Design Document

Group NO.	Group56
Group Members	Haoran Wang, Zijing Wu, Yanqi Xu, Licheng Ma, Jiacheng Zhang, Lihao Xu

CONTENTS

1. Summary of Proposal

1.1 Background, Aim and Objectives

The "Volunteers" software is an innovative platform that designed to serve community groups, charitable organizations, non-profit entities, and university students across the United Kingdom by facilitating volunteer services. The software's main goal is to make the life of activity organizers easier by offering all the necessary activity information in one place, which they can easily access and manage. Furthermore, our project provide an easy way for students to get activity involved and a channel for them to communicate. In the Volunteers, we train an AI model which ensures that organizers are able to find the most suitable volunteers based on their needs. This is achieved through matching the skills and interests of volunteers with what is required for the activities— ensuring an intelligent volunteer matching service. The platform also includes a forum feature that aimed at magnifying the effect of community involvement in volunteer events.

This document serves as the design document, aiming to comprehensively document and detail the design methodologies and decisions adopted throughout the project. It discusses different elements that include software's architecture, functional components, data model design and process design details along with technical implementation plus user interface design. The document offers explicit development directions for team participants, establishing the basis of future enhancements and maintenance activity.

Aims & Objectives

The primary aim of the project is to provide a comprehensive volunteer service platform that addresses the following objectives:

Aims：

1. Provide a one-stop volunteer service platform
2. Meet the needs of social service organizations (e.g., nursing homes, hospitals) for volunteer human resources by utilizing the platform.
3. Enhance the visibility and impact of volunteer service activities.
4. Utilizes layered administrative privileges to ensure system security and stability.

Objectives:

1. Implement a fuzzy search by fuzzy matching search terms with information on volunteer activities that allows users (students) to easily check and sign up for volunteer activities based on their interests (a1)
2. All procedures related to volunteer activities, including enrollment, posting, updating, deletion, and auditing can be unified on the platform (a1) .
3. Organizer account could post volunteer activities and could view and manage information about users who have been matched with posted activities (a2, a4).
4. Organizer account could post volunteer activities and could view and manage information about users who have been matched with posted activities (a2, a4).
5. Implement a reputation monitoring mechanism that can be used to trace and keep a record of the completion of a user's (student) volunteer activities (a4).
6. The system administrator has super-user privileges to view and manage information and activity status of all users including students and organizers within the system and be able to unlock and lock users (a4)

Main Functions and components from initial requirement analysis document：

1. Administrators can view, edit, and delete user details for students and organizers, as well as review organizer registration applications to maintain accurate and secure user data on the platform.
2. Organizers have the ability to post volunteer activities on the platform, which includes entering activity details, scheduling activity times and locations, and recruiting volunteers. They can also manage event registrations and

communicate with participants.

3. Students can browse information about activities within the software, locate events of interest through search and filter functionalities, and sign up to participate in volunteer activities. They also have the opportunity to earn points for their involvement in activities and view a leaderboard of student user points to encourage active participation.
4. The software is integrated with a social sharing module that supports both organizers and students to share their experiences and insights from activities. This fosters communication and interaction among users and enhances the impact of the activities.
5. Both students and organizers are required to register for an account with providing the necessary personal information. Administrators log in to the system using credentials assigned by the software entity to access administrative dashboard.
6. The development of the platform use the WinForms .NET Framework 4.8, which provides a stable and reliable framework for the user interface and supports cross-platform deployment and utilization.
7. The backend logic is implemented using the C# language, leveraging its robust object-oriented features and extensive class libraries to fulfill the system's business logic and data processing functionalities.
8. The data management is handled using a SQL Server database, ensuring the security and reliability of the data. The database is deployed on Alibaba Cloud's Elastic Compute Service (ECS), which offers high-performance and scalable data storage and access services.

1.2 Changes to Specification

1) **Volunteer Activity Sign-up Audit**: We add a new function for auditing volunteer registrations, which allows organizers to audit volunteers application after activity sign-up. Within the system, we designed an AI model that calculates the suitability of volunteers. This AI model will be trained by the student's history of participation and total activity points. It will be integrated with the sign-up audit interface where organizers can view the personal information of applicants and assess the match between the student and the event. This process enables organizers to make an informed decision on whether to approve a volunteer's participation in a given activity. In the initial requirement analysis, we did not design an audit process for volunteer registration. Feedback from organizers indicated that this feature is crucial for ensuring the quality of the events and managing volunteer engagement.
2) **Map Functionality for Activity Post Address Selection:** We add an interactive map function in the organizers' activity posting interface, enabling organizers to select the exact address of the posting activity by clicking on the map or entering geographical coordinates. This functionality is achieved by integrating a Google Maps service API into the software. The inclusion of the map function is a significant enhancement to the activity posting and display process. It can increase convenience for organizers when publishing activities and aids volunteers in better visualizing the event location.
3) **Forum Functionality：** The social sharing module initially outlined in the requirement analysis has been expanded into a comprehensive forum. This forum serves as an interactive platform for both student user and organizer user to engage with one another. It allows users to post topics, share their experiences from volunteer activities, reply to comment posted by other users, and view the participation and feedback from others. The forum integration is designed to more effectively facilitate communication and interaction between users, and enhance the level of engagement in activities.
4) **Transition from WinForms to WPF for Selected Interfaces：** Initially, we decided to developed entirely project using WinForms for all page interfaces. However, to enhance user experience and improve the aesthetic quality of the interface, we have determined that development for the forum and leaderboard interfaces will use WPF. WPF offers a richer set of UI design capabilities compared to WinForms, including superior graphics rendering, animation effects, and customizable controls, which will assist us in creating a more engaging and responsive interface.

2. Software Design

2.1 Anticipated Components of the software

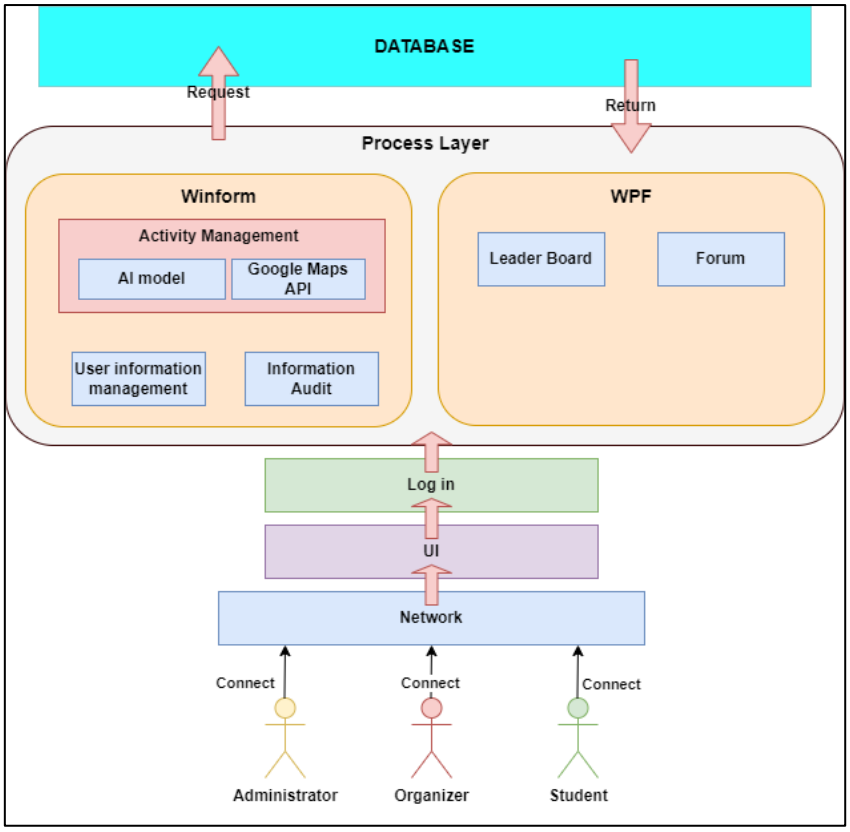


Figure 1 Anticipated Component Relationship

This section provides a detailed introduction to the main anticipated system components of our software. These components collectively form the technological foundation of the software, and providing the necessary functionalities for user interaction, information management, activity management, and system administration.

Anticipated Components:

1. **Database：** Data storage is the core of the Volunteers software. The database is responsible for preserving user information, activity information, and forum interactions. In this project, we use the Relational Database Management System (RDBMS) to organize these data. The data storage is closely integrated with components such as user auditing, activity management, and forum interaction, providing a reliable data support framework for these functionalities.
2. **Login and Registration：** The login and registration system permits new students and organizer to register their accounts, log in, and recover passwords. Meanwhile, it also allows administrators to log in using credentials provided by the software development team. The implementation of this component is based on user information, ensuring the legitimacy and security of user accounts. Upon successful registration, users gain access to their personal information management and can utilize other features of the platform.
3. **User Information management:** The User Information Management component allows users to maintain their personal profiles and account settings. Administrators utilize this component to manage registered users throughout the system, performing account management operations such as adding or removing student information and locking or unlocking accounts. This component is tightly integrated with the data storage to facilitate real-time updates to user information.
4. **Activity Management:** The Activity Management component acts as a bridge for interaction between organizers and volunteers. Within this component, organizers can post new activities, manage posted activities, and process volunteer sign-up application. To ensure the smooth progression of events, the design of this component takes into account its interaction with the User Information Management component.
5. **Audit System：** The Audit System is a pivotal element in the platform's quality control, tasked with ensuring the accuracy and effectiveness of all information needed audit. Administrators employ this component to review and approve organizer registration submissions. Organizers use the system to audit student applications for fit, evaluating whether the volunteers' skills and experiences meet the activity's demands, thus assuring the quality and efficiency of engagement in activities. Upon an activity's conclusion, organizers leverage this component to audit volunteer participation, verify the completion status of the activity, and provide assessments and feedback based on the volunteers' performance. The

Audit System collaborates closely with the User Information Management component to maintain the precision of registered information and aligns with the Activity Management component to ensure a seamless audit process for activity applications and completion.

6. **Point System:** The Point System is designed to motivate students to engage more actively in activities. It automatically updates points based on the user's completion of activity participation and reflects the results on the volunteer leaderboard interface.
7. **Forum:** The Forum serves as an interactive communication platform for users, allowing both students and administrators to participate. It enables users to post messages, share their experiences from activities, reply to topics comments, and view the participation and feedback from others in the community.
8. **Matching AI Model:** The AI Matching Model component utilizes machine learning algorithms to analyze the student's historical activity participation and activity points. It provides organizers with precise volunteer matching information, facilitating the process of finding suitable volunteers for specific activities.
9. **Maps:** The Maps component provides geographical information services to users, assisting them in more intuitively locating the addresses of events. It offers an interactive map view that allows users to easily identify the venue of an activity. This component will be integrated with the Activity Management component, it supplies users with a visual representation of activity address, enhancing users' spatial awareness and navigation to the site.
10. **Network:** The Network component is the core communication infrastructure of the volunteer service software. It is responsible for handling real-time data exchange and synchronization between users. This component leverages efficient and scalable network solutions provided by Alibaba Cloud services to ensure high availability and stability of the platform. By implementing real-time communication, the Network component supports interactive forums, activity management, and other key functionalities, allowing users to remain connected and interact from anywhere.
11. **UI (User Interface):** The UI component serves as the gateway for user interaction with the system. Utilizing WinForms and WPF technologies, a suite of intuitive and user-friendly interfaces will be developed. These interfaces work in concert with all backend components to deliver a seamless user experience.

2.2 DataModel Design

The Global Entity-Relationship (ER) diagram is used to illustrate the logical relationships among entities within the system.

2.2.1 Global ER diagram for volunteer service software

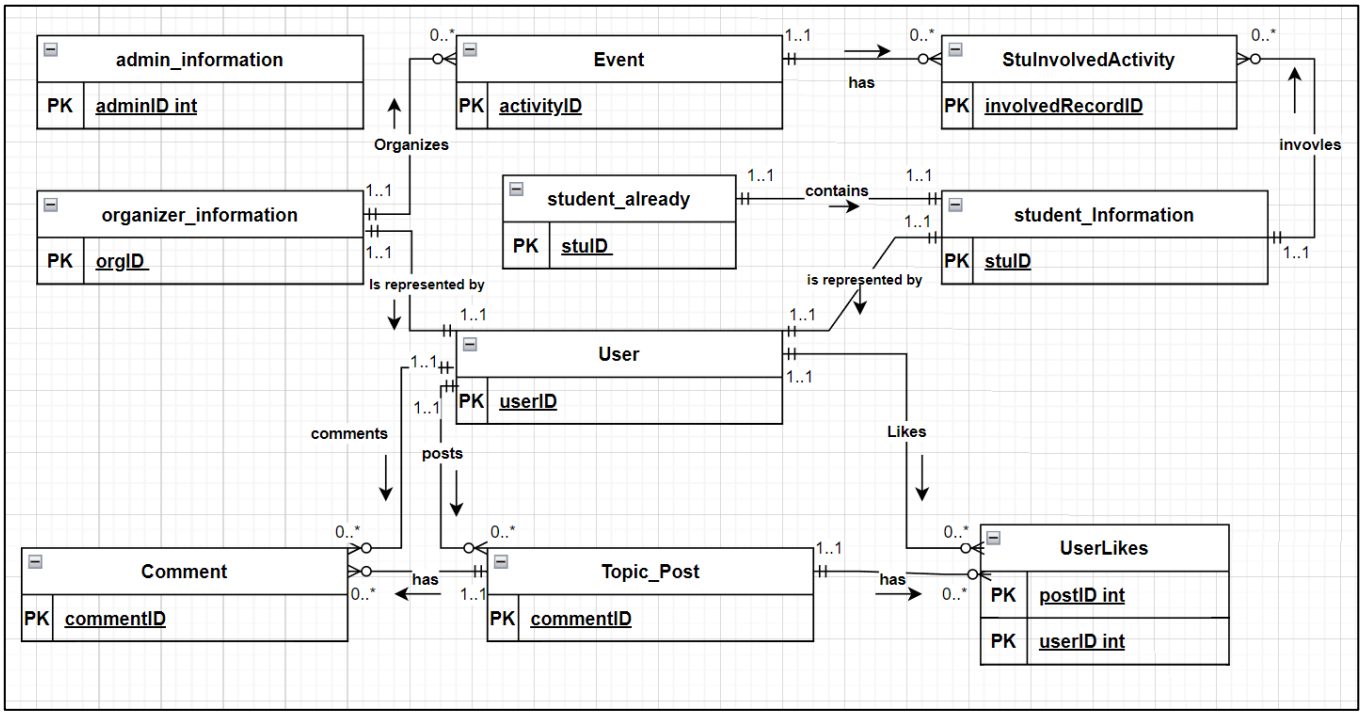


Figure 2 Global ER diagram

After creating the Global ER diagram, it is essential to document each entity's attributes and the relationships between entities using a data dictionary.

2.2.2 Data Dictionaries

Entity and property Table

Entity name	Description	Aliases	Occurrence
admin_information	Holds detailed information about administrator accounts, including administratorID and passwords.	Admin Account, Administrator	A record is maintained for each administrator in the system.
organizer_information	Contains information about organizers, such as unique organizerID, usernames, organizational affiliations, and other details.	Org Account, Organizer	A record is maintained for each organizer in the system.
student_already	Contains basic information about school-wide students who are studying in this campus, such as student ID, full names, and software registered status.	School-wide Student	Each existing student has one corresponding entry in the table.
student_information	Contains detailed records of registered students, such as student ID, student name, nicknames, contact numbers, account passwords, and account status.	Registered Students, Volunteers	A record is maintained for each registered student in the system.
stuInvolvedActivity	Tracks students' participation in activities, including the activity's ID, the student's sign-up application status, and completion status.	Student Activity Participation, Activity Engagement Record	A record is created for each instance of a student's involvement in an activity.
Event	Defines all details of an activity, such as the name, intended number of participants, date, service domain, location, service recipient and description.	Activity	There is a unique record for each activity organized within the system.
User	Provides a unique numeric identifier for each student and organizer in the system	User Account, User	Each user (student or organizer) is represented by one record in the system.
Comment	Records comments posted by users on topic, including the comment's unique ID, the user’s ID, and the text of the comment.	User Comment, Post Discussion	Each comment is given a unique record in the comments table.
Topic_Post	Stores data about topic posts, including a unique post identifier, user’s ID, title, content, and the number of likes received.	Discussion Thread, Topics	Every post made on the forum or discussion board is recorded as a single entry.
UserLikes	Logs instances where users have liked a topic post, noting the post's ID and the user's ID.	Likes, Preferences.	Each like given by a user to a post is recorded as a separate entry.

Entities Relationship Table

Entity	Multiplicity	Relationship	Multiplicity	Entitiy
organizer_information	0..*	Organizes	1..1	Event
organizer_information	1..1	Is Represented By	1..1	User
student_already	1..1	Contains	1..1	student_information
student_information	1..1	Is Compared With	1..1	student_already
student_information	0..*	Involves	1..1	stuInvolvedActivity
student_information	1..1	Is Represented By	1..1	User

stuInvolvedActivity stuInvolvedActivity	1..1 1..1	Involves Participates In	0..* 0..*	student_information Event
Event Event	0..* 1..1	Has Is Organized By	1..1 0..*	stuInvolvedActivity organizer_information
User User User User User	1..1 1..1 0...* 0..* 0..*	Represents Represents Posts Posts Likes	1..1 1..1 1..1 1..1 1..1	student_information organizer_information Comment Topic_Post UserLikes
Comment Comment	1..1 1..1	Is Posted By Involves	0..* 0..*	User Topic_Post
Topic_Post Topic_Post	0..* 0..*	Has Has	1..1 1..1	Comment UserLikes
UserLikes UserLikes	1..1 1..1	Likes Initiated By	0..* 0..*	Topic_Post User

Entity Table

Entity	Attributes	Description	Key	Data type and length	Nulls	Multy-valued
admin_information	adminID	Unique identifier for an administrator.	Primary key	Integer	NO	NO
	adminPassword	Password for the administrator account.		16 Variable Characters	NO	NO
organizer_information	orgID	Unique identifier for an organizer.	Primary key	Auto-Incrementing Integer	NO	NO
	userName	Username of the organizer.		15 Variable Characters	NO	NO
	orgPassword	Password for the organizer account.		16 Variable Characters	NO	NO
	organizationName	Name of the organizer's organization or company.		50 Variable Characters	NO	NO
	email	Contact email for the organizer.		50 Variable Characters	NO	NO
	secureProblem	Security question for account recovery.		40 Variable Characters	NO	NO
	secureAnswer	Answer to the security question.		40 Variable Characters	NO	NO
	certification_graph	Image of organizer's certification.		Varbinary (MAX)	NO	NO
	orgAvatar	Image of organizer's avatar.		Varbinary (MAX)	YES	NO
	userState	Current state of the user account.		Integer	NO	NO
	registerState	Registration status of the user.		Integer	NO	NO
	userID	A representation of organizer user.	Foreign key	Integer	NO	NO

student_already	stuID	Unique identifier for a student.	Primary key	Integer	NO	NO
	stuName	Full name of the student.		30 Variable Characters	NO	NO
	registerStatus	Status of the student's software registration.		Integer	NO	NO
student_Information	stuID	Unique identifier for a student.	Primary key	Integer	NO	NO
	stuName	Student's name.		30 Variable Characters	NO	NO
	nickname	Student's preferred nickname.		20 Variable Characters	NO	NO
	stuPassword	Password for the student account.		16 Variable Characters	NO	NO
	telephoneNumber	Student's contact phone number.		11 Variable Characters	NO	NO
	email	Student's contact email.		50 Variable Characters	NO	NO
	secureProblem	Security question for the student's account.		40 Variable Charcaters	NO	NO
	secureAnswer	Answer to the security question.		40 Variable Characters	NO	NO
	totalPoints	Total activity points earned by the student.		Integer	NO	NO
	stuAvatar	Image of student's avatar.		Varbinary (Max)	YES	NO
	userState	Status of the student's account.		Integer	NO	NO
	userID	A representation of student user.		Integer	NO	NO
StuInvolvedActivity	involvedRecordID	Unique identifier for a student's activity record.	Primary key	Auto-Incrementing Integer	NO	NO
	stuID	Student's unique ID.	Foreign key	Integer	NO	NO
	activityID	Unique ID of the activity participated in.	Foreign key	Integer	NO	NO
	applyStatus	Status of the student's application to the activity.		Integer	NO	NO
	applyTime	Time of the student's application.		Date	NO	NO
	finishStatus	Status of the student's completion of the activity.		Integer	NO	NO
Event	activityID	Unique identifier for an event.	Primary key	Auto-Incrementing Integer	NO	NO
	orgID	Unique ID of the activity organizer.		Integer	NO	NO
	activityName	Name of the activity.		50 Variable	NO	NO

				Characters		
	activityPopulation	Target number of participants for the activity.		Integer	NO	NO
	activityDate	Date of the activity.		Date	NO	NO
	activityAddress	Address where the activity takes place.		100 Variable Characters	NO	NO
	activityContent	Description of the activity.		500 Variable Characters	NO	NO
	serviceDomain	Domain of the service or activity.		30 Variable Characters	NO	NO
	serviceRecipient	Intended recipients of the activity.		30 Variable Characters	NO	NO
	serviceLocation	Specific location of the activity.		30 Variable Characters	NO	NO
	activityGraph	Graphic representation of the activity.		Varbinary (MAX)	YES	NO
	activityStatus	Current status of the activity.		Integer	NO	NO
	numOfInvolved	Number of people successfully involved in the activity.		Integer	NO	NO
	activityPoint	Points awarded for the activity.		Integer	NO	NO
	activityDuration	Duration of the activity.		Integer	NO	NO
	address_Iatitude	Latitude of the activity address.		Decimal	NO	NO
	address_longtitude	Longitude of the activity address.		Decimal	NO	NO
User	userID	A unique numeric code assigned to each student and organizer for identification.	Primary key	Integer	NO	NO
Topic_Post	postID	Unique identifier for a topic post.	Primary key	Auto-Incrementing Integer	NO	NO
	userID	Unique ID of the post's user.		Integer	NO	NO
	topicTitle	Title of the topic post.		50 Variable Characters	NO	NO
	topicContent	Content of the topic post.		500 Variable Characters	NO	NO
	graph	Associated image with the post.		Varbinary (MAX)	YES	NO
	likes_number	Number of likes on the post.		Integer	NO	NO
	createTime	Time the post was created.		Datetime	NO	NO
	videoPath	Path or link to a video in the post.		200 Variable Characters	YES	NO
Comment	commentID	Unique identifier for a comment.	Primary key	Integer	NO	NO
	userID	Unique ID of the user.		Integer	NO	NO
	postID	ID of the post being		Integer	NO	NO

		commented on.				
	commentContent	Text of the comment.		100 Variable Characters	NO	NO
	createdTime	Time the comment was posted.		Datetime	NO	NO
UserLikes	postID	Unique ID of the liked post.	Primary key	Auto-Incrementing Integer	NO	NO
	userID	Unique ID of the user who liked the post.		Integer	NO	NO
	likeTime	Time the post was liked.		Datetime	YES	NO

***Supplement:** Among the entity attributes, there are status-related attributes that the development team identifies using integer-type data. The following are the additional status indicators for each entity:

Entity	Attributes	Description	Status Representation
organizer_information	userState	Organizers' account states.	0 : Normal; 1 : Locked
	registerState	Organizers' registration application states.	0 : Wait for review; 1 : Registration Application Approved
student_already	registerStatus	Students software registration status.	0 : Unregistered; 1 : Registered
student_Information	userState	Students' account states.	0 : Normal; 1 : Locked
StuInvolvedActivity	applyStatus	Student participating activities' sign-up application status.	0 : Not pass; 1 : Pass
	finishStatus	Student activity finish status.	0 : Not finish; 1 : Finish
Event	activityStatus	Activities Status.	0 : Not yet started; 1 : In progressing; 2 : Ended

2.2.3 Logical Table Structure

The data dictionary provides a detailed record of the attributes of each entity within volunteer service software, as well as the relationships between entities. The Logical Table can map these concepts into a logical table structure.

admin_information (adminID, adminPassword) **Primary Key** adminID	**organizer_information** (orgID, username, orgPassword, organizationName, email, secureProblem, secureAnswer, certification_graph, orgAvatar, userState, registerState, userID) **Primary Key** orgID **Foreign Key** userID **references** User (userID)
student_already (stuID, stuName, registerStatus) **Primary Key** stuID	**student_Information**(stuID, stuName, nickname, stuPassword, telephoneNumber, email, secureProblem, secureAnswer, totalPoints, stuAvatar, userState, userID) **Primary Key** stuID **Foreign Key** userID **references** User (ID)
StuInvolvedActivity (involvedRecordID, stuID, activityID, applyStatus, applyTime, finishStatus) **Primary Key** involvedRecordID **Foreign Key** stuID **references** student_Information (stuID) **Foreign Key** activityID **references** Event (activityID)	**Event** (activityID, orgID, activityName, activityPopulation, activityDate, activityAddress, activityContent, serviceDomain, serviceRecipient, serviceLocation, activityGraph, activityStatus, numOfInvolved, activityPoint, activityDuration, address_Iatitude, address_longtitude) **Primary Key** activityID

	Foreign Key orgID **references** organizer_information (orgID)
User (userID, userName) **Primary Key** userID	**Topic_Post** (postID, userID, topicTitle, topicContent, graph, likes_number, createTime, videoPath) **Primary Key** postID **Foreign Key** userID **references** User (userID)
Comment (commentID, userID, postID, commentContent, createdTime) **Primary Key** commentID **Foreign Key** userID **references** User (userID) **Foreign Key** postID **references** Topic_Post (postID)	**UserLikes** (postID, userID, likeTime) **Composite Primary Key** (postID, userID)

2.2.4 Physical Table Structure

Physical tables are designed considering the size of the data and the usage patterns, taking into account physical storage parameters such as indexing, partitioning, and storage engines. Designing strategies for index and query optimization is crucial to enhancing the performance of the database.

```
domain    Administrator_ID                 Integer maximum length 16
domain    Administrator_Account_Password   variable length character string maximum length 16

admin_information (
adminID                Administrator_ID                 NOT NULL,
adminPassword          Administrator_Account_Password   NOT NULL

Primary Key    adminID )
```

```
domain    Organizer_Account_ID          Auto-Incrementing Integer increases from 0
domain    Organizer_User_Name           variable length character string maximum length 15
domain    Organizer_Account_Password    variable length character string maximum length 16
domain    Organizer_Organization_Name   variable length character string maximum length 50
domain    Email                         variable length character string maximum length 50
domain    Account_Secure_Problem        variable length character string maximum length 40
domain    Secure_Problem_Answer         variable length character string maximum length 40
domain    Certification_for_Registration  Varbinary(MAX)
domain    Organizer_Avatar              Varbinary(MAX)
domain    Organizer_User_State          Integer
domain    Organizer_Registration_State  Integer
domain    User_ID Integer               Integer

Organizer_Information(
orgID                  Organizer_Account_ID          NOT NULL,
userName               Organizer_User_Name           NOT NULL,
orgPassword            Organizer_Account_Password    NOT NULL,
organizationName       Organizer_Organization_Name   NOT NULL,
email                  Email                         NOT NULL,
secureProblem          Account_Secure_Problem        NOT NULL,
secureAnswer           Secure_Problem_Answer         NOT NULL,
```

```
certification_graph     Certification_for_Registration     NOT NULL,
orgAvatar               Organizer_Avatar,
userState               Organizer_User_State               NOT NULL,
registerState           Organizer_Registration_State       NOT NULL,
userID                  User_ID                            NOT NULL

Primary Key     orgID
Foreign Key     userID     References User (userID) ON UPDATE CASCADE ON DELETE CASCADE)
```

```
domain     Student_ID                  Integer maximum length 9
domain     Student_Name                variable length character string maximum length 30
domain     Student_Registration_Status Integer

student_already(
stuID              Student_ID                  NOT NULL,
stuName            Student_Name                NOT NULL,
registerStatus     Student_Registration_Status NOT NULL

Primary Key    stuID )
```

```
domain     Student_ID                  Integer Integer maximum length 9
domain     Student_Name                variable length character string maximum length 30
domain     Student_User_Nickname       variable length character string maximum length 20
domain     Student_Account_Password    variable length character string maximum length 16
domain     Telephone_Number            variable length character string maximum length 11
domain     Email                       variable length character string maximum length 50
domain     Account_Secure_Problem      variable length character string maximum length 40
domain     Secure_Problem_Answer       variable length character string maximum length 40
domain     Total_Activity_Points       Integer maximum length 4
domain     Student_Avatar              Varbinary(MAX)
domain     Student_User_State          Integer
domain     User_ID                     Integer

student_Information (
stuID              Student_ID                  NOT NULL,
stuName            Student_Name                NOT NULL,
nickname           Student_User_Nickname       NOT NULL,
stuPassword        Student_Account_Password    NOT NULL,
telephoneNumber    Telephone_Number            NOT NULL,
email              Email                       NOT NULL,
secureProblem      Account_Secure_Problem      NOT NULL,
secureAnswer       Secure_Problem_Answer       NOT NULL,
totalPoints        Total_Activity_Points       NOT NULL,
stuAvatar          Student_Avatar
userState          Student_User_State          NOT NULL,
```

```
userID                  User_ID                 NOT NULL

Primary Key     stuID
Foreign Key     userID  references User(userID) ON UPDATE CASCADE ON DELETE CASCADE )
```

```
domain      Student_Activity_Involved_Record_ID   Auto-Incrementing Integer
domain      Student_ID                            Integer maximum length 9
domain      Activity_ID                           Integer maximum length 5
domain      Activity_Application_Status           Integer
domain      Application_Time                      Date
domain      Activity_Finish_Status                Integer

StuInvolvedActivity (
involvedRecordID    Student_Activity_Involved_Record_ID   NOT NULL,
stuID               Student_ID                            NOT NULL,
activityID          Activity_ID                           NOT NULL,
applyStatus         Activity_Application_Status           NOT NULL,
applyTime           Application_Time                      NOT NULL,
finishStatus        Activity_Finish_Status                NOT NULL,

Primary Key     involvedRecordID
Foreign Key     stuID references student_Information(stuID) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key     activityID references Event(activityID) ON UPDATE CASCADE ON DELETE NO ACTION )
```

```
domain      Activity_ID                 Auto-Incrementing Integer
domain      Organizer_ID                Integer
domain      Activity_Name               variable length character string maximum length 50
domain      Activity_Population         Integer
domain      Activity_Date               Date
domain      Activity_Address            variable length character string maximum length 100
domain      Activity_Content            variable length character string maximum length 500
domain      Activity_Service_Domain     variable length character string maximum length 30
domain      Activity_Service_Recipient  variable length character string maximum length 30
domain      Activity_Service_Location   variable length character string maximum length 30
domain      Activity_Graph              Varbinary(MAX)
domain      Activity_Status             Integer
domain      Number_of_Involved_Student  Integer
domain      Activity_Point              Integer
domain      Activity_Duration           Integer
domain      Address_Iatitude            Decimal
domain      Address_longtitude          Decimal

Event (
activityID          Activity_ID             NOT NULL,
orgID               Organizer_ID            NOT NULL,
```

activityName	Activity_Name	NOT NULL,
activityPopulation	Activity_Population	NOT NULL,
activityDate	Activity_Date	NOT NULL,
activityAddress	Activity_Address	NOT NULL,
activityContent	Activity_Content	NOT NULL,
serviceDomain	Activity_Service_Domain	NOT NULL,
serviceRecipient	Activity_Service_Recipient	NOT NULL,
serviceLocation	Activity_Service_Location	NOT NULL,
activityGraph	Activity_Graph	
activityStatus	Activity_Status	NOT NULL,
numOfInvolved	Number_of_Involved_Student	NOT NULL,
activityPoint	Activity_Point	NOT NULL,
activityDuration	Activity_Duration	NOT NULL,
address_Iatitude	Address_Iatitude	NOT NULL,
address_longtitude	Address_longtitude	NOT NULL

Primary Key activityID
Foreign Key orgID references organizer_information(orgID) ON UPDATE CASCADE ON DELETE NO ACTION)

domain User_ID Auto-Incrementing Integer
User (
userID User_ID NOT NULL
Primary Key userID)

domain	Topic_Post_ID	Auto-Incrementing Integer
domain	User_ID	Integer
domain	Topic_Title	variable length character string maximum length 50
domain	Topic_Content	variable length character string maximum length 500
domain	Topic_Graph	Varbinary(MAX)
domain	Topic_Likes_Number	Integer
domain	Topic_Create_Time	Datetime
domain	Video_Path	variable length character string maximum length 200

Topic Post (

postID	Topic_Post_ID	NOT NULL,
userID	User_ID	NOT NULL,
topicTitle	Topic_Title	NOT NULL,
topicContent	Topic_Content	NOT NULL,
graph	Topic_Graph	
likes_number	Topic_Likes_Number	NOT NULL,
createTime	Topic_Create_Time	NOT NULL,
videoPath	Video_Path	NOT NULL

Primary Key postID
Foreign Key userID references User(userID) ON UPDATE CASCADE ON DELETE NO ACTION)

domain	Comment_ID	Auto-Incrementing Integer
domain	User_ID	Integer
domain	Topic_Post_ID	Integer
domain	Comment_Content	variable length character string maximum length 100
domain	Comment_Created_Time	Datetime

Comment (

commentID	Comment_ID	NOT NULL,
userID	User_ID	NOT NULL,
postID	Topic_Post_ID	NOT NULL,
commentContent	Comment_Content	NOT NULL,
createdTime	Comment_Created_Time	NOT NULL

Primary Key commentID
Foreign Key userID references User(userID) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key postID references Topic_Post(posyID) ON UPDATE CASCADE ON DELETE NO ACTION)

domain	Topic_Post_ID	Integer
domain	User_ID	Integer
domain	Likes_Time	Datetime

UserLikes (

postID	Topic_Post_ID	NOT NULL,
userID	User_ID	NOT NULL,
likeTime	Likes_Time	NOT NULL

Composite Primary Key (postID, userID)

2.2.5 Business Policies & Rules

2.2.5.1 Business Policies

Administrator Related Policies
1. Administrators have to use the predefined account stored in the database to log into the software.
2. Administrators have the capability to manage information of the school-wide students.
3. Administrators are responsible for auditing organizer registration applications.
4. Administrators can manage registered volunteers and organizers' accounts, including viewing, searching for user information, and performing actions such as locking and unlocking user account.

Organizer Related Polices
1. Organizers must register an account by providing basic information and uploading proof of organizational qualifications.
2. After registration, the account requires approval from a software administrator. The account is activated upon successful review and approval.

3.	There are specific character limits for usernames and organization/company names, and certain characters are not permitted.
4.	Organizers must create passwords that meet the particular security requirements.
5.	Organizers have the ability to post new events, view and sort existing events, and review volunteer applications.
6.	Organizers can participate in community discussions by posting on forums, commenting, and liking posts.
7.	Organizers are allowed to update their personal details, however, changes to usernames, email addresses, and qualification documents are not permitted.

Student Related	
1.	Student users have to register a student account to use the volunteer software.
2.	Students are required to provide their real names and student IDs upon registration, which will be cross-verified with the school's student information database.
3.	Students must create passwords that satisfy the specific security requirements.
4.	Students can browse, search, sign up, and view detailed information about activities.
5.	Students can post, comment, like posts, and engage in discussions on the forum.
6.	Students can edit their personal information, with the exception of their full names and student IDs.
7.	Students can view historical forum posts and track the activities they have participated in.
8.	Students can access a leaderboard that ranks volunteers based on their activity points earned.

2.2.5.2 Business Rules

Administrator:

Administrator Account Usage:
Rule: Administrators must log in using an account provided by the software entity.
Database Constraint: The fields adminID and adminPassword in the admin_information table should be set to not allow updates.

School-wide Student Information Management:
Rule: Administrators are allowed to directly interact with the database to manage information of students across the entire institution.
Database Constraint: The student_already table should be structured to provide direct Create, Read, Update, and Delete (CRUD) operations for administrators.

Volunteer Information Management:
Rule: Administrators can view, sort, and search the information of registered volunteer students.
Database Constraint: Indexes should be implemented on the student_information table to optimize operations for searching and sorting.

Student Account Lock/Unlock:
Rule: Administrators have the authority to lock or unlock student accounts as necessary.
Database Constraint: A field should be added to the student_information table to indicate the lock status of an account.

Organizer Registration Audit:
Rule: Administrators need to audit the qualification of organizers, approve or reject their registration applications.
Database Constraint: A foreign key constraint should be added to the organizer_information table, referencing the organizerID to link to their qualification proof image.

Organizer Information Management:
Rule: Administrators can view, sort, and search the information of registered organizers.

Database Constraint: Indexes should be applied to the organizer_information table to enhance the efficiency of search and sort operations.

Organizer Account Lock/Unlock:
Rule: Administrators can lock or unlock organizer accounts.
Database Constraint: A field should be included in the organizer_information table to denote the account lock status.

Organizer

Username and Organization Name Format:
Rule: The username must not exceed 15 characters, and the company/organization name must not exceed 50 characters.
Database Constraint: Enforce length and character restrictions on the userName and organizationName fields in the organizer_information table.

Password Complexity:
Rule: Passwords must be between 8 to 16 characters, include at least one uppercase letter, and must not contain the specific characters "=" or "_".
Database Constraint: Implement regular expression validation on the orgPassword field in the organizer_information table.

Password Reset:
Rule: Organizers can reset their passwords through security question.
Database Constraint: Provide a secure password reset process, and store the security question and answer in the database.

Qualification Proof Upload:
Rule: Organizers must upload a screenshot of their company/organization's certification.
Database Constraint: Validate the file format and size through the front-end form during the registration process, and store the uploaded image in the database on the back end.

Activity Inquiry and Sorting:
Rule: Organizers can search and sort activities by activity name, starting date, or status.
Database Constraint: Implement indexing on the activity name, date, and status fields in the Event table to optimize search and sorting operations.

Activity Publication:
Rule: The publication of an activity must include a title, content, number of participants, and activity points.
Database Constraint: Enforce NOT NULL constraints on the relevant activity fields in the Event table, with the exception of Activity_Graph, which can be NULL.

Activity Status Management:
Rule: Activity status is categorized as "Not Started," "In Progress," and "Ended," with automatic updates based on the current system time.
Database Constraint: Add a field to the Event table to store the activity status and implement the logic for status transitions.

Volunteer Application Audit:
Rule: Organizers can review and decide whether to accept volunteer applications.
Database Constraint: Add a field to the StuInvolvedActivity table to indicate the audit status of the application.

Forum Post Management:
Rule: Organizers can post and delete their own forum posts and edit their own comments.

Database Constraint: Add foreign key constraints in the Topic_Post and Comment tables to associate with the organizer's account and enforce permission checks.

Personal Information Update:
Rule: Organizers can update their personal information, except for the username, email, and qualification proof.
Database Constraint: Implement update permission controls on the updatable fields in the organizer_information table.

Historical Post Management:
Rule: Organizers can view and manage their own historical posts.
Database Constraint: Add a field to the Topic_Post and Comment tables to identify the organizer associated with the post or comment.

Student

Student Account Registration:
Rule: The username must not exceed 20 characters and must not contain the characters "=" or "_".
Database Constraint: Enforce length and character restrictions on the userName field in the student_information table.

Student Information Verification:
Rule: The system automatically verifies the correctness of the student's name and ID upon account registration
Database Constraint: Implement a uniqueness constraint on the student_information table to ensure each student's ID is unique; cross-verify with the school's student information database to ensure the authenticity of the identity.

Password Requirements:
Rule: Passwords must be between 8 to 16 characters, include at least one uppercase letter, and must not contain the characters "=" or "_".
Database Constraint: Apply regular expression checks on the stuPassword field in the student_information table.

Activity Inquiry and Sorting:
Rule: Students can search and sort activities by name, date, or status.
Database Constraint: Implement indexing on the activity name, date, and status fields in the Event table to optimize search and sorting operations.

Activity Registration:
Rule: Students can only sign up for activities with the status "Not Started."
Database Constraint: Add a field to the Event table to store the activity status and implement status transition logic based on system time.

Forum Post Management:
Rule: Students can post new topics, like and comment on topics. Each user can only like a topic once.
Database Constraint: Add fields in the Topic_Post and Comment tables to record likes and comments, and enforce integrity constraints.

Personal Information Modification:
Rule: Students can modify all personal information except for their real name and student ID.
Database Constraint: Allow updates to fields other than the name and ID in the student_information table.

Historical Post Management:
Rule: Students can view, modify, or delete their own forum posts and comments.

Database Constraint: Allow students to update their own comment content in the Comment table.

Activity Participation Tracking:
Rule: Students can view their historical records and current status of participation in activities.
Database Constraint: Record student activity participation information in the StuInvolvedActivity table.

Leaderboard Viewing:
Rule: Students can view the leaderboard of the top 100 volunteers based on activity points.
Database Constraint: Implement a leaderboard view or query to rank volunteers according to points.

2.2.6 Transaction Table

A **Administrator Login:** Administrator accesses the software using valid credentials.

B **School-wide Student Information Management:** This involves actions such as retrieving student data, adding new student records, and deleting existing student records from the system.

C **Registered Student Information Management:** Include reading student information and managing the status of student accounts.

D **Organizer Registration Audit:** The process of reviewing and approving new organizer registration.

E **Registered Organizer Information Management:** Involve reading organizer information and managing the status of organizer accounts, including the ability to lock or unlock these account.

Transaction/ Table	A				B				C				D				E			
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D
admin_information		X																		
organizer_information													X	X	X	X	X	X	X	X
student_already					X	X	X	X												
student_Information									X	X	X	X								
StuInvolvedActivity																				
Event																				
User									X								X			
Topic_Post																				
Comment																				
UserLikes																				

I insert R read U update D delete

F **Student Registration:** The process where students create accounts.

G **Student Login:** Allow students to access their accounts by entering their credentials.

H **Student Password Recovery:** A feature that enables students to recover their passwords in case they forget them, involving identity verification and the setup of a new password.

I **Student Personal Information Modification:** Permit students to update their personal details to keep their records accurate and current.

J **Activity Browsing:** Allow students to explore available volunteer activities, view activity details, and select those they are interested in participating in.

Transaction/ Table	F				G				H				I				J			
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D

admin_information																				
organizer_information																				
student_already		X																		
student_Information	X	X				X				X	X			X	X					
StuInvolvedActivity																				
Event																		X		
User	X																			
Topic_Post																				
Comment																				
UserLikes																				

I insert R read U update D delete

K **Activity Sign up: S**tudent sign up for the activities of their interest after browsing through the available options.

L **Organizer Registration:** Allow new organizers to register an account on the software by submitting their personal and organizational details, including organizational certification.

M **Organizer Login:** Enable organizers to log in to their accounts using their credentials, providing access to manage activities, review volunteer applications, and engage in the forum.

N **Organizer Password Recovery:** Offer organizers a method to recover their passwords if forgotten, involving verification steps and the establishment of a new password.

O **Organizer Personal Information Modification:** Permit organizers to update their personal information.

Transaction/ Table	K				L				M				N				O			
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D
admin_information																				
organizer_information					X	X	X			X				X	X			X	X	
student_already																				
student_Information		X																		
StuInvolvedActivity	X																			
Event		X																		
User					X															
Topic_Post																				
Comment																				
UserLikes																				

I insert R read U update D delete

P **Activity Browsing:** Organizers review available activities and access details to engage with the community and make informed decisions about participation.

Q **Activity Post:** Organizers create and publish new activities for volunteers to find and sign up for, including providing essential information such as activity description, requirements, and timing.

R **Activity Information Modification:** Allow organizers to make updates to the details of existing activities to ensure information remains accurate and relevant.

S **Volunteer Registration Audit:** Organizers review and approve volunteer sign-ups for their activities, ensuring that only eligible and suitable volunteers participate.

T **Activity Completion Audit:** Organizers assess the completion status of an activity, evaluate outcomes, and provide feedback to participants upon the activity's conclusion.

Transaction/	P	Q	R	S	T

Table																				
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D
admin_information																				
organizer_information						X				X										
student_already																				
student_Information														X				X		
StuInvolvedActivity														X	X			X	X	
Event		X			X					X	X			X				X		
User																				
Topic_Post																				
Comment																				
UserLikes																				

I insert R read U update D delete

U **Forum Browsing:** The action of reviewing existing discussions and threads within the forum, allowing users to stay informed about ongoing conversations and community interests.

V **Forum Post Creation, Deletion, and Modification:** Organizers have the capability to start new discussions by posting topics, as well as to delete or edit their own posts as needed.

W **Forum Commenting and Comment Modification:** Enable users to engage with forum posts by adding comments, and to modify or update their own comments after they have been posted.

X **Forum Liking:** Allow users to express their approval or support for forum posts by liking them, which can help highlight valuable contributions and foster a sense of community.

Y **Leaderboard Viewing:** Students can view the leaderboard to see the rankings of volunteers based on activity points.

Transaction/ Table	U				V				W				X				Y			
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D
admin_information																				
organizer_information		X				X				X				X						
student_already																				
student_Information		X				X				X				X				X		
StuInvolvedActivity																				
Event																				
User		X				X				X				X						
Topic_Post		X			X	X	X	X												
Comment		X							X	X	X	X								
UserLikes		X											X	X		X				

I insert R read U update D delete

2.3 Interface Design

2.3.1 Internal Interface Design

In our software solutions, there is often a need to exchange information between different projects. To simplify cross-project data sharing and reduce dependence on external storage, we designed the SharedData, Activity_Shared, and Topic classes to provide a lightweight, thread-safe mechanism for global data sharing.

a. All Projects Sharing

```
…
using System.Threading.Tasks;
```

```
namespace All_Project
{
    public class SharedData
    {
        public static int stuID    { get; set; }
        public    static int orgID    { get; set; }
        public static string userName { get; set; }
    }
}
```

SharedData: As a static class, it stores the basic information of the current user, such as: Student ID, organizer ID and user name. It allows any part of the software to directly read or set shared user information without going through the middle layer, ensuring the thread safety of atomic operations on properties.

b. Activity Relevant Projects Sharing

```
…
using System.Threading.Tasks;
namespace Activity
{
    public class Activity_Shared
    {
        public string Nickname { get; set; }
        public int stuID { get; set; }
        public int ActivityID { get; set; }
        public byte[] ImageData { get; set; }
        public string ActivityName { get; set; }
        public string ActivityDate { get; set; }
        ……
    }
}
```

Activity_Shared: This class encapsulates detailed information about the activities in the system, such as: Activity name, participant-student information, and status. It is used to create and represent activities within the system, which can then be displayed to users through the activity management module.

c. Forum Relevant Projects Sharing

```
…
using System.Threading.Tasks;
namespace Group_template
{
    public class Topic
    {
        public string postID { get; set; }
        public int userID { get; set; }
        public byte[] ImageData { get; set; }
        public int Likes { get; set; }
        ……
    }
}
```

Topic: Represents a topic post in a system's forum or discussion area and contains information about the post, including the post ID, user ID, content, and any associated media. Instances of this class are used to create and

display forum topics so that users can interact with the content.

2.3.2 External Interface Design

2.3.2.1 AI model

In the volunteer activity management field, there is often a challenge in fully considering the personal characteristics of volunteers and the specific needs of the activities. Organizers face the challenge of efficiently and accurately recruiting volunteers that match event needs. Traditional matching methods often rely on manual review and judgment based on experience, which is time-consuming and labor-intensive. To improve the accuracy and efficiency of the matching process, we decided to integrate advanced artificial intelligence models to automate the process. The AI model based on machine learning algorithms will analyze various data points such as volunteer profiles, past participation, skills, and activity requirements to suggest the most suitable candidates. This integration is expected to streamline the recruitment process, reduce organizer workload, and improve the overall quality of volunteer matching by minimizing the impact of subjective bias.

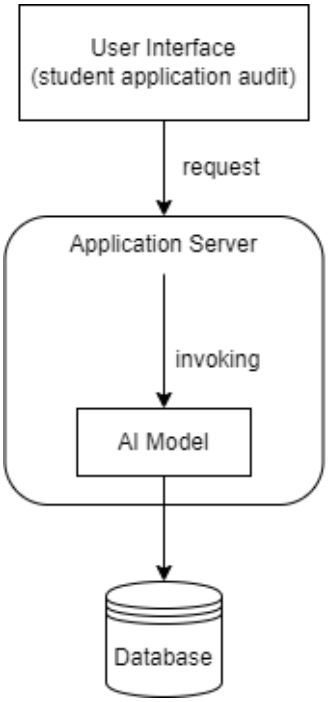


Figure 3 Relationship Between AI model and System

Model Training Method:

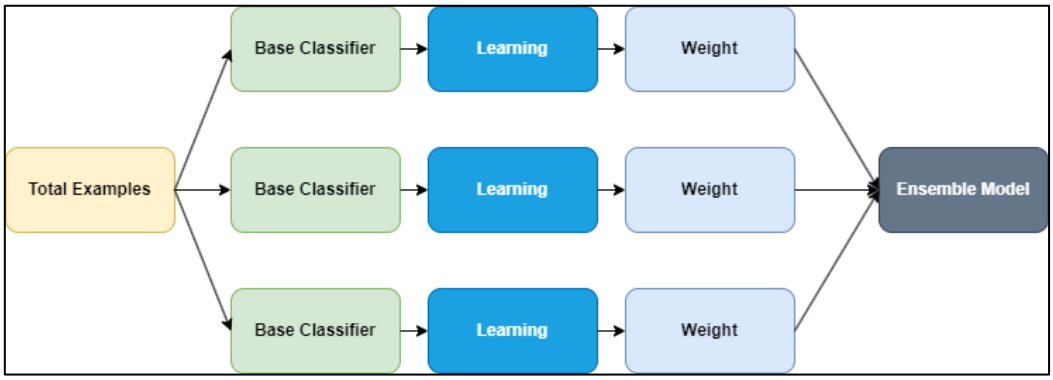


Figure 4 Structure of Decision Tree

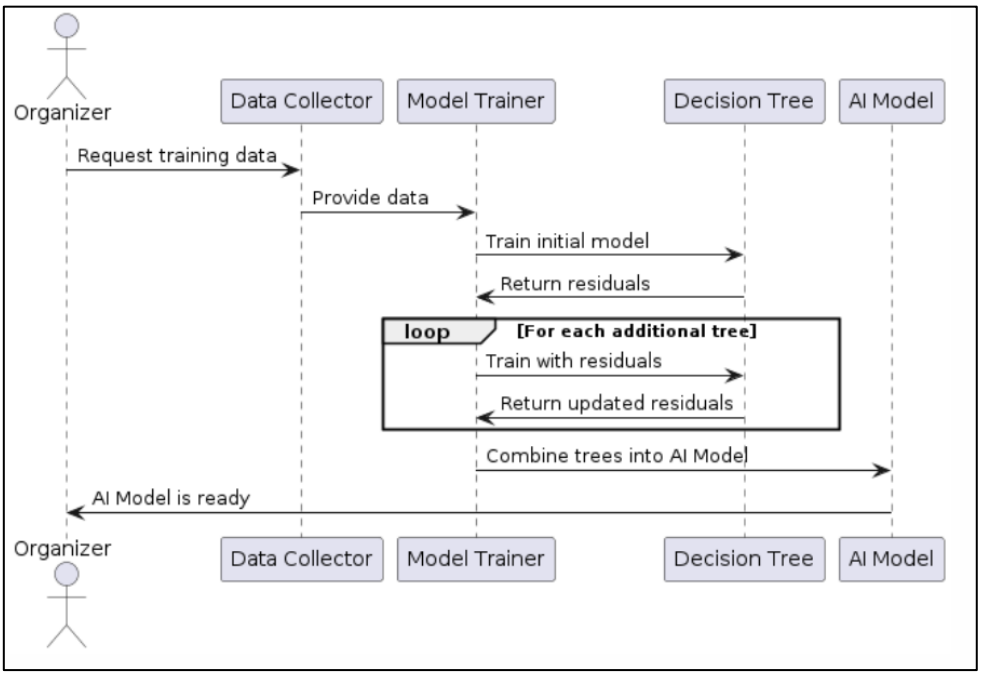


Figure 5 Model Training Sequence Diagram

We chose an AI model based on Gradient Boosting Decision Trees (GBDT), which has significant advantages in handling classification and regression problems, especially when dealing with complex nonlinear relationships. This model uses an ensemble learning method to improve the accuracy of matching calculation by combining the prediction results of multiple classification and regression trees (CART). In terms of model training, we have utilized 200 CART regression trees, with each tree trained on the prediction residuals of the preceding trees. When calculating the matching degree, the model analyzes the historical activity information and total points earned by volunteers, outputting a matching score. During the report generation phase, the system produces an individual report that includes the matching score and graphical statistics.

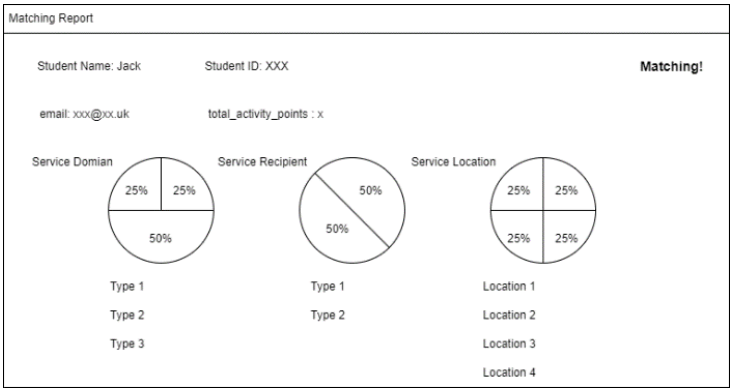


Figure 6 Matching Report View

2.3.2.2 Google Maps API

In order to provide a more intuitive and convenient experience in selecting activity addresses for organizers, we combine the Google Maps API with the organizer's event publishing interface.

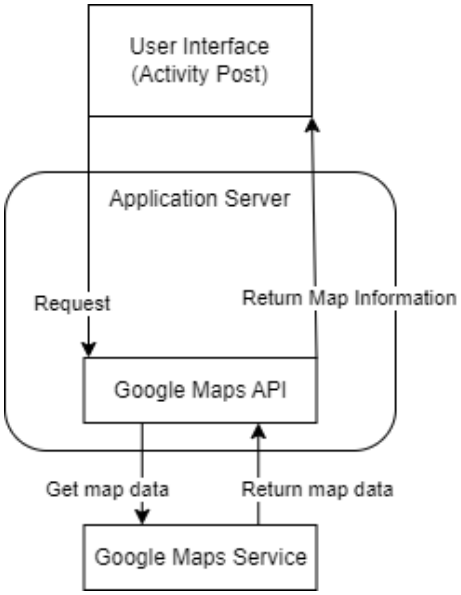


Figure 7 Relationship Between Map API and System

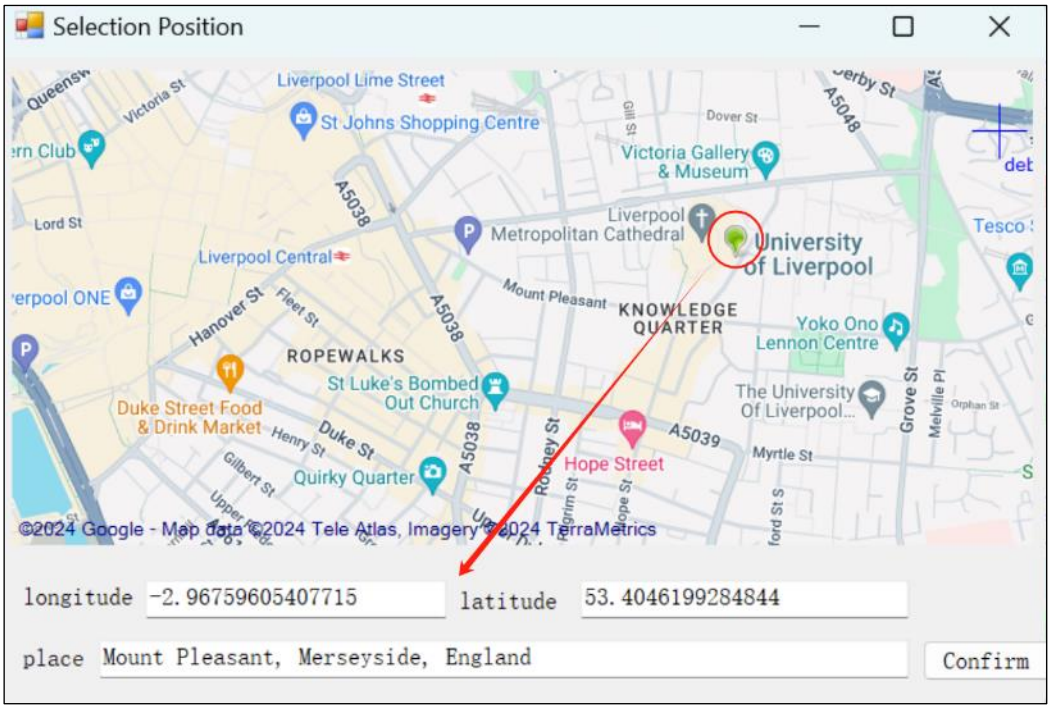


Figure 8 Maps Function View

Technical Implementation:

First, set the map provider of the map control to ensure the user interface can load and display the Google Maps. Upon the user selects an activity address on the map, the FromLocalToLatLng method in the GMap.NET library is employed to accurately convert the relative position of the selected point into actual latitude and longitude coordinates. Then we make a call to the Google Maps API to accept the transformed latitude and longitude parameters of the selected point to retrieve detailed address information.

Pseudocode:

```
Pseudocode
// Set the map control provider to Google Maps
MapControl.Provider = GoogleMapsProvider();
// Allow the user to select a location on the map
var location = UserSelectLocationOnMap();
// Convert the selected location's coordinates to latitude and longitude
```

```
// and call the Google Maps API to get the address information
var addressInfo = GoogleMapsAPI.GetAddressInfo(location.Latitude, location.Longitude);
// Automatically fill the retrieved address information into the form
AutoFillAddressForm(addressInfo);
```

Tips: This pseudocode serves as a high-level representation of the steps involved in the technical implementation and would need to be translated into a specific programming language for execution.

2.4 Process Design

2.4.1 UML use case & functional description

2.4.1.1 User Login Use-case Diagram

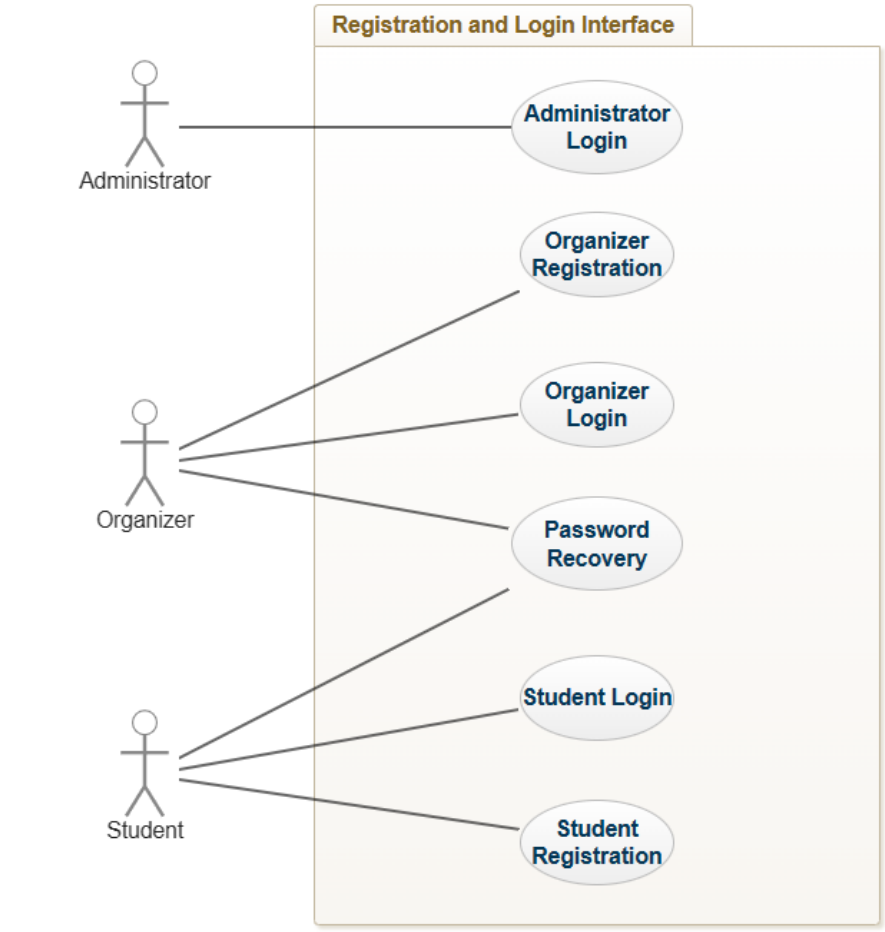


Figure 9 Registration and Login

Functional Description of Registration and Login Interface

UC1 Administrator Login	
Description	The Administrator uses a secure login interface to access the system management features.
Goals	To authenticate and access the administrative dashboard.
Actions	Enter ID and password, submit login request.
Triggers	The need to manage system settings, user accounts, or perform organizer audits.
Information used	Administrator credentials, including administratorID and password.
Information produced	Access to administrative dashboard.

UC2 Organizer Registration	
Description	New organizers register for the software to create and manage volunteer activities.
Goals	To create an organizer account and start posting activities.
Actions	Fill out registration form, submit personal and organizational details, organizational certification.
Triggers	The desire to engage volunteers for activities or causes.
Information used	Personal information, organizational details, organizational certification.
Information produced	New organizer account, pending approval status.

UC3 Organizer Login	
Description	Registered organizers log in to manage their activities and interact with the software
Goals	To manage posted activities, view volunteer applications, post updates and engage in forum.
Actions	Enter credentials, submit login request, navigate to activity management dashboard
Triggers	The need to update activity information, review volunteer applications, or engage with the forum.
Information used	Organizer credentials(including username, password) and activity data.
Information produced	Updated activity information, logs of volunteer interactions

UC4 Student Registration	
Description	New volunteers create an account on the platform to start engaging in volunteer activities.
Goals	To register as a volunteer and express interest in activities.
Actions	Provide personal details, set a username and password, agree to terms of service.
Triggers	The motivation to volunteer, the availability of a sign-up option.
Information used	Personal information(must input full name and stuID), username and password.
Information produced	New student accoun.

UC5 Student Login	
Description	Students log in to explore volunteer opportunities, sign up for activities, and manage their profiles.
Goals	To participate in volunteer activities and track personal involvement.
Actions	Enter credentials submit login request, browse posted activities.
Triggers	The interest in volunteering, the need to sign up for a volunteer activity.
Information used	Student credentials (including stuID, password), activity data.
Information produced	Student profile updates, activity sign-up records.

UC6 Password Recovery	
Description	Users recover their passwords in case they forget their login details.
Goals	To regain access to their account.
Actions	Request password recovery, verify identity, set new password.
Triggers	Forgotten password.
Information used	For organizer user: email, username, security questions For student user: stuID, email, security questions.
Information produced	Reset password link, new password setup.

2.4.1.2 Administrator Interface Use-case Diagram

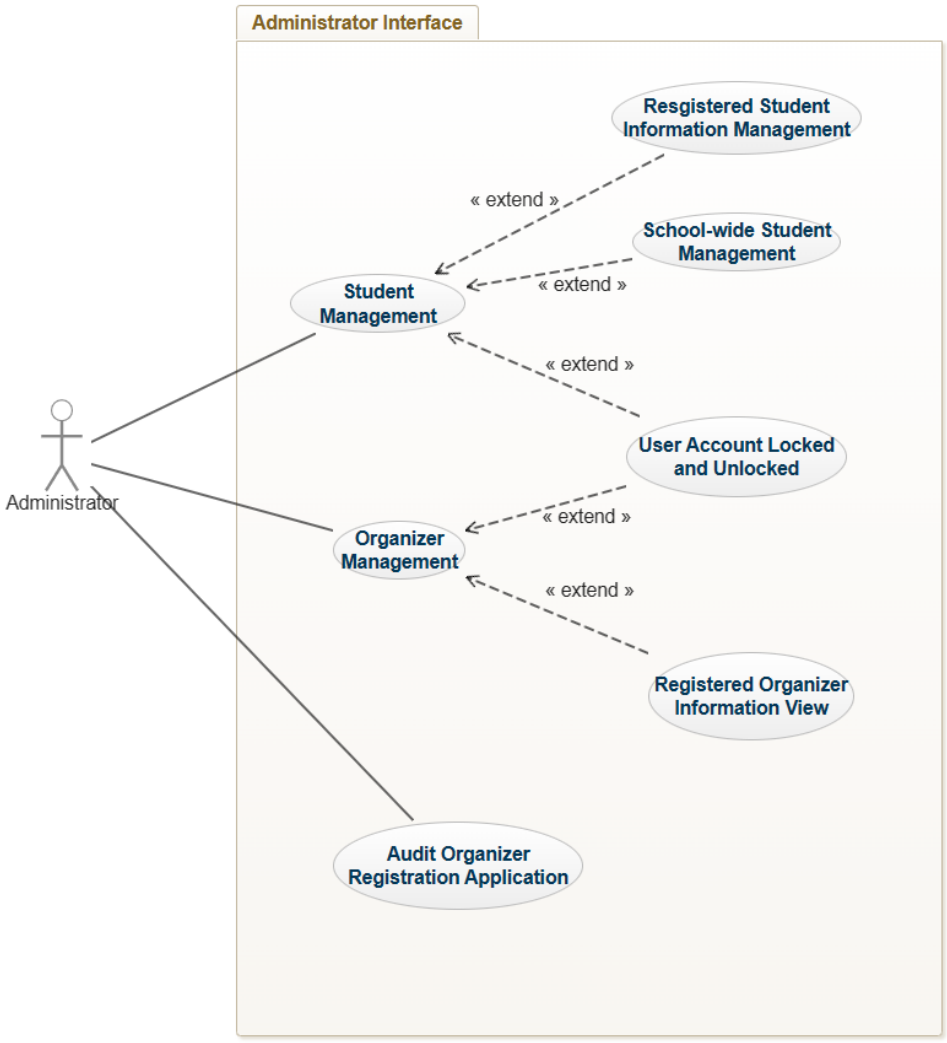


Figure 10 Administrator Interface

UC7 Student Management	
Description	Enables administrators to manage student records, including student information, account status.
Goals	To maintain accurate and current student data for administrative and analytical purposes.
Actions	Add or delete new student , update student information, view student details information, locked or unlocked accounts.
Triggers	Student enrollment, changes in student status, or administrative audits.
Information used	Student personal details, account status.
Information produced	Updated student database, administrative actions.

UC8 Registered Student Information Management	
Description	Allows administrators to manage the personal information of registered students within the system.
Goals	To maintain accurate and up-to-date student records for various administrative tasks.
Actions	View, update student information as necessary.
Triggers	The need to manage student data due to registration, updates, or account issues.
Information used	Student registration data.

Information produced	Updated student records, confirmation of actions taken.

UC9 School-wide Student Management	
Description	Enables administrators to oversee student information at a university-wide level.
Goals	To centralize student data management for consistency and accessibility.
Actions	Access, review, and manage student information across all departments or programs.
Triggers	Administrative requirements for student data at a systemic level.
Information used	School-wide student data.
Information produced	Student management insights

UC10 User Account Locked and Unlocked	
Description	Provides administrators with the ability to lock or unlock user accounts based on predefined criteria or circumstances.
Goals	To ensure account security and compliance with institutional policies.
Actions	Lock or unlock accounts, notify users.
Triggers	Account security breaches, policy violations, or user requests.
Information used	User account status.
Information produced	Record of account status changes and notifications.

UC11 Organizer Management	
Description	Allows administrators to oversee and manage the organizers within the system.
Goals	To ensure that organizers meet the criteria for registering and to manage their information effectively.
Actions	Review organizer registration applications, approve or deny registration, and provide feedback or support as needed.
Triggers	Submission of new organizer applications, organizer requests for support.
Information used	Organizer application data, organizer registration status.
Information produced	Approved organizer list, organizer information update.

UC12 Organizer Information View	
Description	Enables administrators to view detailed information about organizers registered within the system.
Goals	To monitor and assess the information of organizers.
Actions	Access and review organizer information.
Triggers	The need to communicate with them or check their certification.
Information used	Organizer profile data.
Information produced	Overview of organizer information.

UC13 Audit Organizer Registration Application	
Description	Permits administrators to review and approve organizer registration applications to ensure they meet the required standards
Goals	To maintain quality control over organizer registrations.
Actions	Review applications, verify information, and approve or reject registrations.
Triggers	New organizer registration submissions.
Information used	Registration applications, qualification criteria.

Information produced	Decision on registration, communication to applicant, and updated registration status.

2.4.1.3 Organizer interface Use-case Diagram

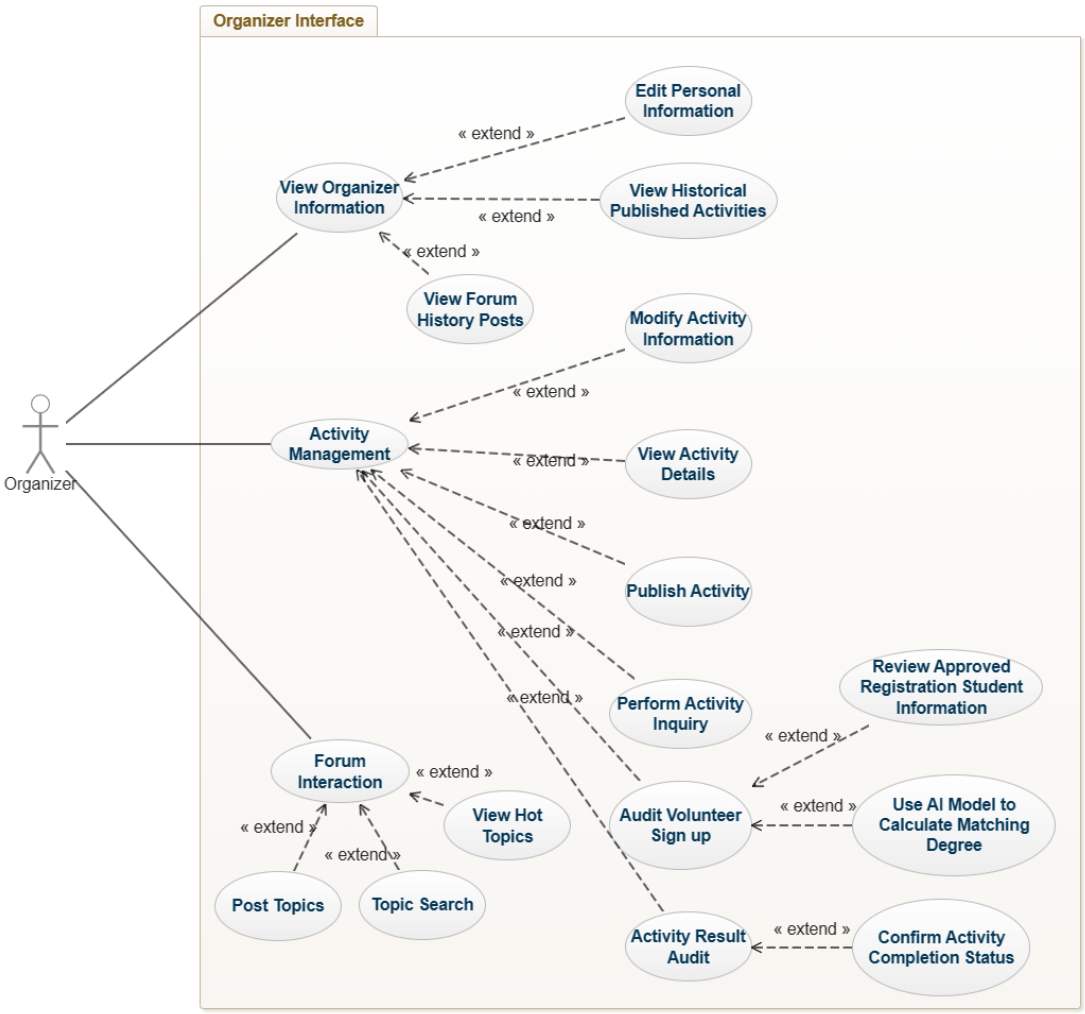


Figure 11 Organizer Interface

UC14 View Organizer Information	
Description	Enables organizers to view and manage their personal details, check their historical activities, on the platform.
Goals	To ensure personal details are accurate and up-to-date
Actions	Display personal information and provide options to edit
Triggers	The need for organizers to verify or update their information
Information used	Organizer's personal details.
Information produced	Updated personal information record.

UC15 Edit Personal Information	
Description	Permits organizers to update their personal information, such as contact details or profile preferences.
Goals	To maintain accurate and current personal records.
Actions	Modify personal details and save changes.
Triggers	Changes in the organizer's personal information.

Information used	Current personal information, new data provided by the student.
Information produced	Updated personal profile in the system.

UC16 View Historical Published Activities	
Description	Enables organizers to view a history of their past activities and outcomes.
Goals	To review previous activities for analysis or reporting
Actions	Access the historical data section and browse past activities.
Triggers	The need to evaluate past success or gather statistics.
Information used	Organizer credentials, date range filters.
Information produced	List of past activities with their details.

UC17 View Forum Historical Post	
Description	Permits organizers to view their history of forum posts and interactions
Goals	To review past contributions to forum discussions
Actions	Navigate to the forum history section and view past posts.
Triggers	Interest in past forum activities or need to reference previous posts
Information used	Organizer's forum activity data.
Information produced	List of the organizer's forum posts and comments.

UC18 Activity Management	
Description	Enables organizers to manage activities from creation to completion, including posting, editing, auditing student sign up, confirming student's activity finish state and monitoring the status of activities.
Goals	To efficiently handle all aspects of activity lifecycle within the system.
Actions	Create new activities, edit existing activities, filter and search activities, audit student activity sign up, confirm student finish state.
Triggers	The need to organize, publicize, or manage student sign up or charity events.
Information used	Activity details, student sign-up information.
Information produced	A comprehensive list or dashboard view of activities with their current status and details, confirmation messages for creation, editing, or deletion of activities, and student relevant status update.

UC19 Perform Activity Query	
Description	Enables organizers to search and inquire about activities that they have published.
Goals	To find and assess activities.
Actions	Input search criteria, review the details.
Triggers	The need to search their activities.
Information used	Search parameters defined by the organizer, including activity name, date, class.
Information produced	A list of activities that meet the search or sort criteria.

UC20 Publish Activity	
Description	Provides the organizer with the capability to create and publish new activities.
Goals	To attract volunteers and initiate community engagement.

Actions	Create a new activity, submit for review, and publish upon approval.
Triggers	The need to introduce new activities to the community.
Information used	Activity information.
Information produced	A published activity available for volunteer sign-up.

UC21 Modify Activity Information	
Description	Permits the organizer to make changes to the details of an existing activity.
Goals	To keep activity information accurate and up-to-date.
Actions	Edit activity details and save the modifications.
Triggers	New information becomes available or errors are found.
Information used	Activity Information.
Information produced	Updated activity details in the system.

UC22 Review Approved Volunteer Sign up	
Description	Enables the organizer to view and manage the list of approved volunteers for an activity
Goals	To manage and monitor volunteer participation
Actions	Review the list of approved volunteers and perform necessary actions like notifications.
Triggers	The completion of the volunteer registration review process
Information used	
Information produced	A curated list of volunteers for the activity.

UC23 Audit Volunteer Sign Up	
Description	Enables organizers to review and approve volunteer sign-ups for their activities.
Goals	To ensure that only eligible and suitable volunteers are participating in the activity
Actions	Access the list of volunteer sign-ups, evaluate each applicant using AI calculating model, and make approval decisions.
Triggers	Receipt of volunteer sign-ups for an activity.
Information used	Volunteer application data, activity requirements.
Information produced	Approved list of volunteers, notification to applicants.

UC24 Use AI Model to Calculate Matching Degree	
Description	Integrates an AI model to assist in evaluating the suitability of volunteers for an activity.
Goals	To streamline the volunteer selection process and enhance match quality.
Actions	Input volunteer and activity data into the AI model and review the output.
Triggers	The need to refine an activity based on feedback or to innovate for better engagement.
Information used	Activity details, feedback, available modules and tools.
Information produced	Matching Report.

UC25 Activity Result Audit	
Description	Allows organizers to audit the results and outcomes of an activity after its completion.
Goals	To assess the success of the activity and gather feedback for future improvements.

Actions	Participant feedback.
Triggers	Completion of an activity.
Information used	Activity outcome data.
Information produced	Audit student finish state.

UC26 Confirm Activity Completion Status	
Description	Lets the organizer mark the student finish situation.
Goals	To officially evaluate an activity finish situation.
Actions	Confirm the activity completion and update student activity status.
Triggers	The end of the activity period.
Information used	Student information, activity information.
Information produced	A record of the activity's completion status and student outcomes.

UC27 Forum Interaction	
Description	Enables organizers to participate in forum discussions related to their activities or interests.
Goals	To engage with the community, share updates, and gather insights.
Actions	Post updates, reply to messages, and create new threads in the forum.
Triggers	Desire to communicate with participants or the community.
Information used	Forum interface, organizer's messages.
Information produced	Forum posts and threads initiated by the organizer.

UC28 Post Topic	
Description	Allows organizers to start new discussions by posting topics in the forum.
Goals	To initiate conversations on subjects of interest or to share updates about activities.
Actions	Compose a new topic, submit it to the forum, and engage with responses.
Triggers	Need to communicate new information or start a discussion.
Information used	Topic content, forum posting information
Information produced	New forum topic visible to the community.

UC29 Topic Search	
Description	Enables organizers to search for specific topics or keywords within the forum.
Goals	To find existing discussions on particular subjects or to monitor community discussions.
Actions	Enter search terms and retrieve relevant forum topics.
Triggers	Need to find information on a specific topic or to monitor discussions.
Information used	Search query, forum database.
Information produced	List of forum topics related to the search query.

UC30 View Hot Topics	
Description	Allows organizers to view the most popular or trending topics within the forum.
Goals	To keep up with community interests and engage in high-priority discussions.
Actions	Access the list of hot topics and participate in or monitor discussions.
Triggers	Interest in current events or popular discussions within the community.
Information used	List of popular forum topics.
Information produced	Display of hot topics with engagement metrics.

2.4.1.4 Student Interface Use-case Diagram

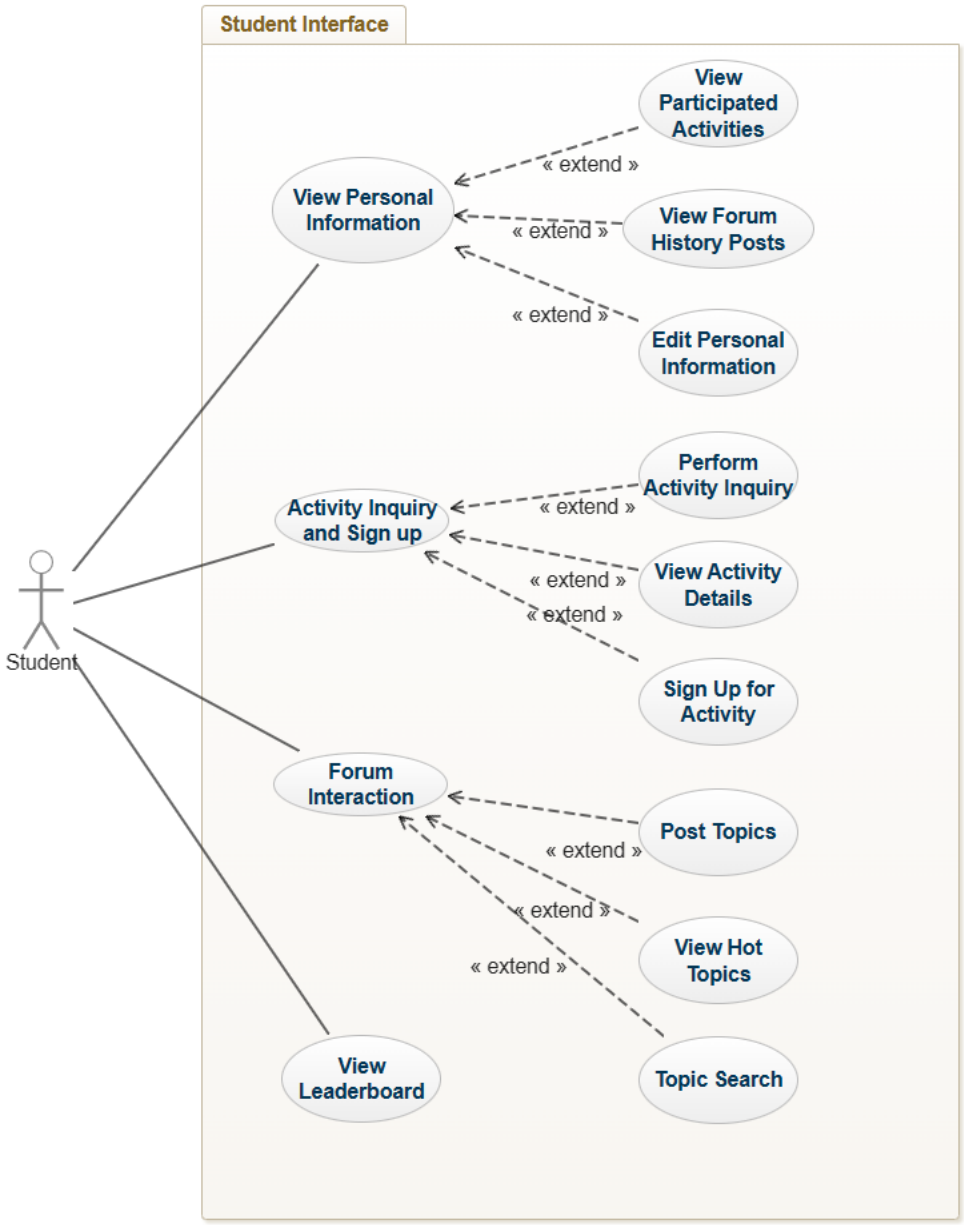


Figure 12 Student Interface

UC31 View Personal Information	
Description	Enables students to view and manage their personal details on the platform.
Goals	To ensure personal details are accurate and up-to-date.
Actions	Display personal information and provide options to edit.
Triggers	The need for students to verify or update their information.
Information used	Student's personal details.
Information produced	Updated personal information record.

UC32 View Participated Activities	
Description	Allows students to review a list of activities they have previously engaged in.
Goals	To reflect on past experiences and to track participation history.
Actions	Retrieve and display a history of the student's activities.
Triggers	The student's request to see their past activities.
Information used	Student account data, activity records.
Information produced	List of past activities with details.

UC33 Edit Personal Information	
Description	Permits students to update their personal information, such as contact details or profile preferences
Goals	To maintain accurate and current personal records
Actions	Modify personal details and save changes.
Triggers	Changes in the student's personal circumstances or information.
Information used	Personal information, new data provided by the student.
Information produced	Updated personal profile in the system.

UC34 Perform Activity Inquiry	
Description	Lets students search for activities that match their interests or availability.
Goals	To find and select activities they wish to participate in
Actions	Input search criteria and view a list of matching activities
Triggers	The desire to get involved in new volunteer activities.
Information used	Activity information, student's search parameters.
Information produced	List of activities that meet the search criteria.

UC35 View Activity Details	
Description	Provides detailed information about a specific activity, including time, location, domain, recipient, point value and requirements.
Goals	To inform students about the specifics of an activity before signing up
Actions	Select an activity and view its full description
Triggers	The need for more information about a particular activity
Information used	Activity details from the database.
Information produced	Display of detailed activity information.

UC36 Sign Up for Activity	
Description	Enables students to register for activities they are interested in.
Goals	To secure a place in a chosen activity.
Actions	Choose an activity and submit a registration request.
Triggers	Finding an activity of interest and wanting to participate
Information used	Activity information, student's personal information.
Information produced	Registration confirmation and updated activity participant list.

UC37 Forum Interaction	
Description	Offers a space for students to engage in discussions, ask questions, and share experiences.
Goals	To build a community and facilitate communication among students.
Actions	Post topics, reply to threads, and create new topics.
Triggers	The need for interaction or to share information with peers
Information used	Forum information, user's text input.
Information produced	New forum posts and thread replies.

UC38 Post Topics	
Description	Allows students to start new discussions on the forum.
Goals	To initiate conversations on topics of interest

Actions	Create a new forum topic and enter a subject and message.
Triggers	Having a new idea or question to discuss with the community.
Information used	Student's input for topic subject and content.
Information produced	New forum topic visible to the community.

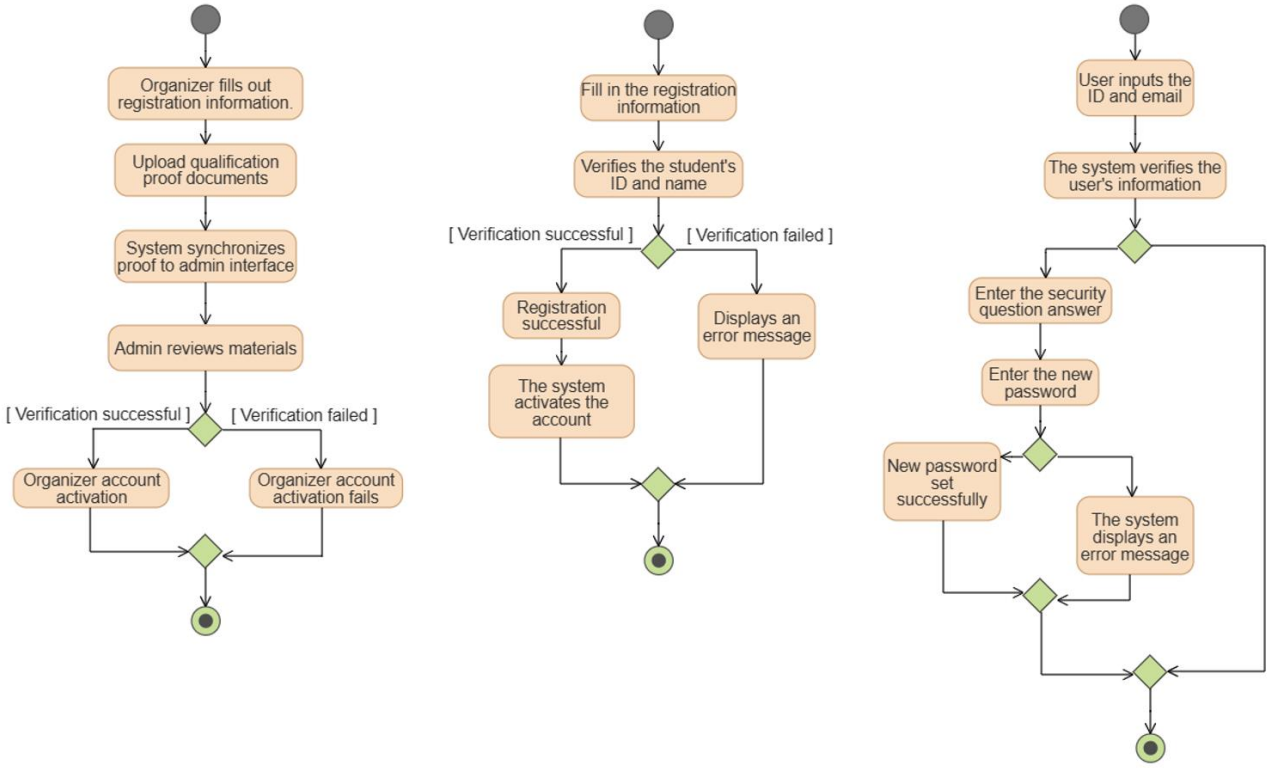
UC39 View Hot Topics	
Description	Highlights popular or trending discussions within the forum.
Goals	To engage with the community on current topics of interest.
Actions	Browse and join in on popular forum discussions.
Triggers	Wanting to participate in active conversations.
Information used	Topic informatino.
Information produced	Increased engagement and activity within hot topics

UC40 Topic Search	
Description	Enables students to search the forum for specific topics or keywords.
Goals	To find existing discussions on particular subjects
Actions	Enter search terms and retrieve relevant forum topics
Triggers	Looking for information on a specific topic that has been discussed.
Information used	Search query, forum information.
Information produced	List of forum topics related to the search query.

UC41 View Leaderboard	
Description	Displays a ranking of students based on their activity participation and points earned.
Goals	To recognize and motivate students for their contributions
Actions	View the leaderboard and see the ranking of participants.
Triggers	Curiosity about one's ranking or the performance of others
Information used	Student information, points system.
Information produced	Display of the leaderboard with rankings.

2.4.2 UML Activity Graph

2.4.2.1 Registration & Login Activity Graph



Graph 1 Organizer Registration Process

Graph 2 Student Registration Process

Graph 3 Password Recovery

Figure 13 Login & Registration Activity Diagram

This section of the document provides a detailed design of the following processes:

1. **Organizer Registration Process:** This graph outlines the steps that an organizers have to take when they register on the software. It includes the information required for registration, such as personal details, organizational affiliation, and necessary qualification. The process would also detail the verification steps, approval workflows, and the final activation of the organizer's account.
2. **Student Registration Process:** Similar to the organizer registration, this process describes how students sign up for the software. It covers the data collection points, such as the student's name, student ID, and contact information. Additionally, it would include the steps for account verification and the process for setting up a new user profile.
3. **Password Recovery Process:** This process is designed to help users recover or reset their passwords in case they forget them. It involves steps such as entering a registered email address, receiving a password reset link, and then creating a new password. The process would ensure security measures are in place to verify the identity of the user requesting the password reset.

2.4.2.2 Student Sign up Activity Graph

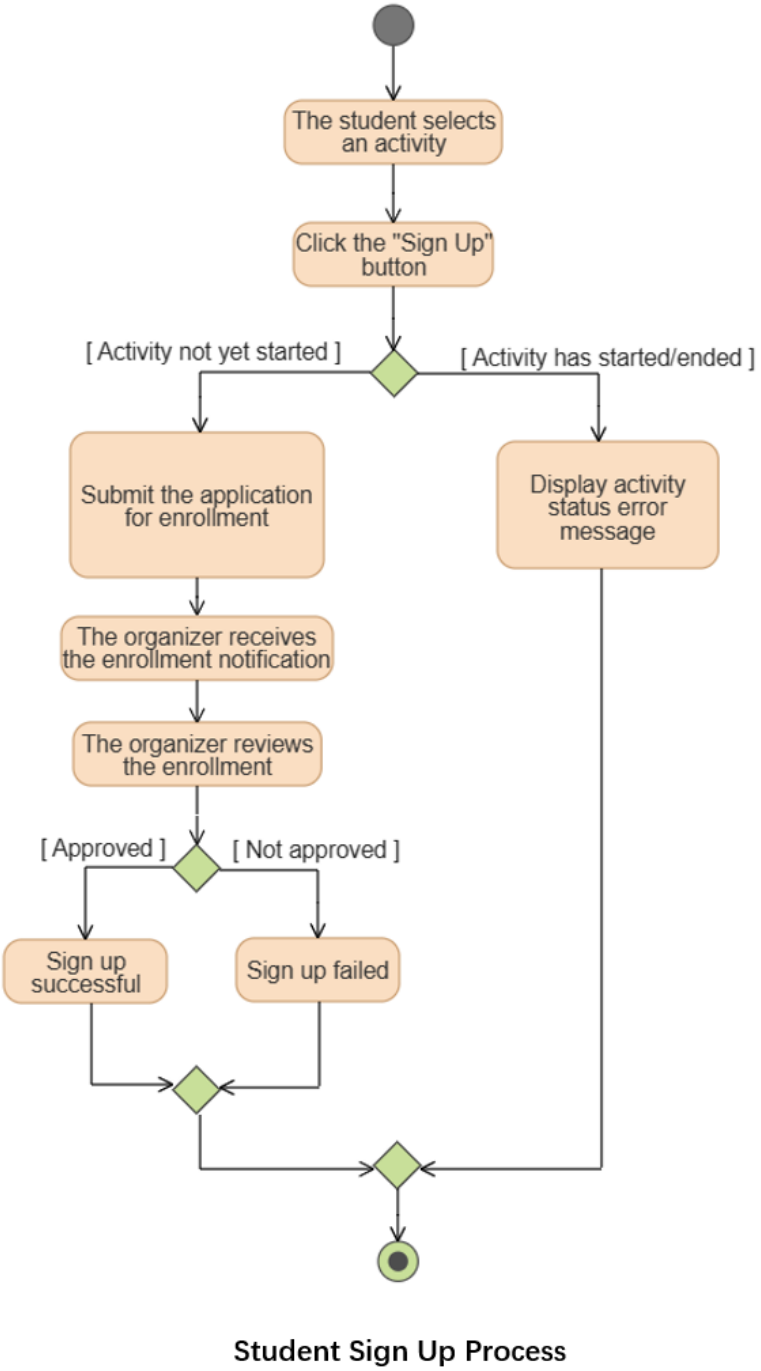


Figure 14 Student Sign-up Process Activity Diagram

This section of the document provides a detailed design of the student sign-up process for activities.

2.4.2.3 Organizer Activity Operation

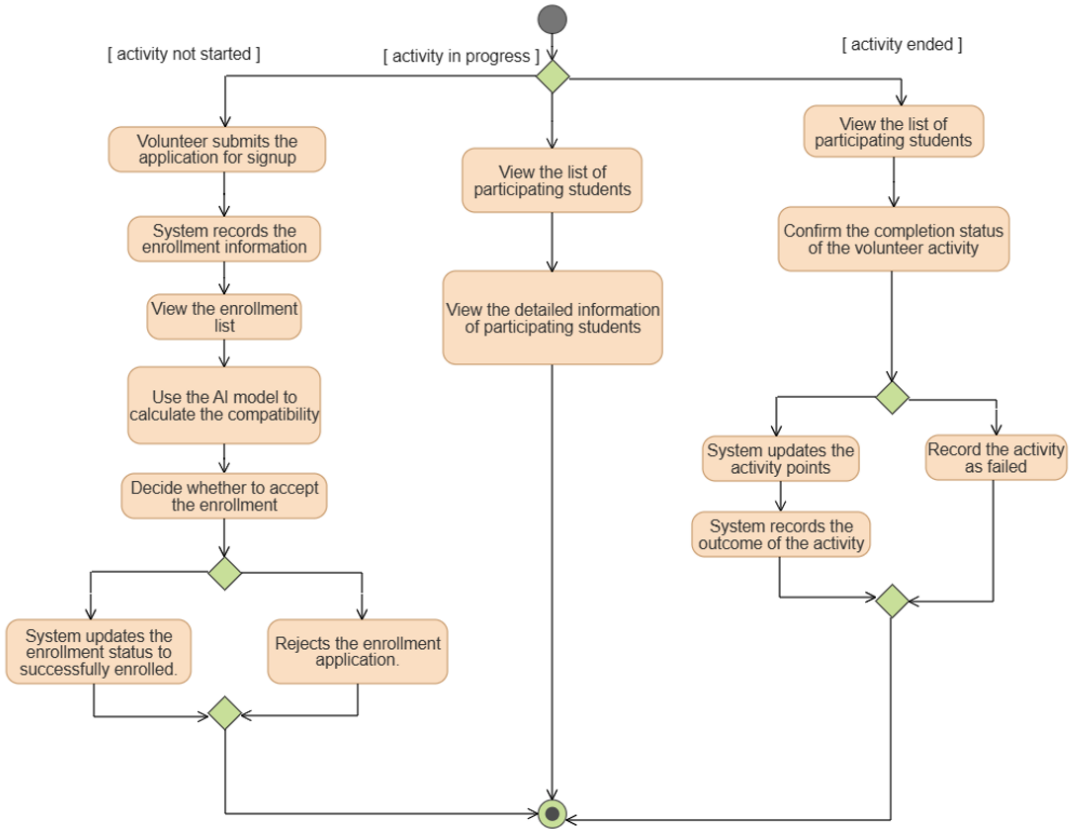


Figure 15 Organizer Activity Operation

This section of the document provides a detailed design of the various operations that organizers can perform based on the different states of an activity. “Activity not started”, “activity in progress”, “activity ended” present three kind of activity status.

2.5 User Interface Design

When designing the UI of the volunteer software, we adopted a user-centered design concept with the goal of creating an intuitive, interactive and responsive user interface. During the design process, we initially used hand-drawn sketches to rapidly iterate through various layout options. Subsequently, we developed interactive prototypes using WinForms technology or WPF within Visual Studio.

2.5.1 Winform Development

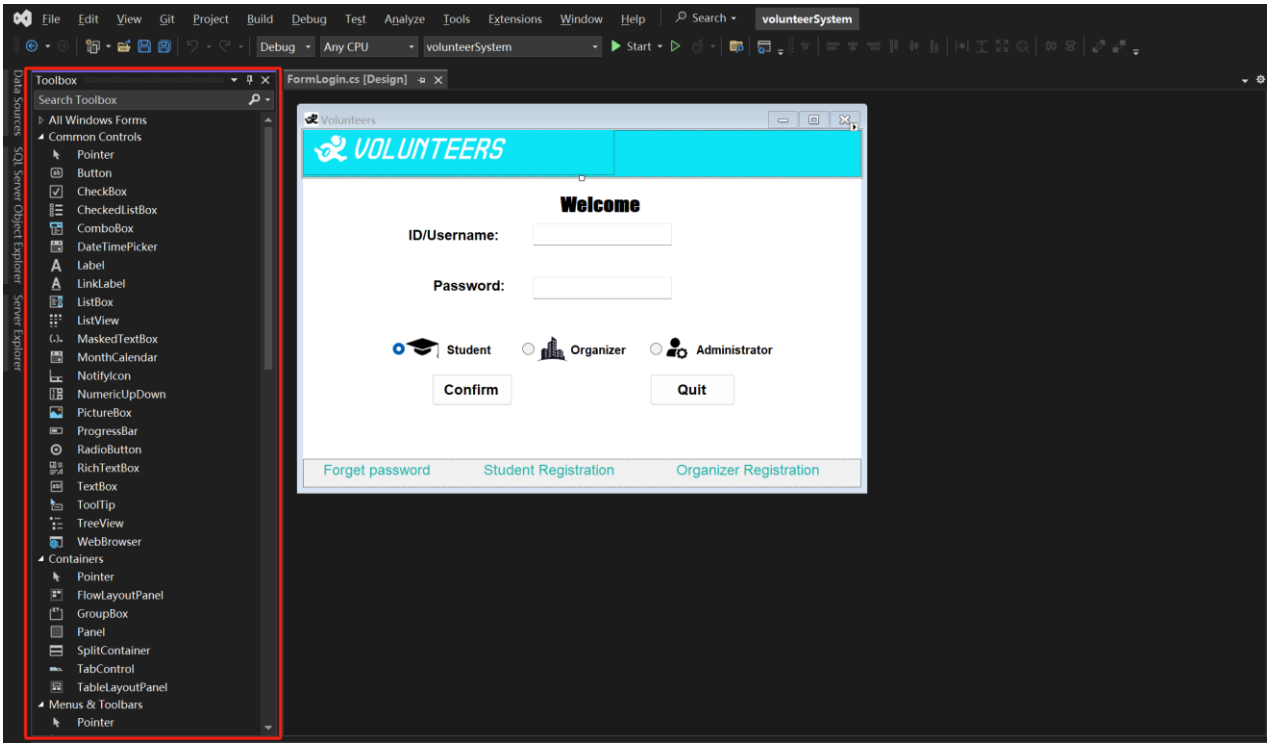


Figure 16 Winform User Interface Development in Visual Studio 2022

In the Volunteers software, we have elected to utilize WinForms for the development:

- The login and registration interfaces.
- The entire suite of administrator interfaces.
- The student interface for personal information management, activity browsing and sign-up.
- The organizer interface for organizer information management and activity management-related interfaces.

Visual Studio provides a rich control library and efficient development process, enabling us to rapidly construct dynamic interfaces corresponding to user operations. The toolbox provides a variety of controls, including buttons, text boxes, and list boxes, that can be used to create complex interface layouts using drag-and-drop operations. Initially, we determined the basic layout of the interface based on requirement analysis and sketched the interface layout by hand. Subsequently, using Visual Studio's WinForms designer, we dragged and dropped controls onto the form and set their properties in the properties window. Additionally, we can adjust the position and style of the controls on the interface through back-end code. After the form is constructed, we bind the corresponding methods to the controls in the event designer and then write the specific business logic code.

The following are some winform interface hand drawings：

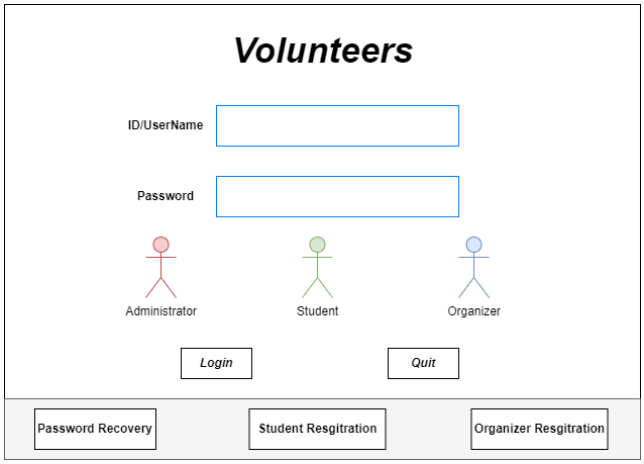


Figure 17 Login & Registration Interface Hand Drawing

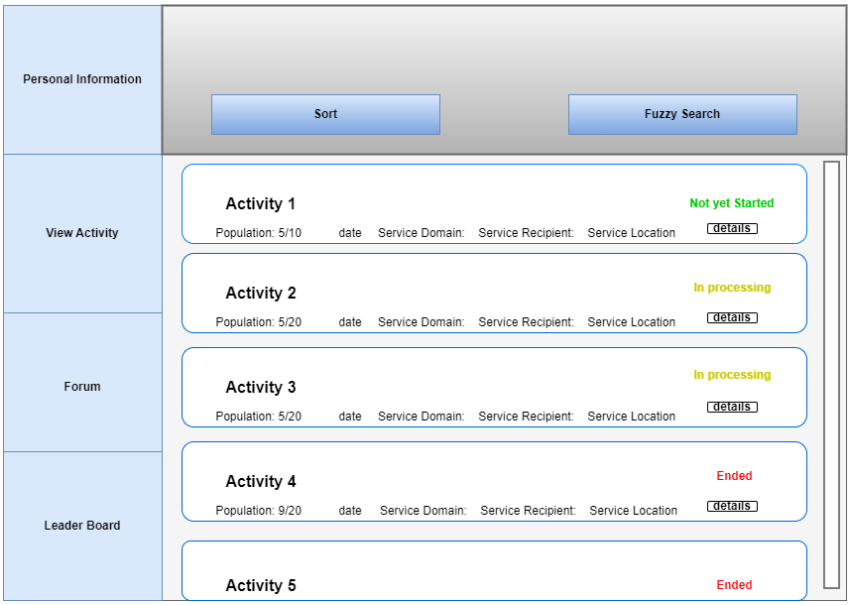


Figure 18 Student Main Dashboard Hand Drawing

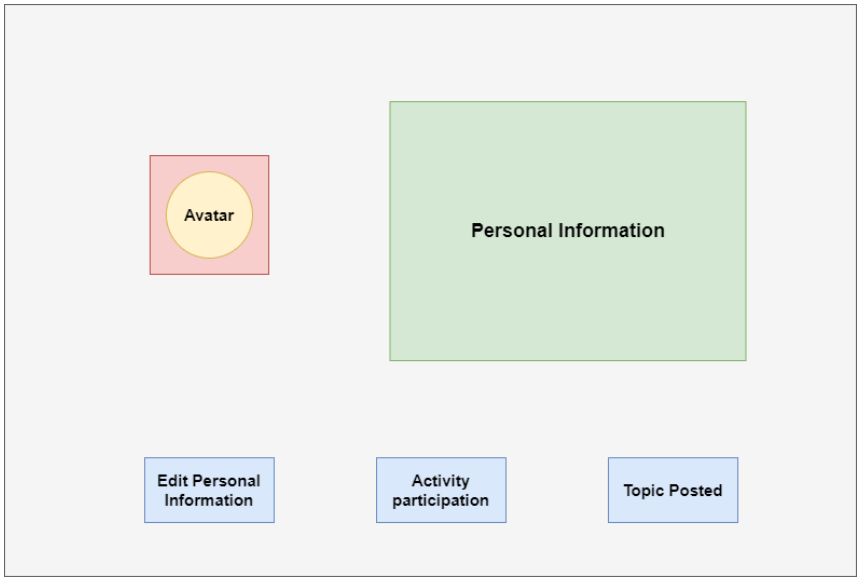


Figure 19 Student Personal Information Hand Drawing

2.5.2 WPF

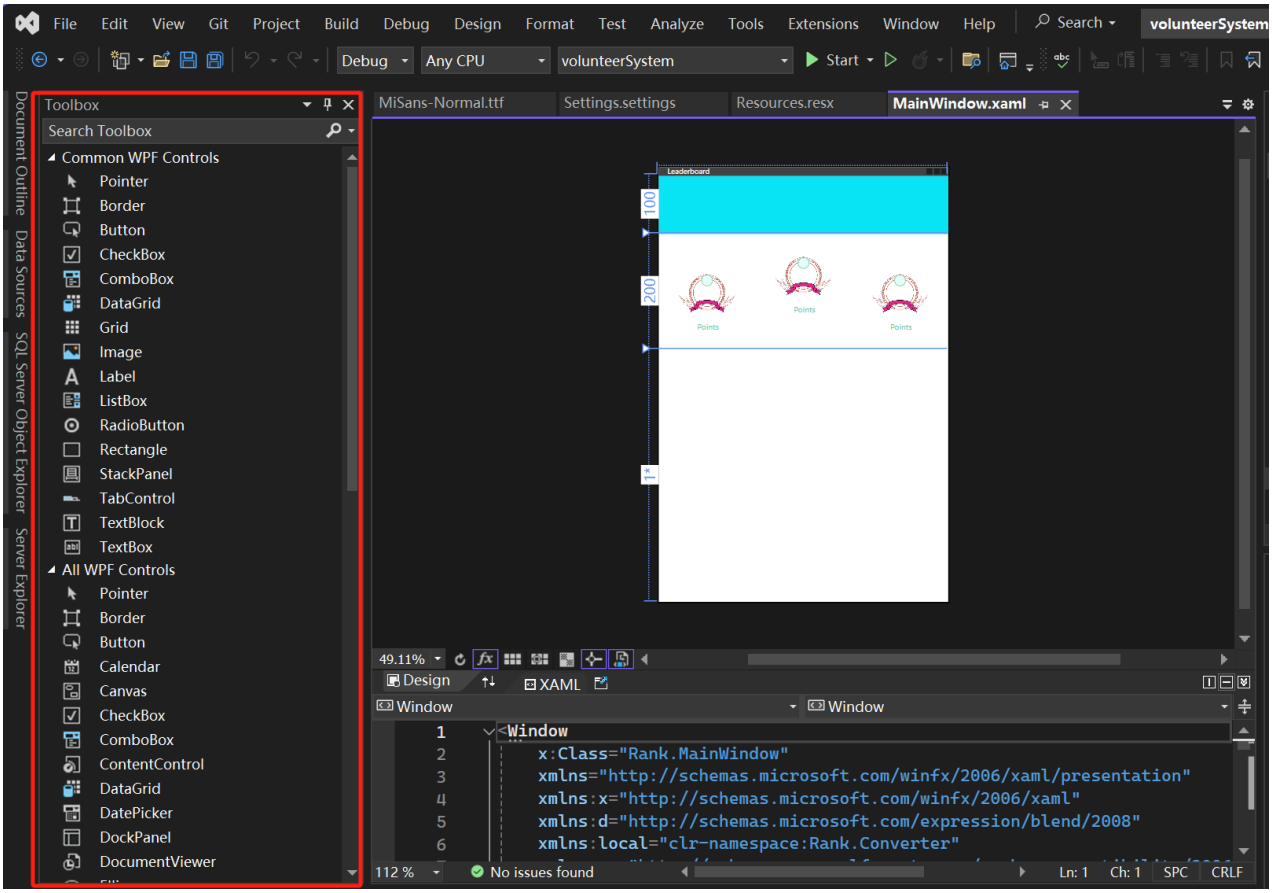


Figure 20 WPF User Interface Development in Visual Studio 2022

In the Volunteers software, we have decided to use Windows Presentation Foundation (WPF) for the development of all forum interfaces as well as the leaderboard interface.

WPF supports hardware-accelerated rendering, which allows for the creation of more sophisticated and responsive user interfaces. Additionally, WPF's XAML (Extensible Application Markup Language) provides a declarative interface design approach that enables us to program complex UI layouts and behaviors. Visual Studio also provides a rich set of WPF controls that can be used to quickly create interfaces through the designer. When developing interfaces with WPF, we first define the root element of the window, then design the basic framework of the interface according to the requirements, and add the necessary controls to the window. Finally, we use WPF's styling and templating features to unify the appearance of the interface. For interaction logic, we implement it in the backend code, linking it with the UI controls through data binding.

The following are some WPF interface hand drawings：

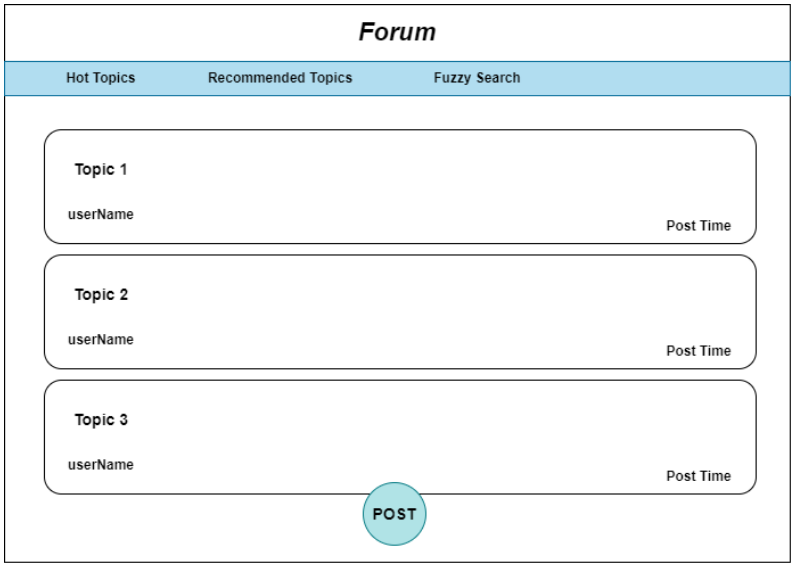


Figure 21 Fourm Interface Hand Drawing

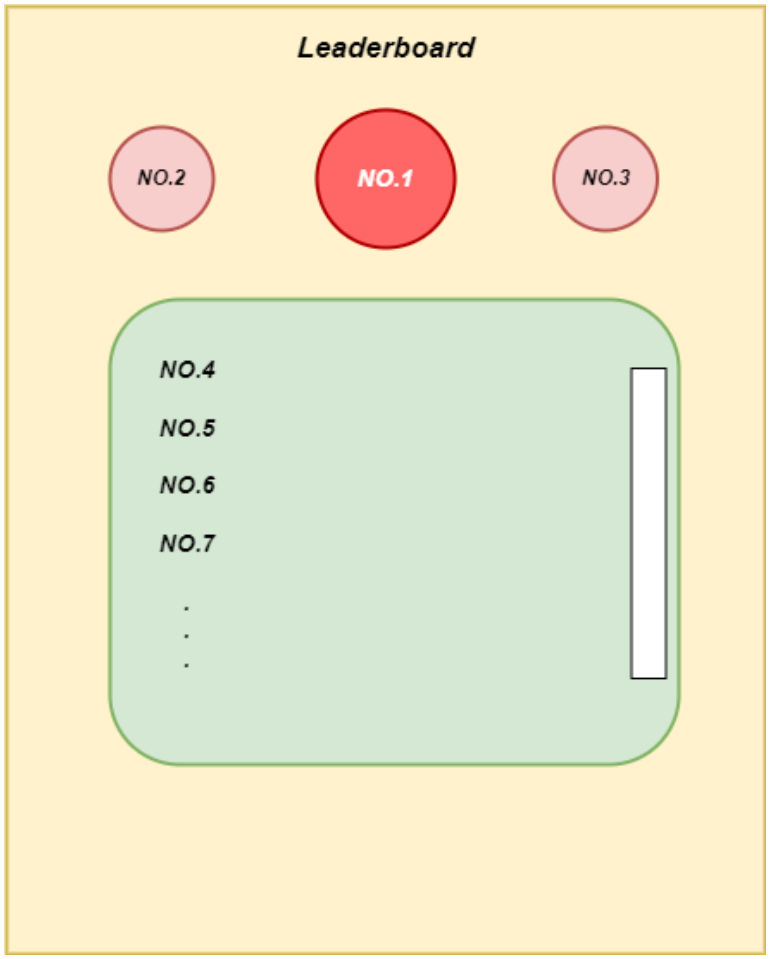


Figure 22 Leaderboard Interface Hand Drawing

2.6 Evaluation Method Design

2.6.1 Functional Testing Design - Black Box Testing

The functional testing strategy aims to verify that each component of the software conforms to the requirements documentation, with a particular focus on the specific needs of different user roles. For the Volunteers software, we have designed the following functional test cases:

Administrator Users:

- Test cases are designed to validate the monitoring and management capabilities of the administrator account, including adding and deleting school-wide student information, managing registered student and organizer

information, and locking and unlocking student and organizer accounts.

- Test the audit process for organizer registration applications, including reviewing submitted content and making approval or rejection decisions.

Organizer Users:

- Test cases are designed to verify the account management and login process for organizers, including uploading supporting materials and the account review process.
- Test activity management, including creating, editing, and updating activity details, uploading images, and updating activity statuses (e.g., changing activity dates).
- Test the audit process for participant applications, including accepting and rejecting student participation requests and viewing participant information.
- Test forum interactions, including the ability to post, edit, and delete topics and comments.

Student Users:

- Test cases are designed to verify the registration and login process for student users, including error handling (such as entering invalid data) and user feedback (such as password error prompts).
- Test the activity browsing and registration process to ensure accurate presentation of activity details and correct response to activity statuses (e.g., preventing registration for full or completed activities).
- Test personal profile management, including changing usernames, uploading or changing profile pictures.
- Test forum interactions, including posting, editing, deleting topics and comments, and searching, browsing, and interacting with posts (such as liking and replying).

Functional Testing Plan Design:

Test tools and methods:

Test Data: Use virtual data to create a controlled testing environment that simulates real-world user behavior and system interactions.

Automated testing: For some test scenarios that are highly repetitive and easy to automate, such as login, registration and basic data operations, automated testing tools are used to perform. This helps improve test efficiency and accuracy.

Manual testing: For functions that require complex user interaction, such as forum interaction and advanced activity management functions, manual testing is performed. This helps to better evaluate the friendliness of the user interface and the user experience during actual use.

Test Feedback Process:

1.Error Handling and Feedback Mechanisms: Ensure clear error messages are provided when users perform incorrect operations, along with guidance for correction.

2.User Feedback: Test the system's response to user inputs, including immediate feedback on success or failure

Test Table Design:

Test Case ID	Description	Objective	Expected Outcome	Actual Outcome	Guarantee	Bug
TC1	…	…	…	…	…	…
TC2	…	…	…	…	…	…

2.6.2 Acceptance Testing Design

Purpose: Acceptance testing is designed to confirm that the software not only meets technical functional requirements but also aligns with the actual needs and operational habits of the users. This strategy is crucial for ensuring the software's functional integrity, operational convenience, and the appeal of the user interface.

Testing Protocol: The acceptance testing protocol includes environmental simulation, participation by multiple stakeholders,

scenario modeling, and long-term operation. This multifaceted approach simulates real-world usage patterns, gathers diverse feedback, and assesses the stability and performance of the software under continuous operation conditions.

2.6.3 Usability Testing Design

Core Objective: The core purpose of usability testing is to ensure that the software interface is friendly and intuitive, so that all types of users can quickly learn and effectively use the software, so as to improve the overall user satisfaction and usability of the product.

Evaluation Techniques: Evaluation techniques include user testing sessions, questionnaires, and metric analysis. These methods together provide an in-depth understanding of user interactions, preferences, and the effectiveness of the user interface.

2.6.4 Compatibility Testing Design

Design Objective: The main goal of compatibility testing is to verify the functional performance of software on different versions of Windows system, to ensure that the software can maintain stability and reliability under various system configurations.

Testing Framework: The framework for compatibility testing is constructed to reflect real-world user experience environments, encompassing a range of operating systems and hardware configurations. This robust framework ensures that the software's functionality remains consistent and undamaged across different platforms.

2.7 Project Team Coordination and Scheduling

After planning the software's testing strategies, another critical component of the project is ensuring effective division of labor and collaboration among team members.

Software Development Allocation:

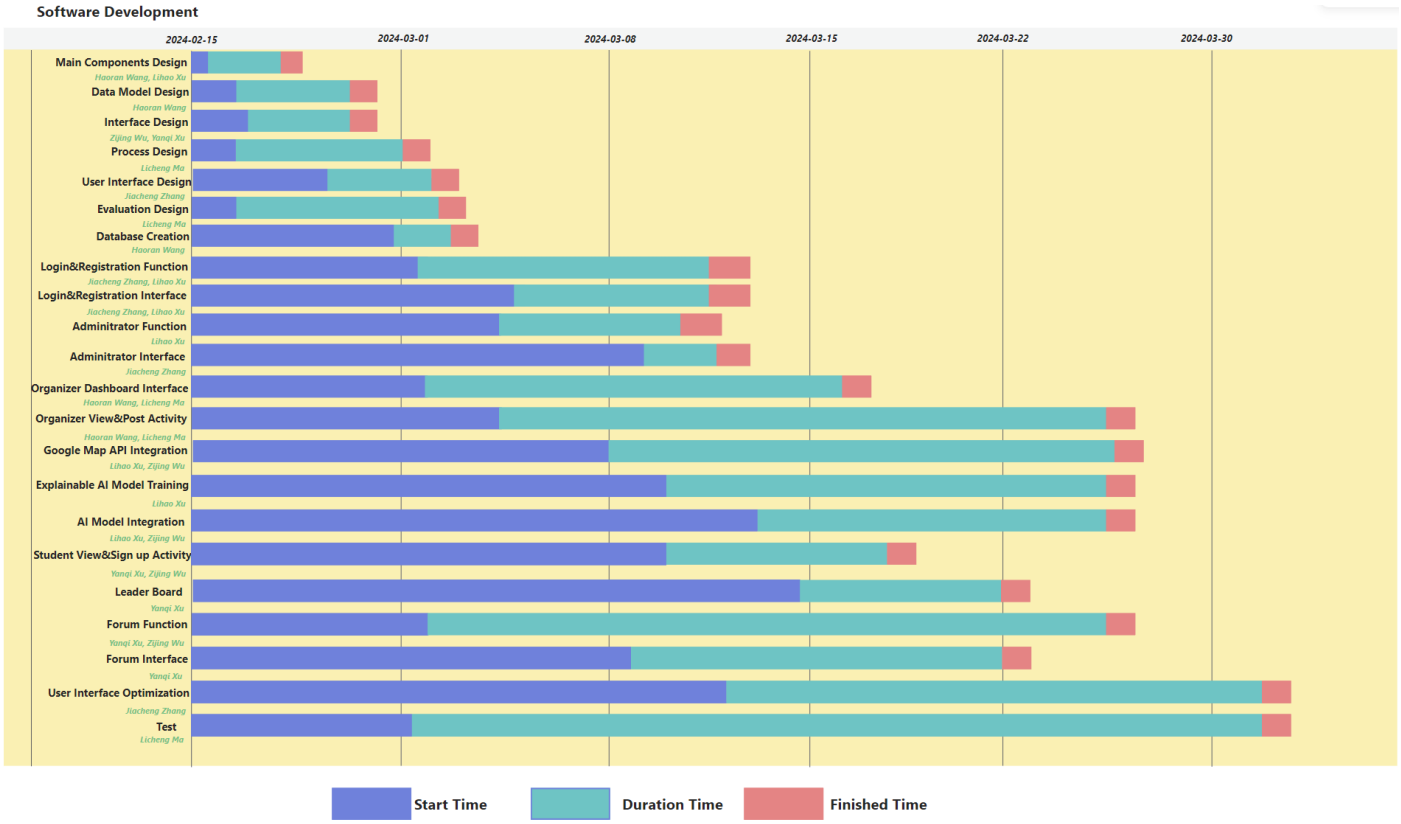


Figure 23 Software Development Allocation

Entire Group Project Allocation:

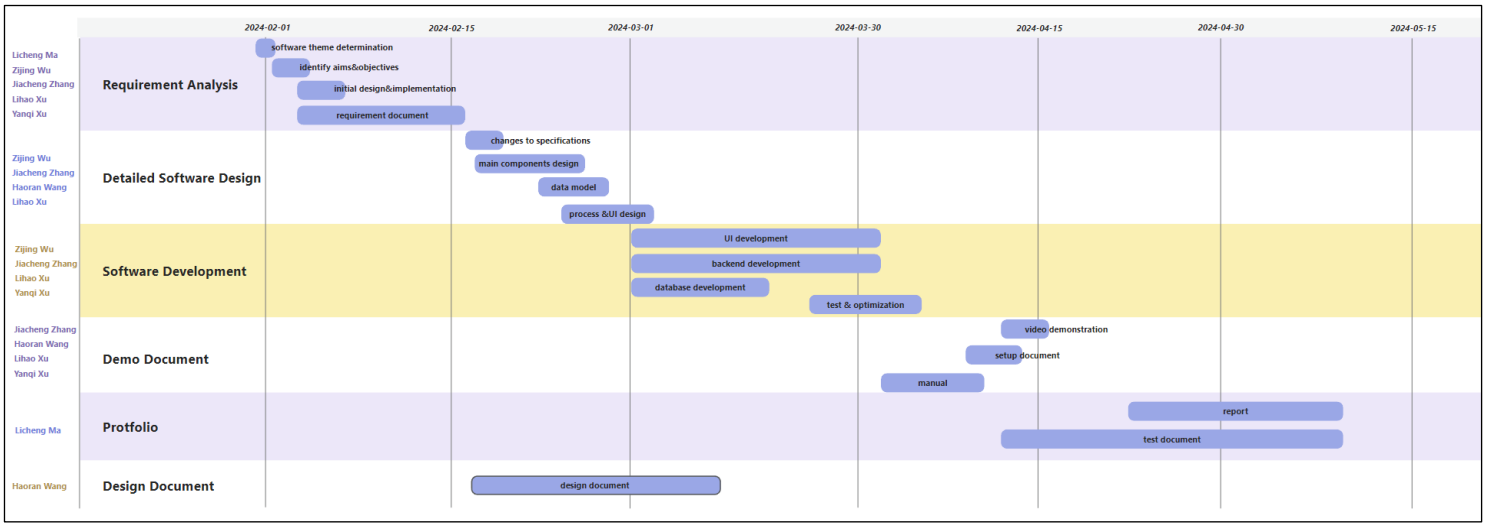


Figure 24 Entire Group Project Allocation