



香港中文大學 (深圳)

The Chinese University of Hong Kong, Shenzhen

SCHOOL OF DATA SCIENCE

CSC4001: Software Engineering

Course Project Report
A Web-based Application: PickUp

Author:

Chen Boyi (Young) 119010010

Zhu Luokai (Luca) 120090460

Zeng Zhuoru (Philip) 119010417

May, 2022

1 Introduction

1.1 Project Overview

It is common that we find it hard to team up with someone who has the same needs or interests as we do. For example, we may want to have a carpooling with others to the airport or train station in order to save money, or sometimes we want to do some sports and want to find more people to join the game. The need for teaming up is common and obvious in our daily life, but at most of the time, this would be troublesome. Our project "PickUp" aims at providing an online platform to help people within a community (e.g. CUHKSZ students) to team up with their fellows to do activities together easily. Users can post their proposal for an activity or their needs to team up to our website (These proposals would be refer to as "PickUp" proposals in the following part of this report). Others who have the same interest or need can contact the post publisher easily within a few minutes and make their appointment.

1.2 Objective

Motivation & Goal

The "PickUp" project is designed to help community members to easily find fellows who have the same need or interest and make their appointment. The system will have two parts, the frontend and the backend. The frontend will enable users to browse and publish posts, get contact to other users, and manage the PickUp proposal they initiated. The backend will provide supports to the management of user data, such as user accounts, posts, and other customer service.

The whole project can be roughly divided into three main components, which are User, Activity and Admin component. Each components can be further divided into several sub-components.

Target User

As mentioned above, The target users will be people within a specific community or geographical region. The reason is that it's easier and more convenient for people in the same community to make appointment. The main user of this application are expected to be students and faculties within CUHKSZ community for this initial state of this project.

What's more, we also provide prime service for users who wants to unlimited number of posting and joining "PickUp" proposals compared to free users who is allowed to post and join limited number of "PickUp" proposals.

1.3 Highlights

- Cloud Deployment

The system has been deployed on Amazon Web Service. The front-end and back-end are deployed on AWS Elastic Compute Cloud, database is deployed on AWS Relational Database Service. You can access our deployed website on <http://54.172.232.138/>

- Project Specific Functions Fulfilling Daily Life Needs

The PickUp app is intended to help solve the real life problems we face, and the appointment making & management system can indeed help users solve their needs in life.

- Pretty User Interface and Color Selection

We have carefully implemented our website to make sure users can easily use the app and the UI components are concise and beautifully arranged.

- NOTICE: Two demo accounts are provided here and you can use them to explore our website on <http://54.172.232.138/>:

- User account

- Email: philip@google.com
 - Password: 123456

- Admin account

- Email: admin1@gmail.com
 - Password: 123456

1.4 Team Introduction:

Our team consists of three students who took CSC4001 this semester.

The rule we assigned the workload is based on the division of different components:

- Zhu Luokai (Luca)
 - User Account & User Info Management
 - Sign Up & Email Verification
 - Log In
 - Setting & Updating Personal Information
 - UI Design
- Zeng Zhuoru (Philip)
 - Activity Management
 - User Activity List
 - Admin Interface
 - Manage User Account (List & Reset Password)
 - Manage Admin Account
 - Database Design & Implementation
 - Cloud Deployment
- Chen Boyi (Young)
 - Activity Management
 - Display All Activities
 - Create New Activity
 - Join Activity
 - Database Design & Implementation

1.5 System Features

The "PickUp" application will have five main features :

- User Account Registration and Login
 - Users can sign up and login to their account via email address
 - Email verification is implemented during sign up
- Account Information Management
 - In the personal homepage, users can manage their account information like avatars, passwords and other personal information
- "PickUp" Proposal Searching and Classifying
 - In the front page, user can search for the "PickUp" proposals they are interested in
- For easy searching all "PickUp" proposals would be classified into specific type using tags
- "PickUp" Proposal Management
 - Users can publish new "PickUp" proposals, manage team members
- Users can participate in other users "PickUp" proposal by registering for that activity
- Admin Management Interface
 - Admin account have access to all user accounts and have authority to modify these accounts, like changing their password or even deleting specific accounts.
 - Admin account can create new admin account and manage existing admin accounts

1.6 Project Statistics

The following is our module specification.

| Module ID | Module name | Frontend Function | Backend Function | Description |
|-----------|------------------|---|---|---|
| 1 | Display Activity | display the available activity information | fetch information and send to front end | fetch activity information from database and display to users |
| 2 | Search Activity | get input about search from user | fetch activity information of given requirement | allow user to search for activities they are interested in |
| 3 | Sort Activity | get input about sort from user | fetch activity information and sort them in given order | allow user to sort the activities in the way they like |
| 4 | Join Activity | check if user has already participated in the activity, and send user id and activity id to backend for further manipulation if user not in activity. | manipulate the user activity relationship table | allow user to join certain activities |
| 5 | Post Activity | gather information of activity from user [227] | receive information from front-end and insert into database | allow user to post their own activities |
| 6 | List Activity | display all the activity one specific user joined | fetch activity information given user_id | fetch activity information of a specific user and display to user |
| 7 | Manage Activity | update the member and status of a specific activity, include delete user, terminate activity, change activity manager | update activity status or member data | update activity status or member data to database |

| Module ID | Module name | Frontend Function | Backend Function | Description |
|-----------|--------------------------|--|---|---|
| 8 | List User Account | display all the user account information | fetch user account information | fetch user account information from database and display on website |
| 9 | List Admin Account | display all the admin account information | fetch admin account information | fetch admin account information from database and display on website |
| 10 | Manage User Account | update user account information | update user password or delete user account | update user password or delete user account in database |
| 11 | Manage Admin Account | update admin account information | update admin account password or delete admin account | update admin password or delete admin account in database |
| 12 | Create New Admin Account | create new admin account | insert new admin account data to admin_account table | fetch admin account and password info from input box and insert into database |
| 13 | Landing Page | display introduction content to our website | / | display introduction content to our website |
| 14 | Display User Information | display user's current information | fetch user information and send to front end | fetch user information by user id from database and display to users |
| 15 | Set User Information | get the new user information about password or user name from user | fetch old password and send to front end | allow user to modify their account and information |
| 16 | Sign-Up | get the new email and password | call the send email function to send a verification email then verify | allow the user to create his own account |
| 17 | Sign-In | get the email and its corresponding password from user's input | check the correctness of the email and the password is paired | allow user to log in to our website |
| 18 | Email Verification | / | send user a email with a register link | send a email to user |
| 19 | Navigation Bar | navigation bar | / | navigation bar |
| 20 | Router | routing/redirecting between different pages | / | navigate between different web pages |
| 21 | Vuex Store | data communication between vue components | / | data communication between vue components |

- Activity display, search, sort and join functionalities
 - Front-end
 - `/client/src/views/UserHomeView.vue` : 632 lines of code
 - Back-end
 - `/client/src/views/homeServer.js` : 239 lines of codes
- Activity post functionality

- Front-end
 - `/client/src/views/ActivityCreationView.vue` : 444 lines of code.
 - Back-end
 - `/client/src/views/homeServer.js` : 239 lines of codes
- Activity list module's implementation is in
 - Front-end
 - `client/src/views/ActivityListView.vue` : 84 lines of code
 - `client/src/components/ActivityList/ActivityList.vue` : 26 lines of code
 - `client/src/components/ActivityList/ActivityListTitle.vue` : 26 lines of code
 - `client/src/components/ActivityList/ActivityListCard.vue` : 59 lines of code
 - Back-end
 - `server/activityServer.js` : 332 lines of code
- Activity Management module's implementation is in
 - Front-end
 - `client/src/views/ActivityManagementView.vue` : 156 lines of code
 - `client/src/components/ActivityManagement/MemberItemForManager.vue` : 76 lines of code
 - `client/src/components/ActivityManagement/ActivityManagementConsole.vue` : 205 lines of code
 - `client/src/components/ActivityManagement/MemberItemForMember.vue` : 76 lines of code
 - `client/src/components/ActivityManagement/MemberList.vue` : 49 lines of code
 - Back-end
 - `server/activityServer.js` : 332 lines of code
- User account list module's implementation is in
 - Front-end
 - `client/src/views/AdminConsoleView.vue` : 50 lines of code
 - `client/src/components/AdminConsole/UserList.vue` : 36 lines of code
 - `client/src/components/AdminConsole/UserListCard.vue` : 50 lines of code
 - `client/src/components/AdminConsole/UserListHeader.vue` : 27 lines of code
 - Back-end
 - `server/adminServer.js` : 273 lines of code
- Admin account list's implementation is in
 - Front-end
 - `client/src/views/AdminAccountListView.vue` : 70 lines of code
 - `client/src/components/AdminAccountList/AdminAccountList.vue` : 25 lines of code
 - `client/src/components/AdminAccountList/AdminAccountListCard.vue` : 45 lines of code
 - `client/src/components/AdminAccountList/AdminAccountListHead.vue` : 24 lines of code
 - Back-end
 - `server/adminServer.js` : 273 lines of code
- User account management module's implementation is in
 - Front-end
 - `client/src/views/UserManagementView.vue` : 16 lines of code
 - `client/src/components/UserManagement/UserManagementConsole.vue` : 202 lines of code
 - Back-end
 - `server/adminServer.js` : 273 lines of code
- Admin account management module's implementation is in
 - Front-end
 - `client/src/views/AdminAccountManagementView.vue` : 16 lines of code
 - `client/src/components/AdminAccountManagement/AdminAccountManagementConsole.vue` : 187 lines of code
 - Back-end
 - `server/adminServer.js` : 273 lines of code
- New admin account creation module's implementation is in
 - Front-end
 - `client/src/views/CreateAdminAccount.vue` : 79 lines of code

- Back-end
 - `server/adminServer.js` : 273 lines of code
- Display User Information module is implemented in
 - Front-end
 - `client/src/views/PersonalInfoView.vue` : 255 lines of code
 - Back-end
 - `server/profileAndSetting.js`
- Set User Information module is implemented in
 - Front-end
 - `client/src/views/Settingview.vue` : 297 lines of code
 - Back-end
 - `server/profileAndSetting.js`
- Sign Up module
 - Front-end
 - `client/src/views/Registerview.vue` : 98 lines of code
 - Back-end
 - `server/index.js`
- Sign In module
 - Front-end
 - `client/src/views/Loginview.vue` : 147 lines of code
 - Back-end
 - `server/index.js`
- Email Verification module
 - `server/index.js`
- Router module is implemented in
 - `client/src/router/index.js` : 92 lines of code
- Vuex store is implemented in
 - `client/src/store/index.js` : 103 lines of code
 - `client/src/store/UserManagement/index.js` : 24 lines of code
 - `client/src/store/AdminAccountManagement/index.js` : 24 lines of code
 - `client/src/store/ActivityManagement/index.js` : 24 lines of code
- Landing page module is implemented in
 - `client/src/views/UserHomeview.vue` : 557 lines of code
- Navigation bar module is implemented in
 - `client/src/App.vue` : 177 lines of code

2 SYSTEM ARCHITECTURAL DESIGN by DFD

2.1 System Architecture

2.1.1 Architecture Overview

Structure by Functionality

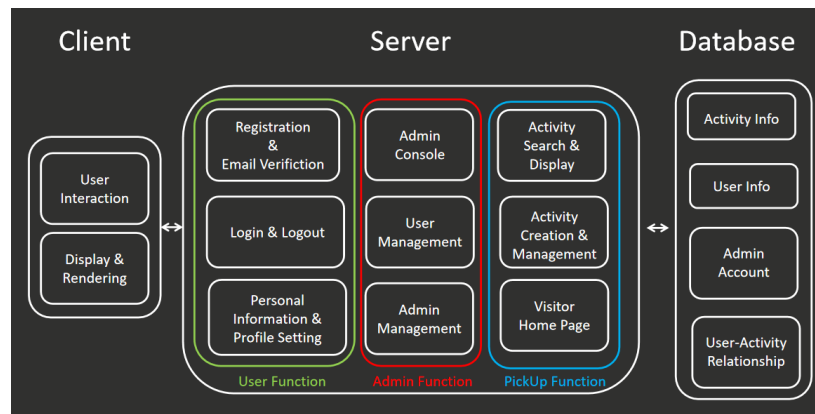


Figure 2.1.1-1

Figure 2.1.1-1 describes the overall architecture design of our system. In the frontend part, users can use their web browser to get access to our system. Our user interface provides users with basic functions like registration, email/message verification, modify privacy settings, etc. Additionally, users will be able to carry out functions related to "PickUp" proposals, like joining, posting and manipulating proposals in the frontend. The functions requiring some calculations or some sorting algorithms are hidden from users. For the server side, there will be 4 tables in the database storing the information of users, activities, user-activity relationship and admin account respectively.

Structure by Technology Stack

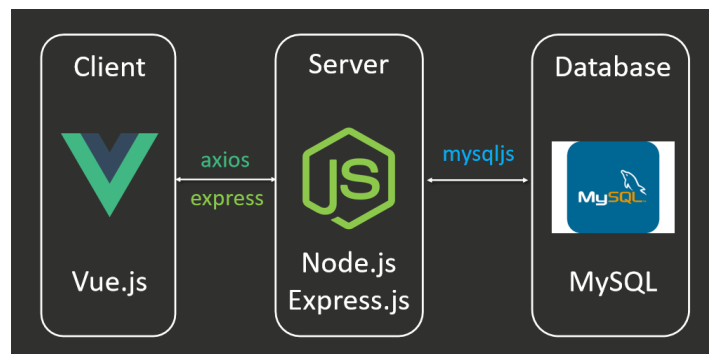


Figure 2.1.1-2

Figure 2.1.1-2 describes the technology stack of this project. Vue.js is picked as the front-end framework. Node.js with Express.js is chosen as the back-end run time environment and framework. MySQL is used as our database. The `axios` and `express` packages are used to conduct data communication between front-end and back-end. The `mysqljs` package is used to setup communication between back-end and database.

Structure by Implementation Specification

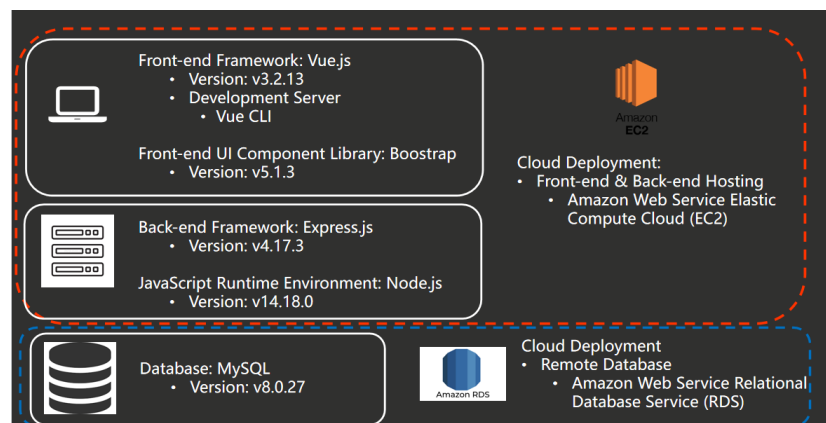


Figure 2.1.1-3

Figure 2.1.1-3 describes the implementation detail of this project. Our project is deployed on cloud. The front-end and back-end is hosted on a Amazon Linux Virtual Machine using Amazon Web Service Elastic Compute Cloud as shown in Figure 2.1.1-4.

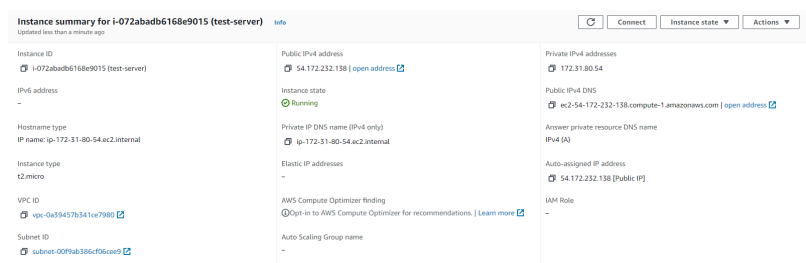


Figure 2.1.1-4

The remote MySQL database is hosted using Amazon Web Service Relational Database Service as shown in Figure 2.1.1-5.

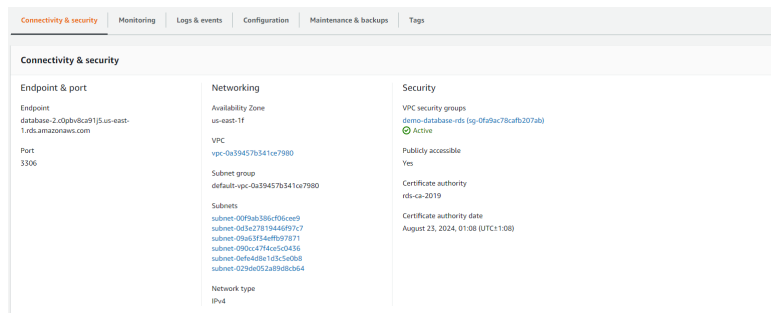


Figure 2.1.1-5

File Structure

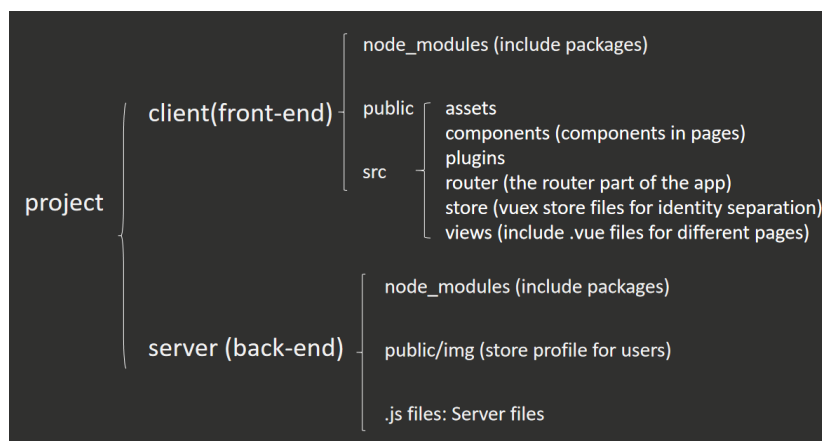


Figure 2.1.1-6

Figure 2.1.1-6 describes the file structure of the project. The `.vue` files in the `/client/src/views` and `/client/src/components` which defines the content and logic of each website page. The `.js` files in `/server` implements all the actions of the back-end, and is responsible for all the data communication between the front-end and database.

2.1.2 Database Design

MySQL is chosen to be our project's database. Four tables are implemented in our project: `activity_info`, `activity_user`, `admin_account`, `user_info`. The specification (Entity Relation Diagram) of these four tables are shown in Figure 2.1.2-1.

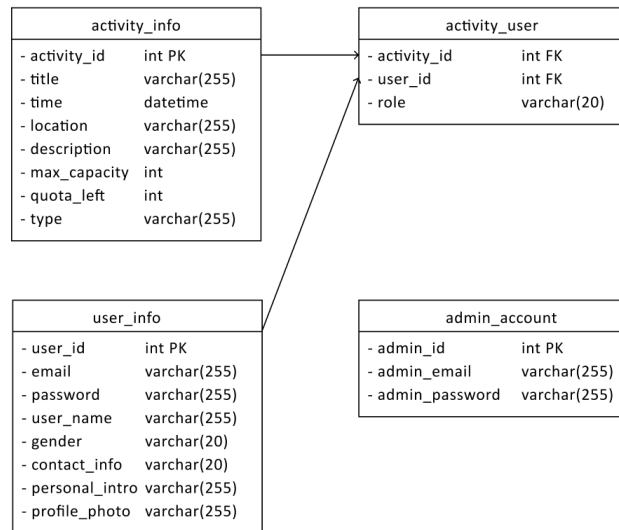


Figure 2.1.2-1

2.1.3 Key Components

In this section, we List all the key components in our project, and the detailed description is in section 3.

- User Account
 - Sign Up & Email Verification
 - Log In
 - Setting & Updating Personal Information
 - Upload Profile Picture
- Activity System
 - Display Proposal
 - Manage User Proposal
 - Search & Sort Proposal
 - Join & Post Proposal
- Admin Interface
 - Manage User Account (List & Reset Password)
 - Manage Admin Account

2.1.4 Key Interface Description

User Account

- Sign Up & Login In
 - This is login page for a user, if the user has never accessed to our website, he will be asked to sign up her account with his emails and his own password.
- Setting & Updating Personal Information
 - This is the setting page for user to modify his personal information and account details. The personal information includes users' usernames, contact information, gender and their own biography.
 - Besides, user can also modify their passwords. However, user must input their original password to confirm the authority instead of changing their password directly. And user has to input their new password twice to prevent the careless typo.
- Email Verification
 - Email Verification Page actually belongs to the sign-up page. Before user sign-up successfully, the server will check the existence and validation of user's account. After user click the email link sent by our server, he will redirect to our home page. The login process is achieved automatically.
- Upload Profile Picture

Activity System

- Display & Search & Post Proposal
 - This page is the one displayed to the user when user log in or view as a guest. It contains frequently used operations related to the “PickUp” proposal, like searching, sorting and joining. And user will also be able to post a new proposal by clicking the post button.
- Manage User Proposal
 - This page list the proposals the user participates in (the user can be the organizer or a member of the activity). User are allowed to get the detailed information about the activity and manage the member of the activity or terminate the activity if he/she is the organizer. Additionally, he/she can quit the activity as he/she likes.
- Join & Post Proposal
 - This page gathers activity information from users and send the information to backend servers for further manipulation.

Admin Interface

- Manage User Account (List & Reset Password)
 - The user account list page displays all the user account information, admin can go to modify user account information by further accessing the user account management page through clicking buttons.
 - The user account management page displays the user information of a specific user. Admin can change the user password or delete the account in this page.
- Manage Admin Account
 - The admin account list page displays all the admin account information. Admin can go further go to modify admin account data by accessing the admin account management page or create new admin accounts in the create new admin account page through clicking buttons.
 - The admin account management page enable admin users to change the password of other admin users or delete other admin user accounts.
 - The create new admin account page enable admin users to create new admin accounts with provided email and initial password.

2.2 DFDs of Key Components

2.2.1 User Account

- Log In & Sign Up & Email Verification

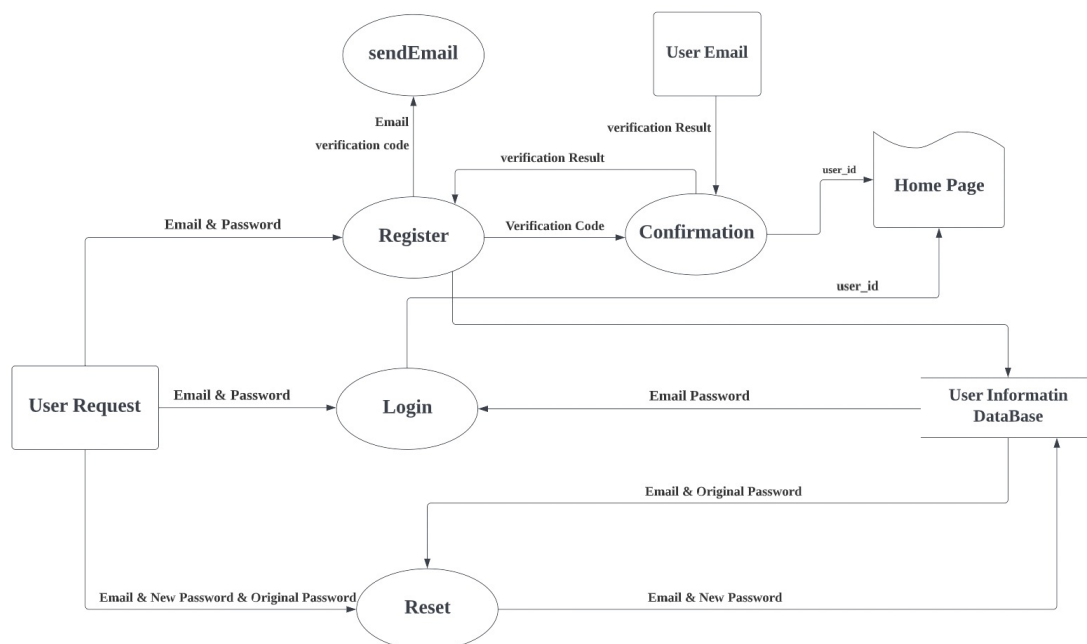


Figure 2.1-1 user_login_dfd

- Setting & Updating Personal Information

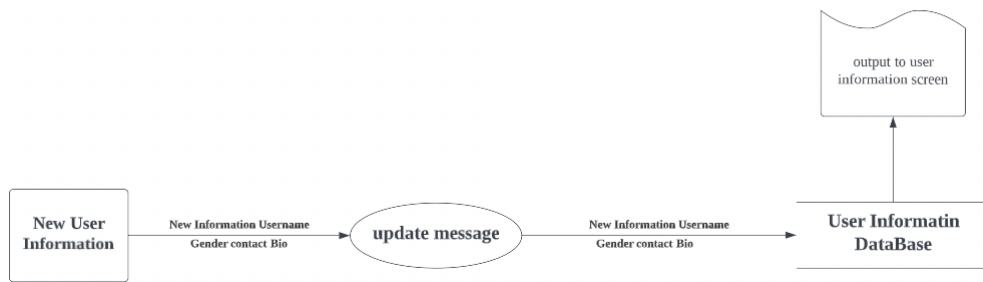


Figure 2.1-2 setting_report

2.2.2 Activity System

- Search & Sort Proposal
 - "PickUp" proposal searching:

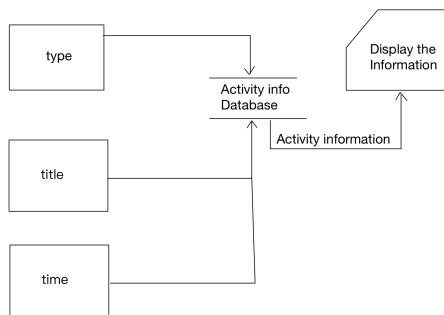


Figure 2.2-1

Figure 2.2-1 shows the data flow of "PickUp" proposal searching procedure, users can give the type, title or time of proposal as input and the function will search the activity_info table in database to fetch the information of the proposals the user is interested in and send it back to the front-end to be display to the user.

- "PickUp" proposal sorting:

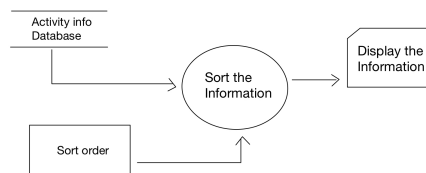


Figure 2.2-2

Figure 2.2-2 shows the data flow of "PickUp" proposal sorting procedure, users can give the kind of order they want the proposals to be sorted, and the function in the backend will receive the order and fetch the proposal information from database. And the function will return a sorted list of proposal information and send it back to the frontend.

- Join & Post Activity
 - "PickUp" proposal joining

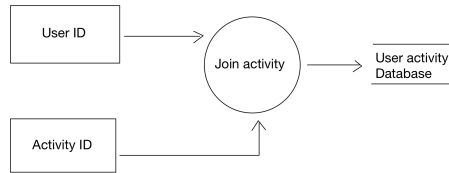


Figure 2.2-3

Figure 2.2-3 shows the data flow of "PickUp" proposal joining procedure, if the user is confirmed to join one proposal, the front-end will send the user_id and the ID of the proposal the user is interested in to the join activity function, and it will insert the participant relation of activity into user activity relationship database.

- "PickUp" proposal posting

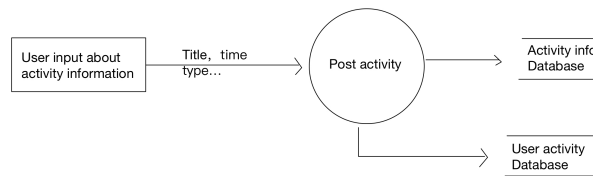


Figure 2.2-4

Figure 2.2-4 shows the data flow of "PickUp" proposal posting procedure, users are able to provide essential information about the proposal, like the type, the title, the time of the activity and the Post activity function will insert the information about the activity into the activity info database and the manager relation of activity into user activity relationship database.

- Manage Activitys
 - Activity List

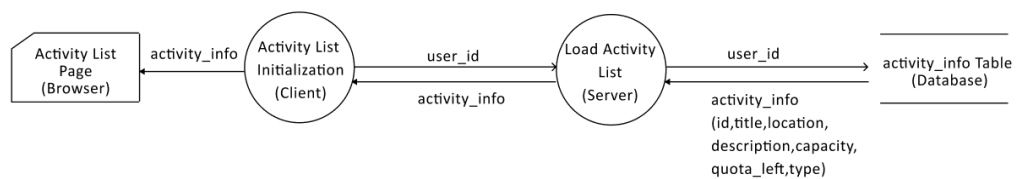


Figure 2.2-5

Data flow of activity list page is shown in Figure 2.2-5. Front-end will send a request to the back-end to collect all the activity data of a specific user. Back-end will access activity data and send back data to the front-end, and the browser will render the content to the page.

- Activity Management

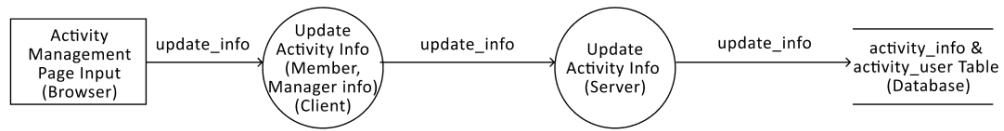


Figure 2.2-6

Data flow of update activity info page is shown in Figure 2.2-6. When update activity info request is triggered by clicking the button on web page, the updated info is sent to the back-end from front-end, and the updated info data will be update into the database from back-end.

2.2.3 Admin User Interface

- Manage User Account & Manage Admin Account
 - Load User/Admin Account Info

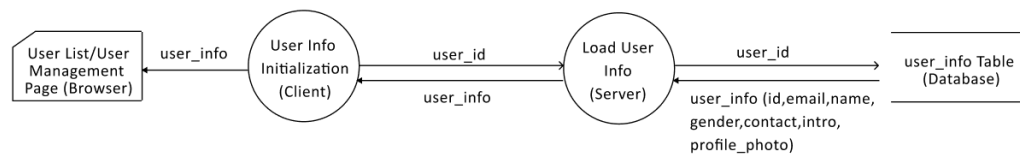


Figure 2.2-7

Data flow of loading user/admin account info is shown in Figure 2.2-7. When a load data request is sent from front-end when the page is accessed, the back-end will forward the query to the user_info table in database. Once the query returns the corresponding value, the user_info data will be forward all the way to the front-end and dispalyed on the webpage.

- Update User Account Info

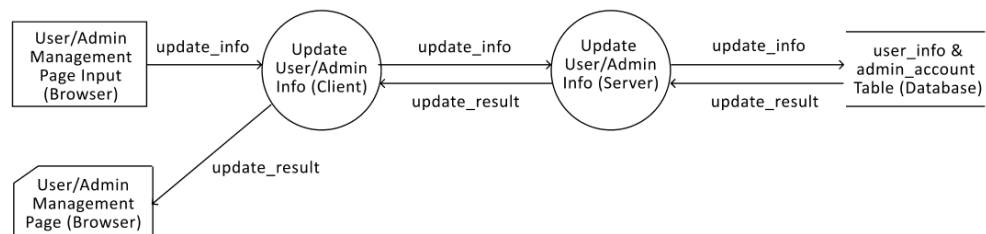


Figure 2.2-8

Data flow of updating user/admin account info is shown in Figure 2.2-8. The update action includes changing the password of user account, deleting user account, create new admin account, change password of admin account and delete admin account. Once the update request is triggered by clicking the button on webpage, the update data will be sent to the back-end and the update query will be conducted into database. Once the update query is finished, an result message will be sent back to the front-end and a pop up window will notify website users the update results.

3 Detailed Description of Components by UML

3.1 Class Diagram

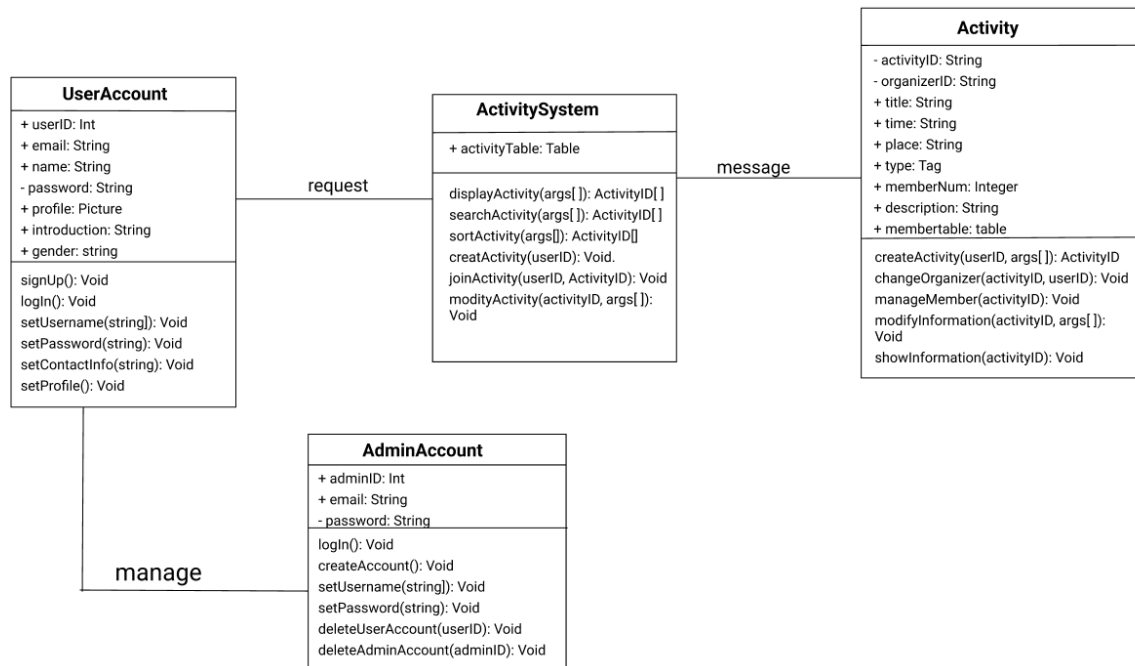


Figure 3.1-1

The above class diagram shows the relations between the classes in our software development design process. This diagram clearly states the essential classes we plane to have in the database.

The UserAccount class is used to store the information about the user, and it provided the basic function for the user to modify their personal information.

And here will be an Activity class as well, which stores the information about the Pickup Proposal. This class also allows the user to modify the activity with the specific activityID and some necessary arguments.

Between the communication of UserAccount and Activity classes, there is a ActivitySystem class. This class stores all the available activities and can receive the requests from users to carry out the "PickUp" Proposal search and display functionality.

Additionally, there is a AdminAccount. It store the information of administrator, the administrator will have the ability to access and modify the information of user account.

3.2 User Account

3.2.1 UML

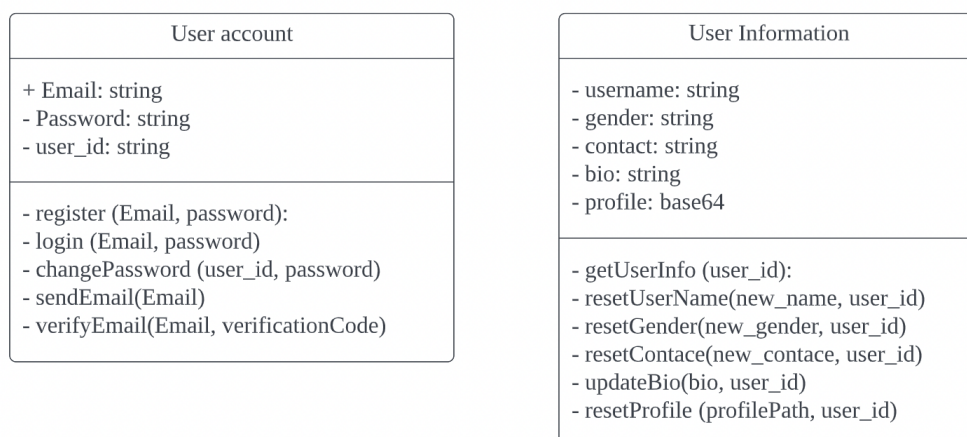


Figure 3.2.1-1 UML of user account

3.2.2 Functionality

- User account
 - This component has three attributes: Email, Password, User_id. User_id is automatically assigned when a new user is created. Besides, it has 5 basic functions to satisfy the needs to operate on these attributes.
 - Function “register” is to get the input of password and email from the front end.
 - Function “login” is to check whether the password and the input is compared then let user log into our website.
 - Function “changePassword” is to change the original password to a new password based on the user_id stored in the website.
 - Function “sendEmail” is used to set email for email verification when a new user attempts to create an account.
 - Function “verifyEmail” is to compare the verification code from the link and the server.
- User Information
 - This component contains five attributes that store user’s basic information. Besides, there are also 6 functions in it.
 - Function “getUserInfo” is to get user’s five information based on the user_id stored in the website. And these five attributes related to the user information will be sent to the front end to show to user.
 - Function “resetUserName” allows users to modify their original username to a new user name.
 - Function “resetGender” allows users to modify their original gender to another genders.
 - Function “resetContact” allows users to modify their original contact information to a new contact information.
 - Function “updateBio” allows users to modify their new biography and interests.
 - Function “resetProfile” allows users to upload a new image as their profiles in our website.

3.2.3 Procedure and Functions

- User Account & User information Management
 - Register page will has two input blocks. It will receive the input from the user and then it will generate a random verification code and call the function “sendEmail” which will send the email to the user’ input email address. And this email contains a link which has the information about the verification code.
 - If the new user clicked the link and our server will compare the information in the link and compare it to stored verification code. If it true, the register process is completed. Besides, the password and the email of the new user will be stored into the data base. During the storing process, the database will also generate a unique user_id for this email.
 - Login page is used to let user login. It has two options. The first one is logging in as a normal user. The other option is logging in as an administrator. Under the input block of there is a register button for new user. The front end receives the input email and password and send it to server. The server will load the password with respect to the input email from the database. If the password in the database is equal to the password from user’s input, the server will redirect the user to the home page with more authority.
- Setting & Updating Personal Information
 - The personal information page will show user’s information to the user. It loads user_id from the website and get this user’s information from the database to the server. After receiving data from the database. The server will send this data to user’s end to show.
 - The setting page is to modify user’s password or other information. The email and user_id cannot be changed once the account is created. If user wants to reset its account’s password. He has to input the original password and the new password in the input box. After server get the three inputs. It will check whether original password is equal to current password. If they are different, the server will reject the request and asked user to input again. Otherwise, server will replace the old password by the new password. Besides, the comparison process of the new password and the confirmed password is done on the front end. If they are not equal, the reset will also be rejected.
 - The modification of other information is less complicated. Once the user clicks the reset button, the information in the input box will be sent to the server. And the server will update the corresponding value in the database.

3.3 Activity

3.3.1 UML

- Use Case Diagram

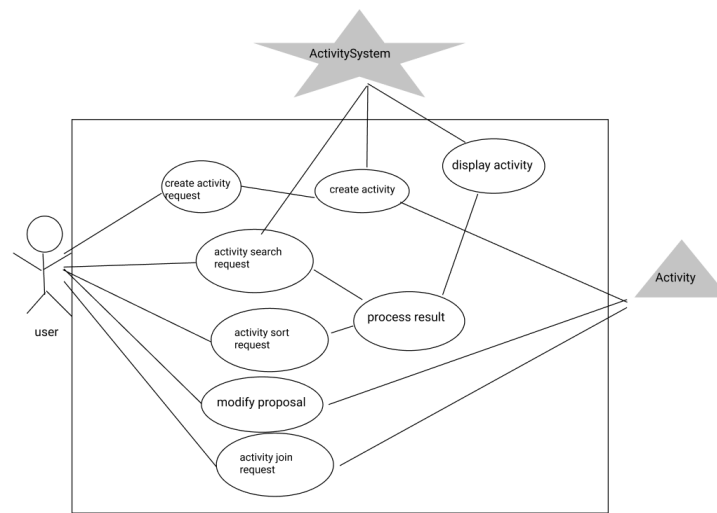


Figure 3.3.1-1 Use Case Diagram

The above use case diagram briefly shows the possible usage of Activity component by users. The functions related to PickUp Proposals all involve three classes: UserAccount, ActivitySystem and Activity class.

If the user posts a proposal search request, the ActivitySystem will go through the database and return the expected result, and there will be a result processing process before the activities are displayed to the user.

If the user posts a proposal sort request, the ActivitySystem will fetch all the activity information from the database, and there will be a result processing process to sort the activity by given order before the activities are displayed to the user.

If the user post a proposal create request, the ActivitySystem will fetch the activity information provided by the user and insert the information into the Activity information table in the database. And the manager relationship between the user and the activity will be inserted into the User Activity relationship table.

- SequenceDiagram

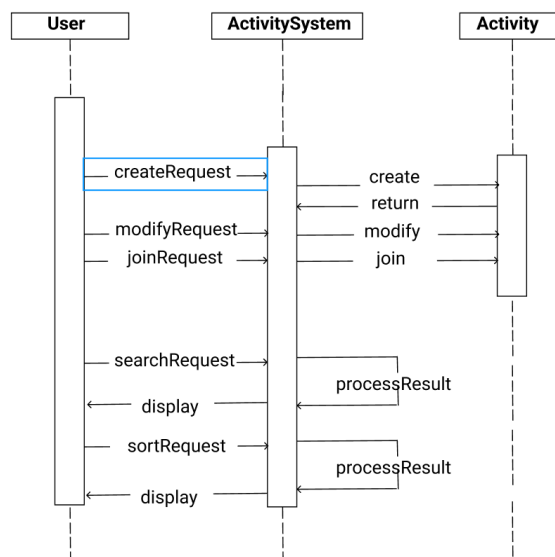


Figure 3.3.1-2 Sequence Diagram

The above sequence diagram describes the full sequence of how a user can interact with the Activity component. As this involves some functionalities related to the component, the details will be discussed in the following section

3.3.2 Functionality & Procedures & Functions

With Activity System component, a user can have ability to access and interact with "PickUp" proposals.

- To Create a new "PickUp" Proposal, the user can click the "Post" button on the home page, and there will be can function checking if the user had logged in or not. Only the user who have logged in can post a new activity. After clicking the button, the user will be redirected to the "activityCreation" page, where users can provide the detailed information about the activity. After that, the user can click the post button on the page to post the activity. During this procedure, the "postActivity()" in the frontend will be called, and the server will receive the request and add information to the tables in the database. The main functions related to this procedure is listed below:

- front end:

```
tryPost() {},  
postActivity() {},
```

- backend:

```
app.post("/postActivity", function(req, res){});
```

- As for searching and Sorting the PickUp Proposal, user can provide the kind of searching he/she want the system to carry out. For example, he/she may want search for activities of "Sports" type, and then he/she click the "search" button, the "searchActivity" function in the frontend will be called, sending a searching request to the backend, there is a corresponding function in the backend ready to fetch the information and sort them. After the sorting procedure, the activity information is send back to the frontend to be displayed to the users. The sorting procedure is similar to the searching procedure, except that the data send to the backend is the order or the sort. The main functions related to this procedure is listed below:

- frontend

```
searchActivity() {},  
searchByDate() {},  
sortActivity() { },
```

- backend

```
app.post("/searchActivity", function(req, res){});  
app.post("/searchByDate", function(req, res){});
```

- The user can also join the activity he/she is interested in. The is a "Activity Square" on the home page, and there are at most 12 "activity card"s on one page. The "activity card" list brief information (only title, type, location and time) about the activity. If the user want to have a look at the detailed information about the activity, he/she may click on the activity card, a "detailed card" will appear on the page, all the information (manager, description, number of members) will be shown on the "detailed card". And there will be a join button on the "detailed card" if the number of members is not full, users may click it to try join that activity. Before joining the activity, there is two checking procedure, first one is to check if the user has logged in, second one is to check if the user has already participated in the activity. The main functions related to this procedure is listed below:

- frontend

```
showDetail(index) {},  
tryJoin() {},
```

- backend

```
app.post('/joinActivity', function(req, res){})
```

- User may also check the proposal he/she participates in and manipulate the proposals he/she organized. The user can go to the "activityList" page to access all the activities he/she participates in. If he/she is a member of the activity, he/she can quit that activity. While if he/she is the organizer of the activity, he/she has the authority to delete members, change organizer, and terminate the activity. The main functions related to this procedure is listed below:

- frontend

```

getActivityInfo() {},
getUserRole() {},
deleteMember(userId) {},
changeManager(newManagerId) {},
terminateActivity() {},
getMemberInfo() {},

```

- backend

```

function get_activity_list(user_id,callback){}
function get_activity_info(activity_id,callback) {}
async function pack_activity_info(user_id){};
function get_user_role(user_id, activity_id, callback) {}
function delete_member(user_id, activity_id, callback) {}
function update_role(activity_id, user_id, new_role){}
function change_manager(activity_id, new_manager_id, old_manager_id){}
function terminate_activity(activity_id) {}
function get_member_list(activity_id, callback){}
function get_user_info(user_id, callback){}
async function pack_member_info(activity_id) {};
app.post("/getActivityInfo", function(req, res){});
app.post("/getUserRole", function(req, res){});
app.post('/deleteMember', function(req,res){});
app.post('/changeManager', function(req,res){});
app.post('/terminateActivity', function(req,res){});
app.post('/getMemberInfo', function(req,res){});
app.post("/getActivityList", function (req, res) {});

```

3.4 Admin Interface

3.4.1 UML

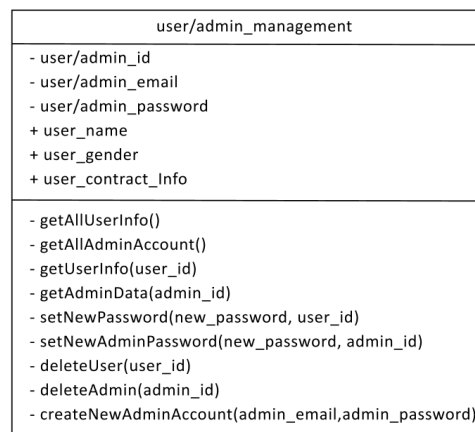


Figure 3.4.1-1 Use Case Diagram

3.4.2 Functionality & Procedures & Functions

- User/Admin Account List
 - To display all the user account information, the `getAllUserInfo()` or `getAllAdminAccount()` function is called. Data of all user will be accessed from `user_info` or `admin_account` database and the data will be passed to the front-end for page rendering. The front-end page is further decomposed into components. The `UserList`, `UserListCard` components are used to dynamically render each of the user info onto the page.
 - To display a single user/admin account's information, the `getUserInfo(user_id)` or `getAdminData(admin_id)` function is called. A single user/admin account's data will be get from the database with provided `user_id` or `admin_id`. The data get will be passed to front-end and rendered on the page.
- User/Admin Account Management
 - Update the user/admin account password. The `setNewPassword(new_password,user_id)`, `setNewAdminPassword(new_password,admin_id)` two functions will update the corresponding password value

in the database when a change new password request is being sent with the provided `new_password` and `id`.

- Delete user/admin account. The `deleteUser(user_id)`, and `deleteAdmin(user_id)` function will be called when a delete account request is sent from the front-end. The corresponding record in the `user_info` or `admin_account` table will be deleted with provided `id`.
- Create new admin account. The `createNewAdminAccount(admin_email,admin_password)` is called if a create new admin account request is triggered. A new record with provided `admin_email` & `admin_password` will be inserted into the `admin_account` table when the request is received by back-end.

4 User Interface Design

4.1 Description of the User Interface

Our website user interface is divided into multiple pages according to different functionalities:

- Landing page
- Login page & Sign Up page
- Activity Display & Management
 - Home page
 - Create new activity page
 - My activity list page
 - Activity management page
- User Info Display & Management
 - Personal information page
 - Setting page
- Admin Interface
 - User account list page
 - User account management page
 - Admin account list page
 - Admin account management page
 - Create new admin account page

All these pages described above will be demonstrated in detail in the following section 4.2. The UI layout is designed by ourselves while some of the UI components are from Bootstrap 5 UI component library.

4.2 Screen Images & Objects & Actions

4.2.1 Login & Sign Up

Login Page is shown in Figure 4.2.1-1.

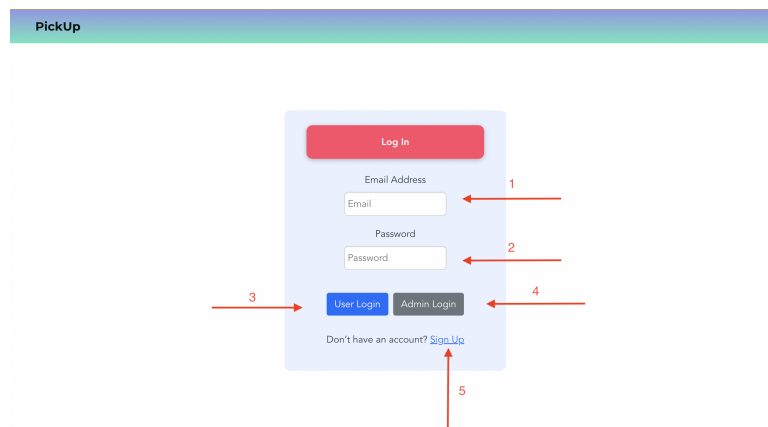


Figure 4.2.1-1 Login Page

Arrow 1 and 2 is where the user inputs its email and password. If click the User Login (3), function “login” will be called to enter the home page. If user is new to our website, he can click the blue words “Sign Up” (5) to register. Administrator can manage the system by click “Admin Login” (4).

Sign Up page is shown in Figure 4.2.1-2

Sign Up

Email Address

Email

Password

Password

Sign Up

Already have an account? [Log In](#)

1

2

3

4

Figure 4.2.1-2 Sign Up Page

The first two input boxes are where the user enters their Email and password. And after click the “Sign Up” (3), user should check its email account. If the user already has an account, he can click the underlined blue words “Log in” (4).

4.2.2 Home Page

PickUp

Home

Activity Management

Personal Center

Login

Welcome to PickUp, Hope you can find friends here.

Now is: 2022/5/12 02:11:54 Thursday

type

Sports

search

Please select one

Sort

12 May 2022

search

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Eat together

type: Meals

time: 2022-05-01

location: Shaw Canteen

title: Travel together

type: Travel

time: 2022-05-22

location: In Shenzhen

title: Travel together

type: Travel

time: 2022-05-22

location: In Shenzhen

1

2

3

4

5

6

Figure 4.2.2-1 Home Page

The above figure is the screenshot of the home.

(Label # 1)User may search for "PickUp" proposal: user should first chooses the kind of search (search by title or search by type) and types in the content in the input box, then he can click the search button. The figure below shows the result of search by type.

type

Sports

search

Please select one

Sort

12 May 2022

search

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

title: Go and Run

type: Sports

time: 2022-05-21

location: Da Yun Sports

Figure 4.2.2-2 Search_By_Type

(Label # 2)User may search for "PickUp" proposal by date: user can click the input box and a table of calendar will appear, then he can choose on date and click the search button next to the input box. The figure below shows the calendar for date selection and the result of search by date:

12 May 2022

search

o and Run

ports

022-05-21

n: Da Yun Sports

2022

Thursday 12 May

May 2022

SUN

MON

TUE

WED

THU

FRI

SAT

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

Figure 4.2.2-3 Calendar

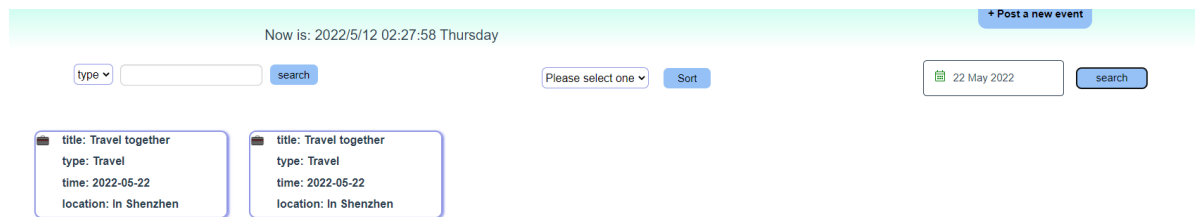


Figure 4.2.2-4 Search_By_Date

(Label # 3) User may sort the proposals by certain order by choosing the sort order and then click the sort button, the figure below shows the result of sort by "MostRecent":

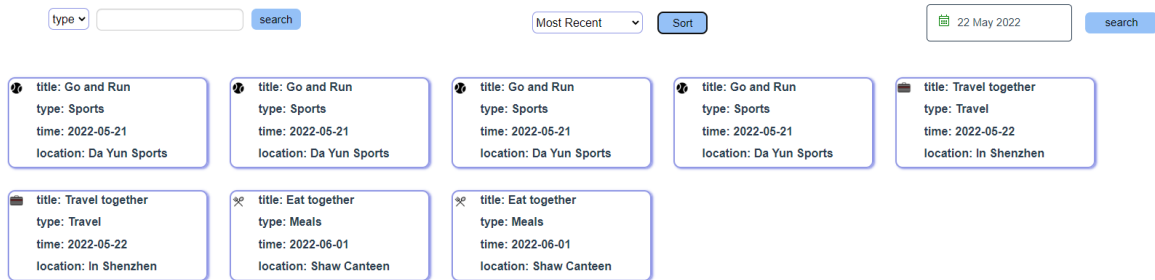


Figure 4.2.2-5 MostRecent

(Label # 4) User may click the post button to post his/her own proposal

(Label # 5 & 6) This is the activity square and activity card showing the brief information about the "PickUp" proposal, after clicking on one activity card, the detail card will be shown on the screen:

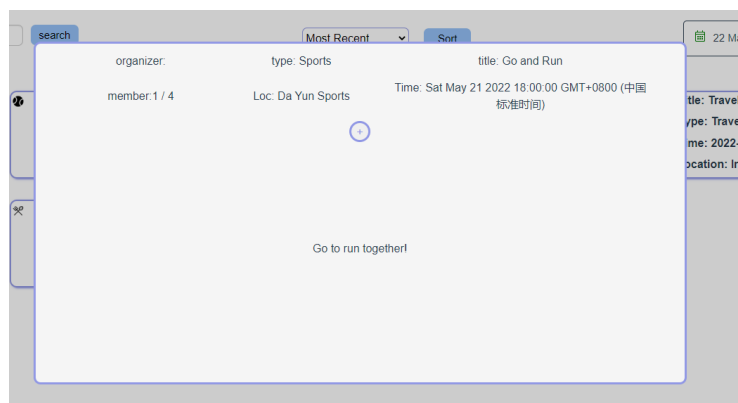


Figure 4.2.2-6 DetailCard

The user may click the "+" button to join the activity.

(Label # 7) These are the buttons for changing the page number. User can click the previous button to go to the previous page of activity square or the next button to go to the next page of activity square. He/she may also go to the page he want to browse by clicking the right most choosing box.

4.2.3 Activity Creation Page

PickUp

[Home](#)
[Activity Management](#)
[Personal Center](#)

Logout

Post a New Event

Post

Activity Type

☐ Sports
☐ Meals
☐ Travel
☐ Shop online
☐ Carpool

Activity Title

Type in your activity title here

Activity Time

12 May 2022

00

00

No. of Members

☐ Two
☐ Three
☐ Four
☐ More

Activity Description

Type in your activity description here ...

Activity Location

Figure 4.2.3-1 Activity Creation

The above figure is the screenshot of the Activity Creation Page. Users can provide the type, title, time, no. of members, description and the location of the activity by clicking the choosing boxes or typing in context into the input boxes. After all information is given, user can click the post button at the right corner of the screen to post the activity.

4.2.4 Activity List Page

Activity list page is shown in Figure 4.2.4-1

- **Label 1** Each of the "rows" are a single record of an activity the user has joined in. The title, time, and remaining quota will be displayed in this page.
- **Label 2** By clicking the button with label 2, user will be redirected to the detail information page of an activity.

| Activity List | | | |
|---------------|--------------------------|-----------------|------------------------|
| Title | Time | Remaining Quota | More Info |
| Go hiking | 2022-04-30T12:00:00.000Z | 4 | Detail |
| Run! | 2022-04-28T15:00:00.000Z | 3 | Detail |
| shop together | 2022-05-12T17:00:00.000Z | 3 | Detail |

Figure 4.2.4-1 Activity List

For any activity, there should be two kinds of role: member and manager.

Activity management page of an activity member is shown in Figure 4.2.4-2.

- **Label 1** Member of an activity can see the list of names of all the other members who also join the same activity.
- **Label 2** Member can choose to quit from the activity by clicking the [Quit](#) button.

Activity Management

Title

Run!

Time

2022-04-28T15:00:00.000Z

Location

University Sports Hall

Type

Sports

Description

Do exercise together!

Max Capacity

5

Quota Left

3

Member List

kevin

philip

Actions

Quit

Back

Figure 4.2.4-2 Activity Management (Member)

Activity management page of an activity manager is shown in Figure 4.2.4-4. Manager of an activity has higher authority than member.

- **Label 1** Managers can delete a member from an activity or set other member as a manager by clicking the drop down button with member's name. A detailed demonstration is shown in Figure 4.2.4-3

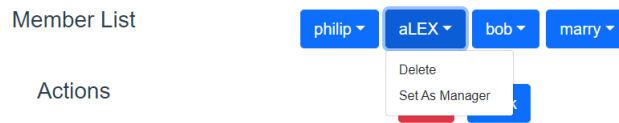


Figure 4.2.4-3 Manage Activity Member

- **Label 2** Manager also can also terminate an activity, by doing so, all the members will automatically be deleted from the activity.
- **Label 3** Managers can quit an activity, but they must make sure they have assigned other members as new manager before they quit. Any activity must have at least one manager.

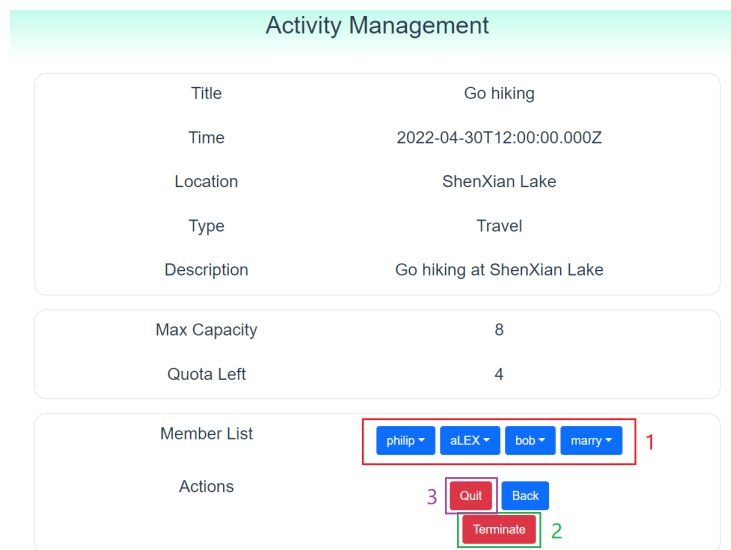


Figure 4.2.4-4 Activity Management (Manager)

4.2.4 Personal Information Page

Personal information page is shown in Figure 4.2.5-1.

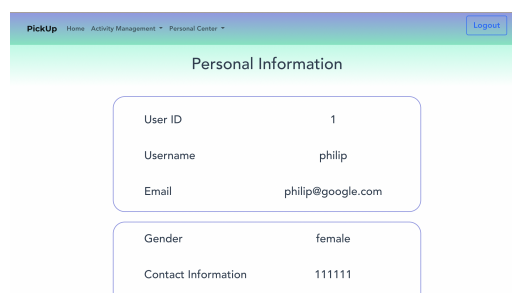


Figure 4.2.4-1 Personal Information

In this page, user can get their basic information of their account. Including user id, user name, user email, user gender, contact info, biography, and set user profile image.

4.2.5 Setting Page

Reset password function is demonstrated in Figure 4.2.6-1.

The screenshot shows the 'Settings' page for 'PickUp'. Under the 'Password Settings' section, there are four input fields and a button. Red arrows and numbers 1 through 4 point to each element: 1 points to the 'Original Password' field, 2 points to the 'New Password' field, 3 points to the 'Confirm new Password' field, and 4 points to the 'Reset password' button. Below the password fields is a 'User Name' section with a label 'Your User Name'.

Figure 4.2.5-1 Reset Password

The first box is to enter the original word (1). And the next two box is to enter the new password for twice (2, 3). After the user inputs all these three boxes, he can click “Reset Password” (4) to update the new password to our system.

Reset user name, contact info and other user information's functionality is shown in Figure 4.2.6-2

The screenshot shows three sections of the settings page: 'User Name', 'Contact', and 'Gender'. The 'User Name' section has a 'New User Name' input field (labeled 1) and a 'Reset User Name' button (labeled 2). The 'Contact' section has a 'New Contact' input field. The 'Gender' section has a label 'Your Gender'.

Figure 4.2.5-2 Reset Other Personal Info

User can modify their user information by typing new data into the input box (1). And after click the “reset user name” or other else, the user information will be updated in our system.

4.2.6 Manage User Account

User account list is shown in Figure 4.2.6-1.

- **Label 1** Each row lists the user account information of a specific user. The user id, user name, user email are listed.
- **Label 2** By clicking the **Manage** button, admin user will be redirected to user management page .

| User List | | | |
|-----------|-----------|-------------------|--------------------------|
| User Id | User Name | User Email | Action |
| 1 | philip | philip@google.com | 2 Manage 1 |
| 2 | aLEX | alex@google.com | Manage |
| 3 | bob | bob@google.com | Manage |
| 4 | marry | marry@google.com | Manage |

Figure 4.2.6-1 User Account List

User management page is shown in Figure 4.2.6-2.

- **Label 1** Admin can reset user's password by entering the new password in the input box and press the **Set** button.
- **Label 2** Admin can delete an user account by clicking the **Delete** button

User Management

| | |
|-----------------------|----------------------|
| User ID | 1 |
| Email | philip@google.com |
| User Name | philip |
| Gender | female |
| Contact Info | 111111 |
| Personal Introduction | I'm an outgoing boy. |

Set New Password For User Set
1

Action 2
Delete
Back

Figure 4.2.6-2 User Account Management

4.2.7 Manage Admin Account

Admin account list is shown in Figure 4.2.7-1

- **Label 1** Admin account info is displayed one row. Admin id, admin account email is displayed.
- **Label 2** By clicking the **Manage** button, admin will be redirected to the admin account management page
- **Label 3** By clicking the **Create New Admin Account** button, admin will be redirected to the create new admin account page

Admin Account List

Create New Admin Account
3

| Admin Id | Admin Account Email | Action |
|----------|---------------------|--|
| 1 | admin1@gmail.com | Manage 1 |
| 2 | admin2@gmail.com | Manage |

Figure 4.2.7-1 Admin Account List

Admin account management page is shown in Figure 4.2.7-2

- **Label 1** Admin user can change the password of other admin user accounts
- **Label 2** Admin user can delete other admin user account

Admin Account Management

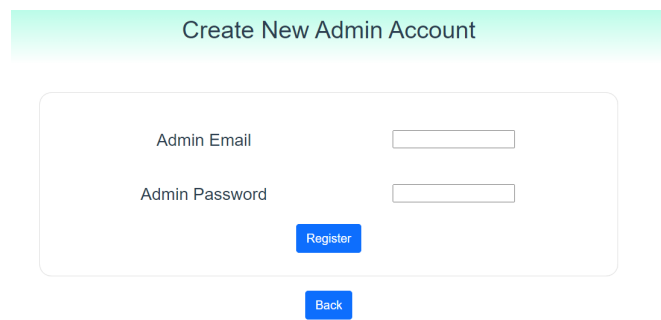
| | |
|-------------|------------------|
| Admin Id | 1 |
| Admin Email | admin1@gmail.com |

Set New Password For Admin Account Set
1

Action 2
Delete
Back

Figure 4.2.7-2 Admin Account Management

Create admin account page is shown in Figure 4.2.7-3. Admin can create new accounts by filling the admin email and admin password inputbox and press the **Register** button. New admin accounts will be automatically created.

A screenshot of a web form titled "Create New Admin Account" in a light green header. The form is enclosed in a light gray rounded rectangle. It contains two input fields: "Admin Email" and "Admin Password", each with a corresponding label to its left. Below the "Admin Password" field is a blue "Register" button. At the bottom center of the form is a blue "Back" button.

Create New Admin Account

Admin Email

Admin Password

Register

Back

Figure 4.2.7-3 Create New Admin Account

4.2.8 Landing Page

A static landing page is made to introduce our website to new users and attract new users to use our application.

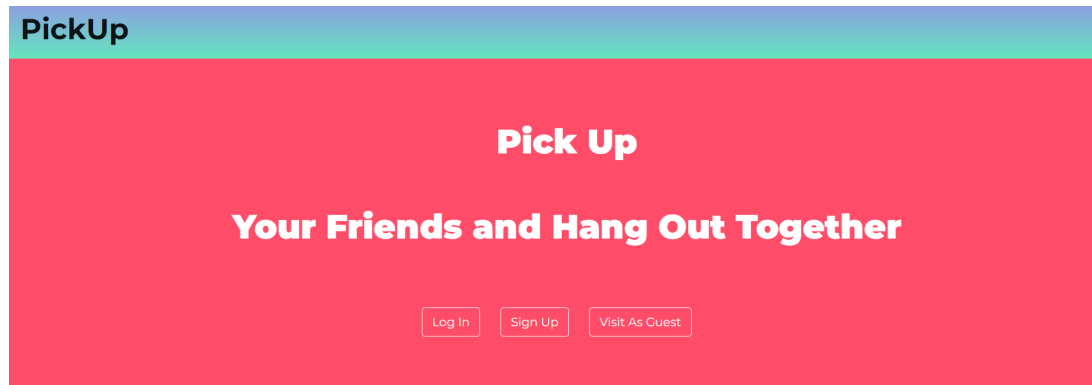


Figure 4.2.8-1 Landing Page

5 Test

5.1 Test Overview and Test Plan

We manually try out different cases of correct and incorrect inputs on both the website pages and directly on the server api. We use postman, an on-line api testing platform to try out different testing cases on our servers' apis directly.

5.2 Case-1-Activity Creation

To test if the activity creation component is robust enough, we designed 7 kinds of test cases to mimic the possible input of users, some representative cases are listed below:

| case | type | title | number | other number | location | description | expected result | real result |
|------|--------|-------|--------|--------------|-------------|-------------|--------------------------------------|--------------------------------------|
| 1 | null | a | 2 | null | canteen | aaa | warning of null value of type | warning of null value of type |
| 2 | Sports | null | 3 | null | Sports Hall | bbb | warning of null value of title | warning of null value of title |
| 3 | Meals | b | null | null | canteen | ccc | warning of null value of number | warning of null value of number |
| 4 | Sports | c | 4 | null | null | ddd | warning of null value of location | warning of null value of location |
| 5 | Meals | d | null | 5 | canteen | null | warning of null value of description | warning of null value of description |
| 6 | Sports | e | null | 6 | Sports Hall | eee | warning of null value of type | warning of null value of type |
| 7 | Meals | f | null | 7 | canteen | fff | post successfully | post successfully |

6 Lessons Learned

From a structural level, we learned to select and connect different frameworks together (front-end, back-end, database). From an website implementation level, we learned the basic components of a web page: HTML, CSS, JAVASCRIPT. We lerned to use Vue.js to implement different logic and page navigation in pages with vue components, vuex package and vue-router package. To implement the communication mechanism between front-end and back-end we learned to use axios and express.js package, and to transfer data between MySQL database and back-end we learned to use mysqljs package. We learned to design a robuts database and write efficient MySQL querys during the interaction between back end and MySQL database. We learned to deploy a website to the cloud through Amazon Web Service console and API. We learned to host web page front-end and back-end with Apache web server and pm2 process control. We learned to host our remote database using Amazon Web Service Relational Database Service. From version control perspective, we learned to use git and github operation to carry out team cooperation better. We learned to test our server API using Postman.

7 Conclusion

Our project is a website based application which is intended to help people solve their need of team up with others with the similar needs quickly on line. The entire project has the front-end (client), back-end (server), database three parts. Front-end is built on Vue.js and back-end is built on Node.js with Express.js. Database used is MySQL. The project is deployed on line useing Amazon Web Services. The front-end and back-end is hosted using Amazon Web Service Elastic Compute Cloud, and the database is hosted remotely using Amazon Web Service Relational Database Service.

Besides from fulfilling all the requested functionalities, such as login, sign up, email verification, admin management, our website provides a user friendly solution to meet up with our initial target. We implement an easy to use interface for users to quickly find partners to team up with for their interests. We also implement a complete user and admin account separation and isolation system, so that our project can be further used in other communities.