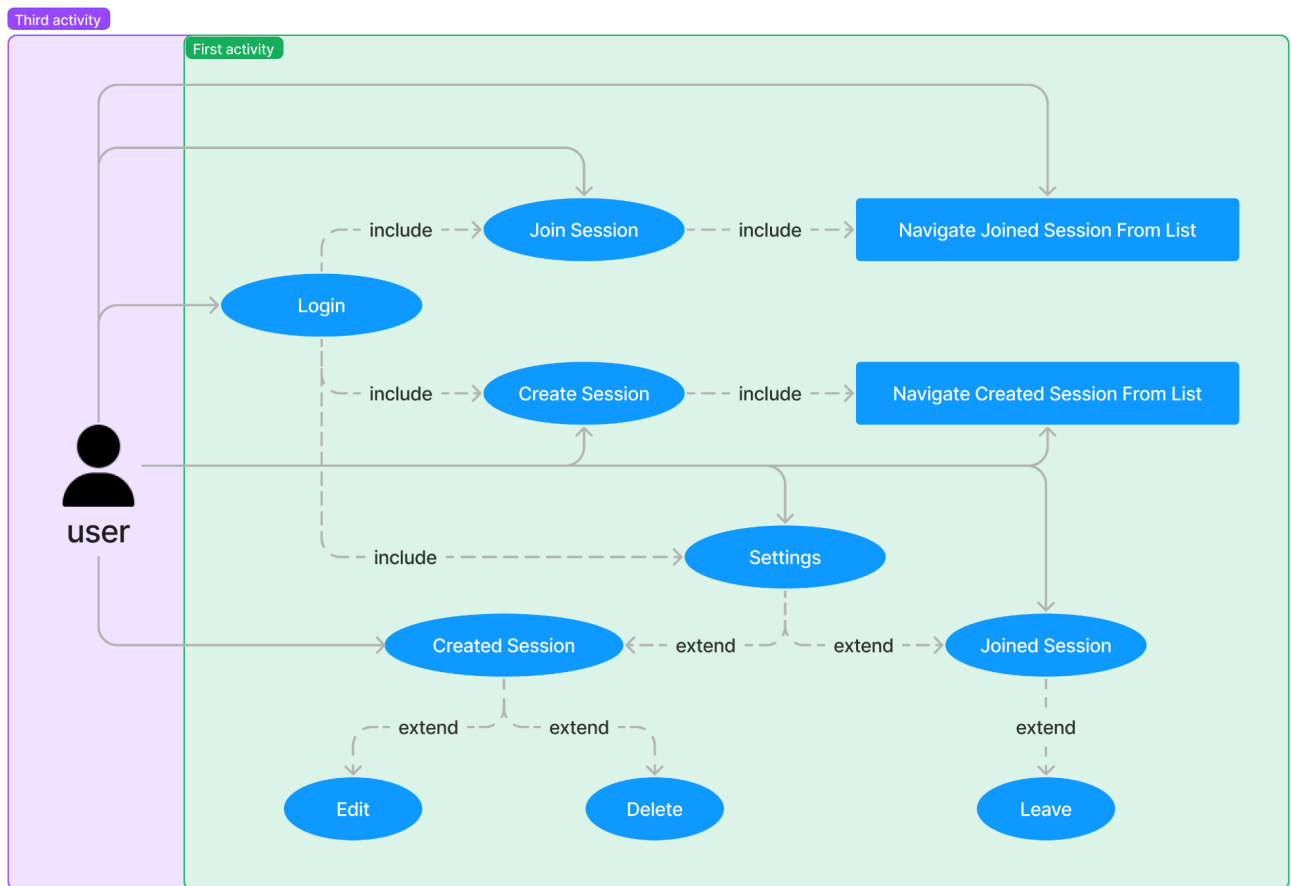


Figure 3.1.2 – Session Management System Use Cases.

This is a chat system between two users. This app uses Web Socket for communication between two users asynchronously. Web Socket is a computer communications protocol, providing full-duplex communication channels over a single TCP connection. Here Web Socket connects two user ports in a channel using Django channel. In the Django channel Redis will work as a message broker and cache data store memory.

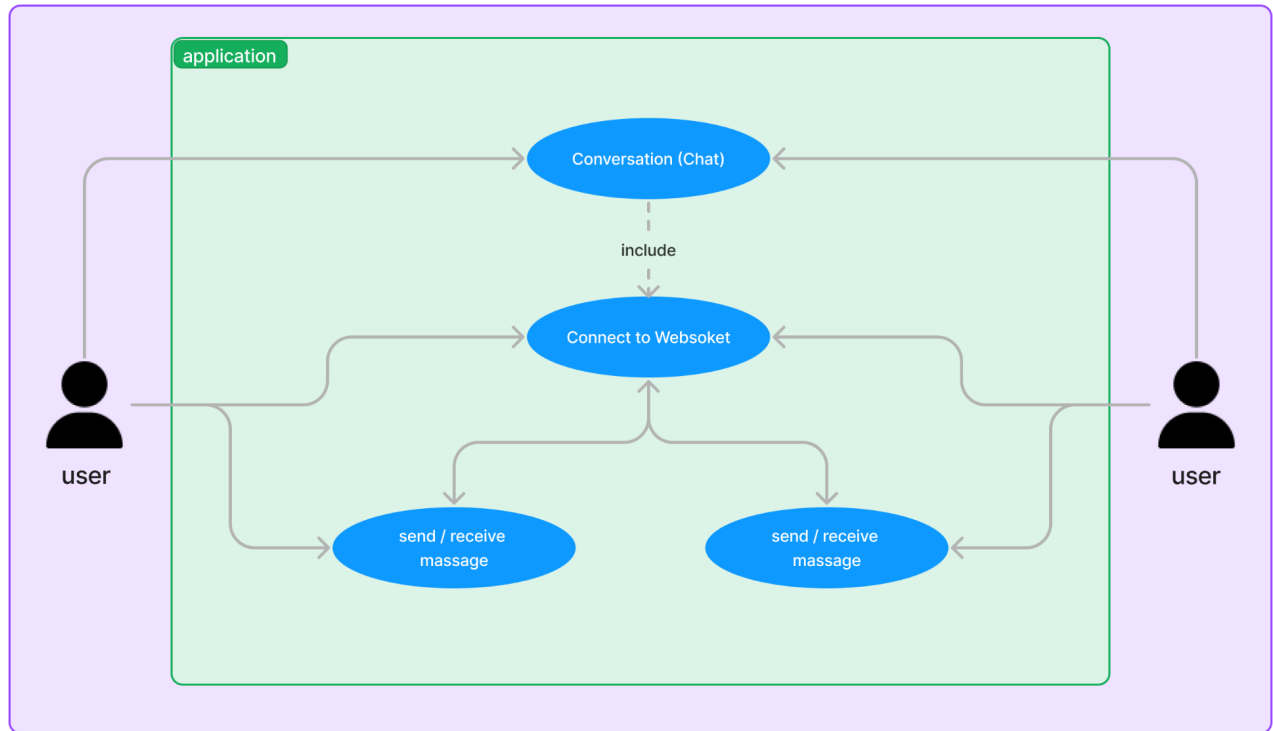


Figure 3.1.3 – Real-time Chatting System Use Cases.

This is a single session use case diagram. Host and session members can create posts with files or without files. also both of them can upload files. Host and session members can operate CRUD functionality of their own post in the session. Host will be allowed to Delete all the posts in the session including the session member's post. Host can not be allowed to update a session member's post. Host and session member can visit other session member profiles from the session member list. only the Host can remove or block a session member from the session. host and session member can comment on all the posts and edit or remove the comment of their own. Post owner can only remove the comment of others.

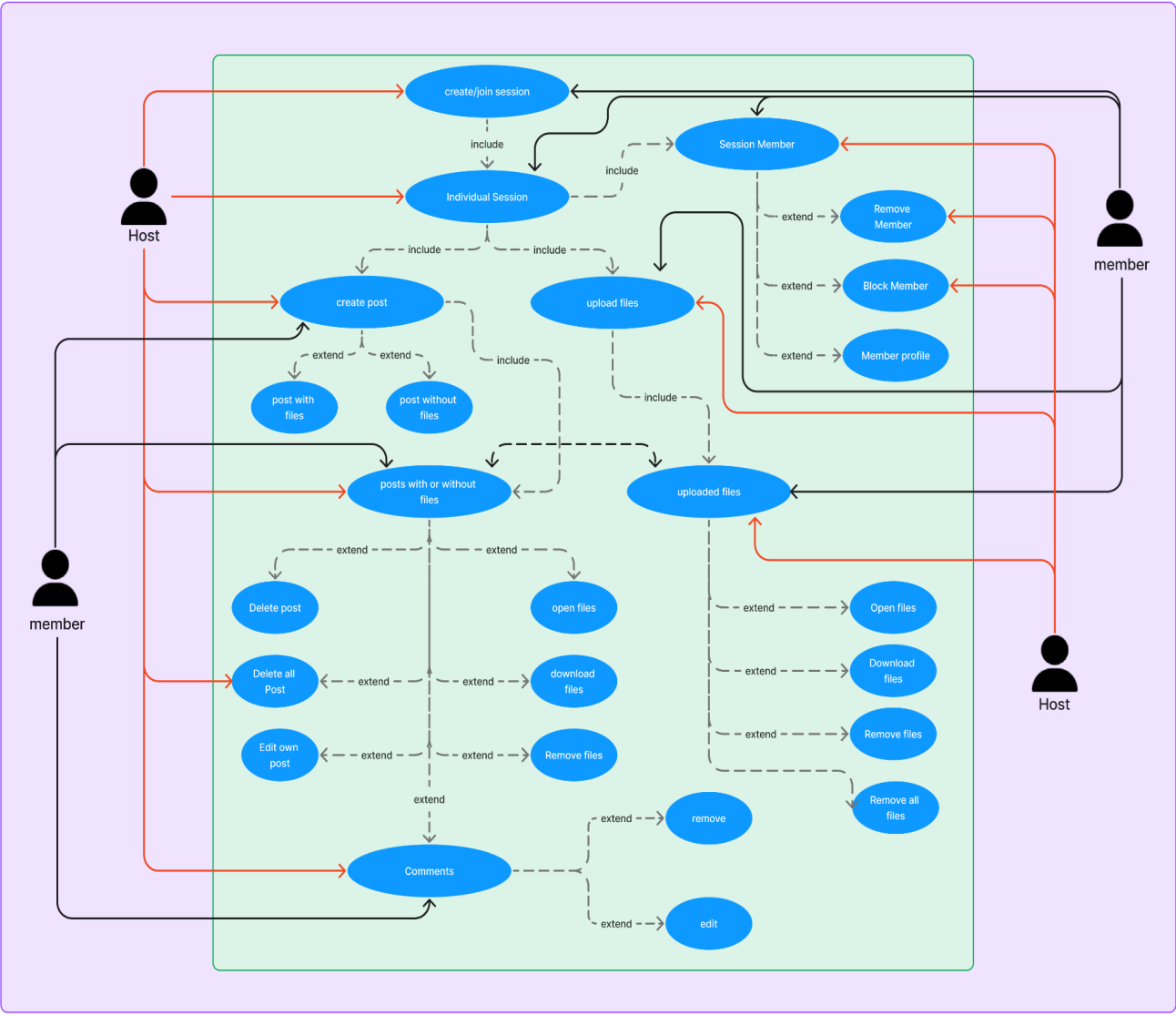


Figure 3.1.4 – Post and File Management Use Case Diagram in Individual Session.

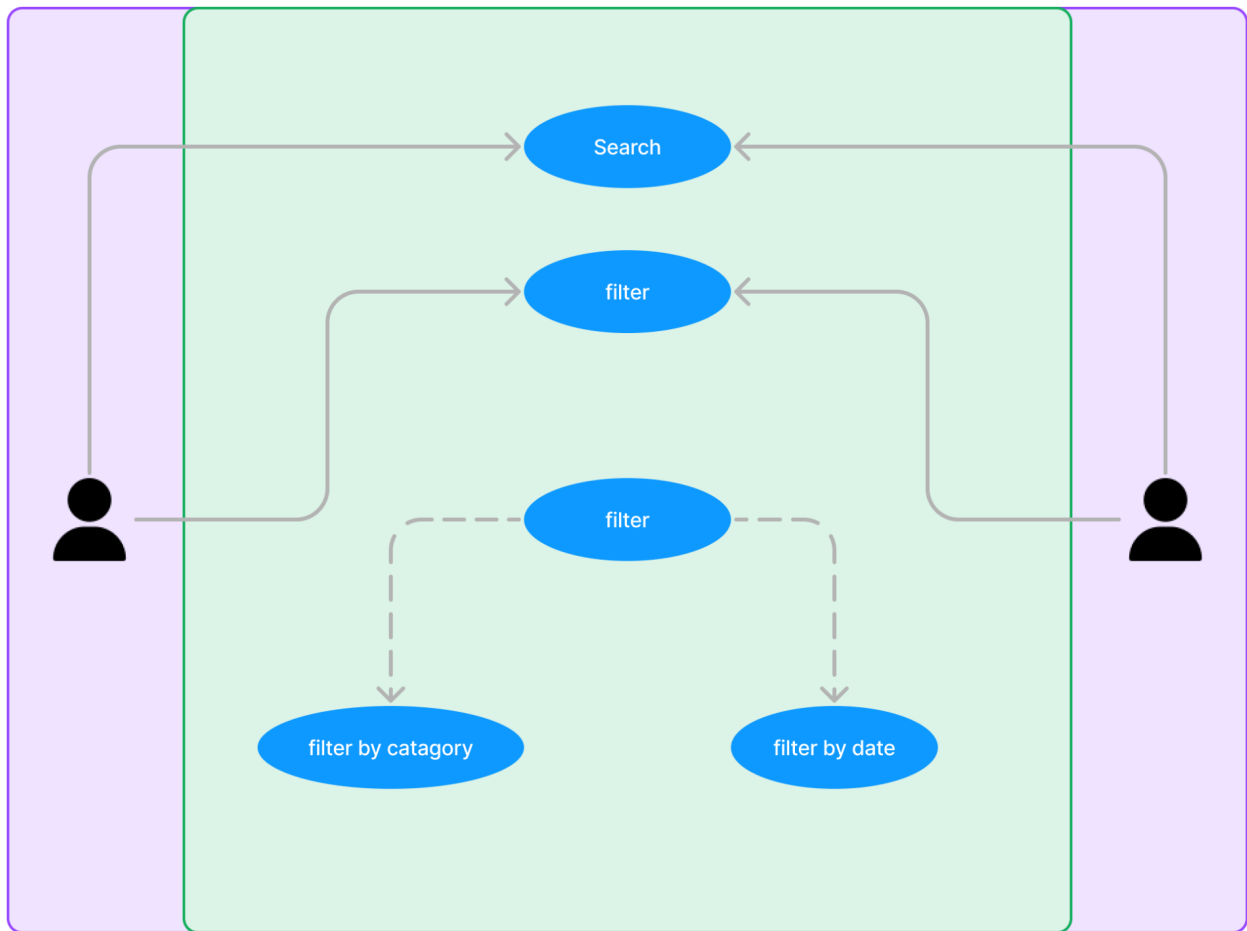


Figure 3.1.5 – Search and Filtering System Use Case

This is a notification app which will notify all kinds of action between two users. This helps users about actions to him by other users. This system has a lot of back-end functionalities, to make this real time we followed a lot of mechanisms. Some of this back-end was handled by Django, we just followed their structure.

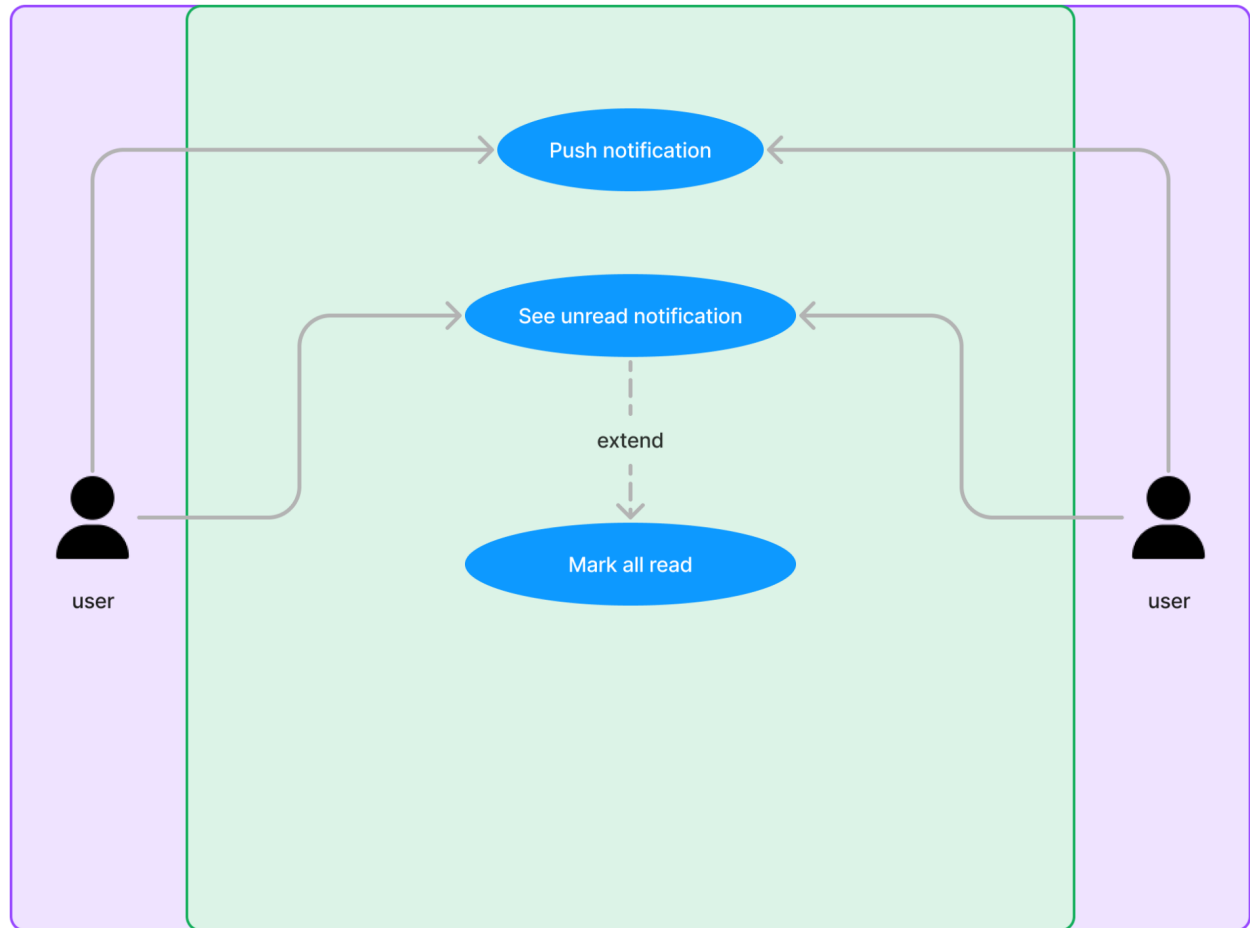


Figure 3.1.6 – Notification System Use Case

3.2 Entity-Relationship Diagram

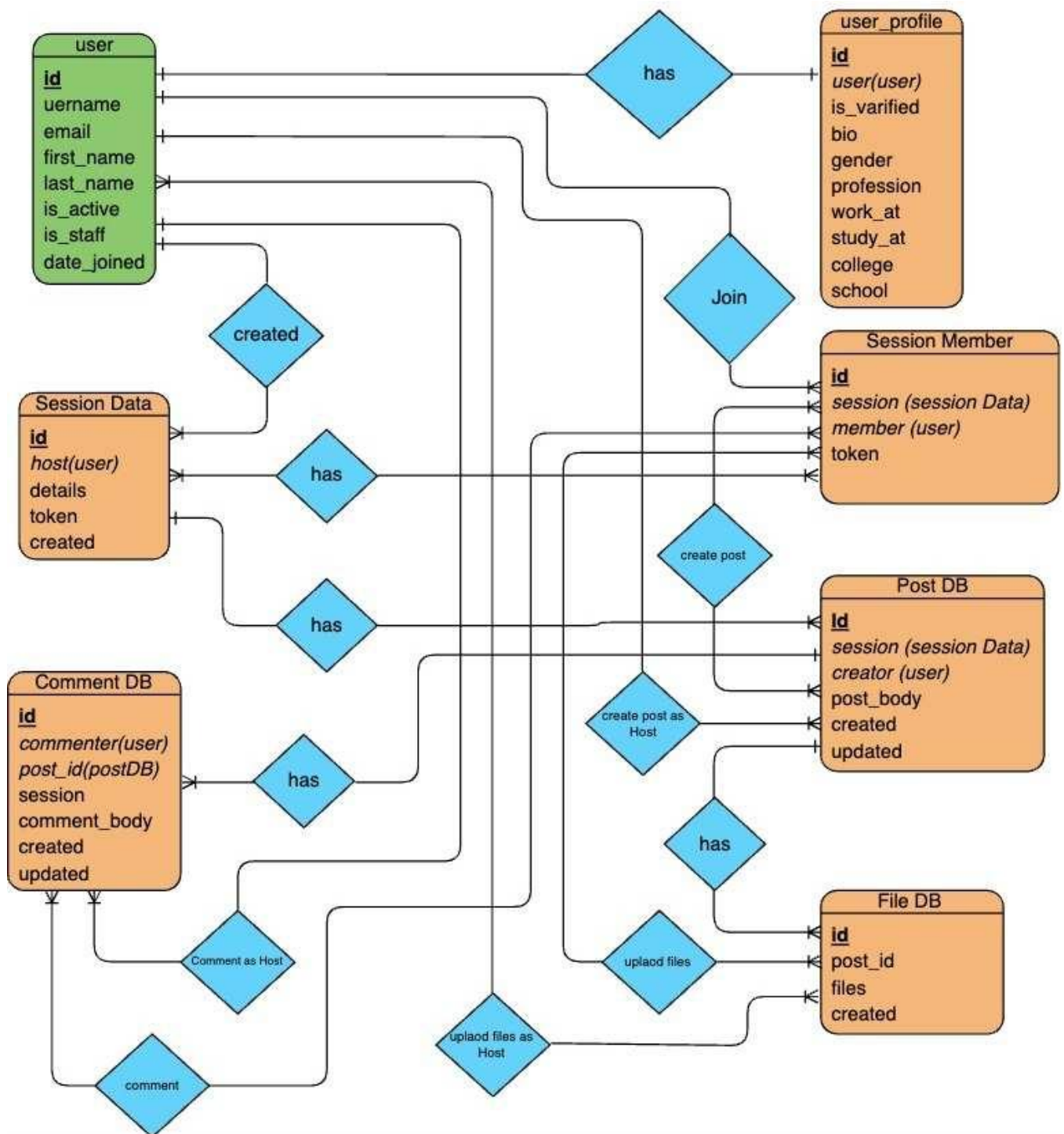


Figure 3.2.1 - Entity Relationship Diagram of the total system.

3.3 Data Flow Diagram

Data Flow Diagram Level - 0

A level 0 Data Flow Diagram (DFD) is an overview of the system's major processes and the flow of data between them. The system involves the "CSBox" functionality, which includes creating sessions, joining sessions, and a file uploading mechanism. If a user creates the session then the user role of the session will be Session host and when another user uses the token then he will act like session member.

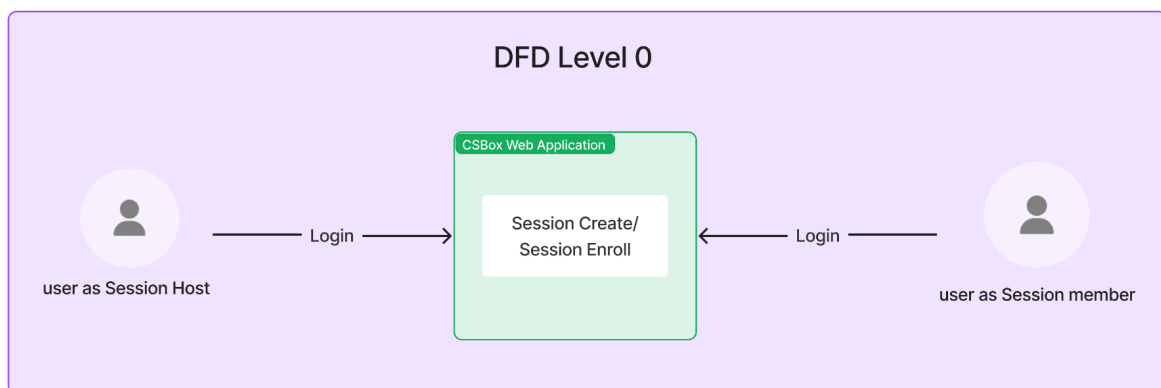


Figure 3.3.1 – Data Flows Diagram Level - 0

Data Flow Diagram Level - 1

This is CSBox level 1 Data Flow Diagram. If a user has no account then he/she needs to create an account. After creating a new account, the user needs to verify the registration process otherwise the user will be unable to access the application. After verifying the registration process, users can login and navigate to the application home page. users can create a session or join an existing session with a valid session token. From the created session list or joined session list user can navigate to an individual session. If a user creates the particular session then the user role will be “Host” with extra permission. If a user joins the session with a token then the user role will be “session member” and the session member user has some limitations. Session host and session member can create a post upload file and visit other session member profiles. host and session member can operate CRUD functionality of their own post and upload files.

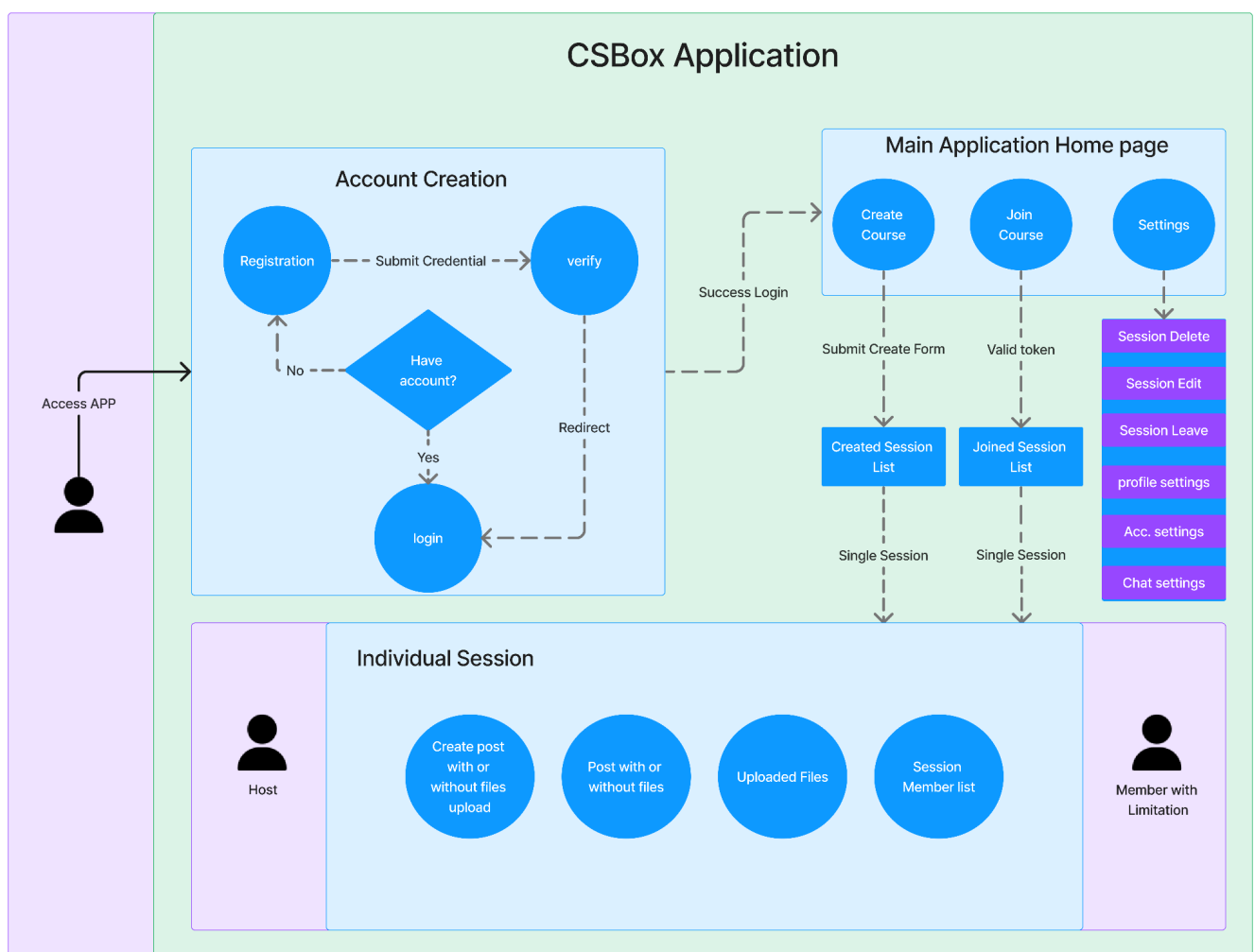


Figure 3.3.2 – Data Flow Diagram Level - 1

Data Flow Diagram Level - 2

This is CSBox level 1 Data Flow Diagram. If a user has no account then he/she needs to create an account. After creating a new account, the user needs to verify the registration process otherwise the user will be unable to access the application. After verifying the registration process, users can login and navigate to the application home page. users can create a session or join an existing session with a valid session token. From the created session list or joined session list user can navigate to an individual session. If a user creates the particular session then the user role will be “Host” with extra permission. If a user joins the session with a token then the user role will be “session member” and the session member user has some limitations. Session host and session member can create a post upload file and visit other session member profiles. host and session members can operate CRUD functionality of their own post and upload files.

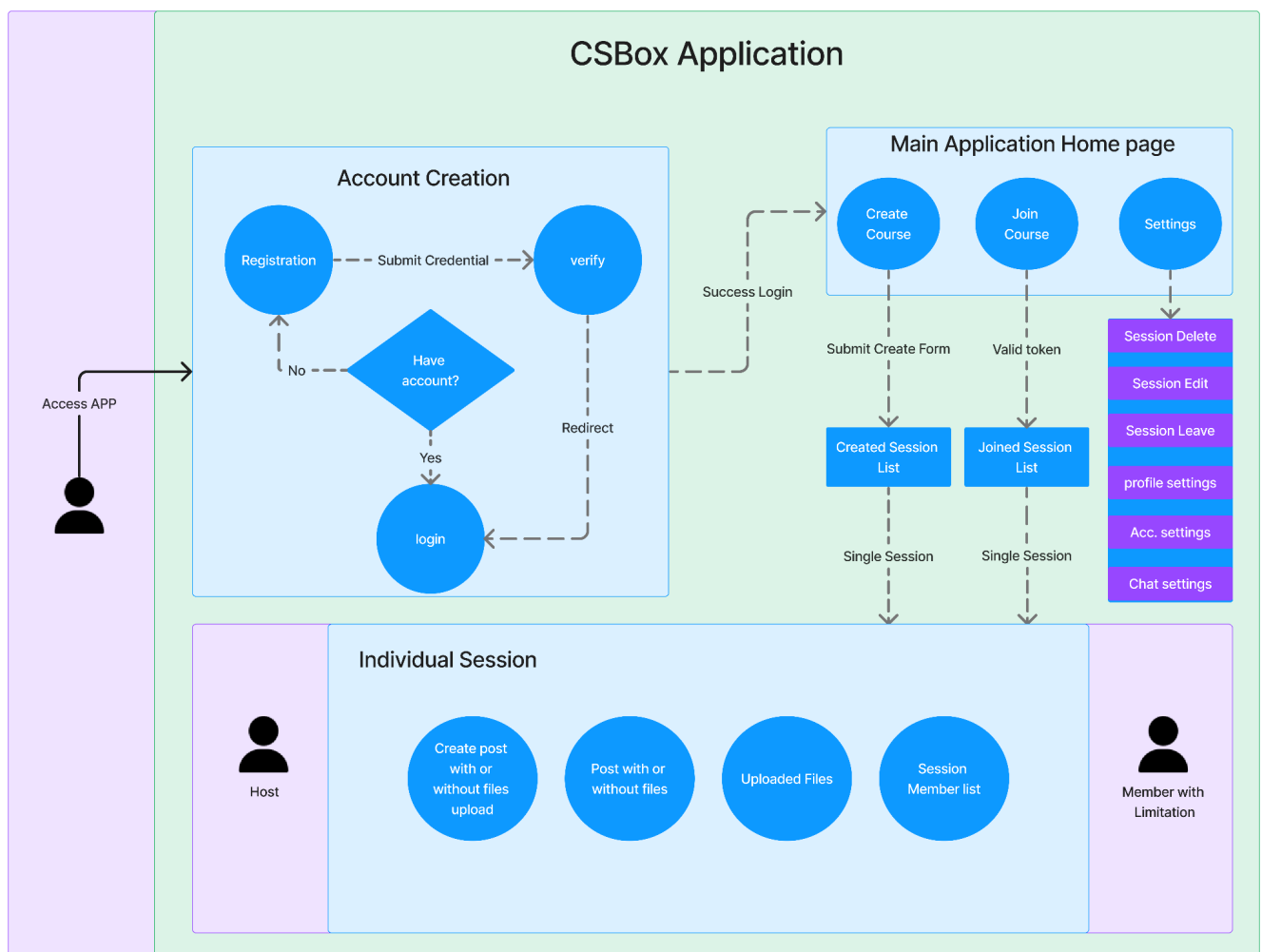


Figure 3.3.2 – Data Flow Diagram Level - 2

4 Implementation

4.1 Configuration

In this application we will use Django 4.0 as server side framework and PostgreSQL as database. So first ensure that python 3.8 and Django and PostgreSQL and a browser in your device.

Open terminal and open VS Code or other code editor.

Write those commands in terminal to start this project -

```
mkdir Project Folder
```

```
django-admin startproject FSMS
```

```
python -m venv env
```

```
pip install django
```

```
pip install pillow
```

Put full project in this directory

```
pip freeze > requirements.txt
```

```
pip install -r requirements.txt
```

and install other modules if necessary.

Alright, next start the virtual environment from the terminal as our development environment. `source env/bin/activate`

Open PGAdmin Make a database in PostgreSQL name 'FSMS' or you can configure another name from settings.

Let's make database migration for

```
PostgreSQL python manage.py
```

```
makemigrations
```

```
python manage.py migrate
```

```
python manage.py runserver
```

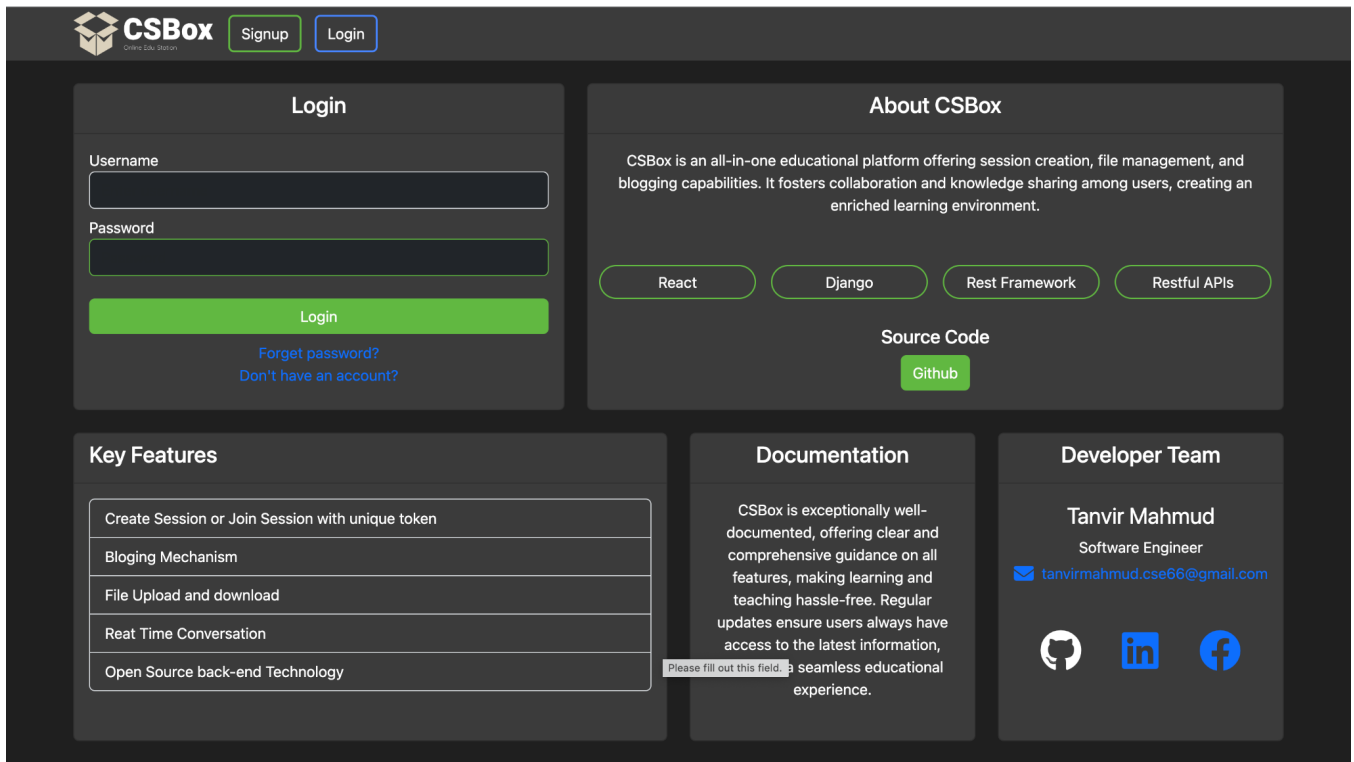
Congratulations, your server will be running!

Open Chrome/Firefox/Safari or any other safe browser to open a running server.

You can find your running server in your local server, type this url in your browser address bar `http://127.0.0.1:8000/` and you will see the homepage of this system.

Note: There may be dependencies on python packages, please install those packages in a virtual environment.

4.2 Interface



The landing page features a dark theme with a top navigation bar containing the CSBox logo and 'Signup' and 'Login' buttons. The main content is divided into several sections: a 'Login' form with fields for Username and Password, a 'Forgot password?' link, and a 'Don't have an account?' link; an 'About CSBox' section describing the platform's capabilities and listing technologies like React, Django, Rest Framework, and Restful APIs; a 'Source Code' section with a 'Github' button; a 'Key Features' section listing features like session creation, blogging, file upload, real-time conversation, and open-source technology; a 'Documentation' section with a description of the platform's documentation and a 'Please fill out this field.' error message; and a 'Developer Team' section for Tanvir Mahmud, a Software Engineer, with contact information and social media links.

CSBox Offers Edu Station

[Signup](#) [Login](#)

Login

Username

Password

[Login](#)

[Forget password?](#)
[Don't have an account?](#)

About CSBox

CSBox is an all-in-one educational platform offering session creation, file management, and blogging capabilities. It fosters collaboration and knowledge sharing among users, creating an enriched learning environment.

[React](#) [Django](#) [Rest Framework](#) [Restful APIs](#)

Source Code

[Github](#)

Key Features

- Create Session or Join Session with unique token
- Blogging Mechanism
- File Upload and download
- Real Time Conversation
- Open Source back-end Technology

Documentation

CSBox is exceptionally well-documented, offering clear and comprehensive guidance on all features, making learning and teaching hassle-free. Regular updates ensure users always have access to the latest information, a seamless educational experience.

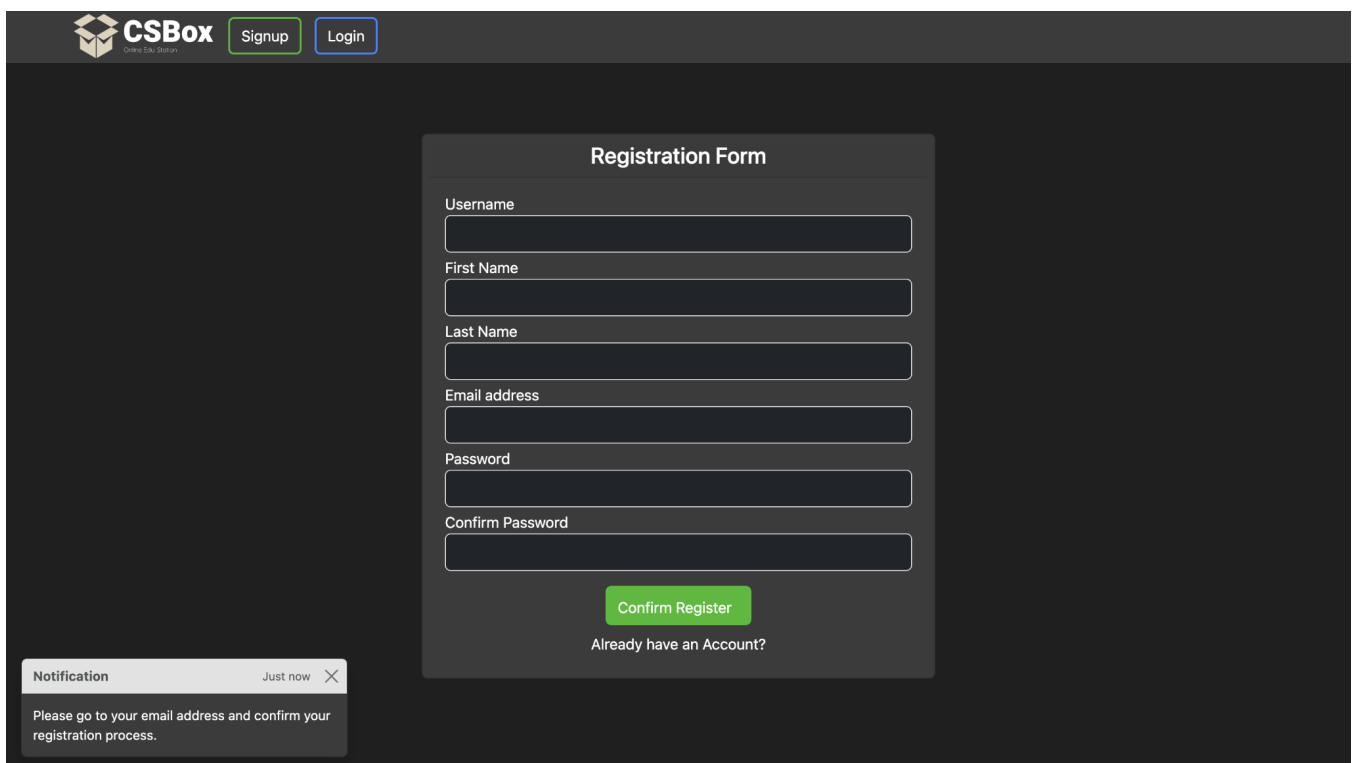
Please fill out this field.

Developer Team

Tanvir Mahmud
Software Engineer
tanvirmahmud.cse66@gmail.com

[Github](#) [LinkedIn](#) [Facebook](#)

Figure 4.2.1 – Interface for user as landing page



The registration form is a central modal with a dark background. It contains fields for Username, First Name, Last Name, Email address, Password, and Confirm Password. A 'Confirm Register' button is at the bottom. Below the button is a link 'Already have an Account?'. A notification box at the bottom left states: 'Please go to your email address and confirm your registration process.'

CSBox Offers Edu Station

[Signup](#) [Login](#)

Registration Form

Username

First Name

Last Name

Email address

Password

Confirm Password

[Confirm Register](#)

[Already have an Account?](#)

Notification Just now [×](#)

Please go to your email address and confirm your registration process.

Figure 4.2.2 – Interface for user registration form

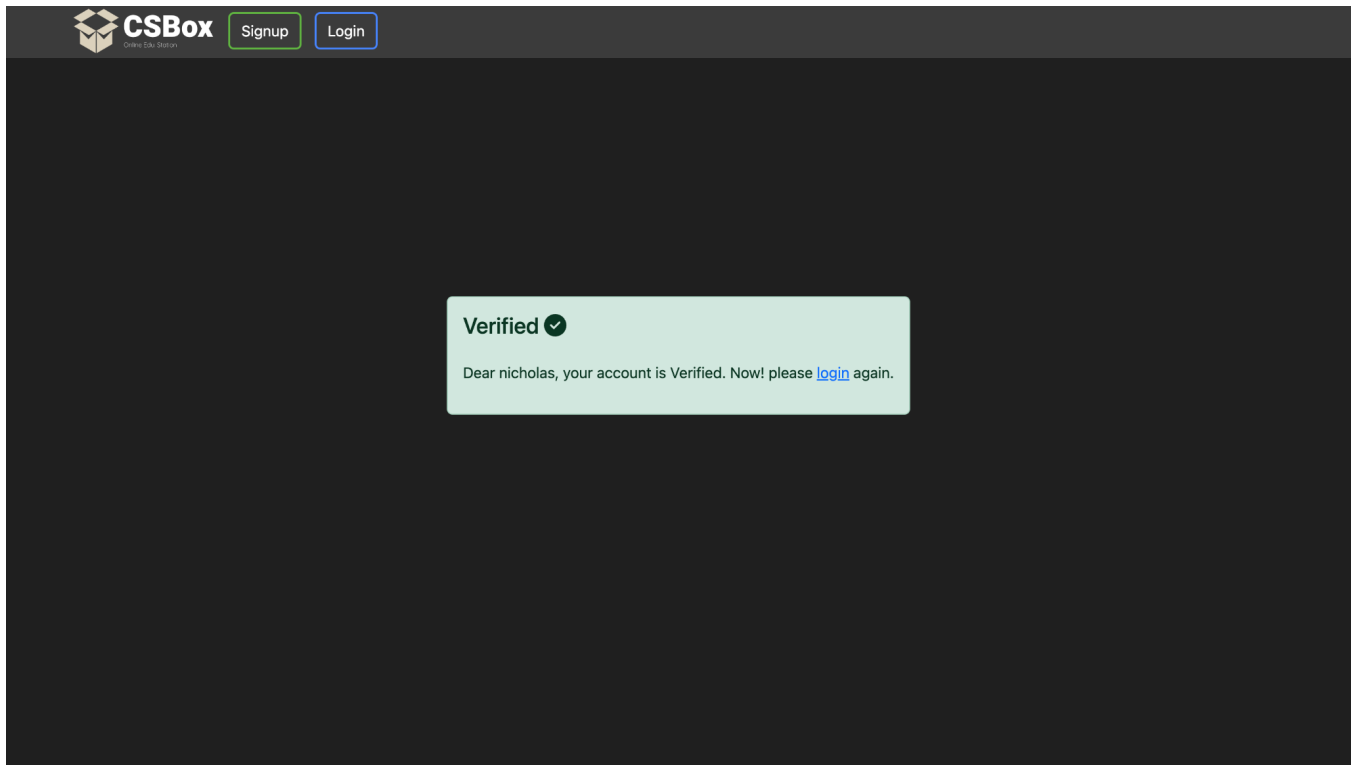


Figure 4.2.3 – Interface for confirmation of verified users.

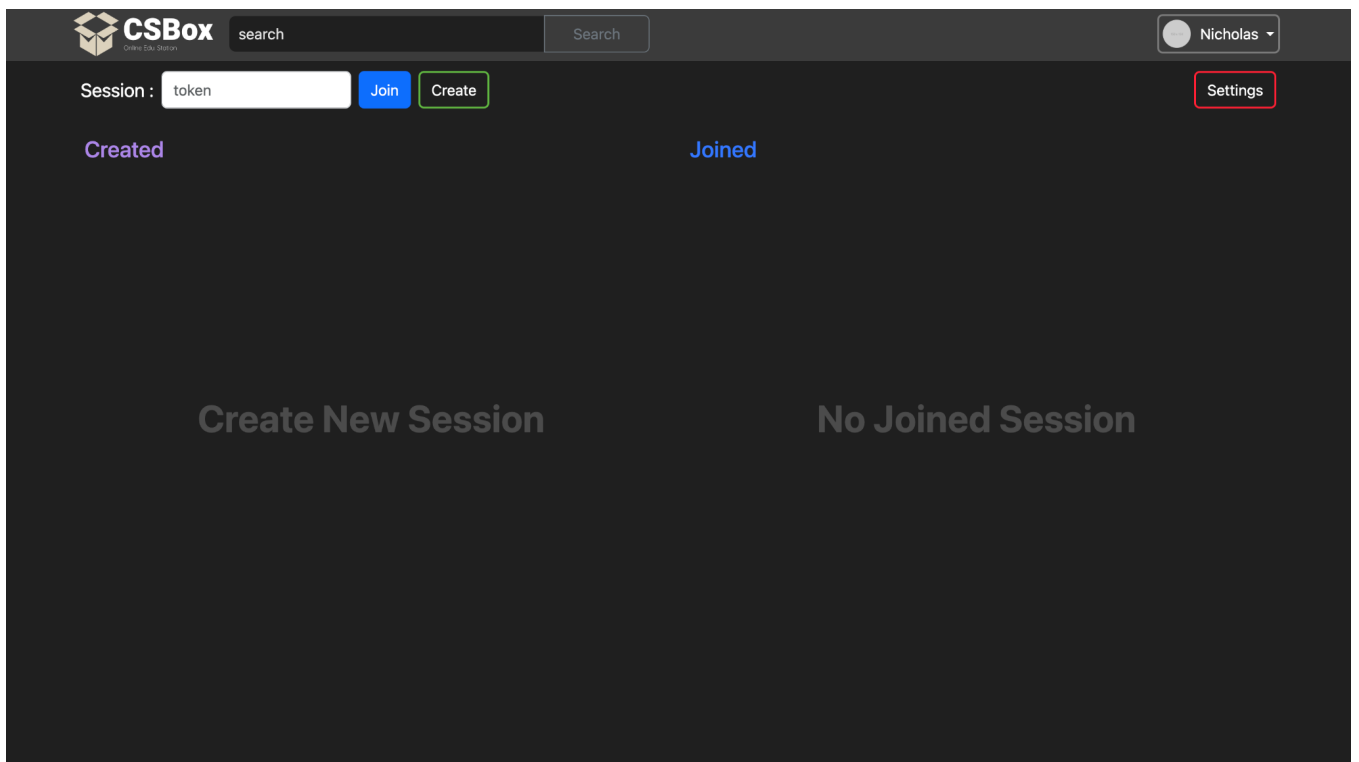


Figure 4.2.4 – Interface for user landing home page

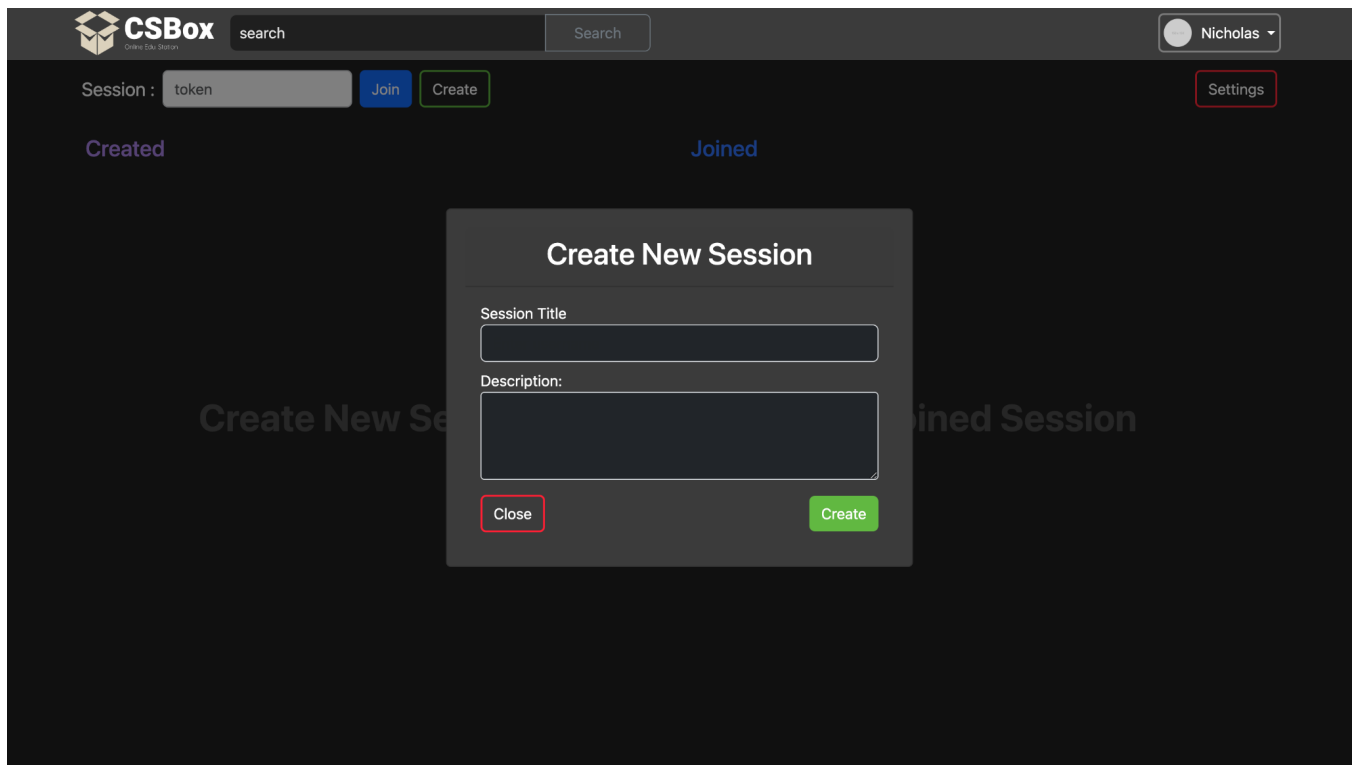


Figure 4.2.5 – Interface for creating Session.

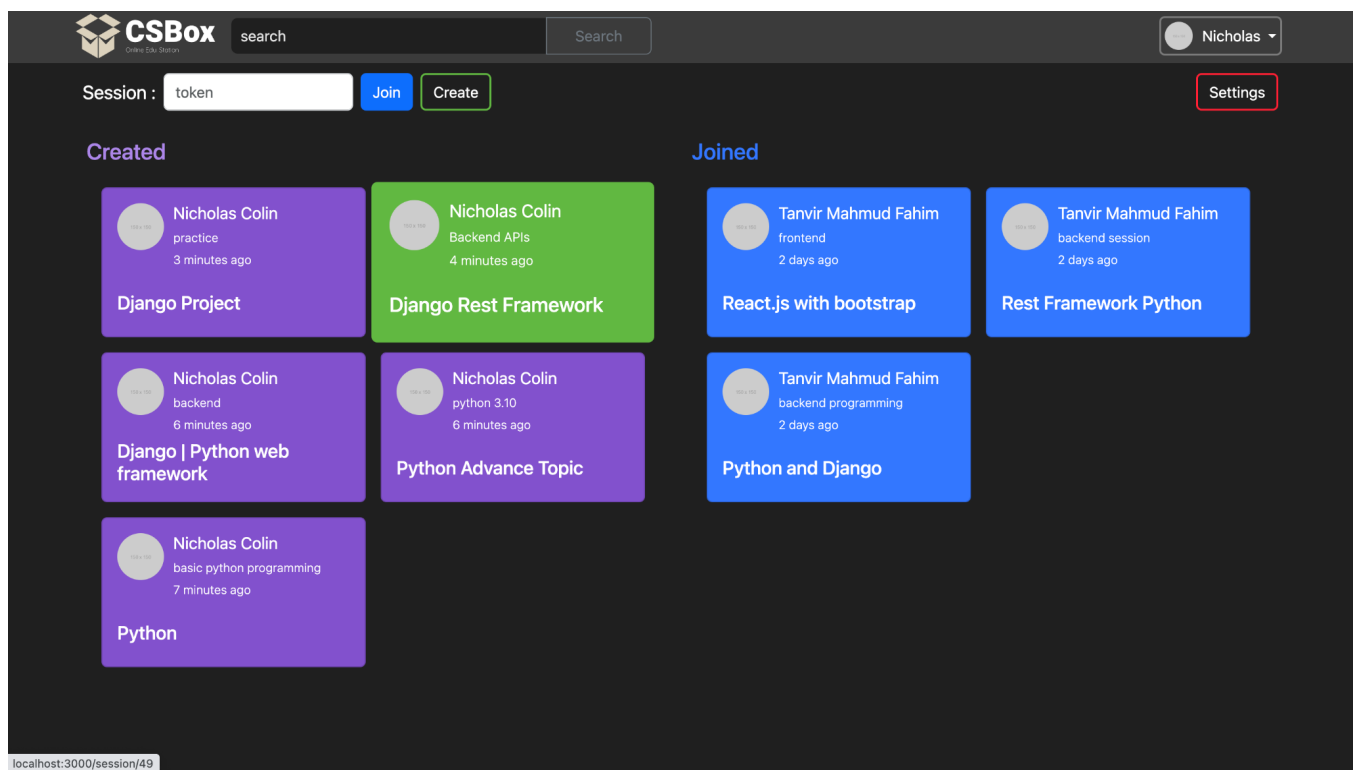


Figure 4.2.6 – Interface for created and joined session.

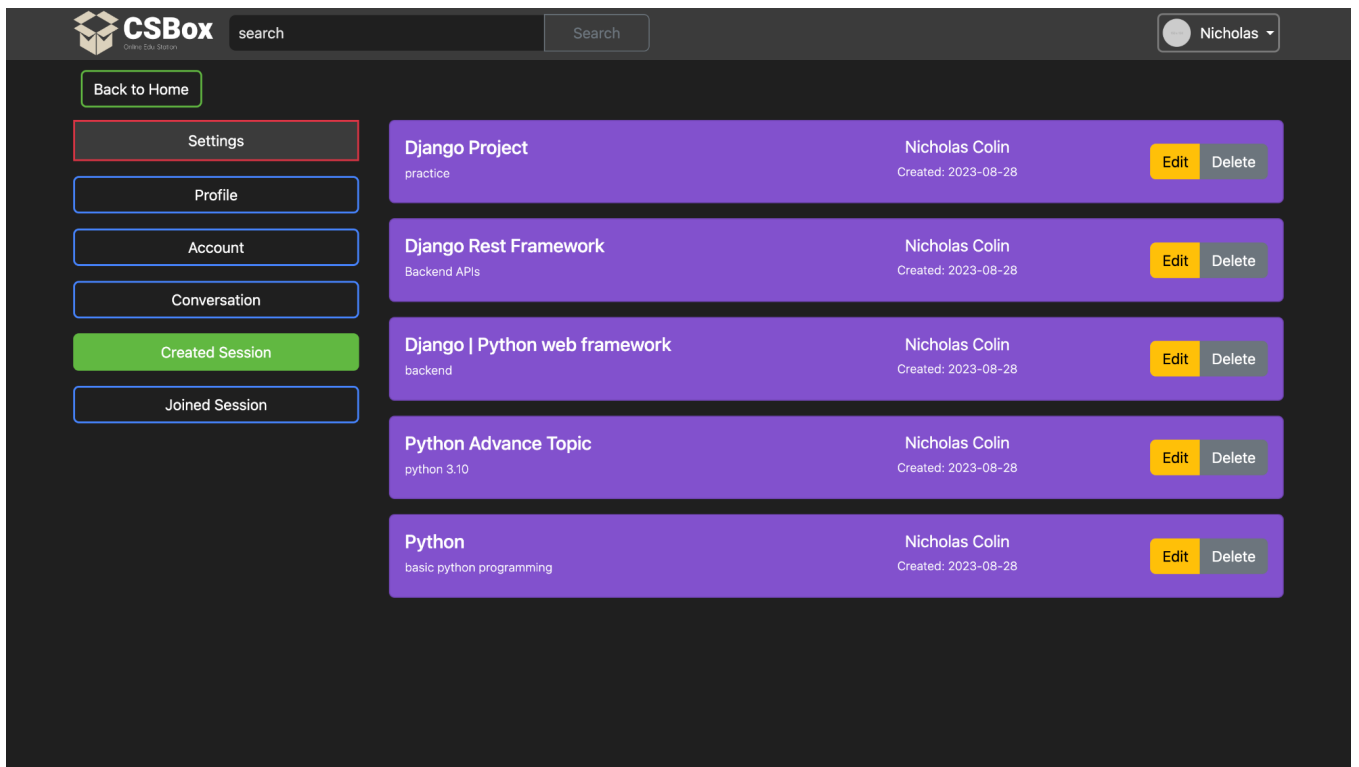


Figure 4.2.7 – Interface for settings (created session).

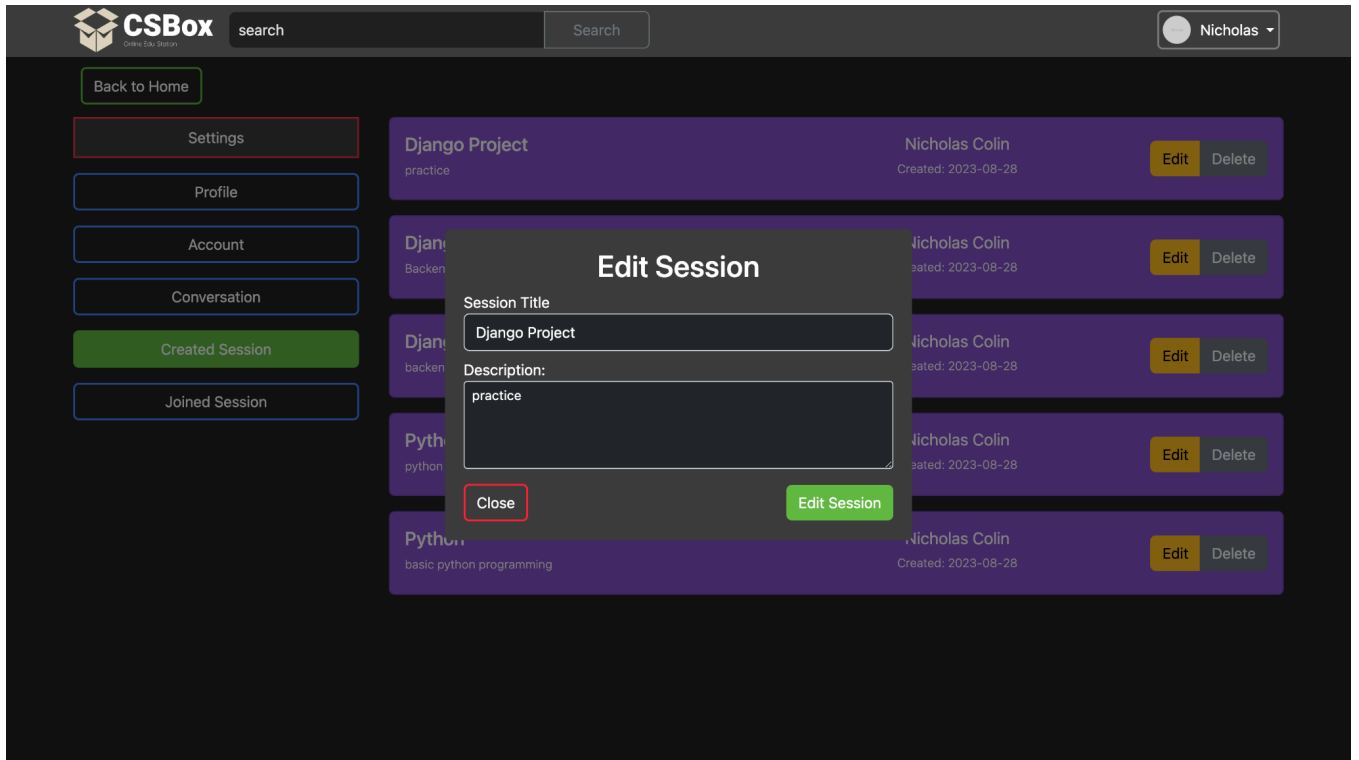


Figure 4.2.8 – Interface for Edit Created Session.

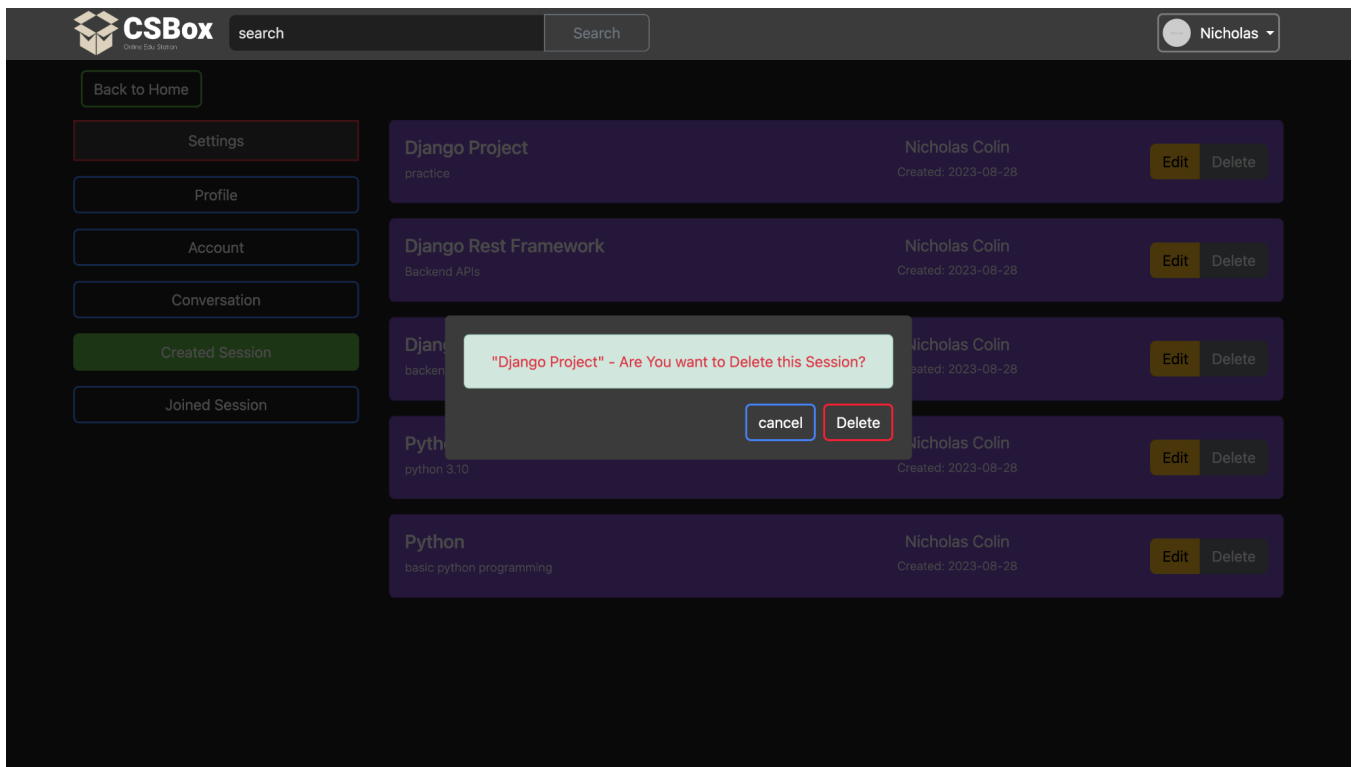


Figure 4.2.9 – Interface for Delete Created Session.

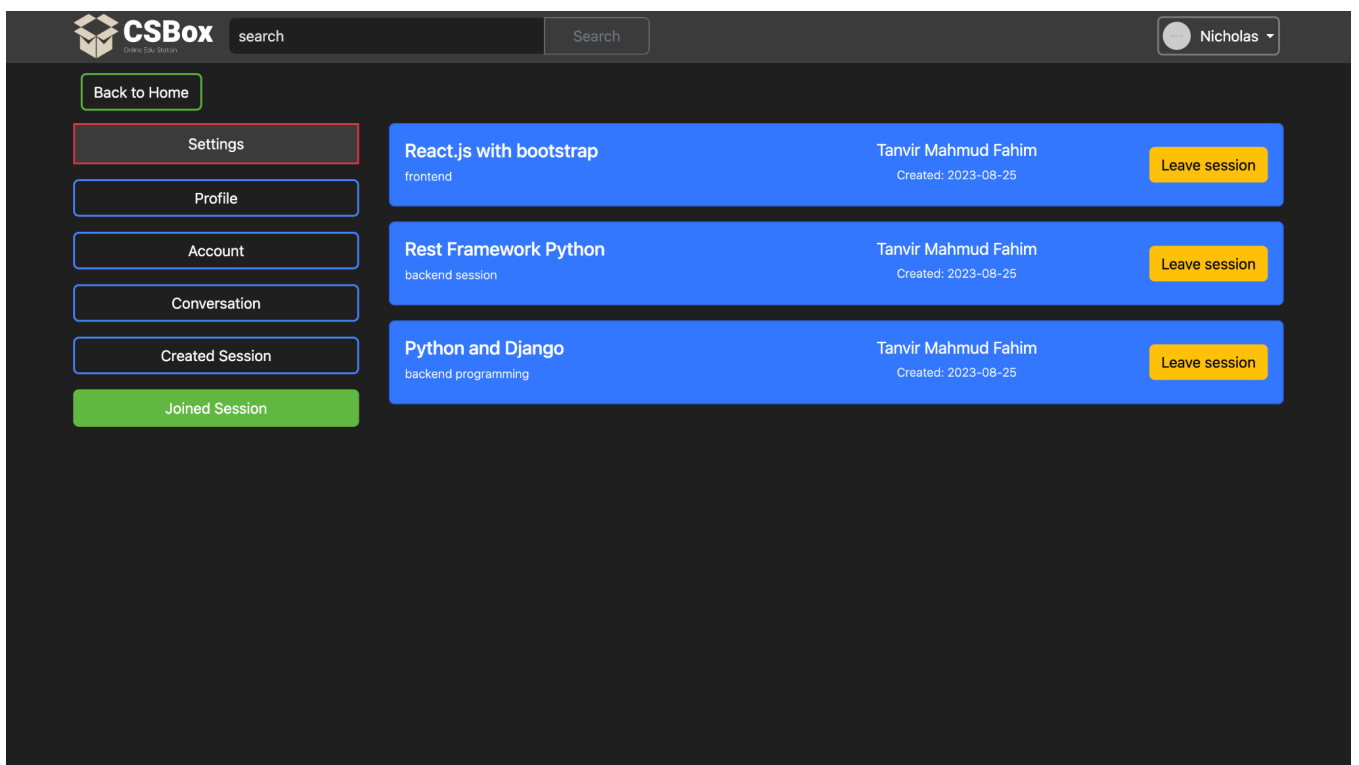


Figure 4.2.10 – Interface for settings (joined session).

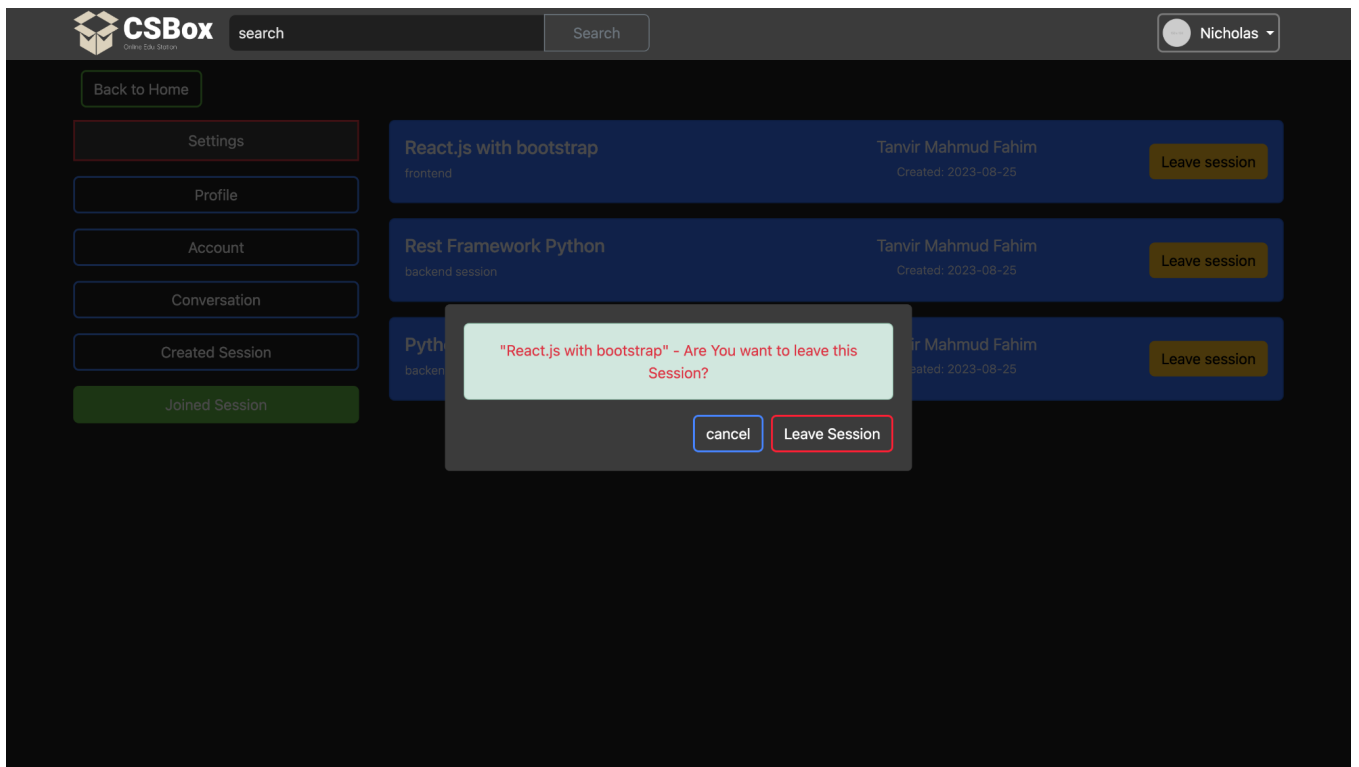


Figure 4.2.11 – Interface for Leave Joined Session.

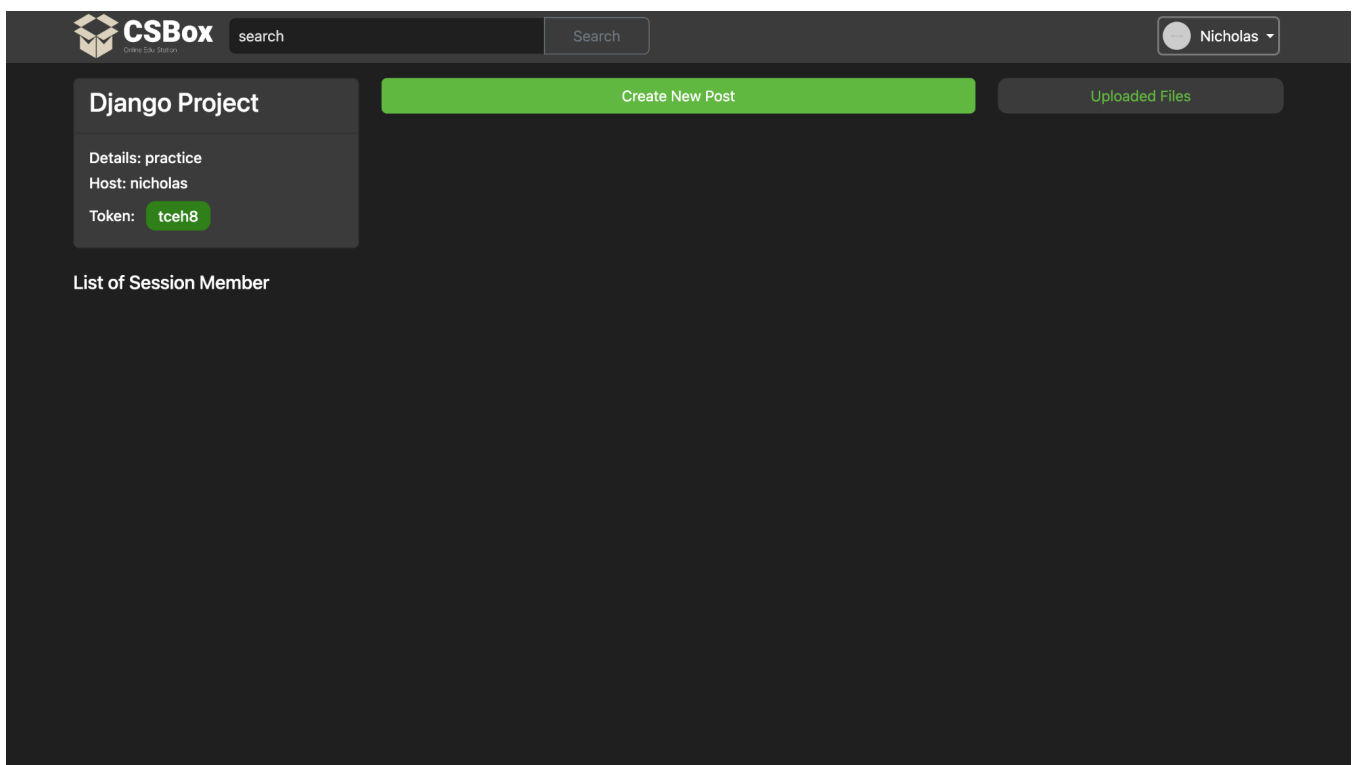


Figure 4.2.12 – Interface for Individual Session Landing page.

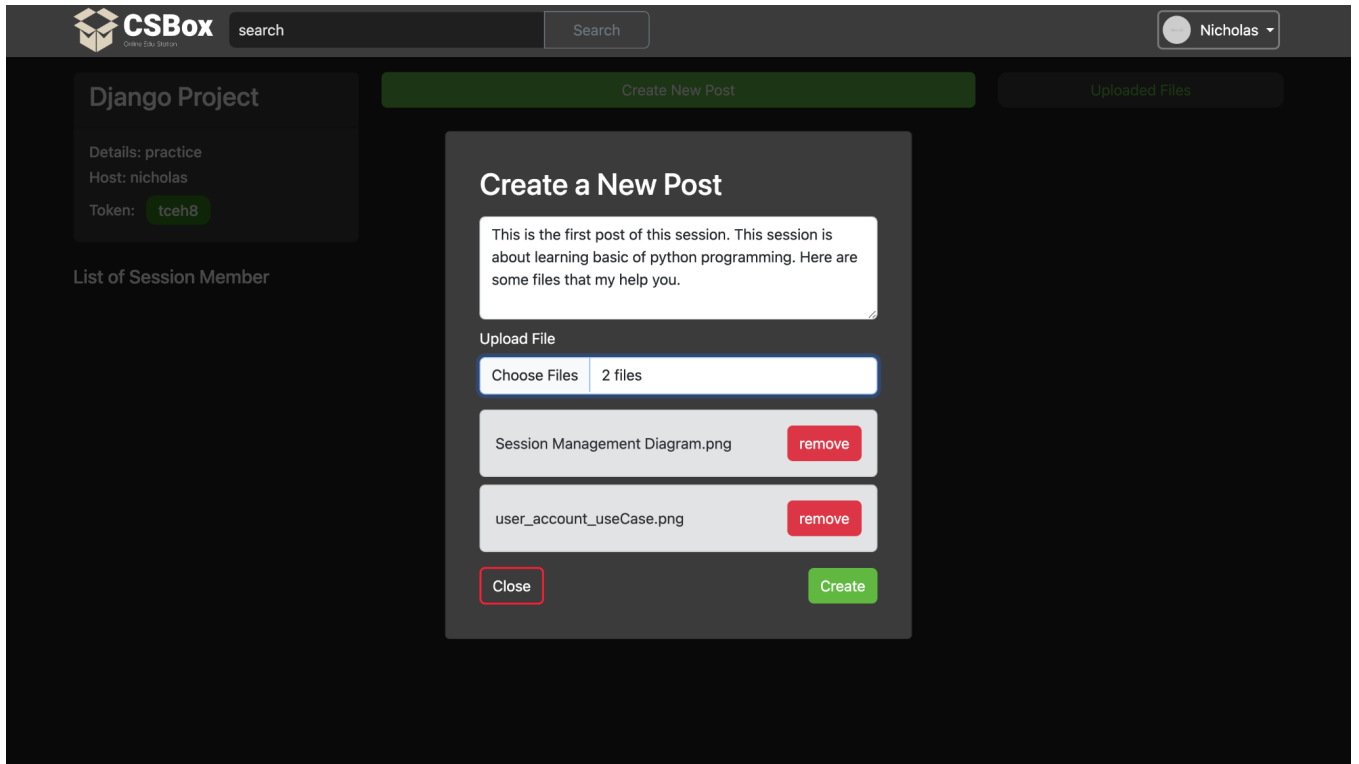


Figure 4.2.13 – Interface for Creating post in Session.

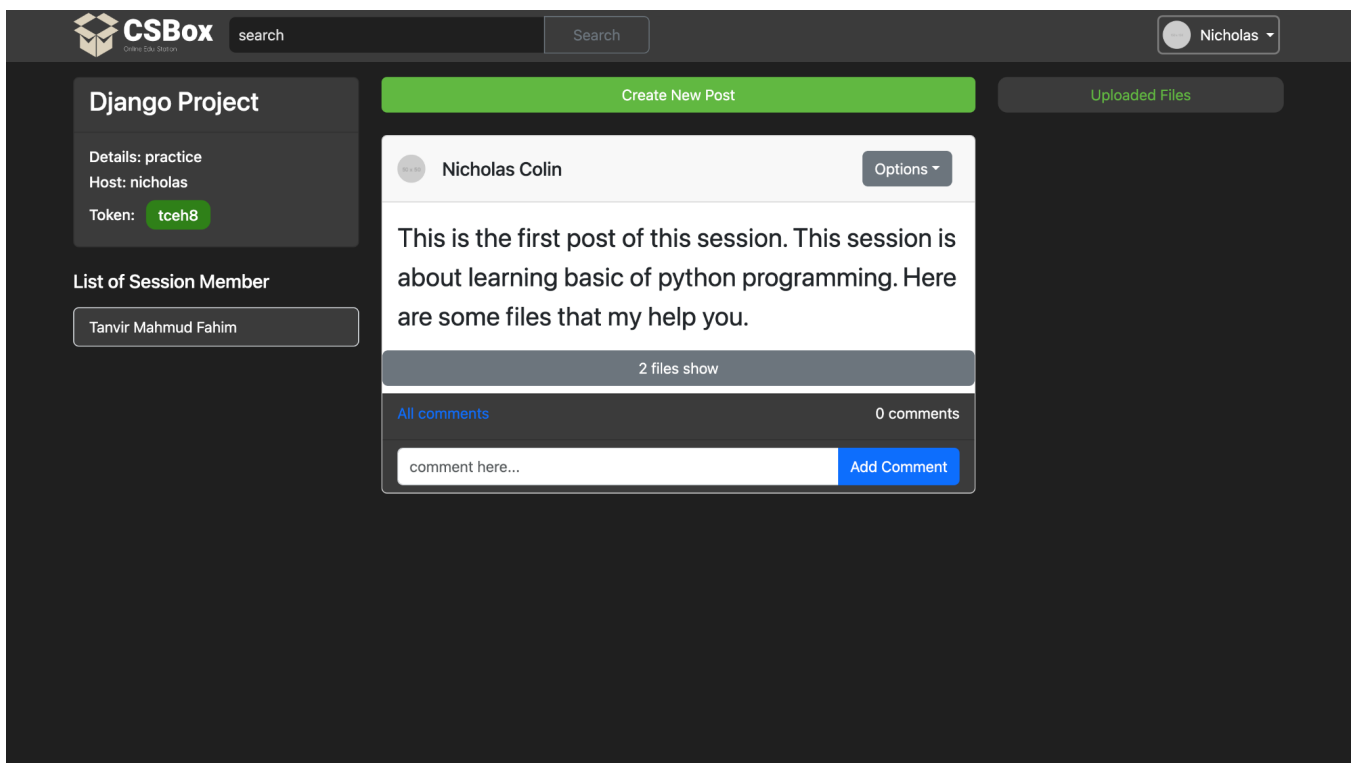


Figure 4.2.14 – Interface for Session post.

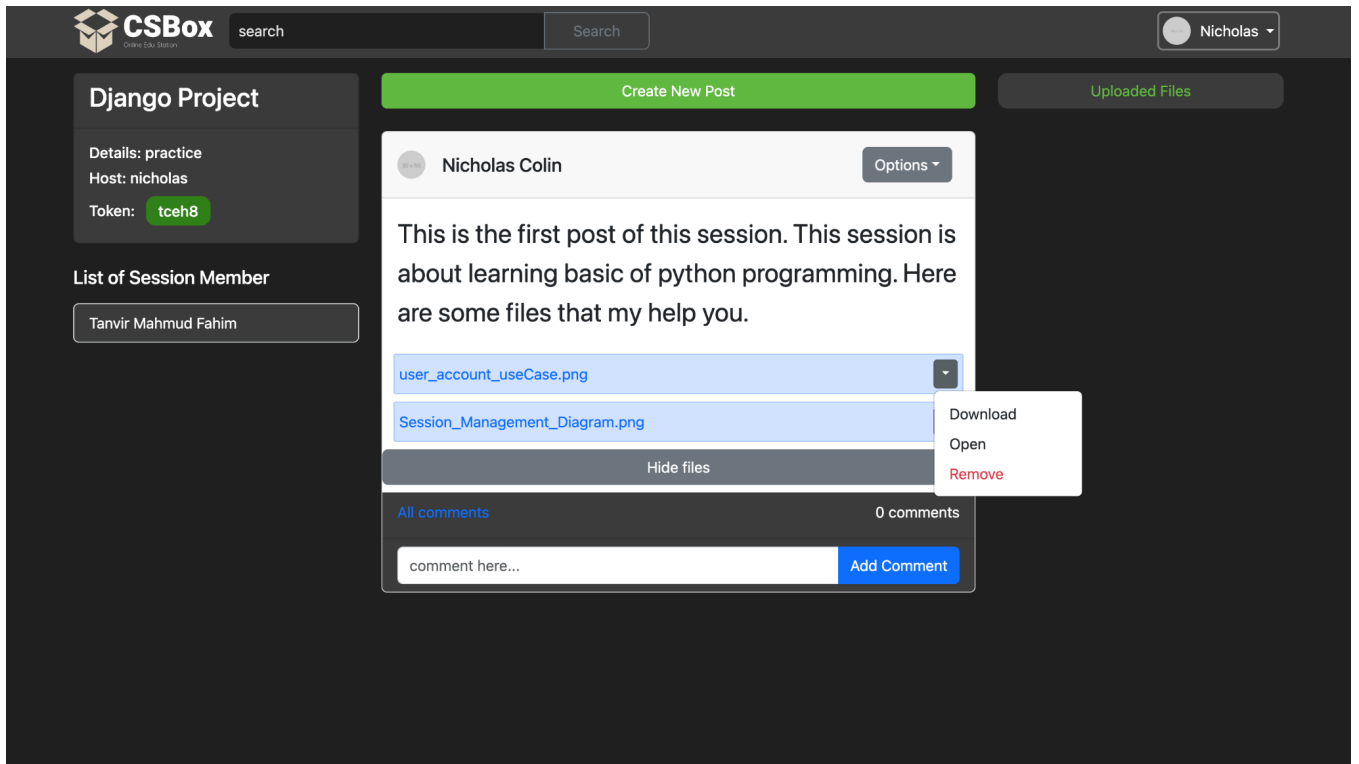


Figure 4.2.15 – Interface for Post’s file access.

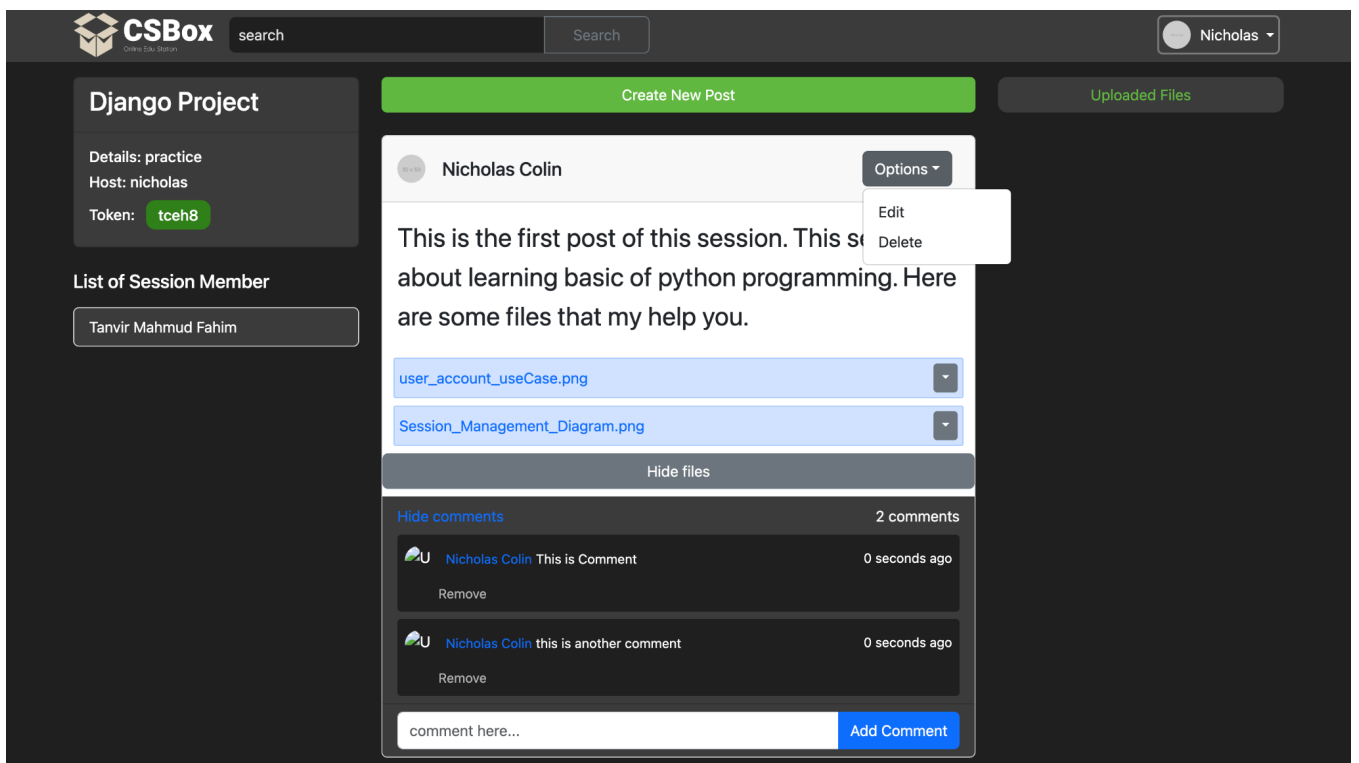


Figure 4.2.16 – Interface for post CRUD accessibility

5 Conclusion

After this successful project, we achieved our final goal. This is a huge milestone for us, we developed our requirements and needs. There were lots of problems, challenges and bugs and errors in our development time but we took the challenge and did it. We motivate our-self “There is always a way”. There will be a lot of updates and optimization in our next SDLC iteration.

5.1 *Future Work*

Lot of Research & Development we make our product workable and after reaching this goal, we have another plan to make this project workable in the real world. We will host this application in a cloud space, and we will buy a business domain for this. We will make marketing in our connections and we will try to find funds and venture capital for our project.

References

- [1] Django was invented to meet fast-moving newsroom deadlines, while satisfying the tough requirements of experienced web developers.
[Online] Available: <https://www.djangoproject.com/start/overview/>
- [2] Django Tutorial Part 5: Creating our home page
[Online] Available: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Home_page
- [3] ASGI is a spiritual successor to WSGI, the long-standing Python standard for compatibility between web servers, frameworks, and applications.
[Online] Available: <https://asgi.readthedocs.io/en/latest/introduction.html>