

CloudLGU

Final Report

Group 1

Gao Ziqi	118010077
Xia Hanyang	118010339
Wang Mingjie	119010300
Zhang Yizhan	119010450

School of Data Science
Chinese University of Hong Kong, Shenzhen

May 13, 2022

Contents

1	INTRODUCTION	1
1.1	Project Overview	1
1.2	Objective	1
1.3	Highlights	1
1.4	Project Statistics	3
2	System Architecture Design	5
2.1	System Architecture	5
2.2	Integrate Front-End with Back-End and Database	6
2.2.1	Front-End and Back-End	6
2.3	Deployment	7
3	Detailed Description of Components by DFDs and UMLs	7
3.1	User Login and Registration	8
3.2	Office Time Appointment	10
3.3	Posting and Commenting System	12
3.4	Administrator	14
4	UI Design	14
4.1	Log in	14
4.1.1	Exist User Login	14
4.1.2	Reset Password	15
4.1.3	Register New User	15
4.2	Forum	15
4.2.1	Show All Posts	16
4.2.2	Add Post	16
4.2.3	Show Post	17
4.3	Office Time	17
4.3.1	Search Week	18
4.3.2	Search Prof	18
4.3.3	My	19
4.3.4	My Office Time	19
4.4	Personal Center	21
4.4.1	Basic Information	21

4.4.2	Change Password	21
4.5	Manager User	22
4.5.1	Delete User	22
4.5.2	Edit User	23
5	TEST	24
5.1	Test Overview & Test Plan	24
5.2	Case 1:Postman Test	24
5.2.1	Create Post	25
5.2.2	Show all Post	25
5.3	Case 2: Mock.js Test	26
5.3.1	Log in	26
5.3.2	Get User Profile	26
5.3.3	Log Out	27
5.4	SonarQube Scanning for Code Style	28
6	Lessons Learned	29
7	Conclusion	30

1 INTRODUCTION

1.1 Project Overview

Our project, CloudLGU (meaning "the Chinese University of Hong Kong, Shenzhen on the cloud") aims to provide a platform for all students and faculty members in CUHK(SZ) to communicate, share and help each other in the hope that every CUHK(SZ)er can meet on the cloud, stand together and feel the warmth of the community especially in this difficult time shrouded by the COVID.

Our group members include Wang Mingjie (ID: 119010300), Gao Ziqi (ID: 118010077), Xia Hanyang (ID: 118010339) and Zhang Yizhan (ID: 119010450). Mingjie and Yizhan implemented the front-end of the website while Ziqi and Hanyang implemented the back-end of the system. Like the codes, the report is written by us together where every one writes about the part of system he implements. Every one contributes equally to the project.

1.2 Objective

Our product design idea comes from the online forum "LGULife", which provides a platform for CUHK(SZ) students to post and reply to messages. Our goal is to provide a similar system focusing on information exchange. Our system targets the faculty members as well. We would also include a section showing the covid dynamics, which will focus on Shenzhen and the campus's relevant policies. We hope to provide an online platform with clean and user-friendly UI for students and the faculties where they can get and share information with no barriers.

Our system is mainly composed of 4 parts, including registration (sign up) and login, office time booking system, forum system, and user management system. Every one with a CUHK(SZ) email account can register and log in to our web platform. They can also modify their profile. The administrator can modify the profile of every user, including the name, password, introduction and identity. The professors can post their available time to communicate with students and the students can make appointments with the professors. All the users can post and comment messages as our system aims to provide an information-sharing platform. We would post an article in the forum regarding the pandemic situation in the mainland, the campus's prevention measures and policies in our website as well. More details of these features will be introduced in later sections.

1.3 Highlights

Forum One major feature of our website is our forum where every one can post articles and reply to other articles. We aim to provide an information-sharing platform. We may see lots of research assistants recruitment advertisements from all kinds of WeChat groups. Or perhaps, we may see many WeChat

groups to resell some spare and idle "gadgets." Usually, those WeChat groups are not specific to share those information, so it is relatively easy to miss those information. Hence, we want a platform to pool all those recruitment or "Wanted" to break the "Information Fort" and offers unimpeded information to the students and faculties, so that the users can find all the information they may be interested in without trolling their chat history. On the other hand, we observe that due to the unpredictable pandemic situation, the campus's policies regarding preventing and controlling the pandemics, include policy of returning to and leaving the campus, change very frequently. Many students would ask the tutors in WeChat group for detailed prevention control measures. At the same time, the campus have different policies for students returning to campus from different regions according to the classification of cities by the government. However, the campus won't declare the classification explicitly and the students should check by themselves. All this may cause low efficiency, misunderstanding problems. So we decide to post an article showing the pandemic dynamics in Shenzhen and the campus's detailed policies.

Office Time System The other major feature of our website is our office time system where professors can create office time when they are available and students can reserve available office times. In CUHK(SZ), contact and communication between students and faculties are always welcomed. However, students may find it "hard" to contact the professors. If one student is not familiar with one professor, it is not that appropriate to use WeChat because WeChat's casual social contact product positioning. On the opposite, it is not efficient enough to contact your professor especially when you want to make an appointment with him to request for a meeting, since you need to confirm the time of each other, let alone the email etiquette. Hence we would include a office time booking system and both CUHK(SZ) students and professors would enjoy our system.

Expected Customers and Users The targeted users of our system are students and faculty members of CUHK(SZ). Every one with a CUHK(SZ) email account would be welcomed to join the community. In view of the above features, we intend to develop an one-stop online information-sharing platform. The platform provides information regarding office time of professors, communication between community members and pandemic dynamics. We hope that our platform, which collects information distributed in different positions, can help CUHK(SZ)ers to get better and more timely information.

Programming Features

- **Dynamic router and navigation bar based on user identity.** Each user has one of following three identities: administrator, faculty and student. Users with different identities have different interfaces. For example, only the administrator has permission to check and modify profile of all user. One way to handle this is to write three different router and navigation bars and use the right

one when the user logs in. However, in our project, we write one only router in which each route has a *roles* attribute indicating the kinds of user who can access this route. When the user logs in, the routes would be generated automatically based on user's identity. Then the navigation bar would be generated based on the router. We also have a router permission check function. Every time the user change route, the program would check if he/she has the right to access the page. If not, the user would be logged out immediately. We also set a whitelist (like log in and register), pages in which can be accessed in any case.

- **Request interceptor.** Front-end process would frequently get or post data from or to the back-end and then the database. For each Ajax request, before the request API is called, we would first add a *token* header to the request. The *token* is the unique ID of each user. After the data was sent to front-end from back-end, a code was first examined before the data can be used by the process. If an error code was detected, which means the user can not access the data, either because of lack of permission or account anomaly, the user would be directly logged out. If the code indicates success, the data will be passed to the process for its purpose. Notice this interceptor with its added header and return code will not be explicitly declared in our API document.

1.4 Project Statistics

The statistics of our code comes from the static analyzer called SonarQube. The installation of this tool will be introduced later. And we can get **following statistics** (Note that the all codes in .Vue file, including HTML codes and CSS codes, are considered as JavaScript code):

Language	Lines of Code
JavaScript	4100
Python	1100
CSS	346
HTML	167

Table 1: Statistics (LOC) of Programming Language

We have 28 non-trivial functions in the front-end user interface. **Below table** show these functions.

ID	Function	Description
1	Login	Users can login if they already have an account
2	Forget password	Users can reset password with verification code sent by email
3	Register	Users can register with their CUHK email
4	Modify password with old password	Users can reset password with old password
5	Get user list	Administer can get the list of users and their details
6	Search user	Administer can search user by user name, email and identity
7	Sort user list	Administer can sort the user list by user name, email and identity
8	Reset user profile	Administer can reset user profile
9	Delete user	Administer can delete user account
10	Get profile info	User can view their profile
11	Update user profile	User can updatere their profile
12	Create time slot	Professors can create their office time
13	Delete time slot	Professors can delete their office time
14	Update time slot	Professors can update their office time
15	Book office time	Students can book office time
16	Search by professor's name	Students can search office time slot by professor's name
17	Search this week	Return all the professors who have office hour this week
18	Professor check office time	Professors can update their office time
19	Student Check Office Time	Students can update their office time
20	Create new post	Users can add new post to the forum
21	Delete post	Users can delete post on the forum
22	Update post	Users can update post on the forum
23	Show post	Users can view all details of a single post
24	Show all posts	Users can view all posts in the forum page
25	Create new comment	Users can add new comment to the posts
26	Delete comment	Writer can delete comment written by himself/herself
27	Update comment	Writer can update comment written by himself/herself
28	Logout	User can logout from the system

Table 2: Function Points

2 System Architecture Design

2.1 System Architecture

The functions of our systems have been shown in our DFDs, there is no need to further explain here. Hence we have below **technical architecture diagram**.

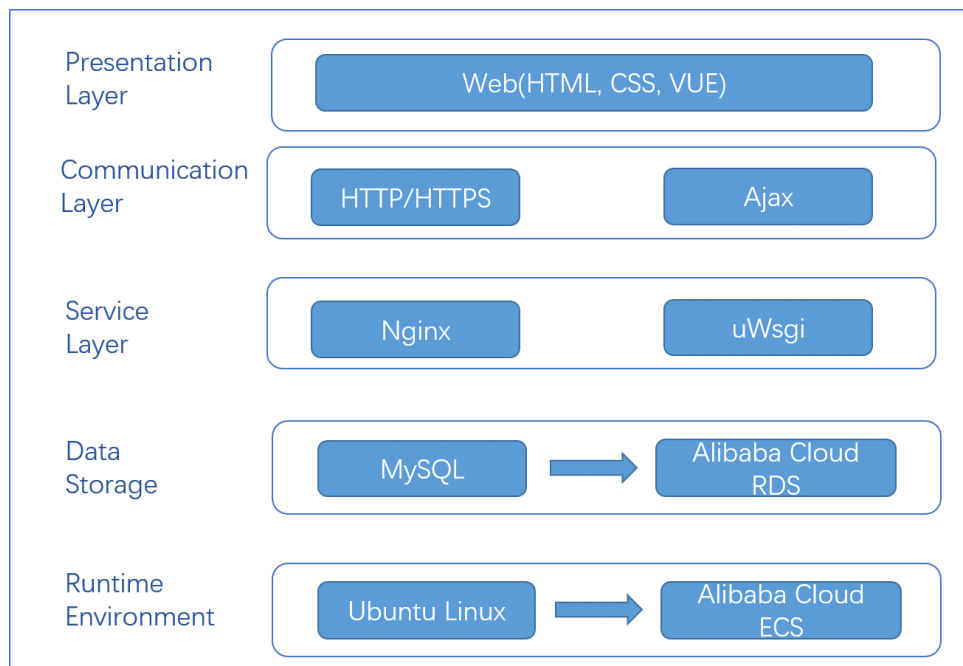


Figure 1: Technical Architecture

Our project is firstly deployed on our local machine. Then we deploy our project to the CentOS Server with Nginx and uWSGI.

We use MySQL to save relevant data for the data storage part, and work as our database management system (DBMS).

The front-end part is deployed on Nginx, while the Django python back-end part is first deployed using uWSGI. All the requests to the back-end will be passed to the uWSGI server with the reverse proxy feature of Nginx. Also, this kind of architecture has good expandability, where one can easily balance the load by adding more servers. Also, we have finished the deployment documents, and this can be used to deploy our projects to any Linux server or cloud servers like EC2. The deployment documents will be introduced in the next subsection.

In the communication layer, we use Ajax's GET and POST methods to cooperate with the reverse proxy feature to make the request and respond procedure with the back-end server.

Vue.js is used to build the user interfaces, which makes on top of standard HTML, CSS, and JavaScript, and provide a declarative and component-based programming model. See **whole sever-network**

architecture.

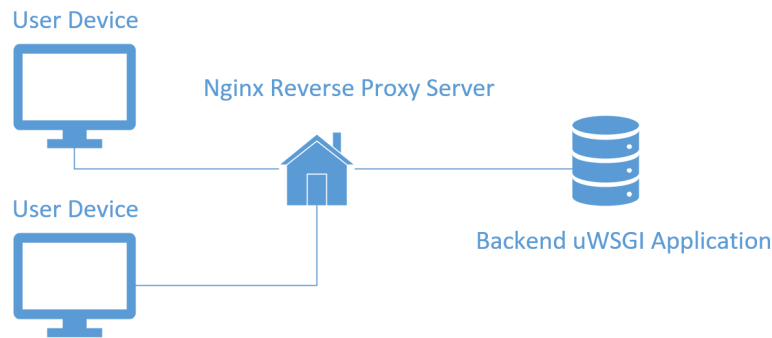


Figure 2: Server-Network Architecture

2.2 Integrate Front-End with Back-End and Database

2.2.1 Front-End and Back-End

Deal with CORS Problem

Note that we can use reverse proxy of Nginx to deal with this problem. Please have a look at the reverse proxy's setting in *nginx.conf*:

```
location /django_api {
    proxy_pass http://10.30.201.28:8080/api;
    proxy_set_header    Host      $host;
    proxy_set_header    X-Real-IP  $remote_addr;
    proxy_set_header    X-Forwarded-For  $proxy_add_x_forwarded_for;
    proxy_set_header    X-Forwarded-Proto https;
}
```

Also, we need to update the configuration in *./vue.config.js* of our Vue client like this:

```
proxy: {
  '/api': {
    target: 'http://10.30.201.28:8080/api', (this is the back-end port)
    ws: true,
    changOrigin: true,
    pathRewrite: {
      '^/api': ''
    }
  }
},
```

For the django part, remember to add the CORS related middleware into the *settings.py* and we are all done.

Application Interface

Interfaces are one of the most important parts for the software development, since they work as a bridge to connect the client and server. To achieve this goal, we have very detailed API document saved in our repository: [API Document](#)

Here is an excerpt of our API documents:

Office Time Part

Create Office Time Slot

API Address: `/api/officetime/prof/create`

Send

- `otDate`: Date of the office hour (String YYYY-MM-DD)
- `otStartTime`: Start Time of the wanted office time (String hh-mm)
- `otEndTime`: End time of the wanted office time (String hh-mm)
- `otLocation`: Where to hold the office time. (String)
- `Professor_userID`: Professor's userID (int sent from the backend during the login part)

Return

- `otID`: Office time ID stored in the Database (Int)
- `status`:
 - success: bool (success or failure of the create operation)
 - response: String (report success/failure reason (time conflict, location conflict))

2.3 Deployment

You can choose to deploy our projects locally by starting the server and client separately with our [README for Django](#) and [README for Vue](#).

In addition, we have also prepared detailed instructions for the deployment with Nginx and uWSGI. Due to the limited space, this part can be got from the Markdown file here: [Deployment Document with Nginx and uWSGI](#).

The above deployment documentation can also be used if we want to deploy our project into the cloud where we only need to take care about the IP address of cloud servers and databases.

3 Detailed Description of Components by DFDs and UMLs

The class diagram below shows the relations between the classes in our software development design process. Students and Professors are accounts that inherit the Users accounts. The difference is the user

identity. All office time, posts and comments will be stored in the corresponding table with an unique ID as the primary key. Username is not unique here.

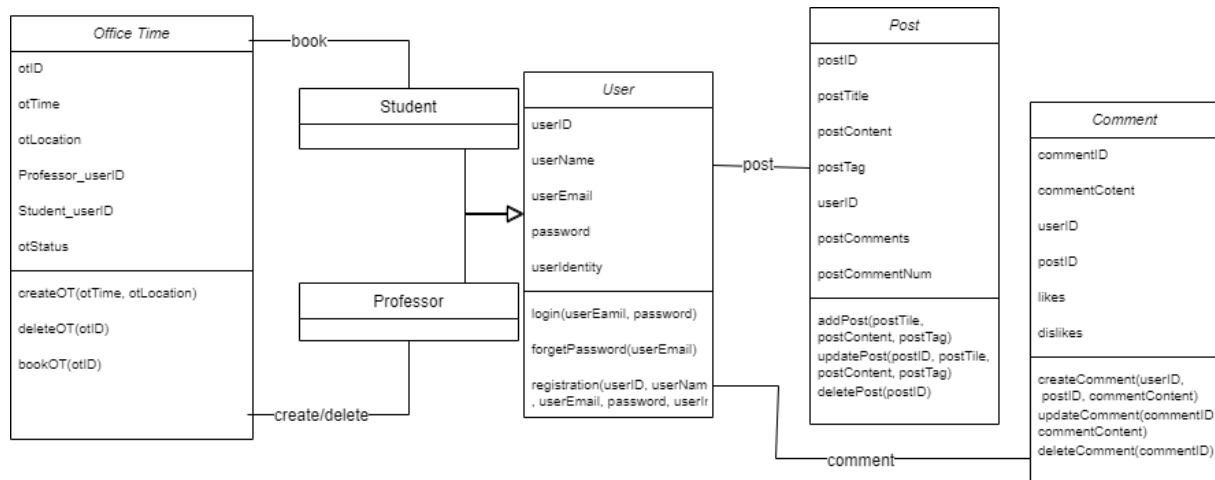


Figure 3: Class Diagram

The following user case diagram shows all of the major functions for our three kinds of users: Students, Faculty, and Administrator, and those functions will be introduced in detail later:

3.1 User Login and Registration

As shown in the DFD of user login and registration, users need to go over a user registration process before they could use our system. To create an account, we requires users to enter their CUHK(SZ) email to sign up. The email format should be "xxx@link.cuhk.edu.cn (students email)" or "xxx@cuhk.edu.cn (faculty)". Then, the terminal will send a verification email with verification code to the registered email, and they are required to enter the verification code in the UI to continue register.

If users have any login problems, our system provides a function for users to change their password. These users will be asked to type in their email for receiving an email with verification code to reset their password. Then by submitting the new password and the correct code, our database will update the user's data in the password field.

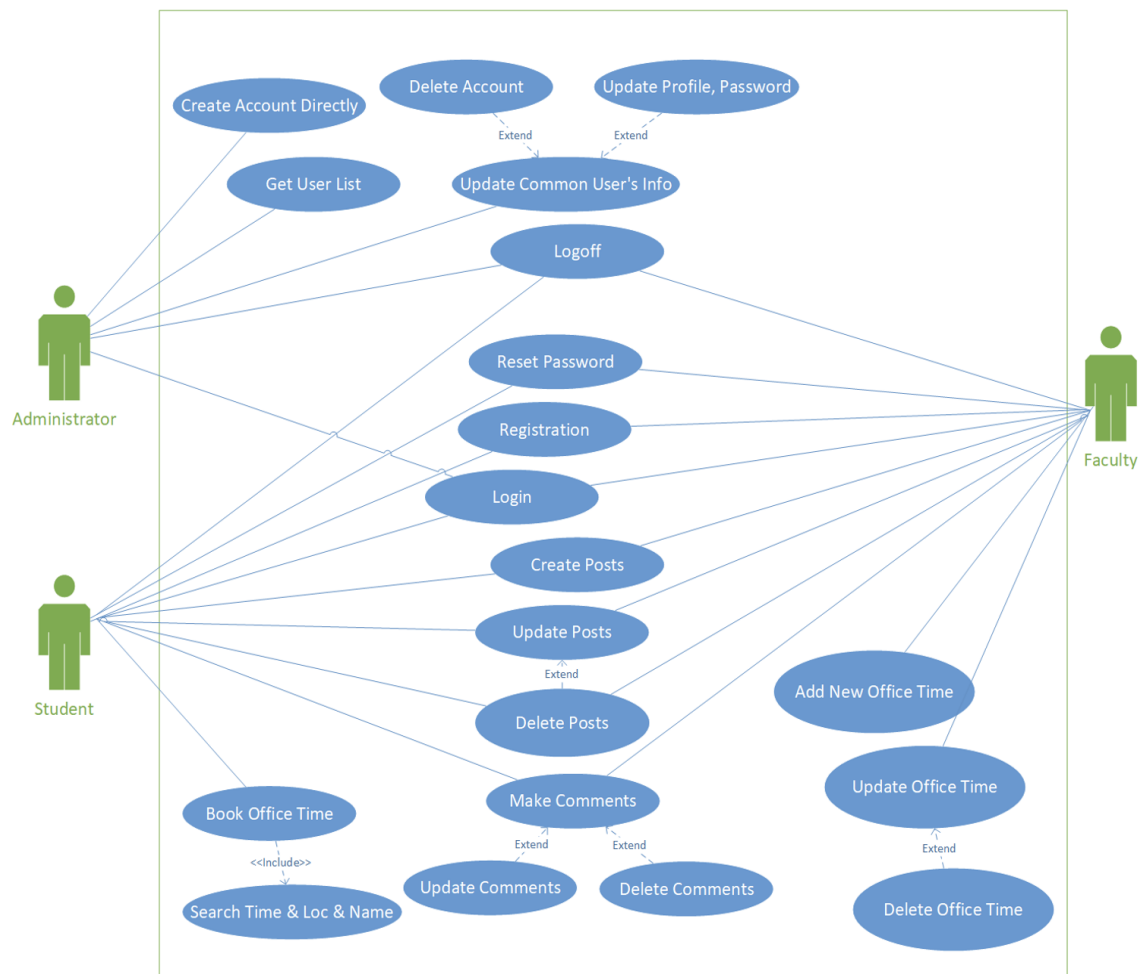


Figure 4: Use Case Diagram

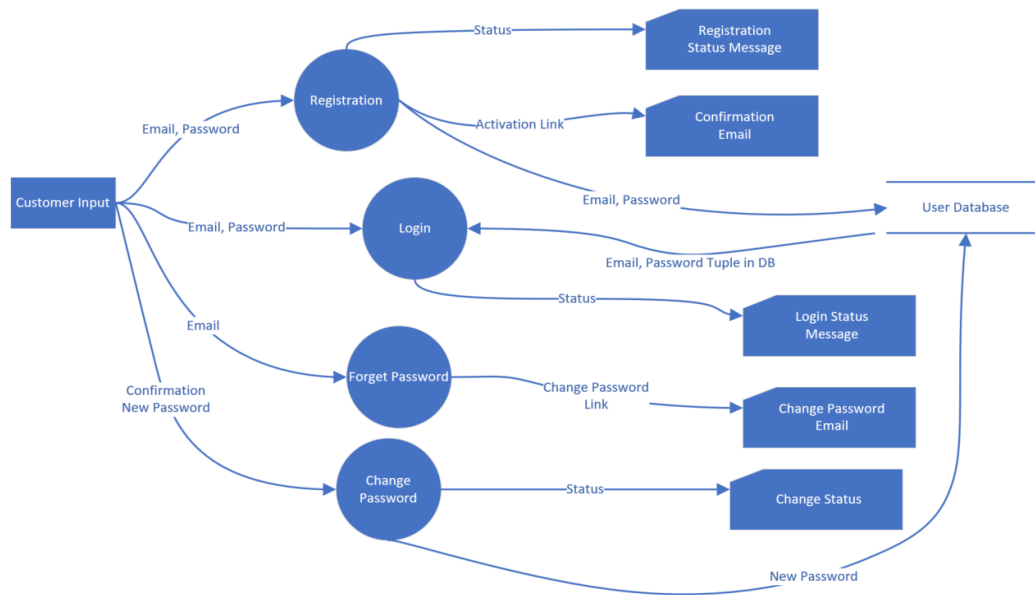


Figure 5: Data Flow Diagram: User Login and Registration System

3.2 Office Time Appointment

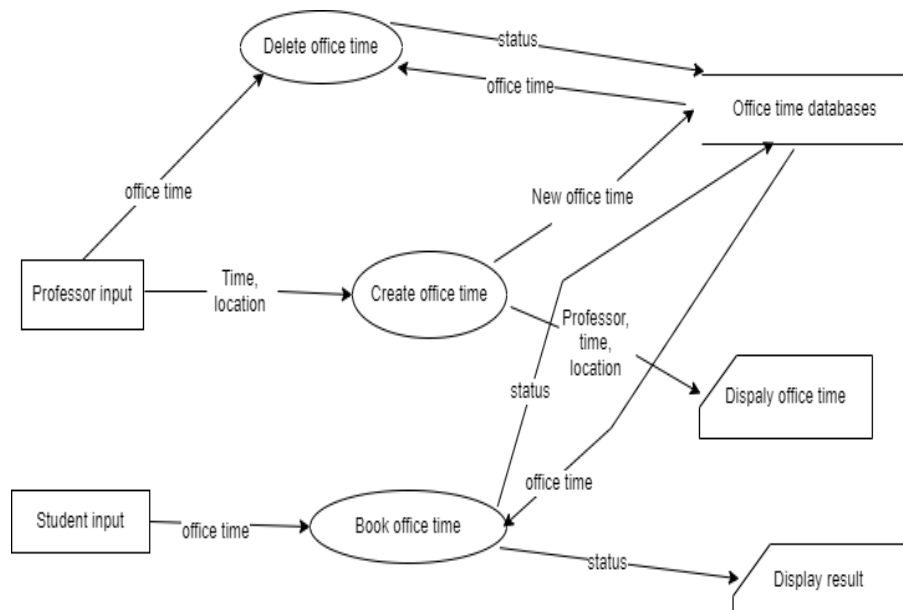


Figure 6: Data Flow Diagram: Office Time Booking System

The DFD of office time appointment system is shown above. Only professors or instructors offer office time to students. They should provide time and location of an office time whenever creating one. They can also delete the office time opened. The system will display all available office time for students to book. Also, students have two options to find office time. Firstly, they can get the lists of professors who will hold office time this week. Secondly, student can search the professor's time slots with professor's name. Each time a student tries to book an office time, the system would check it then return the

result of booking.

The following Finite-State Machines diagrams shows how the Office Time system work for **professors** and **students**:

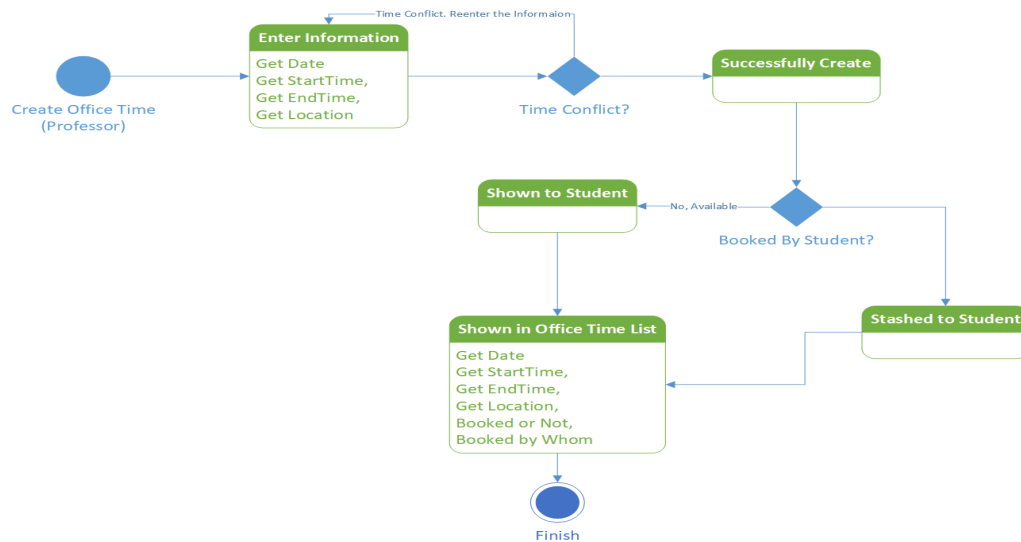


Figure 7: FSM: Office Time Booking System for Professor

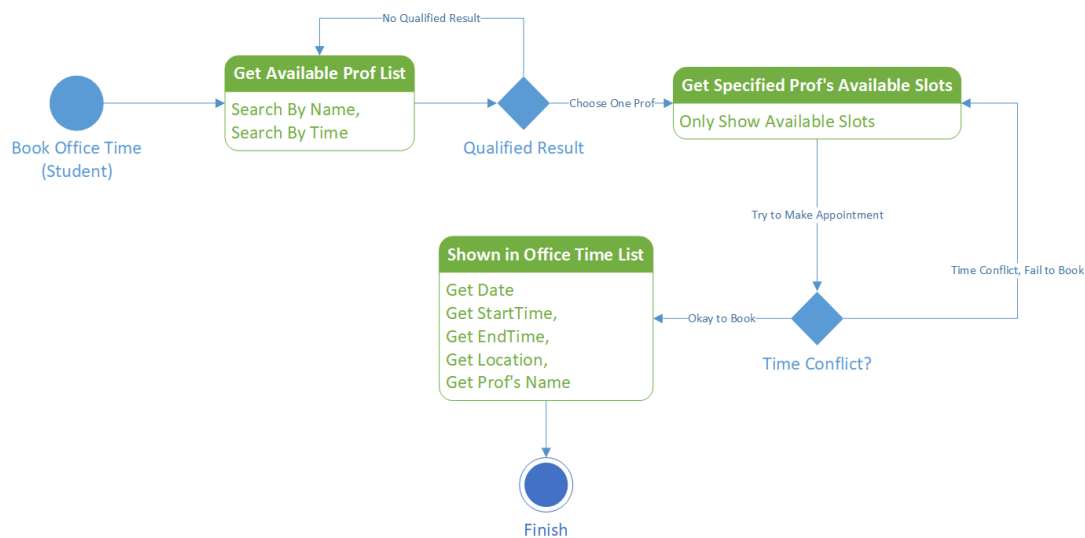


Figure 8: FSM: Office Time Booking System for Student

The following sequence diagram shows the timeline of the office time appointment system. Firstly, the instructor will post their time and location. Also, the instructor can update or delete the unoccupied time slots. After the professor post their office time slots, students can search by name, time, and location to find available office time and make appointment. Then, the student will get the confirmation, and the instructor can see every slots status (whether the slots are booked and booked by whom).

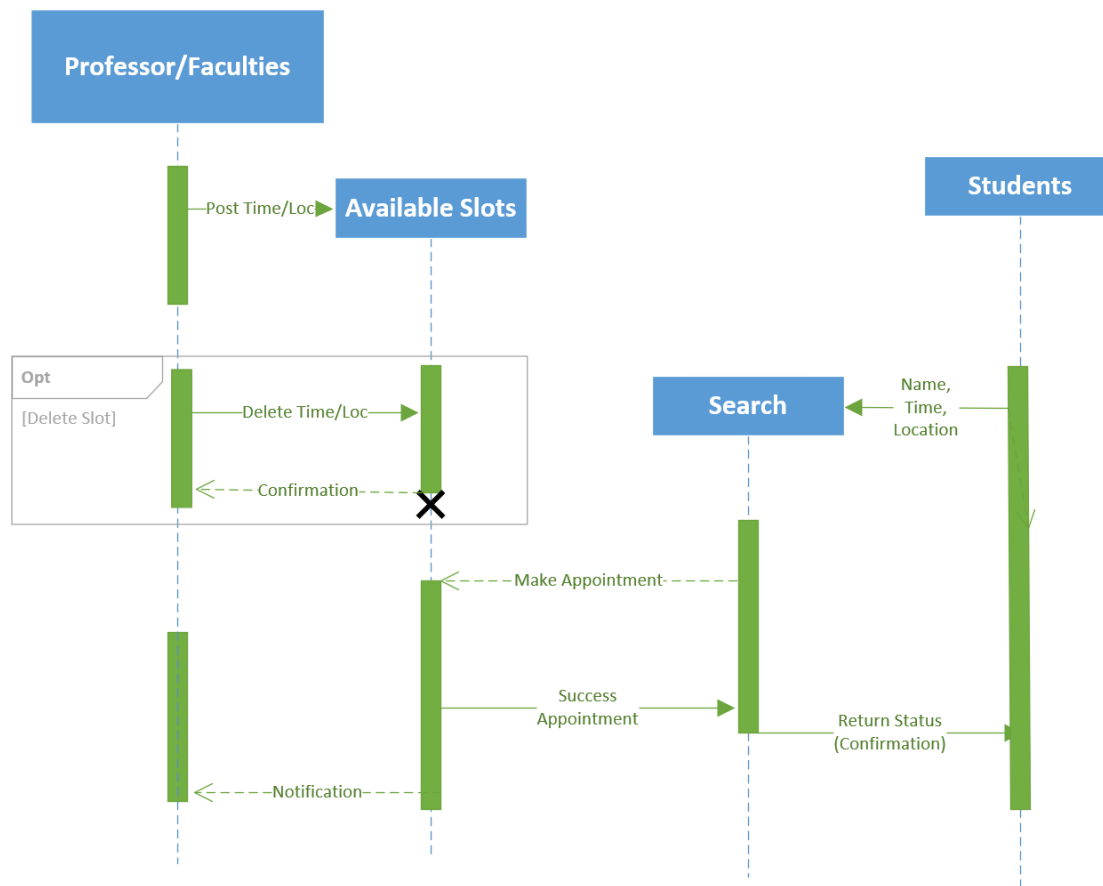


Figure 9: UML: Office Time Booking Sequence Diagram

3.3 Posting and Commenting System

As shown in the **DFD**, every user of the platform can post new messages and comment on existing posts. They can also update or delete anything posted or commented by themselves. Our posts have two general tags: TOP, DISCUSSION, HELP, RECRUIT. And they are ordered by the updated time. That is, the latest updated post will be shown first. The **following sequence diagram** shows the process of interactions between the users and our servers. Any user can create their posts, and the poster can update or close their posts at any time. Other users can make comments under the posts. Also, they can update or delete their posts. All the comments belong to the post. If the post is close, other users cannot make new comments under that post.

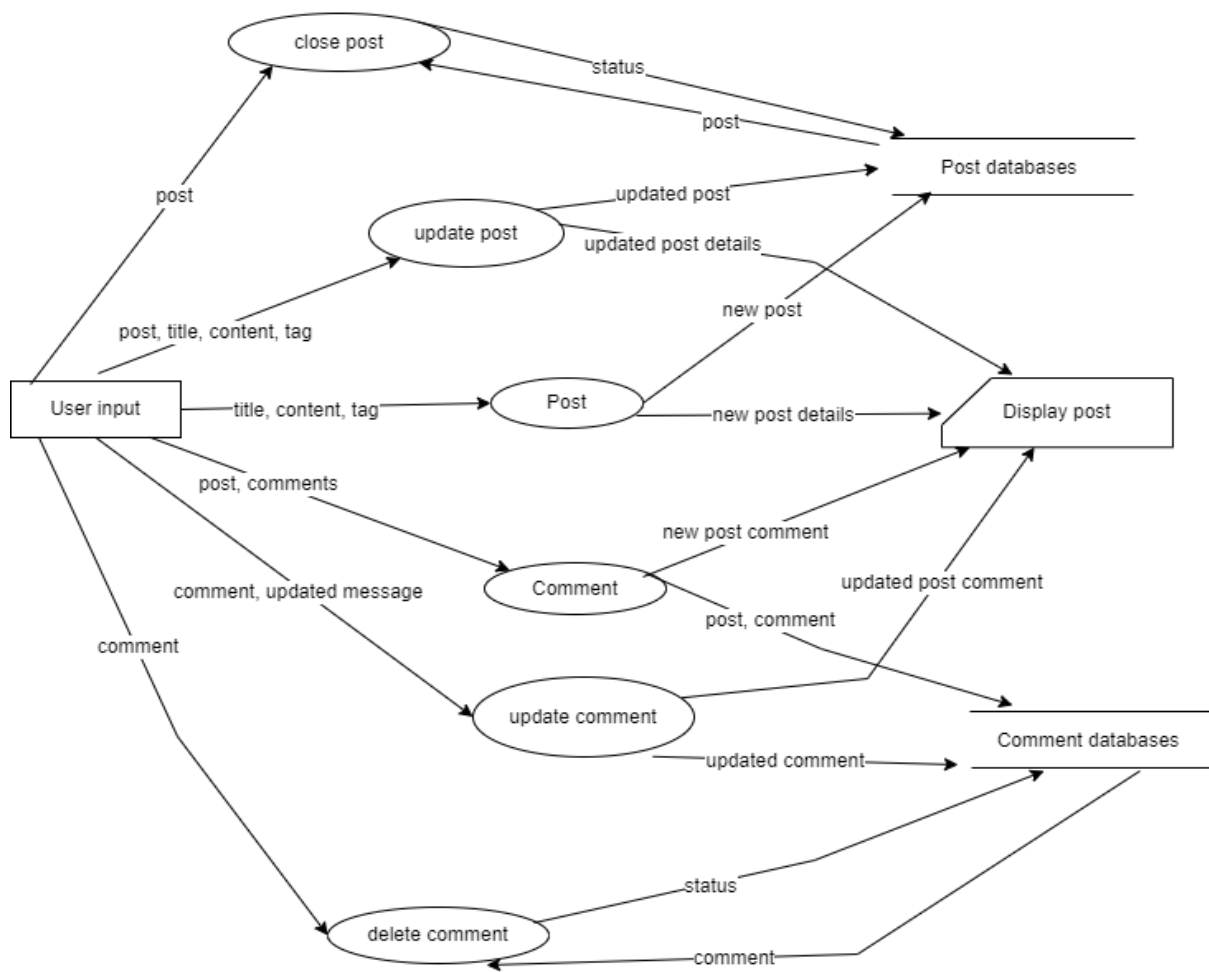


Figure 10: Data Flow Diagram: Posting and Commenting System

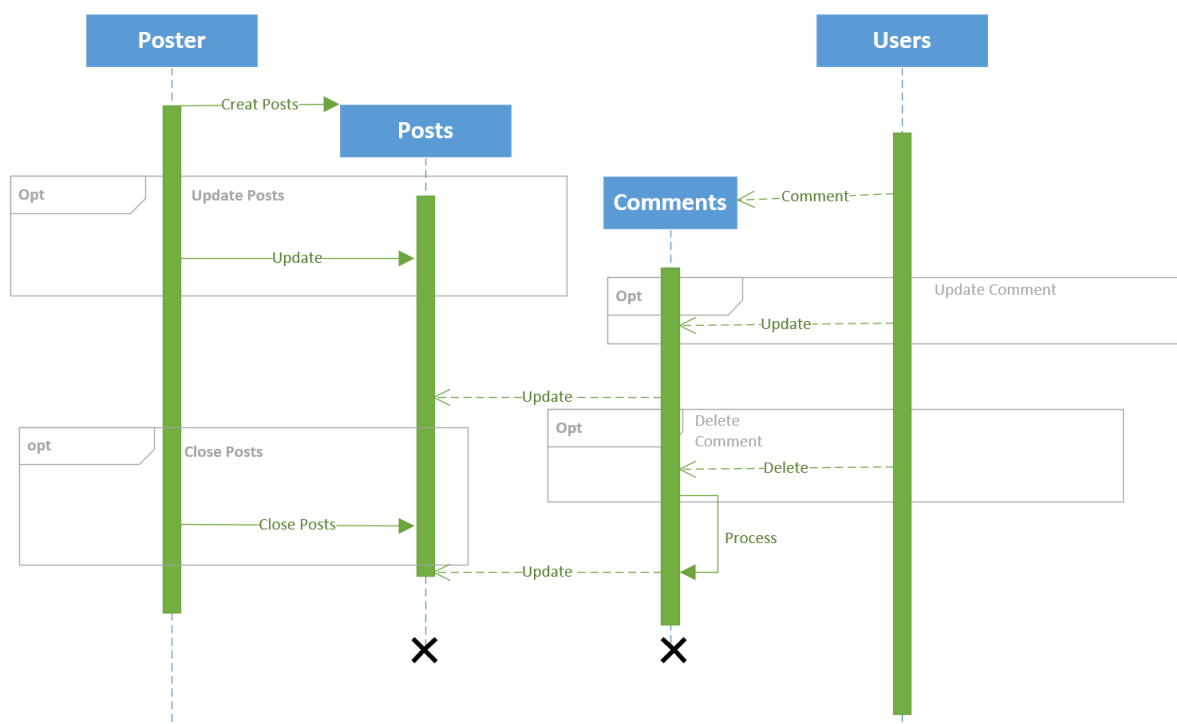


Figure 11: UML: Post and Comment Sequence Diagram

3.4 Administrator

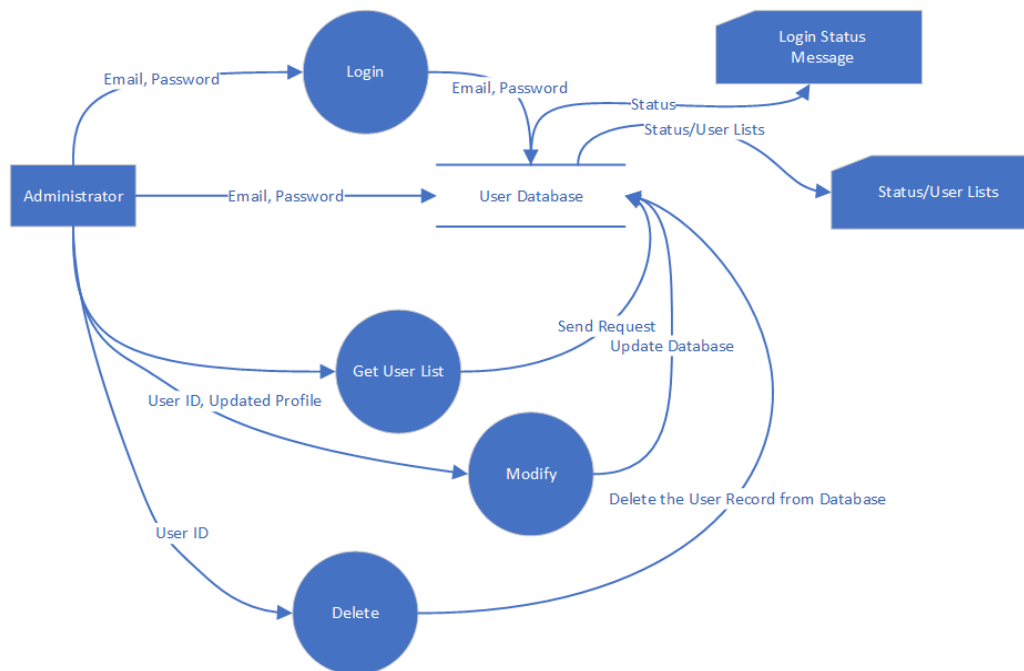


Figure 12: Data Flow Diagram: Data Flow Diagrams for Administrator

As shown in the **DFD**, our administrator can get the user's list and adjust their profile and password. Administrator are also authorized to delete user's accounts. Note that administrator account can only be registered through Django's superuser or MySQL.

4 UI Design

In this section, we present the user interface of our project. This should guide any user to use our platform.

4.1 Log in

The start page is **shown below** for existing user login, there are also two options below. Click "Don't have an account" to register page to create a new account. Click "reset password" to reset password through e-mail.

4.1.1 Exist User Login

Input valid user email, password and verification code, then click "Login" (see **screenshot**).

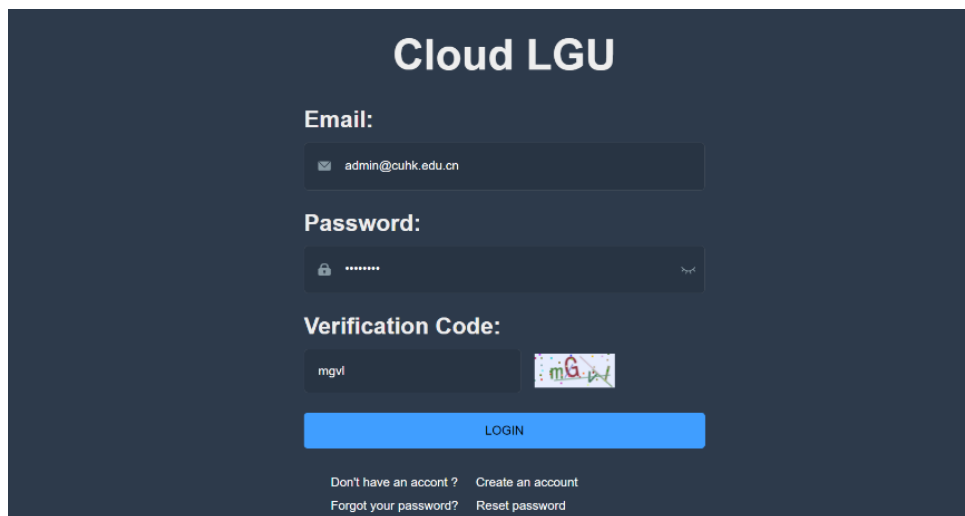
The image shows the login interface for Cloud LGU. It has a dark blue background with white text. At the top, it says "Cloud LGU" in a large, bold font. Below that, there are three input fields: "Email:" with the value "admin@cuhk.edu.cn", "Password:" with masked characters "*****", and "Verification Code:" with the value "mgvl". To the right of the verification code field is a small image of a verification code. Below the input fields is a blue "LOGIN" button. At the bottom, there are two links: "Don't have an account? Create an account" and "Forgot your password? Reset password".

Figure 13: Log in

4.1.2 Reset Password

Input your e-mail and click "get verification". then, your email will receive an email with code. Input the code and click "next step" to enter new password (see [screenshot on the left](#)).

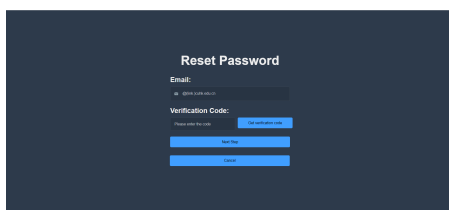
The image shows the "Reset Password" form. It has a dark blue background with white text. The title "Reset Password" is at the top. Below it, there are three input fields: "Email:" with the value "admin@cuhk.edu.cn", "Verification Code:" with the value "mgvl", and a "Next Step" button. Below the input fields is a blue "Next Step" button.

Figure 14: Reset password

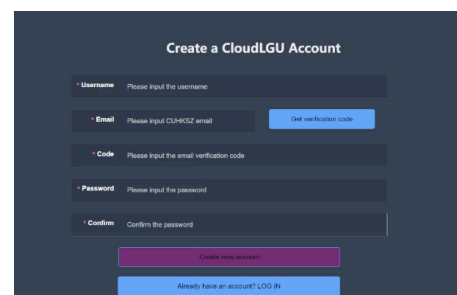
The image shows the "Create a CloudLGU Account" form. It has a dark blue background with white text. The title "Create a CloudLGU Account" is at the top. Below it, there are five input fields: "Username" (Please input the username), "Email" (Please input CUHK email) with a "Get verification code" button, "Code" (Please input the email verification code), "Password" (Please input the password), and "Confirm" (Confirm the password). Below the input fields are two buttons: "Create new account" and "Already have an account? LOG IN".

Figure 15: Register new user

4.1.3 Register New User

Input username you want and your email, then click "get verification code" to receive a code from email. enter the code and your password twice to confirm them. Finally, click "create" to create your account. If you do not want to create, click "LOG IN" to login page (see [screenshot on the right](#)).

4.2 Forum

This part works for all users including students and professors, to see all posts with special tags, especially TOP tag posting information about campus epidemic prevention. Besides, you can post by yourself and comment posts.

4.2.1 Show All Posts

The only page that can be directly arrived from the navigation bar in forum part. The head of this page have five buttons, including All, Discussion, Recruit, Help and Post Now (see [the screenshot](#)).

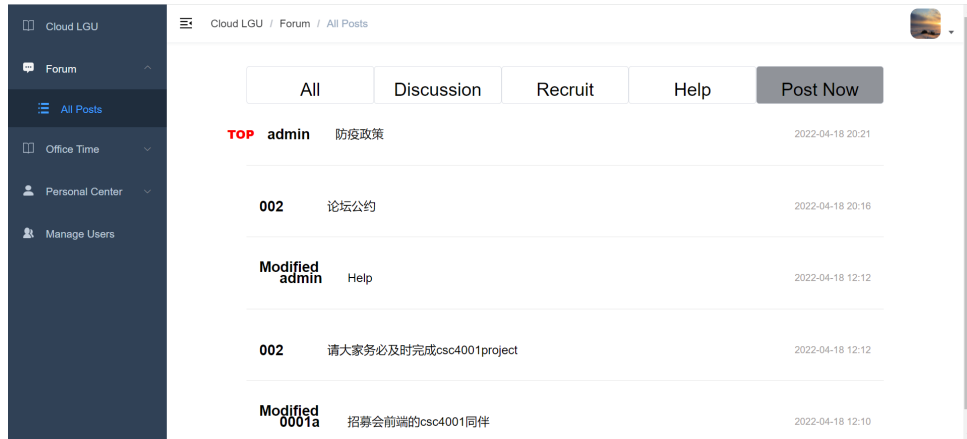


Figure 16: All posts

Click first four buttons can change the post list below. "All" shows all posts with a TOP at the top of list. Click "Discussion", "Recruit" and "Help" to show posts with these special tags, meanwhile, the TOP post is still on the top.

Click "Post Now" to jump [add post page](#).

4.2.2 Add Post

Input title, pick a tag and input content, then click "Post" button.

If you do not fill all these three part, you cannot submit successfully, the page will ask you to fill them

The screenshot shows the 'New Post' form. It has three main input areas: a title field with the text 'Test', a tag dropdown menu currently showing 'Discussion', and a large content text area with the text 'Check!'. At the bottom left, there is a red error message that says 'please enter content'. To the right of the error message are two buttons: 'Post' (in blue) and 'cancel' (in white).

Figure 17: Add post

4.2.3 Show Post

This page shows a detail page for a post article. You can see the content of the post and all the comment after that. Users can do comment on it and delete your own comment.

You can click the button "back" on the left top to go back to the posts list.

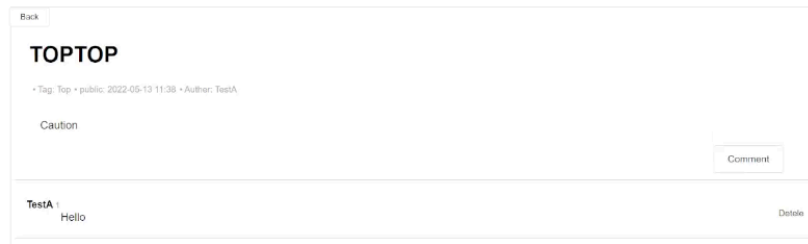


Figure 18: Show post, delete comment

Comment

To comment the post, click the "Comment" button on the right bottom of the article. Then there will be a **input box**. After type in the content, click "comment" below to submit your comment. And click "cancel" to cancel comment, then the input box will disappear.

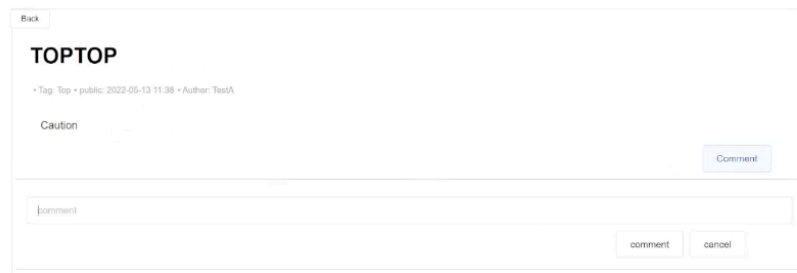


Figure 19: comment

Delete Comment

This can only work if you are the commenter of this comment. If so, there will be a "delete" button on the right of comment, click it to delete.

4.3 Office Time

For students, you can search available OTs this week with all professors, or search one professor. And you can book OT and check them in your reservations.

For professors, you can check you OT information (when, and booked or not), and you can also create

new OT slot for your free time and delete them.

4.3.1 Search Week

The first page in the OT reservation part, which is used for student. It shows all available professors in this week, as a form of week calendar. Example figure below shows that the professor named "admin" has OT on Tuesday and OT on Thursday. To be mentioned, this will not show the number of OT for each professor in each day. If the professor "admin" has three OTs on this Thursday, the page has the same display.

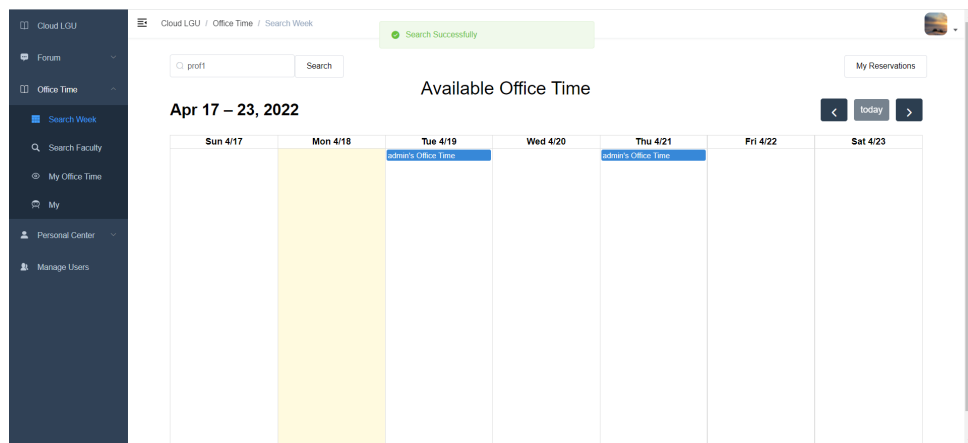


Figure 20: Student OT search week

There are three buttons on the top right of calendar. Click the left arrow button to see available OT in last week and right arrow button for the information in the next week. and click the button "today" to back to this week.

To get detail information for available professor and book his/her OT, click the blue box with his/her name. Then, it will go to the booking page for each professor, which has the same result by searching professor's name directly.

There is a search box on the left top of this page, by entering the whole name of professor you want to book, and clicking the button "Search" to the booking page for the professor.

There is a "My Reservation" button on the right top, click to jump to "My" page.

4.3.2 Search Prof

This page is used for student to search detailed OT for each professor and book OT.

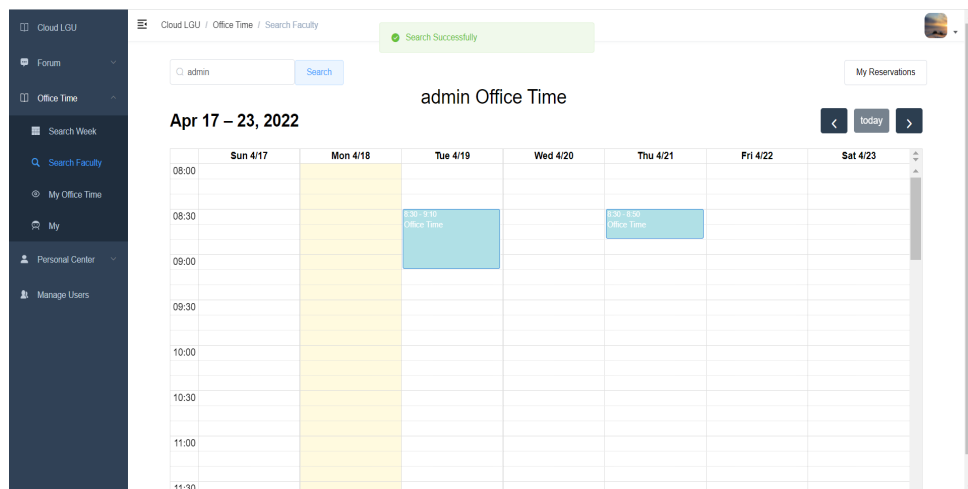


Figure 21: Student OT search week

Search

Enter the whole name of the professor you want, click "Search". If your input is valid and the professor has OT, the calendar below will show available OT for this professor. If there is no available OT or this professor does not exist, there will be an alert.

Book OT

To book the available OT, click the event (blue box). There will be an alert to ask to confirm. Once you click Yes, the OT is booked by you and it is no longer available for others, then you can check your reservation is "My" page.

4.3.3 My

This page shows the reserved Office Time for student user.

Each box is an OT booked by you, which shows the start time, end time and information of professor. Move the mouse on the event to show detailed information like location. There are three buttons on the top right of calendar. Click the left arrow button to see your reserved OT in last week and right arrow button for the information in the next week. and click the button "today" to back to this week.

There is a search box on the left top of this page, by entering the whole name of professor you want to book, and clicking the button "Search" to the booking page for the professor.

4.3.4 My Office Time

This page is used for professor to check your weekly OT, add a new available OT and delete an OT. Every deep blue box means an OT slot booked, light blue box means an available OT slot.

There are three buttons on the top right of calendar. Click the left arrow button to see OTs in last week

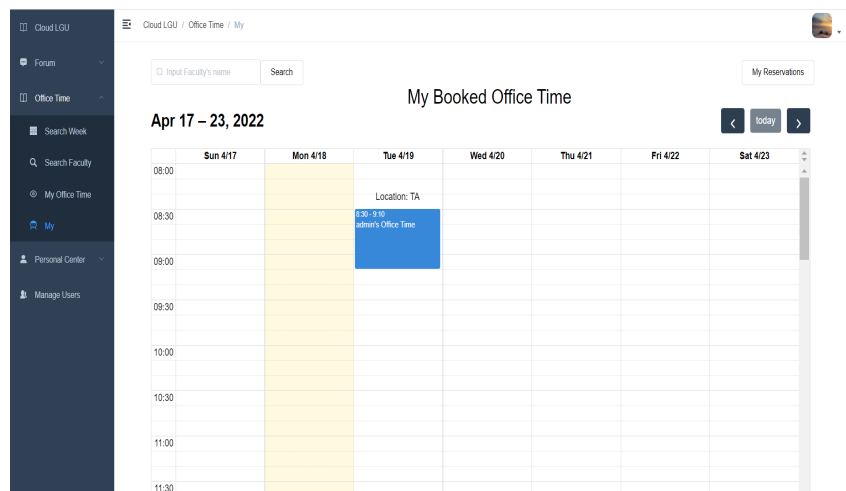


Figure 22: Student OT search week

and right arrow button for the information in the next week. and click the button "today" to back to this week.

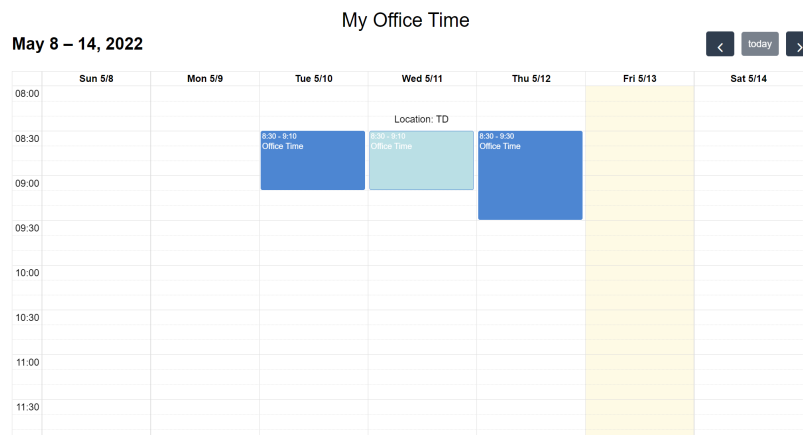


Figure 23: professor check OT

Create OT

A professor can click any blank time slot in the calendar to create new Office Time directly. Then, there will be an alert to ask for time duration as a number of minutes. If input minutes is valid, then you need to input the location. For example, if you click the blank slot start at 8:30 and input 30 as duration, then you create a new available OT from 8:30 to 9:00 at this day.

Delete OT

Professor can delete his/her OT whether it is already booked or not. Just click an OT slot in the calendar and confirm in the alert, you can delete it.

4.4 Personal Center

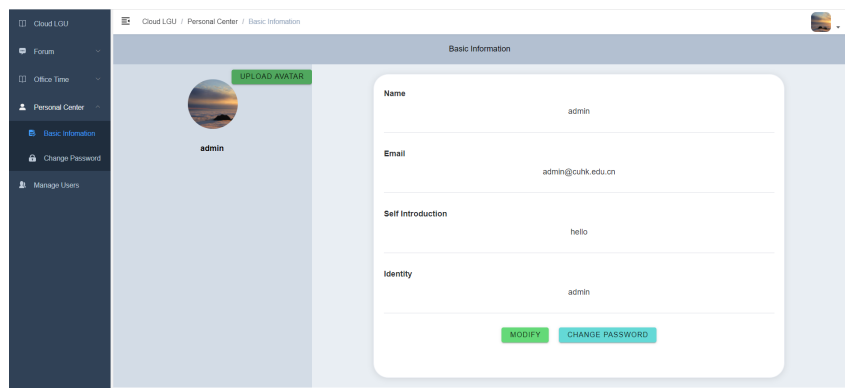


Figure 24: basic information

4.4.1 Basic Information

You can see your **basic information** including avatar, name, email, introduction and identity.

You can also **change your avatar** by clicking "upload avatar" and pick one from your computer. You can also **change your username and introduction** by clicking "modify" and "submit" to save.

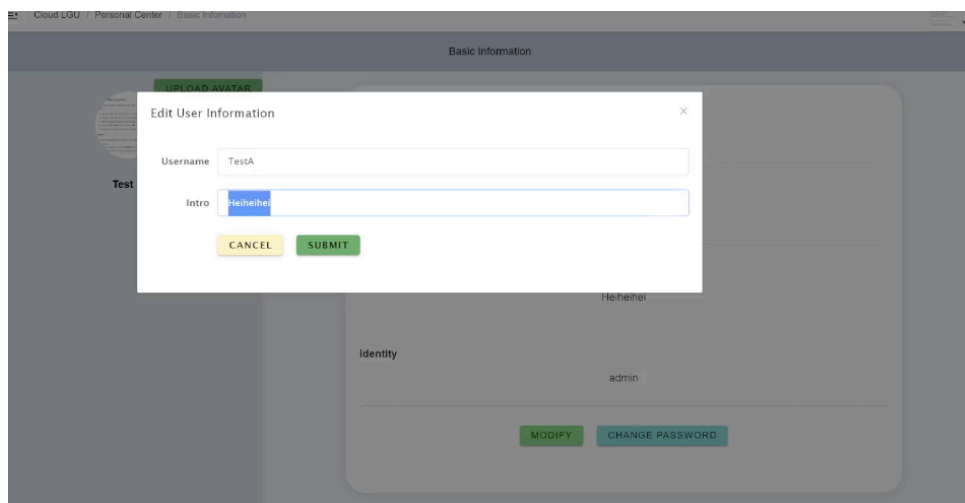


Figure 25: modify information

4.4.2 Change Password

You can change your password **in this page** if you can remember your password. Input your old password and input your new password twice, click "Confirm" to save. You can also try to use your email rather than the old password to prove your identity.

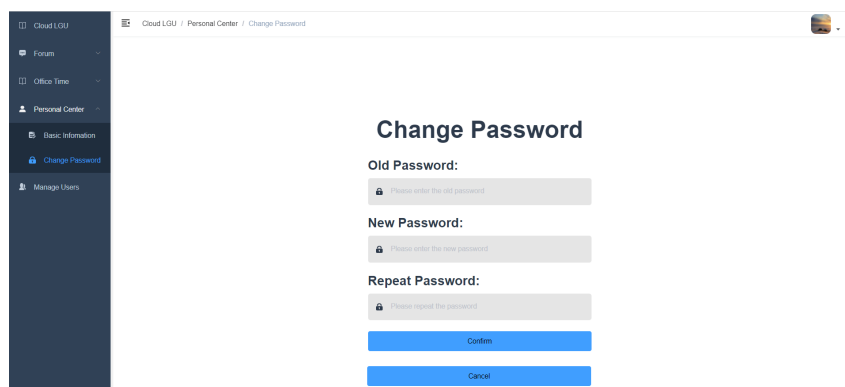
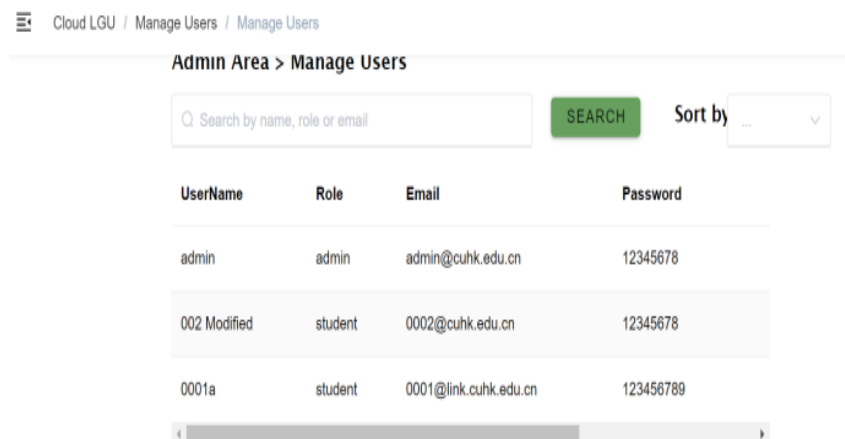


Figure 26: change password

4.5 Manager User

This part can only be seen and check by admin, to see **all information of all users**. The admin can search the user by name, email or role. The admin can also sort the list.



UserName	Role	Email	Password
admin	admin	admin@cuhk.edu.cn	12345678
002 Modified	student	0002@cuhk.edu.cn	12345678
0001a	student	0001@link.cuhk.edu.cn	123456789

Figure 27: see information

4.5.1 Delete User

Click the setting button at the right end of each user and click Delete to delete the user.

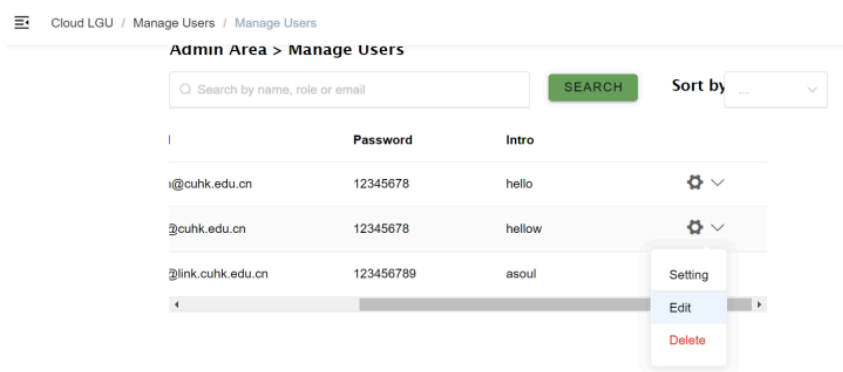


Figure 28: delete user

4.5.2 Edit User

Click the setting button at the right end of each user and click Edit to edit all information of the user except their email. click "submit" to confirm.

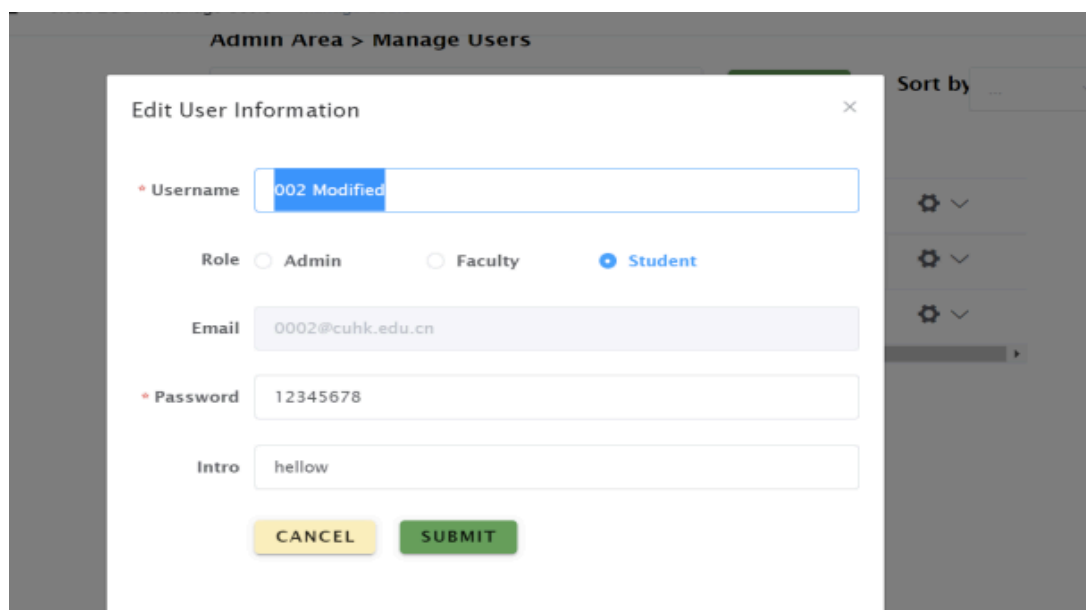


Figure 29: edit information

5 TEST

5.1 Test Overview & Test Plan

Timeline	Stage	Tasks
3.24-4.15	Develop Modules (Before the Integration)	Black Box API Test with Postman for Each API Front End API Test with MOCK Stub Code Review Between Group Members User-Friendly Check Keep Scanning the Code to Ensure Readability and Security
3.24-4.15	API Document Design (Before the Integration)	Check the Rationality of Each Design of API
4.16-4.18	After the Integration of Front-End and Back-End	Black Box Functional Test Compatibility Test Performance Test with PageSpeed by Google

Our test plan consists of mainly three parts. During our development stage, we will conduct regular code style check with SonarQube. For the front-end, we will keep tracking whether the UI is user-friendly, and whether all of the buttons, routers are set correctly. Also, we use Mock Stub to test our front-end's functions.

As mentioned before, we maintain an API document recording the address of the API, the inputs and outputs of that API. We keep updating those interfaces so that all of the interfaces can meet our requirements. In order to test the functionality of those APIs, we use Postman to test whether the output in back-end is in accordance with our API document and use Mock.js to test front-end. We will show some test samples below.

5.2 Case 1:Postman Test

We use postman to test the Back-end functions before connecting with the front end UI. And here is part of our test result. However, because our API is different from the API of postman, this kind of test does not fit to our project now, the code of input part need to be changed if we still want to test in this way.

5.2.1 Create Post

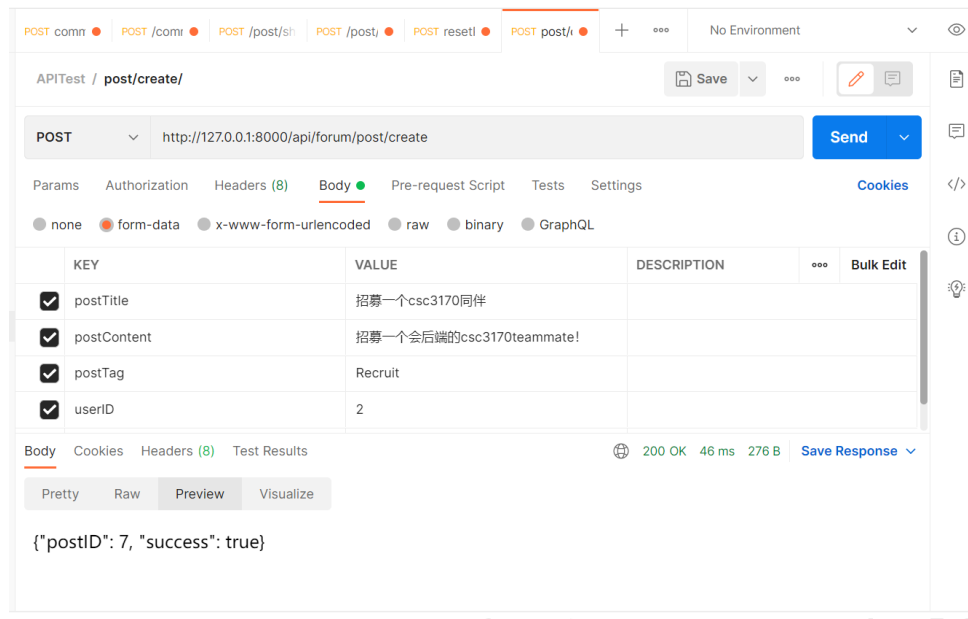


Figure 30: Test case for function 'Create_new_post'

Purpose

Function 'Create_new_post' can let the users to add new posts to the forum.

Inputs

1. Title of the new post
2. Content of the new post
3. Tag of the post
4. poster's ID

Expected Outputs and Pass/Fail Criteria

If the new post is created successfully, back end will return id of the new post and a 'successful:true'. If the new post does not be created, this function will return 'successful:false'.

5.2.2 Show all Post

Purpose

Function 'Show_all_post' can let the users view all posts from the forum.

Inputs

No input needed.

Expected Outputs and Pass/Fail Criteria

If the function get all posts successfully, back end will return a list contains all posts and 'successful:true'. Otherwise this function will return 'successful:false'.

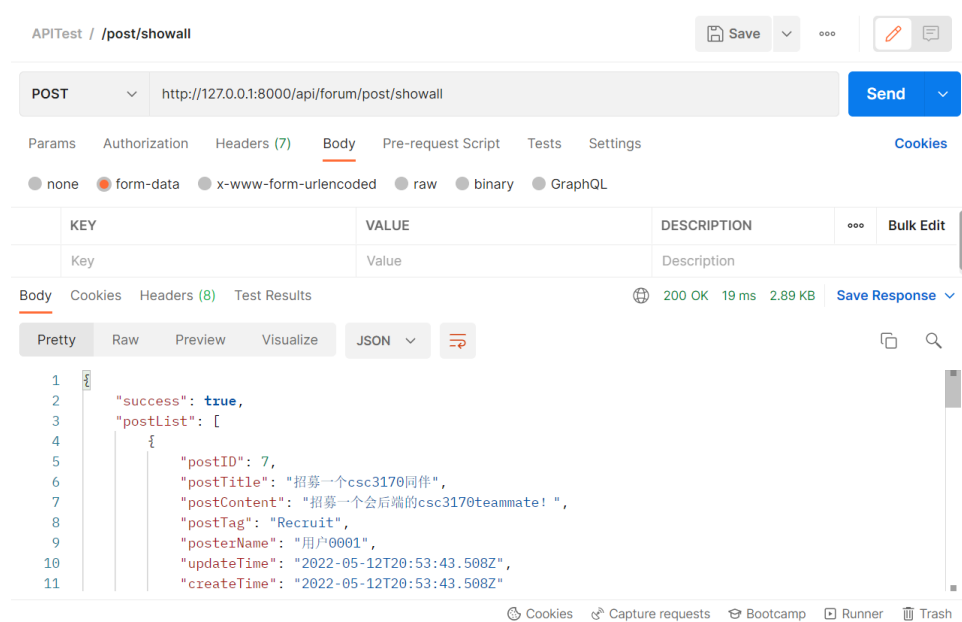


Figure 31: Test case for function 'Show_all_post'

5.3 Case 2: Mock.js Test

Before the integration of front-end and back-end, we use Mock.js to establish **mock database on the left** and API for front-end functionality test. We show the test cases of user operations here.

5.3.1 Log in

Purpose Test user login function

Inputs User email and password. We will test at least four cases: admin account, student account, professor account and invalid account.

Expected Outputs and Pass/Fail Criteria Whether the use email and password are correct or not. If the email and password match, the user can log in to the website. Otherwise, an error code would be returned and an error message would be generated. The criteria is whether can log in or not and if logged in, whether the home page would be displayed and corresponding router would be generated. See **the code on the right**.

5.3.2 Get User Profile

Purpose This API intends to retrieve the basic information of one user.

Inputs User's unique token stored in Cookies. This is API is called automatically by the program.

Expected Output and Pass/Fail Criteria Expected output is the profile of the user, including the email, name, identity, introduction, avatar url, and return code. Then the data would be stored in Cookies and shown on the page. Criteria is simply whether or not the front-end gets the data and display them suc-

```
const tokens = {
  admin: {
    token: 'admin-token'
  },
  student: {
    token: 'student-token'
  },
  faculty: {
    token: 'faculty-token'
  }
}

const users = {
  'admin-token': {
    roles: ['admin'],
    introduction: 'I am a super administrator',
    avatar: 'https://api.wallstcn.com/f778738c-e4f8-4870-b634-56703b4acafe.gif',
    name: 'Super Admin',
    email: 'admin@cuhk.edu.cn'
  },
  'student-token': {
    roles: ['student'],
    introduction: 'I am an student',
    avatar: 'https://api.wallstcn.com/f778738c-e4f8-4870-b634-56703b4acafe.gif',
    name: 'Normal student',
    email: 'student@cuhk.edu.cn'
  },
  'faculty-token': {
    roles: ['faculty'],
    introduction: 'I am a professor',
    avatar: 'https://api.wallstcn.com/f778738c-e4f8-4870-b634-56703b4acafe.gif',
    name: 'Normal professor',
    email: 'faculty@cuhk.edu.cn'
  }
}
```

Figure 32: Mock Database

```
module.exports = {
  // user login
  {
    url: '/vcloudlgU/user/login',
    type: 'post',
    response: config => {
      const { useremail } = config.body
      const token = tokens[useremail]

      // mock error
      if (!token) {
        return {
          code: 60204,
          message: 'Account and password are incorrect.'
        }
      }

      return {
        code: 20000,
        data: token
      }
    }
  }
},
```

Figure 33: Mock User Log in API

cessfully. See [the code on the left](#).

5.3.3 Log Out

Purpose This API lets user to log out of the website.

Inputs None currently. Directly log out. Local Cookies would be dispatched.

Expected Output and Pass/Fail Criteria User should be logged out of the website and can not visit any page unless re-log in. See [the code on the right](#).

```
// get user info
{
  url: '/vcloudlgU/user/info\.*',
  type: 'get',
  response: config => {
    const { token } = config.query
    const info = users[token]

    // mock error
    if (!info) {
      return {
        code: 50008,
        message: 'Login failed, unable to get user details.'
      }
    }

    return {
      code: 20000,
      data: info
    }
  }
},
```

Figure 34: Mock Get User Profile API

```
// user logout
{
  url: '/vcloudlgU/user/logout',
  type: 'post',
  response: _ => {
    return {
      code: 20000,
      data: 'success'
    }
  }
}

// user logout
logout({ commit }) {
  return new Promise(resolve => {
    removeToken() // must remove token first
    resetRouter()
    commit('RESET_STATE')
    resolve()
  })
},
```

Figure 35: Mock User Log Out API and Local Function

5.4 SonarQube Scanning for Code Style

SonarQube is a static code analyzer to measure the reliability, security and maintainability of all the languages in the project. And we will use this tool to assure our code style. For python code we take PEP8 and PEP257 style. For JavaScript code, we take Airbnb style. Then after the analysis, we can get the following result:

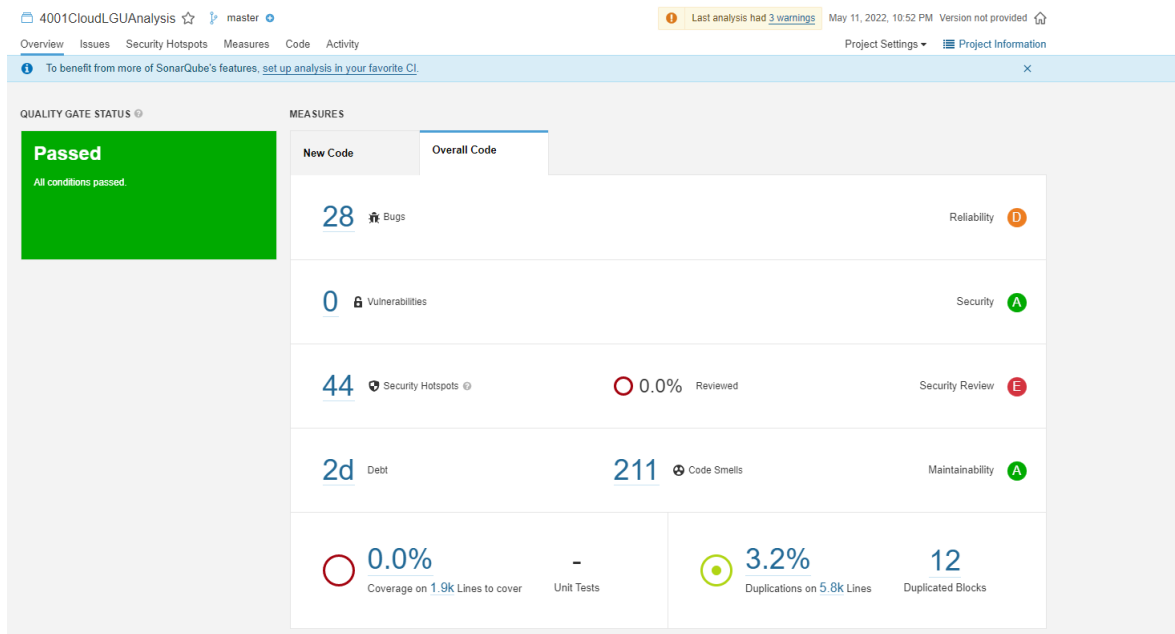


Figure 36: SonarQube Scanning Result

And for every smells (about coding style for readability), bugs, and vulnerability (for security), we can get the problems and suggestions to fix them.

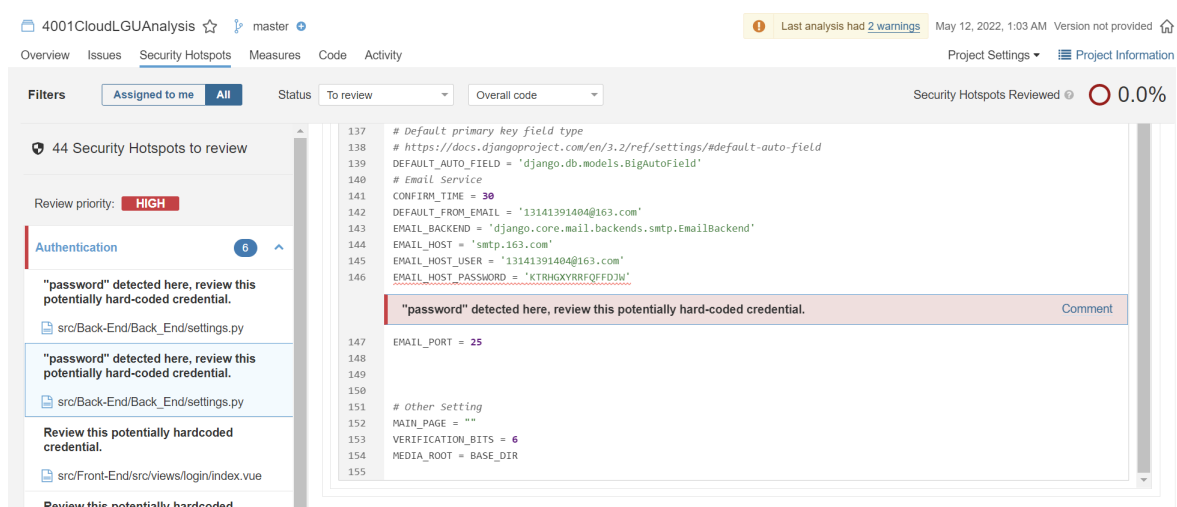


Figure 37: SonarQube Example for Problem

Recommended Secure Coding Practices

- Store the credentials in a configuration file that is not pushed to the code repository.
- Store the credentials in a database.
- Use your cloud provider's service for managing secrets.
- If a password has been disclosed through the source code: change it.

Compliant Solution

```
import os

username = os.getenv("username") # Compliant
password = os.getenv("password") # Compliant
usernamePassword = 'user=%s&password=%s' % (username, password) # Compliant{code}
```

Figure 38: SonarQube Example for Suggestions

6 Lessons Learned

Xia Hanyang 118010339: Most of the programming work I faced before this class was developing a single feature by myself rather than in a group project. Therefore, my ability to cooperate with teammates through documents has been dramatically improved. I think the compositions of our group are pretty complete and meticulous, and all members can develop according to the documents and put forward suggestions for modification in time. Of course, there is no denying that there is still room for improvement in our project management. For example, we did not timely communicate with each other on some details, which wasted some energy on repetitive work.

Wang Mingjie 119010300: Though I've done multiple group projects and had an internship of Vue.js development, this is the first time I try to develop a whole website with my teammates from nothing. I realize how important it is for teammates to communicate and cooperate with each other. I can't imagine how can I manage to finish this whole project by my own. As for building a web platform, I find out the importance of keeping coherency between front-end and back-end design, especially for the API. To achieve coordination, the developer indeed need communicating with each other. I also find out that a good design document can really save a lot of time and unnecessary efforts.

Gao Ziqi 118010077: Working as a group leader is another kind of experience. Besides the assigned work, I have to organize the time schedules, and organize regular meetings with my team members to assure the speed and quality of our development. Also, a leader need to assign the work clearly so that every member can have work to do and learn something from the project. Milestone is a very useful tool. Those small goals can make sure our project are in the right direction and development pace. One pity for this project is that we do not spend much time on software testing although all of us know the importance of testing. If I have another chance to be the leader, I will emphasize the testing, also try my best to facilitate the in-group communication, and do less about the trivial tasks but spend more time on more important tasks like design, teamwork schedule and work assignments.

Zhang Yizhan 119010450: I learned a lot to group up and complete a website project with all of the codes created by ourselves. I had got a big map of how a software is and the structure and process of implementation and testing. Besides, reparing design document before implementation is one of the most important task in a group project, especially for the course of software systems. It is always complex to orginze and connect every componoents together. Only based on the design document like api design, can we face less problems when mergeing the code and execute the whole program later. I really appreciate all the communication and cooperation done with my group mates.

7 Conclusion

The project developed a platform to help professors interact with students more efficiently, espially under such a hard time for everybody. The platform has two main functions: one is to help professors and students agree on office hours; Another is to realize the campus personal interaction forum. To be mentioned, we also update the campus epidemic prevention information in the TOP of our forum to make some effects. On the front end, the project provides a practical, convenient, beautiful dynamic web page; the developer builds an efficient database on the back end. The project management of this project is generally successful. Code and all kinds of documents in <https://github.com/118010077/SE-CloudLGU> can see. Generally speaking, the software developers have made a decent effort and achieved admirable accomplishments, which deserve a decent score.