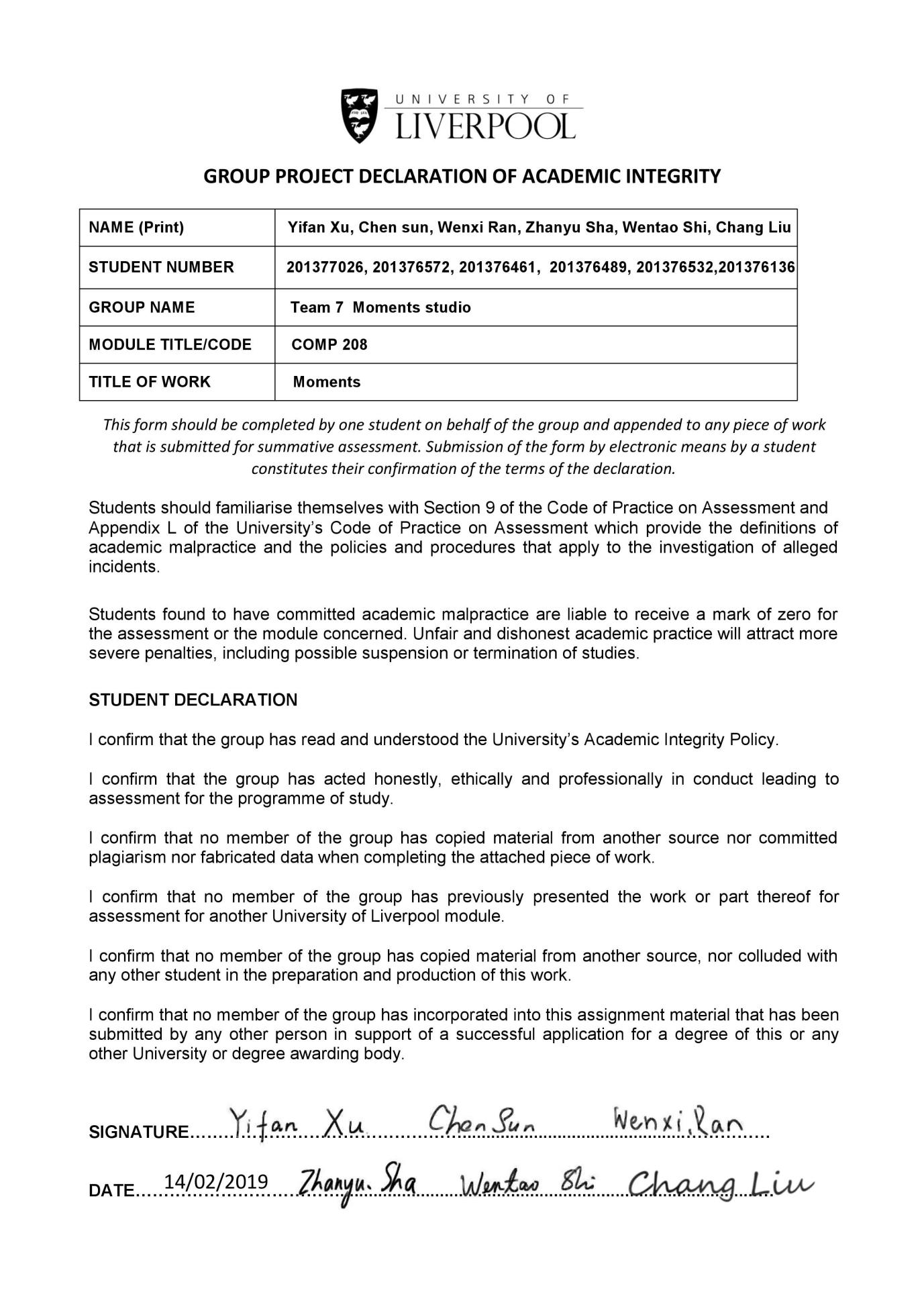
****

**Design Walkthrough**

Moments

****

**COMP208 Team 7**

Yifan Xu

Chen Sun

Chang Liu

Wenxi Ran

Zhanyu Sha

Wentao Shi

15th Mar 2019

*Preface*

In requirement analysis period, we have settled the essential and desirable website functions, target users and different requirements referring to different user groups. The content we researched in the requirement section benefits to the project design period. In the design stage, we will build various models based on the requirement walkthrough to help us have a better understanding of the product. In the implementation stage, the website will be produced according to the models we produced on this period.

The models will be divided into two parts respect to logical and physical aspects and mainly around data, process and transaction. Expound of our design will go to technically by degrees and will involve four key project components: database, society management user interface, general user interface (student) and university department administrator user interface.

Contents

**[Chapter 1](#_Toc31883_WPSOffice_Level1)** **[5](#_Toc31883_WPSOffice_Level1)**

[Proposal Summary](#_Toc19341_WPSOffice_Level2) [6](#_Toc19341_WPSOffice_Level2)

[1.1 Project statement summary](#_Toc23298_WPSOffice_Level2) [6](#_Toc23298_WPSOffice_Level2)

[1.2 Requirement proposal adjustment](#_Toc23204_WPSOffice_Level2) [6](#_Toc23204_WPSOffice_Level2)

[1.3 Design document overview](#_Toc9581_WPSOffice_Level2) [7](#_Toc9581_WPSOffice_Level2)

**[Chapter 2](#_Toc19341_WPSOffice_Level1)** **[7](#_Toc19341_WPSOffice_Level1)**

[Architecture](#_Toc11516_WPSOffice_Level2) [8](#_Toc11516_WPSOffice_Level2)

[2.1 Network Architecture](#_Toc23298_WPSOffice_Level1) [8](#_Toc23298_WPSOffice_Level1)

[2.2 System Architecture](#_Toc23204_WPSOffice_Level1) [9](#_Toc23204_WPSOffice_Level1)

[2.3 Components](#_Toc9581_WPSOffice_Level1) [10](#_Toc9581_WPSOffice_Level1)

[2.4 ­­Component Interface](#_Toc11516_WPSOffice_Level1) [11](#_Toc11516_WPSOffice_Level1)

[2.5 Structure and navigation](#_Toc20414_WPSOffice_Level1) [13](#_Toc20414_WPSOffice_Level1)

**[Chapter 3](#_Toc10135_WPSOffice_Level1)** **[15](#_Toc10135_WPSOffice_Level1)**

[Use Case](#_Toc14211_WPSOffice_Level2) [15](#_Toc14211_WPSOffice_Level2)

[3.1 Use case Diagram](#_Toc10067_WPSOffice_Level2) [15](#_Toc10067_WPSOffice_Level2)

[3.2 Sequence diagrams](#_Toc17377_WPSOffice_Level2) [16](#_Toc17377_WPSOffice_Level2)

**[Chapter 4](#_Toc23134_WPSOffice_Level1)** **[18](#_Toc23134_WPSOffice_Level1)**

[Use Interface Design](#_Toc748_WPSOffice_Level2) [19](#_Toc748_WPSOffice_Level2)

4.1 UI: Homepage [2](#_Toc748_WPSOffice_Level2)1

4.2 UI: Login and Register [2](#_Toc748_WPSOffice_Level2)2

4.3 UI:  Personal society/activity quick navigation [2](#_Toc748_WPSOffice_Level2)3

4.4 UI: Comment [2](#_Toc748_WPSOffice_Level2)4

4.5 UI:Search Result [2](#_Toc748_WPSOffice_Level2)5

4.6 UI:Search Result  [26](#_Toc748_WPSOffice_Level2)

4.7 UI:Event Page  [2](#_Toc748_WPSOffice_Level2)7

**[Chapter 5](#_Toc14211_WPSOffice_Level1)** **[29](#_Toc14211_WPSOffice_Level1)**

[Algorithm design (Pseudocode)](#_Toc26065_WPSOffice_Level2) [29](#_Toc26065_WPSOffice_Level2)

[5.1 Search matching algorithm](#_Toc6517_WPSOffice_Level2) [29](#_Toc6517_WPSOffice_Level2)

[5.2 User-based collaborative filtering](#_Toc31248_WPSOffice_Level2) [31](#_Toc31248_WPSOffice_Level2)

[5.3 Trending association algorithm](#_Toc12124_WPSOffice_Level2) [33](#_Toc12124_WPSOffice_Level2)

**[Chapter 6](#_Toc10067_WPSOffice_Level1)** **[35](#_Toc10067_WPSOffice_Level1)**

[Database Design](#_Toc27924_WPSOffice_Level2) [35](#_Toc27924_WPSOffice_Level2)

[6.1 Entity-relationship diagram](#_Toc12870_WPSOffice_Level2) [35](#_Toc12870_WPSOffice_Level2)

[6.2 Logical table structure](#_Toc25102_WPSOffice_Level2) [35](#_Toc25102_WPSOffice_Level2)

[6.3 Physical table structure](#_Toc9552_WPSOffice_Level2) [36](#_Toc9552_WPSOffice_Level2)

[6.4 Transaction Matrix](#_Toc12713_WPSOffice_Level2) [42](#_Toc12713_WPSOffice_Level2)

[6.5 Data Dictionary](#_Toc24689_WPSOffice_Level2) [45](#_Toc24689_WPSOffice_Level2)

**[Chapter 7](#_Toc17377_WPSOffice_Level1)** **[48](#_Toc17377_WPSOffice_Level1)**

[Evaluation Design](#_Toc5654_WPSOffice_Level2) [48](#_Toc5654_WPSOffice_Level2)

[7.1 Functional Testing](#_Toc748_WPSOffice_Level1) [48](#_Toc748_WPSOffice_Level1)

[7.2 Performance Testing](#_Toc15772_WPSOffice_Level1) [51](#_Toc15772_WPSOffice_Level1)

[7.3 Usability Testing](#_Toc17479_WPSOffice_Level1) [51](#_Toc17479_WPSOffice_Level1)

[7.4 Recoverability Testing](#_Toc29062_WPSOffice_Level1) [52](#_Toc29062_WPSOffice_Level1)

[7.5 Security Testing](#_Toc13753_WPSOffice_Level1) [53](#_Toc13753_WPSOffice_Level1)

[7.6 Database Evaluation design](#_Toc5697_WPSOffice_Level1) [54](#_Toc5697_WPSOffice_Level1)

**[Chapter 8](#_Toc31346_WPSOffice_Level1)** **[56](#_Toc31346_WPSOffice_Level1)**

[Business Rules](#_Toc14043_WPSOffice_Level2) [56](#_Toc14043_WPSOffice_Level2)

**[Chapter 9](#_Toc21742_WPSOffice_Level1)** **[57](#_Toc21742_WPSOffice_Level1)**

[Review Plan](#_Toc14428_WPSOffice_Level2) [57](#_Toc14428_WPSOffice_Level2)

[9.1 Completed Tasks](#_Toc19866_WPSOffice_Level2) [57](#_Toc19866_WPSOffice_Level2)

[9.2 Deliverable](#_Toc12558_WPSOffice_Level2) [57](#_Toc12558_WPSOffice_Level2)

[9.3 Future Plan](#_Toc3037_WPSOffice_Level2) [58](#_Toc3037_WPSOffice_Level2)

[9.4 Gantt Charts](#_Toc24540_WPSOffice_Level2) [59](#_Toc24540_WPSOffice_Level2)

**[Chapter 10](#_Toc9985_WPSOffice_Level1)** **[63](#_Toc9985_WPSOffice_Level1)**

[Bibliography](#_Toc6860_WPSOffice_Level2) [63](#_Toc6860_WPSOffice_Level2)

**[Chapter 11](#_Toc26662_WPSOffice_Level1)** **[64](#_Toc26662_WPSOffice_Level1)**

[Appendix](#_Toc10959_WPSOffice_Level1) [64](#_Toc10959_WPSOffice_Level1)

Chapter 1

Proposal Summary

**1.1 Project statement summary**

*“As a result of the development of computers and communications during the last several decades, information overload has become a significant concern, in which case the amount of information that worth attention outdistances the individual's ability on dealing with it[1].”*

This is a brief statement of the background, aims and the objectives of the project. Due to the information overload problem mentioned above, **Moments** website aims to help students find excting integration information efficiently and also strengthen the communication between universities and students. It allows users to share their opinions via comment and chatroom board and also support search bar to help them quickly gain information, such as campus events and social activities.

**1.2 Requirement proposal adjustment**

After having the feedback from the reviewers and a comprehensive comparison of the website with others, we adjust the original requirements document considering the following aspects: target market, smarter recommendation system with designed Al algorithm, UI design conforming for a higher level of user interaction. The original target users of the system are the students, student societies and clubs, department offices at the University of Liverpool. According to the feedback saying that: “The system might be fit into be used in any higher education environment”, the users might not only be limited to students at the University of Liverpool. As for the Al algorithm part, instead of using a questionnaire to get the users’ preferences, we decide to apply the users’ research or joined activities that stored in the databases and then train the model.

*“we found that transferring useful information is essential for a website, building a strong connection between users and the website.”*

Besides, aims to arise user’s interest and attract users to stay continuously in the website, we make some adjustment on the enhance the interaction between the user and the website. Referencing some popular Live social media nowadays, such as Instagram Live, Facebook Live platform, users could use ‘emoticon package’ during the Live or chatting. Based on the above interaction form, we will design a live chat box which appears as soon as the society post a new event, such as competition or voting. It allows users to share their feelings and see different people’s reaction instantly.

The comments and feedback for an event/ society/ department might also be the main contents that users are interested in, so we will develop a hot search algorithm aiming to present hottest topics to customers and update daily. It gives the user the opportunity to seamlessly interact with the event publisher and build a community around the event or department.

**1.3 Design document overview**

The design document aims to provide an overall architecture of the Moments’ website system to guide the implementation of the product, including architecture design, algorithm design( Pseudocode), User Interface design, database design, evaluation design, followed by business rule and review plan.

Chapter 2

Architecture

This section of the report will set out of the underlying architecture design for the website. It will focus on two central aspects of architecture: Network Architecture and the system architecture

**2.1 Network Architecture**

The web server of the Moments website is based on the Tencent Cloud Server in Frankfurt, Europe, which provides storage, and facilitates communication between clients. The domain and the linker of the website are www.momentists.com.

The limited network size of this server is 2M and only allows 34 people to log in at the same time(the server depending on users’ local internet speeds and varies from browsing different website types). According to this inherently limited network size, in the future developing stage, we will update the brands and the level of the server.

Considering the security of the network, during chatroom function, the users might send messages in the societies chartroom. The public key and private key will encrypt these messages and also apply RSA signature methods. It is quite essential and necessary to protect network security and users' information.

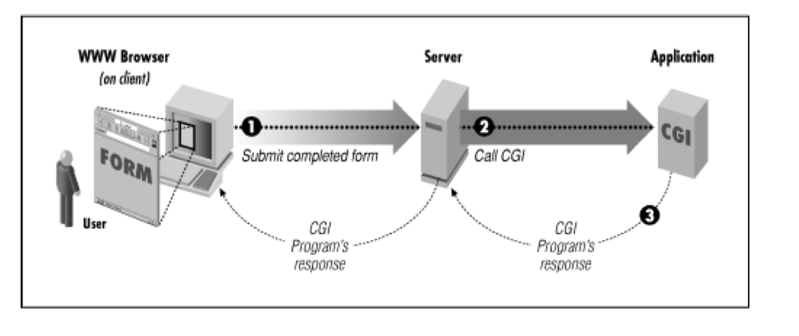


Figure 1. Network Processing design

Perl and CGI programming also will be implemented for network data processing. Here are some following steps. Firstly, the web client will generate a client request, for example, from an HTML form, and sends it to a web server. The server will then select a CGI program to handle the request, converts the client request to a CGI request, executes the program. The CGI program then processes the CGI request and the server passes the program’s response back to the client.

**2.2 System Architecture**

**2.2.1 System boundary diagram**

Here are some primary languages that might be used in designing the parts of the system.

• Presentation: the HTML website for users

• Logical: PHP /javaScript/Perl

• Database: MySQL server

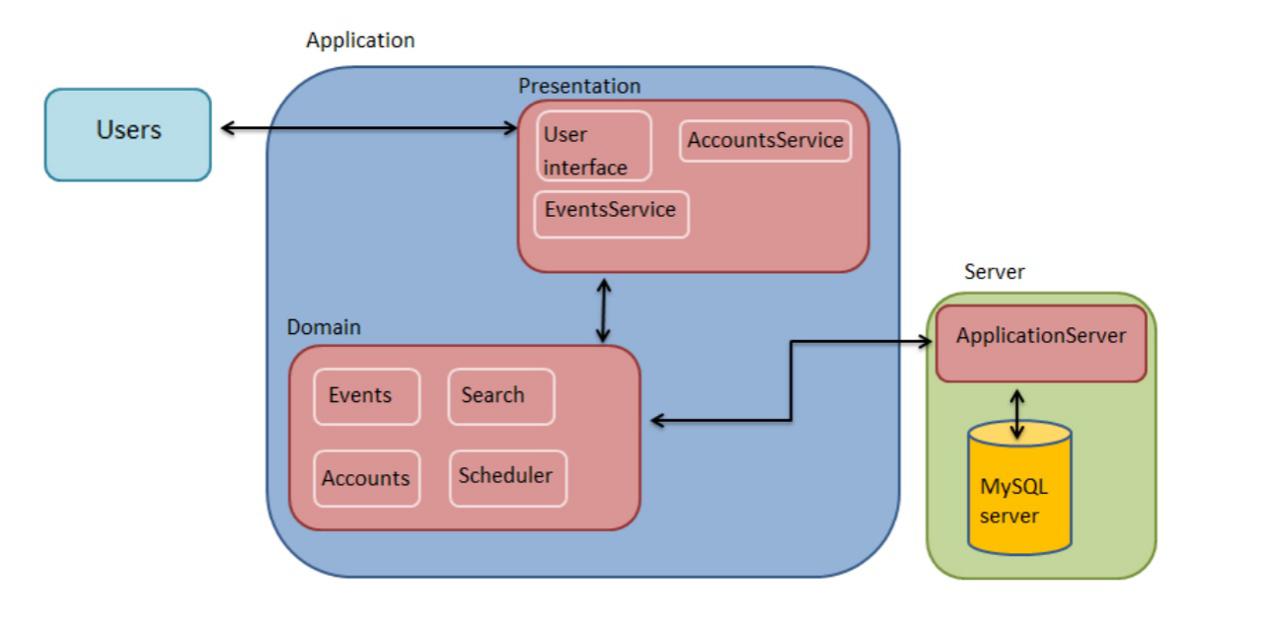
****

Figure 2: System Boundary Diagram

**2.2.2 System architecture design**

*Model-View-Controller (MVC) architecture separates the user’s actions and what they see from the data that is held in the system in system design and implementation of the user interface layer.*

By Applying Model-View-Controller (MVC) architecture, the system for Moments’ website will have three main layers, including the client, the application server and the database. Each layer contains several functions and interacts as little as possible with the rest of the system. The strength of using this architecture is that the system becomes flexible and reusable, which means the system could be changed partly without reprocessing the entire application.

The object-oriented approach will also be applied in this section to illustrate the inheritances and interactions inside the system to decrease the repetition and redundancy of the databases. Additionally, the university department offices will be views as particular student societies.

**2.3 Components**

Based on the Object-Oriented approach, the Moments website system is split into five components, which presents a clear view of how the system operates and interacts with different areas of the system.



Figure 3. Component Graph

**Registration and Accounts**

This component deals with the account details and registration of all users as well as logging in and out of the system.

**Student**

The student is the primary target user in our system. This component is set to deal with all related information to each student.

**Society / University Department**

Society and university department are two vital users of our system. As the website functions, we defined for society and university department are quite similar, we regard them as one system component here. The society/university department part deals with the related work for society and department. The functions are like launching events, editing homepage and chatting room are included in this component.

**Server Operator**

Server operator, as well as being a particular type of user of the system, will be a part of the system that deals with the daily events related to the *Moments* website itself. This component is designed to deal with the server operator website functions.

**Event**

This part of the system deals with the system activities related to the events published by society/university department.

**2.4 ­­Component Interface**

**2.4.1 Registration and Accounts**

**a.Registration Interface**

The registration interface is composed of a simple registration form which allows users to fill in. After verifying the user identity with some information provided by the university successfully, the user could be authorised to log into the website system by inputting their user ID and password.

**b.Accounts**

There are three kinds of accounts: students, society management and university department administrator. All account information updating, register original information, account looking through information etc. will be stored on each account.

**2.4.2** **Society/ University Department**

**a. Society Interface**

Society interface will display the related information of the specific society and might be the most interactive webpage on the system. In society interface, there will be a social group chatting room, brief event exhibition members list.

**b. University Department Interface**

University department interface will provide various information about official notification or activities held by university departments. Registered users could also make comments on the button of the department interface.

**2.4.3 Server Operator Interface**

An interface for a server operator to reply daily questions from the user, checking the published website materials and launching website official information. The interface could only be view by the website server operator.

**2.4.4 Personal Interface**

The personal interface is designed to display information related to the user. The user could edit their personal information by merely clicking the edit button. The user could also quickly jumping into their joined society or activity page by clicking on the society or activity itself.

**2.4.5 Event Interface**

The registered visitors (mainly are the students) could go through the event interface to register for an activity efficiently using a quick book button. Users also could make comments under the event homepage.

**2.5 Structure and navigation**

The diagrams below present the process map and navigation structure of the website. In the process map, there are two states of the system: Log in state and visiting browser state, followed by some essential functions. The navigation structure clearly shows the basic structure of the webpage(which pages are accessible from other pages).

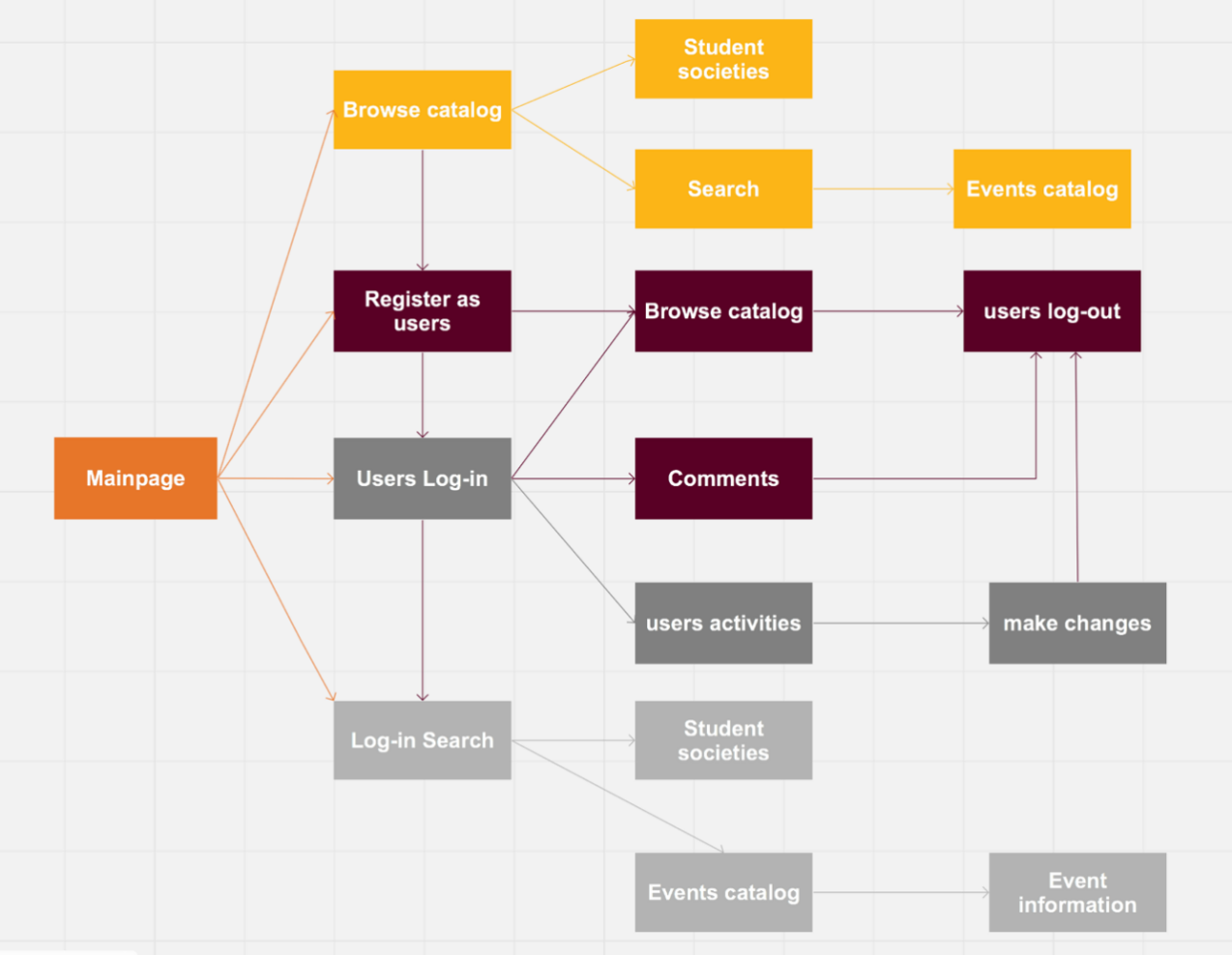
****

Figure 5. Processing map

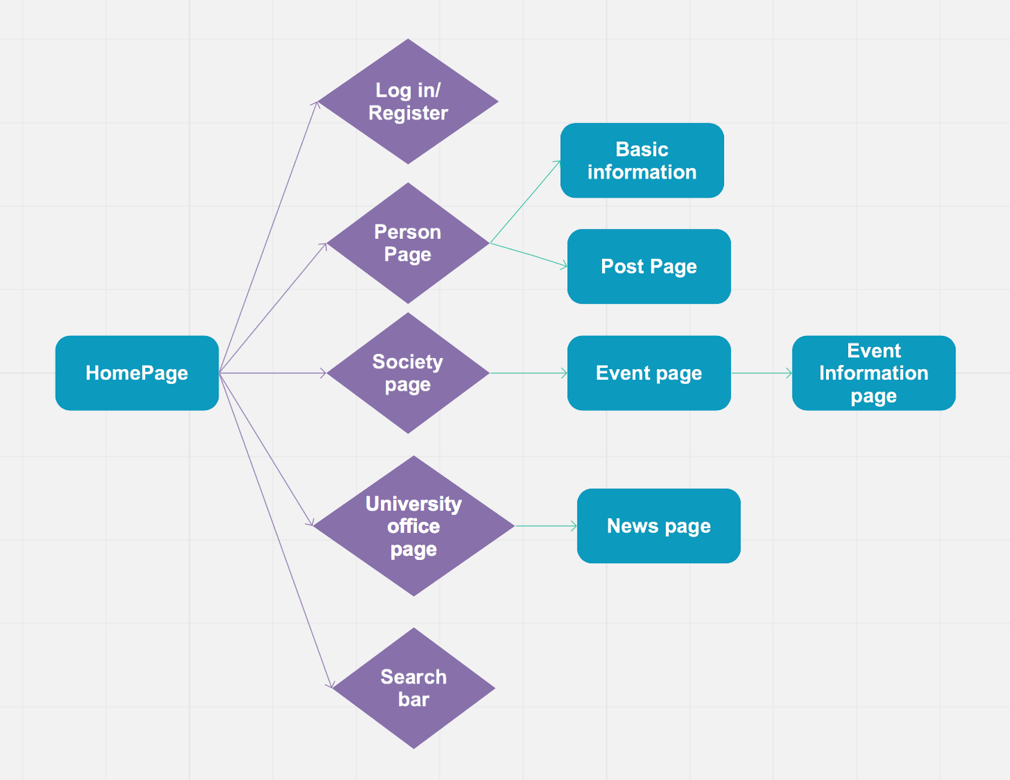


Figure 6. navigation structure chart

Chapter 3

Use Case

**3.1 Use case Diagram**

Here are a use case diagram displaying the relationship between users and how they could use the system properly, followed by a sequence diagram to augment the use case diagrams. The system would require to have three main types of users and an operator. Compared to the original use case diagram, there are three main adjustments.

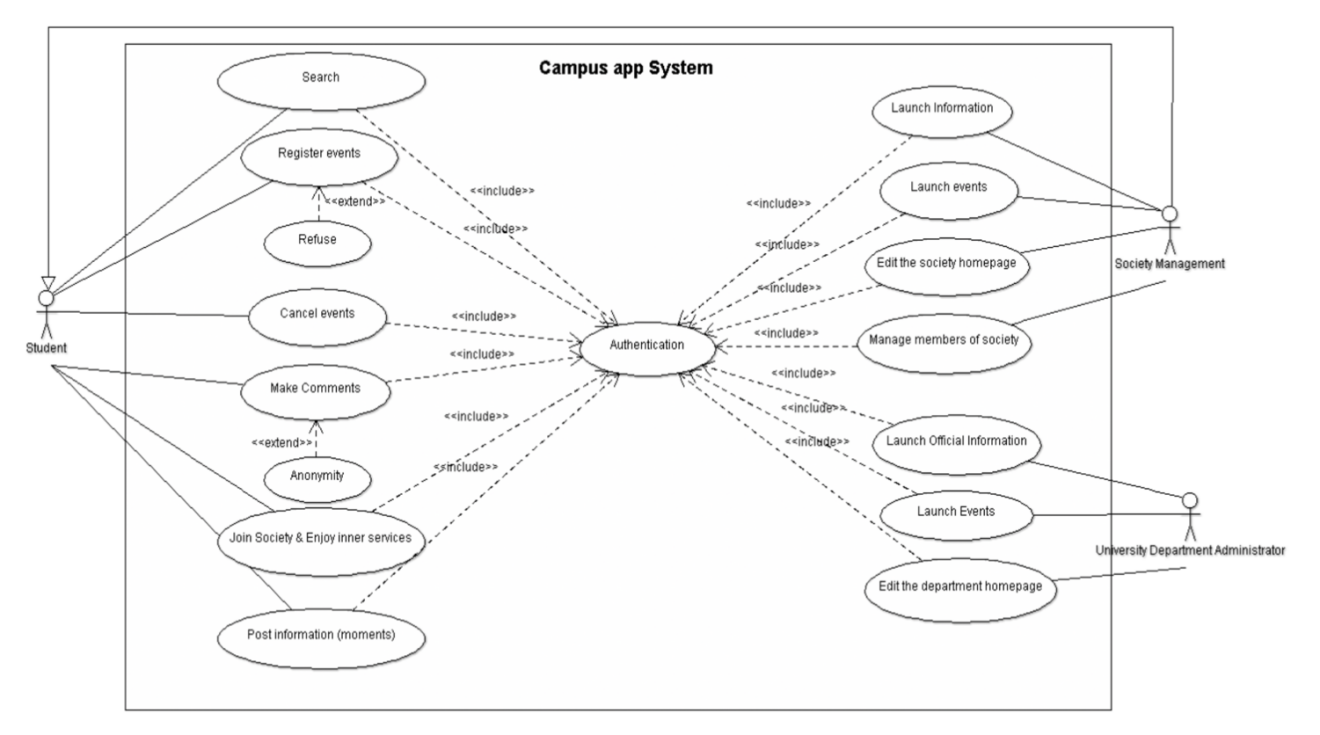


Figure 7: Use case Diagram(1)

Firstly, the website provides a 'Search' function to all kinds of users no matter logged in, and also we change the user 'Student' to 'Website Visitor'. Secondly, the diagram adds a new user named 'Server Operator' who is mainly responsible for maintaining the website(including providing website consultation, launching website maintenance information etc.). There are four use cases of the Server operator: Check application material, Deleting message, Contact and Launch website notification. More details could be found in the use case description part (in the appendix).

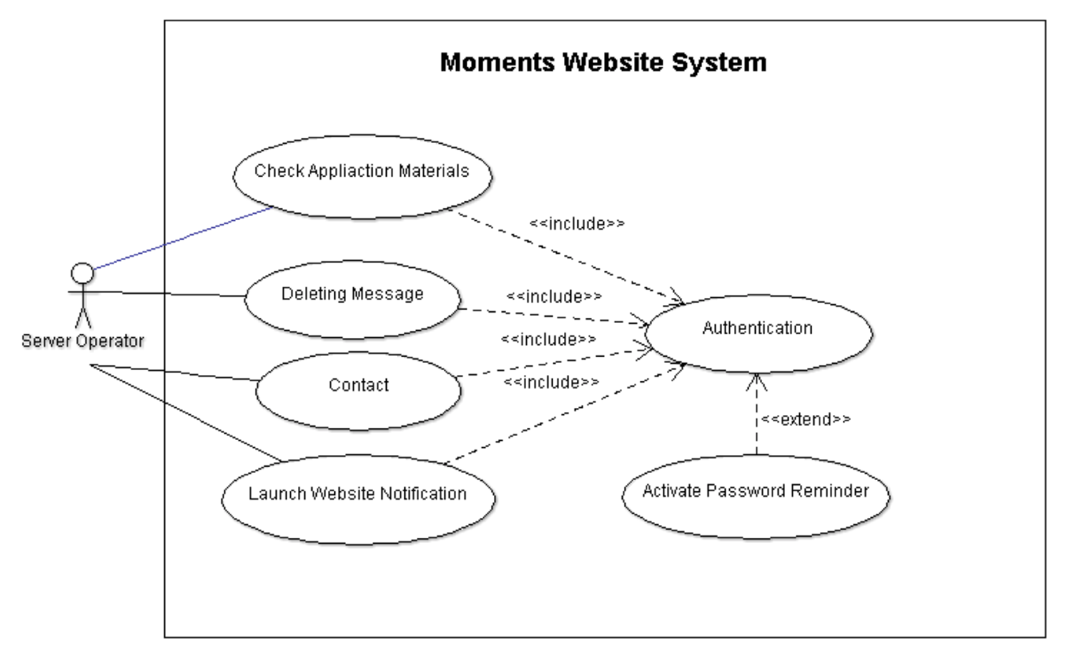


Figure 8: Use case Diagram(2)

Finally, the new use case diagram adds an extendable use case to 'Authentication'. We set the 'Activate password reminder' function to help users remind the correct credentials or reset the password directly.

**3.2 Sequence diagrams**

The following UML sequence diagrams shows the flow of activity for the main user activity: register and login, join club and take part in event and the club manager create the event.

**Register and login**

The diagram below shows the process of register and login. One user: the students. Three objects: the client for each user the server of the whole system and the database. The students first enter the username and password from client to server. The server sends the information to the database. After the information adds into the database, the students register successfully. About the login section, the user sends the request from the client and server, and the server sends the message to the database to check the information. Then the database sends a permit or rejects response back to the server. Then the server passes the message to the receiver. If the student information is verified, the student login successfully.

****

Figure 9:Register and login sequence Diagram

**Join the clubs and events**

The diagram below shows the case when a student attempts to join a club or an event. The student sends request from client to the server. After the server sends the request to database and database record the information return the successful message to the student’s client. The fail case will return refuse information to the user.

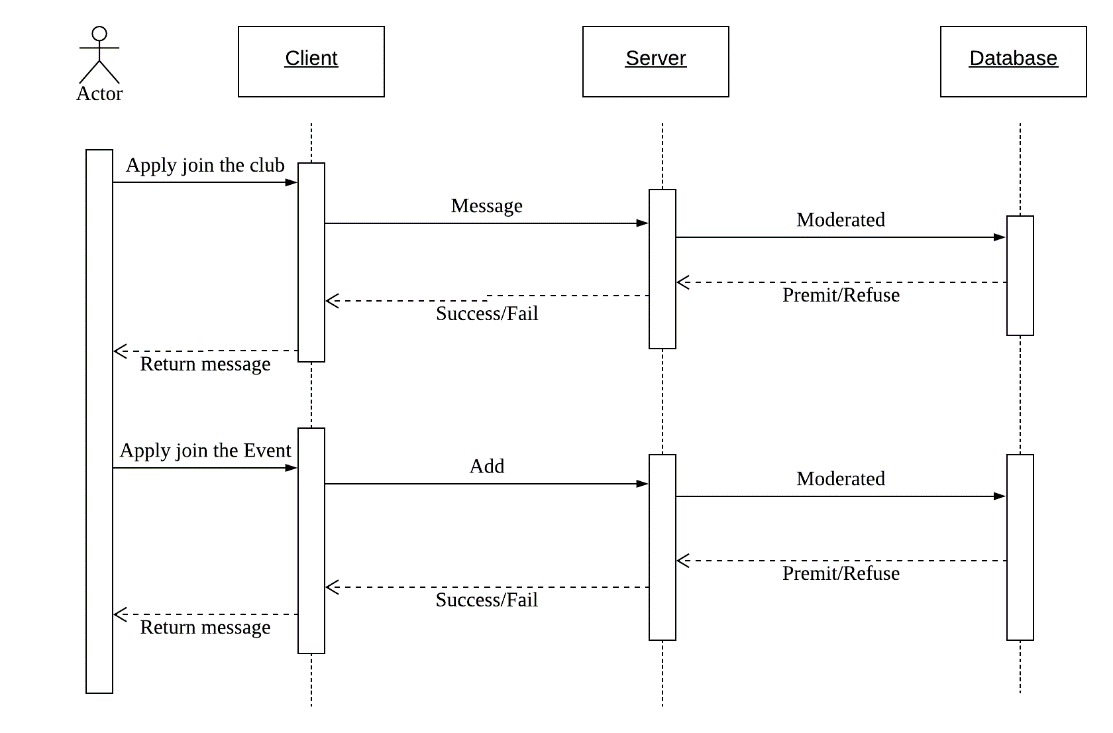


Figure 10:Join the clubs and events sequence Diagram

**Create event**

The diagram below shows the case when a club wants to create an event. The club manager will send the request of forming a club account from the client to the server. The database records the information of the event and permits the club to publish the new event on the website.

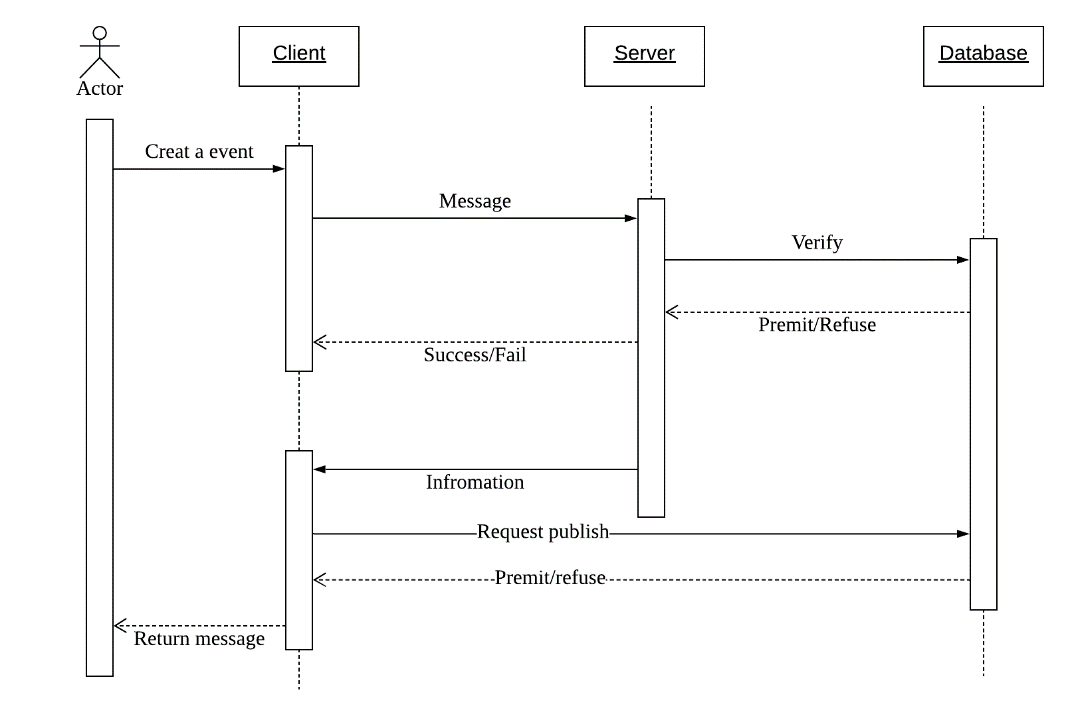
****

Figure 11:Events sequence Diagram

Chapter 4

Use Interface Design

This part will set of some mock examples of User Interface for our Moments’ website, relating to the basic homepage, society page, log in/ register page, event page, personal page, chatting room part etc. A detailed navigation and explain to the web page elements to the web page will also be delivered in this part.

* 1. **UI: Homepage**

**4.1.1 Brief Description**

This part will set of some mock examples of User Interface for our Moments’ website, relating to the basic homepage, society page, login/ register page, event page, personal page, chatting room part etc. Detailed navigation and explain to the web page elements to the web page will also be delivered in this part.

**4.1.2 UI Prototype**

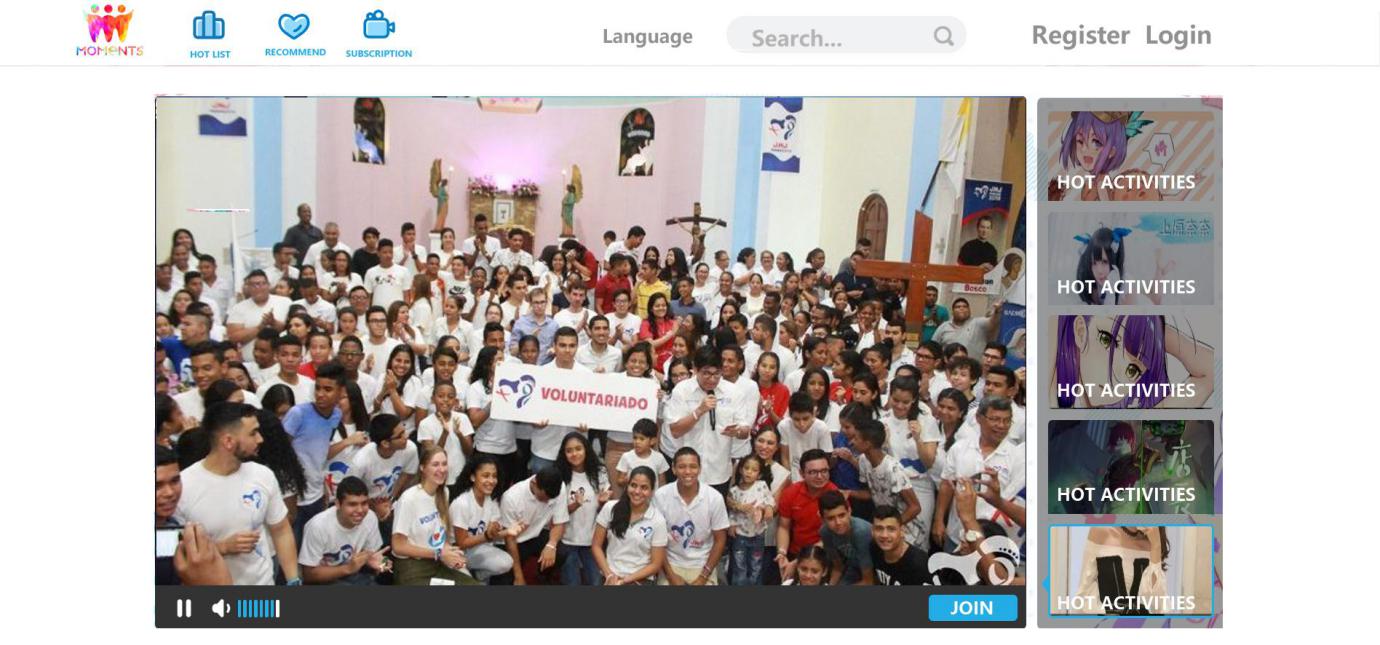


Figure 12.HomePage(1)

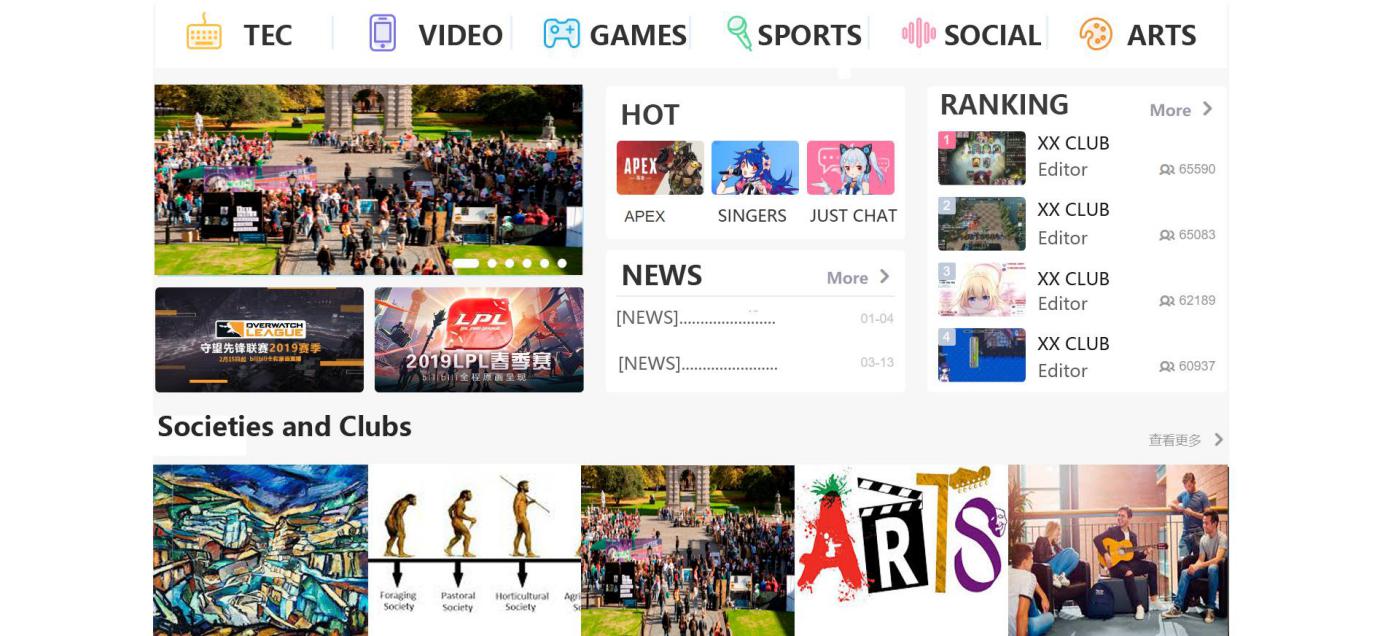


Figure 13. HomePage(2)

**4.1.3 Navigation**

From the UI prototype, we could see that the homepage is mainly divided into three parts. First, the navigation bar is set on the top of the homepage, provides the functions of logging in, registering, website language switching and searching. Also, users could be looking through the hottest event ordering list, the recommendation content to the user and subscriber content by clicking Hot List, Recommend and Subscription button respectively.

The second section is the hot content part, followed the navigation bar. This part mainly pushes the most popular contents. The contents are displayed with the form of an image panel. The images will scroll and update to the latest events content automatically. Clicking the image will lead users to jump to the webpage about the particular event.

Next part will deliver societies and clubs. A series of keywords(e.g. games, arts) are listed in a row firstly. Users could scan the societies they are interested in by pressing the topic button. Moreover, to give a more straight impression about the societies to the website visitor, we also set each society as a panel with an image which could be the link connected to the specific society page.

**4.1.4 Elements**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| *Moments* Logo | JLabel |
| Hot List & Recommend & Subscription | JLabel |
| Language | JLabel |
| Search bar | JTextField / JButton |
| Register & Login | JLabel / JButton |
| TEC/ VIDEO/GAMES etc... | JLabel |

* 1. **UI: Login and Register**

**4.2.1 Brief Description**

The following prototype shows the login and register interface. The Login page allows users to log into the system using their accounts and passwords.

**4.2.2 UI Prototype**

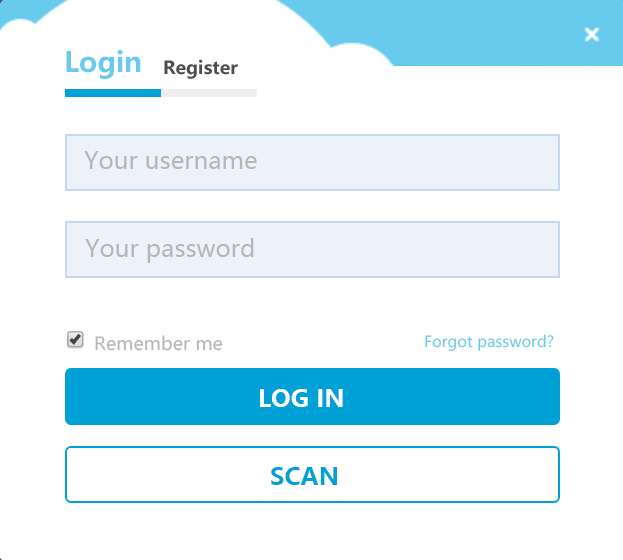


Figure 14. Log in

* + 1. **Navigation**

By clicking the *Login* button on the right top of the webpage, users could enter the login interface. In this interface, users could log into the system by clicking the *LOG IN* button. User could also use the QR code to login after pressing *SCAN* button. The userID and password could be kept by ticking *Remember me.* Moreover, if the user forget his/her password, he/she could click *Forget Password* to find their pin.

**4.2.4 Element**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| Username & Password | JTextField |
| Register & Login & Scan | JLabel / JButton |
| Remember me | JButton |
| Forget password | JLabel |

* 1. **UI: Personal society/activity quick navigation**

**4.3.1 Brief Description**

Personal society/activity quick navigation is an interface which helps the user to find their societies and their latest events quickly. Personal society/activity quick navigation will appear when the user clicking ‘mine’ button on the right top corner after the user have logged into the system.

**4.3.2 UI Prototype**

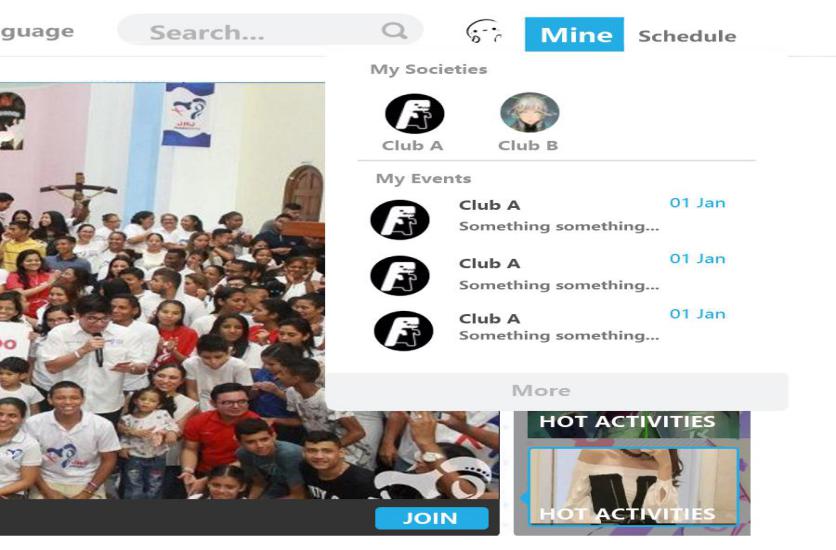


Figure 15.Personal quick navigation

**4.3.3 Navigation**

When the logged user clicks ‘mine’ button, there will be a drop down bar to show all societies and related new activities to clubs. When users click the club head portrait(or labels) of the club, the webpage will jump to the particular society webpage or the event webpage.

**4.3.4 Element**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| Mine | JLabel/ JButton |
| Club | JLabel / JButton |

* 1. **UI: Comment**

**4.4.1 Brief Description**

This mock-up shows the comment section of the homepage. The comments are usually about the society, department or the specific event. The comment section could appears on the bottom society homepage, department homepage and activity homepage.

**4.4.2 UI Prototype**

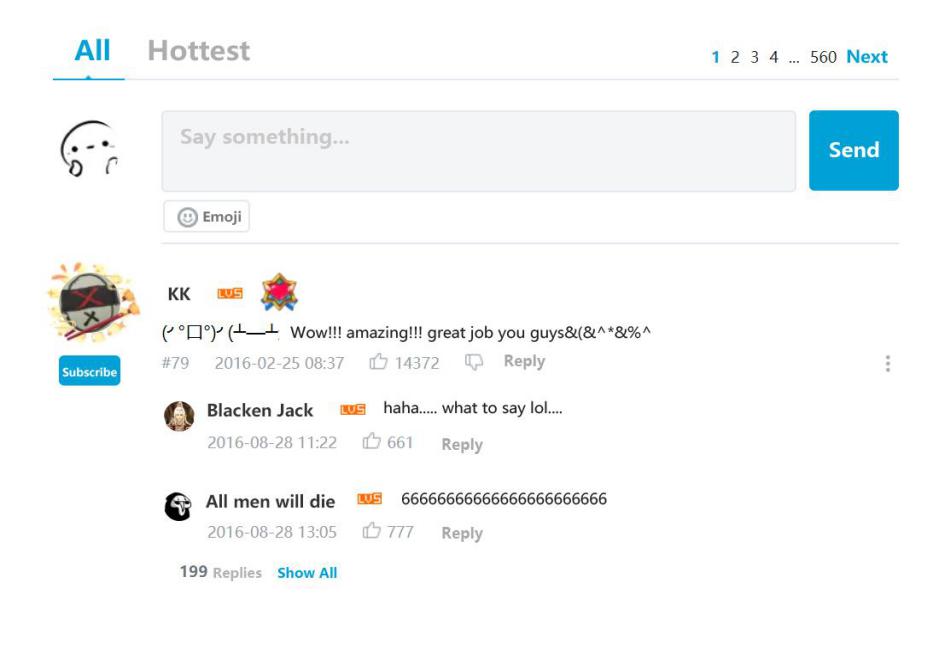


Figure 16.Comment

**4.4.3 Navigation**

The comment section is divided into two parts. The first half part shows that the user could post a comment and the second part displays other comments written by other users. When the user finished their comments, they can share their ideas with others by clicking the ‘Send’ button. Users could also agree to others’ comments by pressing ‘like’ button and discuss with other users by choosing the ‘Reply’ button. Comments could also be pushed forward when the number of likes for comments increased.

**4.4.4 Element**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| Send | JLabel/ JButton |
| Like | JLabel / JButton |
| Reply | JLabel / JButton |
| Say something | JTextArea |
| Subscribe | JLabel / JButton |
| Hottest & All | JLabel / JButton |

* 1. **UI:Search Result**

**4.5.1 Brief Description**

This part shows the jumped interface after searching the keyword. The interface shows all possible results related to the keyword (Including societies and clubs, events etc).

**4.5.2 UI Prototype**

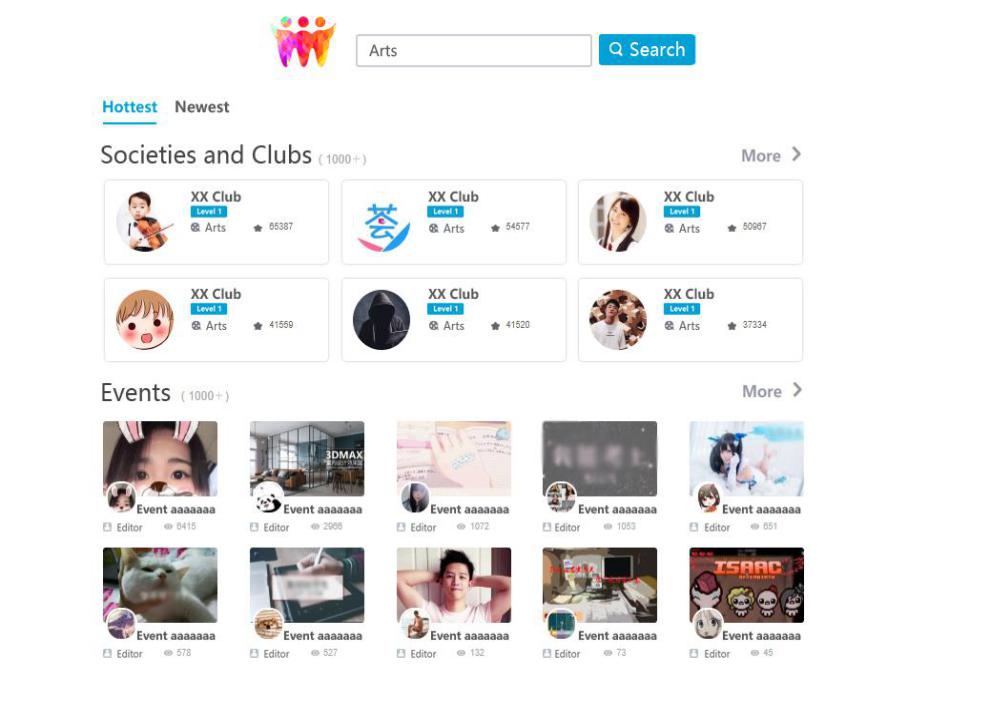


Figure 17. Search result

**4.5.3 Navigation**

The searching result interface will come out after searching the keyword and clicking the ‘search’ button. When the search-result interface shown, users could compare the search results with the information the user wants to find. If there are one matches the target information, the user could click the image/label to find more details about the particular information. If users do not find the target information, users can click ‘more’ to let the website display more information.

**4.5.4 Element**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| Search | JLabel/ JButton |
| More | JLabel / JButton |
| Newest & Hottest | JLabel / JButton |
| SearchText | JTextArea |

* 1. **UI: Society Page** 
     1. **Brief Description**

This mock-up displays the information of one particular society. Users could enter the particular society page when they click the society label or head portrait.

* + 1. **UI Prototype**



Figure 18.Society Page

* + 1. **Navigation**

The society web page has two main sections. The left part displays the social events or activities, and the right-hand side has a society group chatting room which provided for users to burn questions and share ideas with each other. On the left-hand side, the images of society(or club) events will scroll continuously. On the right side, the users who have logged in and joined the society could chat with their friends in the group chat. All the message sent by the user are listed on the right, and all messages from other people are listed on the left. All messages are displayed with time order.

**4.6.4 Element**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| Like | JLabel/ JButton |
| Share | JLabel / JButton |
| Send | JLabel / JButton |
| Moments | JLabel / JButton |
| Home | JLabel / JButton |
| Events | JLabel / JButton |
| Videos | JRadioButton |
| WritingBox | JTextField |
| Setting | JLabel / JButton |

* 1. **UI:Event Page**

**4.7.1 Brief Description**

This mock-up displays the information of one particular event. Users could enter the particular event page when they click the event label or head portrait.

* + 1. **UI Prototype**

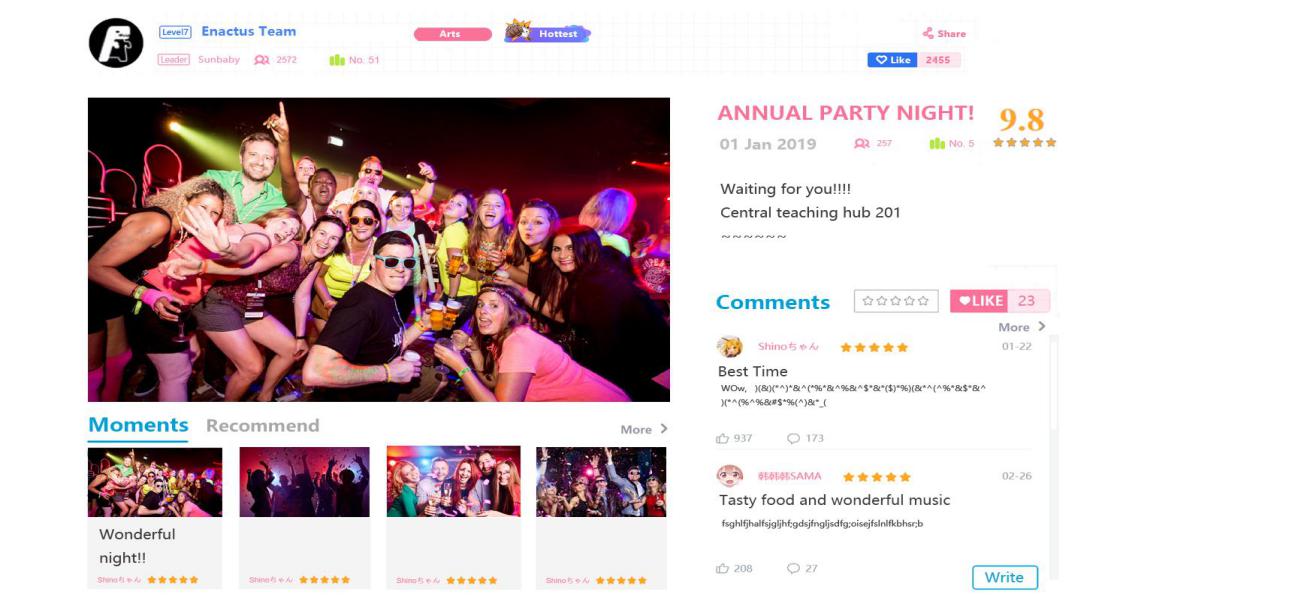


Figure 19.Event Page

**4.7.3 Navigation**

The event web page is split into two main sections. The left part displays the event photos. The images of the events scroll continuously just as the society web page. Under the display of event photos, there are some recommendation events provided here. The user could click Recommend button to view more similar event. On the right-hand side of the webpage, the comments about the event will be listed. Users could write comments by clicking the ‘write’ button. To see more details about the comment section, could see the UI: Comment section.

**4.7.4 Element**

|  |  |
| --- | --- |
| Webpage Element | GUI Element |
| Like | JLabel/ JButton |
| Share | JLabel / JButton |
| Write | JLabel / JButton |
| Moments | JLabel / JButton |
| Recommend | JLabel / JButton |
| More | JLabel / JButton |
| Comments | JRadioButton |

Chapter 5

Algorithm design (Pseudocode)

**5.1 Search matching algorithm**

This algorithm will be used in the search bar function. When the users input the key words in the search bar, the system could quietly match the related socities or events.

**5.1.1 Method 1**

String original

String pattern

Integer count = 0

do

for( u = 0; v < len(original) - len(pattern); u++ )

if( original[i] == pattern[0] )

for( u = 1; v < len(pattern); v++ )

while( orginal[u + v] == label[v] )

++count

**5.1.2 Method 2**

Following is the econd implementation method about string match algorithm:

String\_Match(A,n,B,m)

Input: A(a string the length is n) and B (a string the length is m), A is the original and B is PATTERN.

Output: start(The first subscript of B in A, that is, B is the first subscript of the substring starting from A[start] in A )

Begin

     i = 1, j =1,  start = 0;

     while  start == 0 and i <= n do

           if B[j] == A[i] then

                 j = j + 1;

                 i = i + 1;

           else

                j = next[j] + 1;

               if j == 0 then

                   j = 1;

                   i = i + 1;

               end if;

           end if;

           if j == m + 1 then

              start = i - m;

           end if;

     end while;

End;

**5.1.3 Label and event matching**

Each student can choose their interests at the registration and in their information list. We can give each society a label, and each event will include a list of labels which is a list of keywords about the event. To work out which events to recommend to the student. One of the factors is relevancy the seething and club of events.

String label

String pattern, firstPattern, lastPattern

Integer count = 0

Boolean match = false

do

for(pattern = firstPattern to pattern = lastPattern)

while( match = (pattern != label) )

pattern.nextPattern()

return match

**5.2 User-based collaborative filtering**

Similar statistics are used to obtain neighbouring users with similar hobbies or interests, so it is called User-based collaborative filtering or Neighbor-based Collaborative Filtering.

**5.2.1 Method steps**

1. **Collect user information**

Collect information that can represent the interests of the user. The general website system uses scoring or giving evaluations, which is called "active scoring." The other is “passive scoring”, which is based on the user's behaviour pattern, and the user completes the evaluation instead of the user and does not require the user to score or input evaluation data directly.

1. **Nearest neighbour search (NNS)**

The starting point of user-based collaborative filtering is that another group of users with the same interests and interests of users is to calculate the similarity between the two users. For example, find n users with similar interests and A, and use their score on M as the score prediction of A to M. Generally, different algorithms are selected according to different data. Currently, the similarity algorithms used are the Pearson Correlation Coefficient, Cosine-based Similarity, Adjusted Cosine Similarity (adjusted Cosine Similarity). ).

The User-Based collaborative filtering algorithm first searches for other users similar to the new user based on the user's historical behaviour information. At the same time, based on the evaluation information of other similar users for other items, the current new user may like the item.

Given the user score data matrix R, the user-based collaborative filtering algorithm needs to define a similarity function s: U×U→R to calculate the similarity between users and then calculate the recommendation result based on the score data and the similarity matrix.

1. **How to choose the appropriate similarity calculation method**

In collaborative filtering, an important part is how to choose the appropriate similarity calculation method. The two commonly used similarity calculation methods include the Pearson correlation coefficient and cosine similarity. The formula for calculating the Pearson correlation coefficient is as follows:

Where denotes an item, such as a envet; denotes a set of items evaluated by the user ; denotes a set of items evaluated by the user; denotes a score of the user for the item ; denotes a score of the user for the item ; represents the average rating of user ; represents the average rating of user.

The formula for calculating the cosine similarity is as follows:

1. **Calculate the predicted score of user for unrated items**

Another important step is to calculate the predicted score of users for unrated items. First, according to the similarity calculation in the previous step, the neighbor set of the user is found, where represents a neighbor set and represents a user set. Then, combined with the user rating data set, predicted the score of the user to the , and the calculation formula is as follows:

Where represents the similarity between user and user .

**5.3 Trending association algorithm**

The main idea is to calculate the popularity of each association, and the program will select the top 10 communities on top of the popular community.

To calculate the popularity of one association, let i denotes th day before now. Let w(i) indicates the number of times in the th day before now, and the popularity of the community is . In that case, satisfies this relation:

The above formula indicates that for each community, the program takes the number of searches of the association in the last five days and then calculates the total weight according to a certain weight, and then takes ten most significant most massive total weights as the popular association.

In order to execute algorithm, **search** table is required to be created and it contains following column

association\_ID: as foreign key of **association** table

association\_name: the name of association corresponding association\_ID

search\_date: time period corresponding to the number of lookups

search\_number: the number of times the community is searched corresponds to **search\_date**

The following **physical table structure** is

domain General\_number1 UNSIGNED integer maximum integer 65535

domain General\_text2 variable length text string maximum length 50

domain Date datetime format: 0000-00-00 00:00:00

search(

association\_ID General\_number1 NOT NULL,

association\_name General\_text2 NOT NULL,

search\_date: Date NOT NULL DEFAULT: 0000-00-00 00:00:00, search\_number: General\_number3 NOT NULL DEFAULT 0)

Foreign Key association\_ID References association(ID) ON DELETE CASCADE ON UPDATE CASCADE

The update interval is tentatively set to 1 day. If the update time is required to be shortened, the time field in the database also needs to be specifically subdivided.

As for the specific update time, the program is tentatively set at 4 am, because the user may still be active at 0:00 am.

**5.3.1 pseudocode**

static Date date=now //date stores the value of date in now , which is accurate to day

array Integer weight[] // Total weight of the association

/\*

The number of times that the club was searched on a certain day

\*/

array Integer number[]

array String top[] // Used to display the top 10 popular clubs

static update() // Update popular communities and add new tuples to the database

{

connect database;

/\*

Create a heap data structure, set the root element to the maximum value

\*/

build heap;

for(i=1;i<=sum(association);i++)

{

for(j=1;j<=5;j++)

{

number[i]=exec(“SELECT number FROM search WHERE search\_date=date-i;”)

}

weight[i]=5\*number[1]+4\*number[2]+3\*number[3]+2\*number[4]

+number[5]

insertItem(weight[i]) // Insert weight[i] into the heap data structure

name=exec(“SELECT **association\_name** FROM association WHERE **association\_ID**= i;”)

exec(“INSERT INTO search VALUES(i, name,date,0);”)//

}

for(int k=1;k<=10;k++)

{

top[k]=removeMax()

}

display top10 associations

}

Chapter 6

Database Design

**6.1 Entity-relationship diagram**

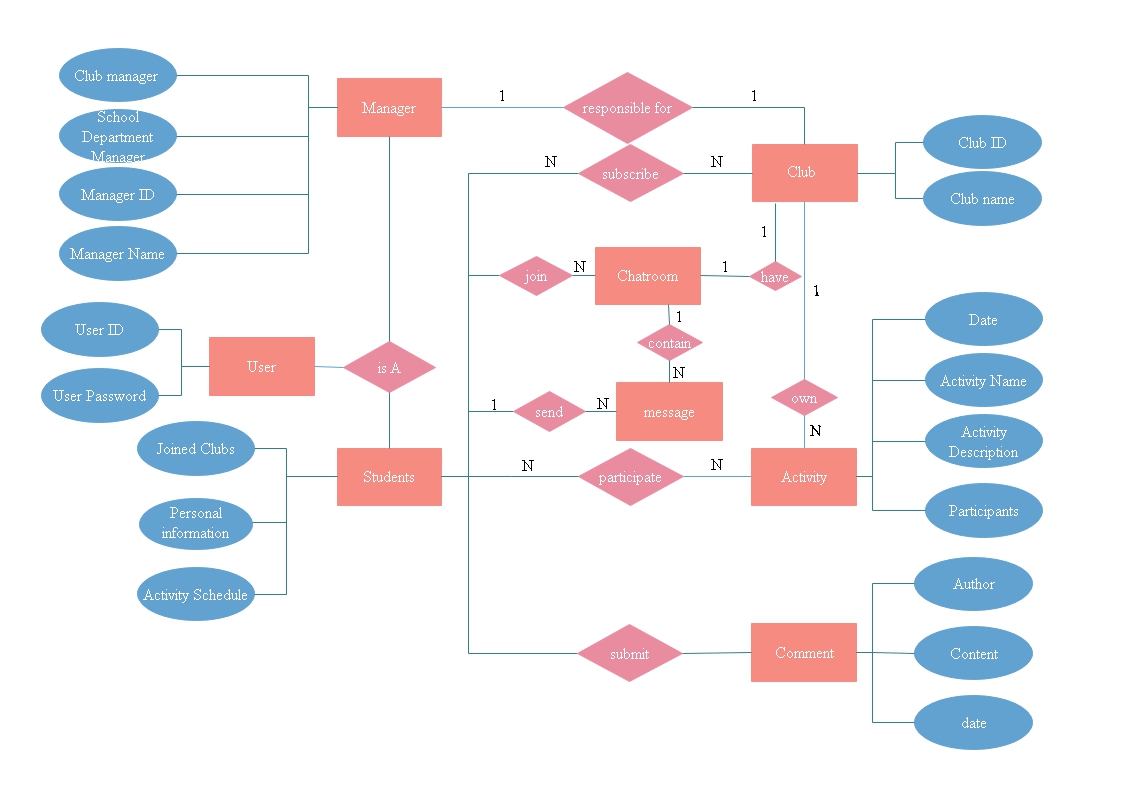
****

Figure 20. Entity-relationship diagram

**6.2 Logical table structure**

|  |
| --- |
| **User**(ID,user\_login,user\_pass,user\_email,user\_registered,display\_name,profile\_name)  **Primary Key**  ID |
| **Usermeta**(umeta\_id,user\_id,meta\_key,meta\_value)  **Primary Key**  umeta\_id  **Foreign Key** user\_id **references** user(ID) |
| **Association**(association\_ID,association\_name,memberNum,association\_description,chatroom\_ID,is\_department)  **Primary Key**  association\_ID  **Foreign Key** chatroom\_ID **references** user(ID) |
| **Association\_touser**(touser\_ID,association\_ID,chatroom\_ID,user\_ID,user\_level,user\_sendLastTime)  **Primary Key**  touser\_ID  **Foreign Key** user\_ID **references** user(ID) |
| **Event**(event\_ID,event\_name,event\_date,event\_description,association\_ID,association\_name,event\_status)  **Primary Key**  event\_ID  **Foreign Key**  association\_ID **references**  association(association\_ID) |
| **Eventmeta**(eventmeta\_ID,event\_ID,meta\_key,meta\_value)  **Primary Key**  eventmeta\_ID  **Foreign Key references** event(event\_ID) |
| **Comment**(comment\_ID,comment\_association\_ID,comment\_event\_ID,comment\_display\_name,comment\_author\_IP,comment\_date,comment\_content,comment\_agent,comment\_user\_ID,)  **Primary Key**  comment\_ID  **Foreign Key** comment\_association\_ID **references** association(association\_ID)  **Foreign Key** comment\_event\_ID **reference** event(event\_ID)  **Foreign Key** comment\_user\_ID **reference** user(ID) |
| **Groupmsg**(msg\_ID,chatroom\_ID,msg\_senderID,msg\_sendername,msg\_senderIP,msg\_date,msg\_content,msg\_agent)  **Primary Key**  msg\_ID  **Foreign Key** msg\_senderID  **references** user(ID) |

**6.3 Physical table structure**

In order to having a better performance, indexing and foreign key will be implemented based on the MySQL database.

**About file organization and index**

Usermeta, event meta, association\_touser table are organised by hash file because the primary key of these tables has no other meaning except to distinguish between tuples. The most critical role of usermeta and event meta is to query for the correspondent relationship between meta\_key and meta\_value based on user\_id or event\_ID. Therefore, using a hash file organisation can speed up query efficiency.

The user, association, association\_touser, event, comment, group msg tables use sequential (sort) files. The primary keys of these tables have no other meaning except distinguishing between tuples with other tables. For example, tables called user have a column namely user\_login. Since the user name cannot be the same, user\_login can be used to distinguish different tuples, but to increase the readability of the database by the administrator, the ID primary keys of these tables are retained. And the primary key of these tables is sorted in increasing order to increase the readability for the administrator's to lookup for the database. If the amount of data is large in the future, the query efficiency will be decreased. Other methods can be used to improve the file structure. Although the readability is reduced, the query efficiency can be improved.

**Index**

User: user\_login serves as the index and display\_name serve as the full-text index

Usermeta: user\_id serves as index

Association:association\_name serves as a full-text index

Association\_touser: association\_ID and user\_ID serve as indexes separately.

Event: association\_ID serve as index and event\_name serve as full-text index.

Eventmata: event\_ID serve as an index

Comment: comment\_association\_ID and comment\_event\_ID serve as indexs separately.

Groupmsg: chatroom\_ID and msg\_date serve as indexes respectively

**Database user view**

About login

CREATE VIEW web\_login as

SELECT ID

FROM user

WHERE user\_login=’admin’ and user\_pass=’123456’

WITH CHECK OPTION;

About viewing user information

CREATE VIEW web\_checkuserinfo as

SELECT a.user\_email, a.display\_name, a.profile\_name, b.meta\_key, b.meta\_value

FROM user a, usermeta b

WHERE a.user\_id=1 and a.user\_id=b.user\_id;

Find a society by name

CREATE VIEW web\_search\_association as

SELECT association\_name

FROM association

WHERE MATCH(association\_name) AGAINST("computer")

WITH CHECK OPTION;

Find activities by name

CREATE VIEW web\_search\_event as

SELECT event\_name

FROM event

WHERE MATCH(event\_name) AGAINST("computer competition 2019")

WITH CHECK OPTION;

View society

CREATE VIEW web\_check\_association as

SELECT association\_name, memberNum, association\_description

FROM association

WHERE association\_id=’1’

WITH CHECK OPTION;

View event

CREATE VIEW web\_check\_event as

SELECT a.user\_email, a.display\_name, a.profile\_name, b.meta\_key, b.meta\_value

FROM association a, event b

WHERE a.user\_id=1 and a.user\_id=b.user\_id;

Join the chat room

CREATE VIEW web\_joinchat as

SELECT chatroom\_ID

FROM association

WHERE association\_id=’1’

WITH CHECK OPTION;

View chat room messages and users

CREATE VIEW web\_check\_event as

SELECT a.user\_ID,a.user\_level, b.msg\_sendername,b.msg\_date,b.msg\_content

FROM association\_touser a, groupmsg b

WHERE a.chatroom\_ID=1 and a.chatroom\_ID=b. chatroom\_ID and TIMESTAMPDIFF(DAY, NOW(),a)<=3;

**Design security mechanisms**

In the website, apart from the website administrators, all users are integrated into a user entity (here named test) to undertake the manipulation including querying the community, viewing information, and modifying the description. The user of that database has manipulated permissions of SELECT, INSERT, UPDATE, and DELETE for the table. However, when the database operates, the backend of the website will strictly check whether the user has this permission to perform this operation. At the same time, the backend will check for user input (if any) to prevent the database from malicious attacks. Afterwards, if there are special requirements, other permissions will be given to this user, and this user will be given a strong password.

A website developer and database designer will directly use root to modify the schema library designed by the website. The required tables will be created in advance, so that the test user does not have the permission to modify the database structure.

**About redundancy**

Considering the efficiency of queries and other factors, the database table for our web will not conform to the third standard form, because conforming to the third standard form will result in more queries needed to join multiple tables, which will result in slower query speed. More amount of data will result in a more noticeable effect.

For example, the association\_name attribute in the event table is redundant. However, if it is cancelled, database administrators have to combine two tables for the query.

Alternatively, when querying for the message record of the chat room, the display\_name of the messaging user needs to be displayed, so a msm\_sendername field is added into the group msg table involved. Although the redundancy is increased, the user table is not required to be connected. .

**About Interface design**

Because we use the php language to write the website backend and the database is written in MySQL, and the version after php5.0 provides the mysqli interface to connect to the MySQL database. So we use the mysqli interface to connect to the database

In the php language, the statement that connects to the database is

$conn = new mysqli($hn,$un,$pw,$db);

Where $hn is the database address, $un is the username, $pw is the password, and $db is the schema.

*The first table presented the general constraints for database domain parameters*

Name of domain constraints

domain General\_number1 UNSIGNED integer maximum integer 65535

domain General\_number2 UNSIGNED integer maximum integer 255

domain General\_number3 UNSIGNED integer maximum integer 1.67e+7

domain General\_number4 UNSIGNED integer maximum integer 4.29e+9

domain User\_name variable length character string maximum length 20

domain General\_character1 variable length character string maximum length 255

domain General\_character2 variable length character string maximum length 100

domain General\_character3 variable length character string maximum length 30

domain General\_text1 variable length text string maximum length 65535

domain General\_text2 variable length text string maximum length 50

domain Date datetime format: 0000-00-00 00:00:00

Table 1.Domain constraints

**6.3.1 User\_table design**

|  |
| --- |
| user(  ID General\_number1 NOT NULL AUTO\_INCREMENT,  user\_login User\_name NOT NULL,  user\_pass General\_character1 NOT NULL,  user\_email General\_character2 NOT NULL,  user\_registered Date NOT NULL,  display\_name General\_character3 NOT NULL,  profile\_name General\_character2 NOT NULL)  Primary Key ID |
| usermeta(  umeta\_id General\_number1 NOT NULL AUTO\_INCREMENT,  user\_id General\_number1 NOT NULL,  meta\_key General\_character1 NOT NULL,  meta\_value General\_text1 NOT NULL)  Primary Key umeta\_id  Foreign Key user\_id References user(ID) ON DELETE CASCADE ON UPDATE CASCADE |

**6.3.2 Society\_table design**

|  |
| --- |
| association(  association\_ID General\_number1 NOT NULL AUTO\_INCREMENT,  association\_name General\_text2 NOT NULL,  memberNum General\_number1 NOT NULL DEFAULT 1,  association\_descriptionGeneral\_text1 NULL,  chatroom\_ID General\_number1 NULL,  is\_department General\_number2 NOT NULL DEFAULT 0)  Primary Key association\_ID  Foreign Key chatroom\_ID References user(ID) ON DELETE SET NULL ON UPDATE CASCADE |
| association\_touser(  touser\_ID General\_number1 NOT NULL AUTO\_INCREMENT,  association\_ID General\_number1 NOT NULL,  chatroom\_ID General\_number1 NULL,  user\_ID General\_number1 NOT NULL,  user\_level General\_number2 NOT NULL DEFAULT 1,  user\_sendLastTime Date NOT NULL DEFAULT: 0000-00-00 00:00:00)  Primary Key touser\_ID  Foreign Key association\_ID Reference association(association\_ID) ON DELETE CASCADE  Foreign Key user\_ID References user(ID) ON DELETE CASCADE |

**6.3.3 Event\_table design**

|  |
| --- |
| event(  event\_ID General\_number3. NOT NULL AUTO\_INCREMENT,  event\_name General\_text2 NOT NULL,  event\_date Date NOT NULL DEFAULT: 0000-00-00 00:00:00  event\_description General\_text1 NULL,  association\_ID General\_number1 NOT NULL,  association\_name General\_text2 NOT NULL,  event\_status General\_number2 NOT NULL DEFAULT 0)  Primary Key event\_ID  Foreign Key association\_ID Reference association(association\_ID) ON DELETE CASCADE  eventmeta(  eventmeta\_ID General\_number3 NOT NULL AUTO\_INCREMENT,  event\_ID General\_number3 NOT NULL,  meta\_key General\_character1 NOT NULL,  meta\_value General\_text1 NOT NULL)  Primary Key eventmeta\_ID  Foreign Key event\_ID Reference event(event\_ID) ON DELETE CASCADE ON UPDATE CASCADE |

**6.3.4 Comment\_table design**

|  |
| --- |
| comment(  comment\_ID General\_number4 NOT NULL AUTO\_INCREMENT,  comment\_association\_ID General\_number1 NOT NULL,  comment\_event\_ID General\_number3 NOT NULL,  comment\_display\_name General\_character3 NOT NULL,  comment\_author\_IP General\_character2 NULL,  comment\_date Date NOT NULL DEFAULT: 0000-00-00 00:00:00,  comment\_content General\_text1 NULL,  comment\_agent General\_character1 NULL  comment\_user\_ID General\_number1 NOT NULL)  Primary Key comment\_ID  Foreign Key comment\_association\_ID Reference association(association\_ID) ON DELETE NO ACTION ON UPDATE CASCADE  Foreign Key comment\_event\_ID Reference event(event\_ID) ON DELETE NO ACTION ON UPDATE CASCADE  Foreign Key comment\_user\_ID Reference user(ID) ON DELETE CASCADE ON UPDATE CASCADE |

**6.3.5 Chartroom\_table design**

|  |
| --- |
| groupmsg(  msg\_ID General\_number4 NOT NULL AUTO\_INCREMENT,  chatroom\_ID General\_number1 NULL,  msg\_senderID General\_number1 NOT NULL,  msg\_sendername General\_character3 NOT NULL,  msg\_senderIP General\_character2 NULL,  msg\_date Date NOT NULL DEFAULT: 0000-00-00 00:00:00,  msg\_content General\_text1 NULL,  msg\_agent General\_character1 NULL)  Primary Key msg\_ID  Foreign Key msg\_senderID Reference user(ID) ON DELETE NO ACTION ON UPDATE CASCADE |

**6.4** **Transaction Matrix**

**Data Entry**

1. Enter the details of a new client.
2. Enter the details of a new society.
3. Enter a new activity to a society.
4. Enter the details of a new activity.
5. Enter a new member to a society.
6. Enter a new comment to an activity.
7. Enter a group chat message.

**Data Update/Deletion**

1. Update/delete the details of a client.
2. Update/delete the details of a society.
3. Update/delete the details of an activity.
4. Update/delete the details of a comment.

**Data Queries**

The database should be capable of supporting the following sample queries:

1. List the details of a society.
2. List the details of an activity.
3. List the members of a society.
4. List the societies that a client has joined in.
5. List the activities that a client has participated in.
6. List the comments of an activity.
7. List the group chat dialogues.
8. List the most popular activities.
9. List the most popular societies.

**I=Insert, R=Read, U=Update, D=Delete**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Transaction/ Table** | **(a)** | | | | **(b)** | | | | **(c)** | | | | **(d)** | | | |
|  | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** |
| **User** | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Usermeta** | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Association** |  |  |  |  | x |  |  |  |  | x | x |  |  |  |  |  |
| **Association\_touser** |  |  |  |  | x |  |  |  |  | x | x |  |  |  |  |  |
| **Event** |  |  |  |  |  |  |  |  | x |  |  |  | x |  |  |  |
| **Eventmeta** |  |  |  |  |  |  |  |  | x |  |  |  | x |  |  |  |
| **Comment** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Groupmsg** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Transaction/ Table** | **(e)** | | | | **(f)** | | | | **(g)** | | | | **(h)** | | | |
|  | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** |
| **User** |  | x | x |  |  | x |  |  |  | x |  |  |  | x | x | x |
| **Usermeta** |  | x | x |  |  | x |  |  |  | x |  |  |  | x | x | x |
| **Association** | x |  |  |  |  | x |  |  |  |  |  |  |  | x | x | x |
| **Association\_touser** | x |  |  |  |  | x |  |  |  |  |  |  |  | x | x | x |
| **Event** |  |  |  |  |  | x | x |  |  |  |  |  |  | x | x | x |
| **Eventmeta** |  |  |  |  |  | x | x |  |  |  |  |  |  | x | x | x |
| **Comment** |  |  |  |  | x |  |  |  |  |  |  |  |  | x | x | x |
| **Groupmsg** |  |  |  |  |  |  |  |  | x |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Transaction/ Table** | **(i)** | | | | **(j)** | | | | **(k)** | | | | **(l)** | | | |
|  | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** |
| **User** |  | x |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| **Usermeta** |  | x |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| **Association** |  |  | x | x |  | x |  |  |  |  |  |  |  | x |  |  |
| **Association\_touser** |  |  | x | x |  | x |  |  |  |  |  |  |  | x |  |  |
| **Event** |  | x | x | x |  |  | x | x |  | x | x |  |  | x |  |  |
| **Eventmeta** |  | x | x | x |  |  | x | x |  | x | x |  |  | x |  |  |
| **Comment** |  | x | x | x |  | x | x | x |  |  | x | x |  | x |  |  |
| **Groupmsg** |  |  |  |  |  |  |  |  |  |  |  |  |  | x |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Transaction/ Table** | **(m)** | | | | **(n)** | | | | **(o)** | | | | **(p)** | | | |
|  | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** |
| **User** |  | x |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| **Usermeta** |  | x |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| **Association** |  | x |  |  |  | x |  |  |  | x |  |  |  |  |  |  |
| **Association\_touser** |  | x |  |  |  | x |  |  |  | x |  |  |  |  |  |  |
| **Event** |  | x |  |  |  |  |  |  |  |  |  |  |  | x |  |  |
| **Eventmeta** |  | x |  |  |  |  |  |  |  |  |  |  |  | x |  |  |
| **Comment** |  | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Groupmsg** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Transaction/ Table** | **(q)** | | | | **(r)** | | | | **(s)** | | | | **(t)** | | | |
|  | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** | **I** | **R** | **U** | **D** |
| **User** |  | x |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| **Usermeta** |  | x |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| **Association** |  |  |  |  |  |  |  |  |  | x |  |  |  | x |  |  |
| **Association\_touser** |  |  |  |  |  |  |  |  |  | x |  |  |  | x |  |  |
| **Event** |  | x |  |  |  |  |  |  |  | x |  |  |  | x |  |  |
| **Eventmeta** |  | x |  |  |  |  |  |  |  | x |  |  |  | x |  |  |
| **Comment** |  | x |  |  |  |  |  |  |  | x |  |  |  |  |  |  |
| **Groupmsg** |  |  |  |  |  | x |  |  |  |  |  |  |  |  |  |  |

**6.5 Data Dictionary**

|  |  |  |  |
| --- | --- | --- | --- |
| Entity Name | Description | Aliases | Occurrence |
| User | A person who use the web | Another user | A user can join a society and can take part in an event and can quite the society and be the manager of the society |
| Society | A group of the same person who has the same hobby | Club | The society allows people join the society and can make an event. |
| Events | A society make an event at some time and some place. | Party | Which is take by a society and accept the people comment who has taken part the events. |
| Comments | People can make comments after join an event. | Review | When the people take part in an event and he can make comments. |
| Message | The record of the message in the chart room | Record | People can discuss in the chart room |
| Manager | The person who manager the club. | Administrator | Manager can manage the club, the people in the club and the events. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute name | Description | Aliases | Found in entity | Occurrence |
| user\_ID | Uniquely identifiers the users | primary key | user | Every user has an user ID |
| user\_login | Username of the user use to login. | Username | user | Every user has an username |
| user\_pass | The password of the user | Password | user | Evert user has an password |
| user\_email | The email of the user | Email | user | Every user has an email |
| user\_Name | The user’s name | Name | user | Every user has a name |
| user\_gender | The user’s gender | sex | user | Every user has a gender |
| user\_birthday | Date of birth | DOB, birthdate | user | Every user has a date of birth |
| association\_ID | Uniquely identifiers the society | Society | society | Every society has an user ID |
| association\_name | Special name of every society | Society name | society | Every society has a name. |
| memberNum | The number of society member | Member number | society | Every society has different member number |
| association\_description | The summery of the society. | Description | society | The summery of the description to tell people what they do |
| chatroom\_ID | Uniquely identifiers the society chatroom . | Chatroom | society | Every society has a chartroom. |
| event\_ID | Uniquely identifiers the events | Event | event | Every event has a id, |
| event\_name | The subject of the event | Event name | event | Event event has a subject. |
| event\_date | The date of the events | Date, time | event | Event need to ensure the time. |
| event\_location | The location of the event. | Location | event | Event need to ensure the location |
| event\_status | Every event has the status not held, in progress, finish. | Status | event | Every event has the status not held, in progress, finish. |
| meta\_key | Events public key | Public key | event | Key is shown how to connect the event in the database. |
| comment\_ID | Uniquely identifiers the comments | Comments | comment | Every comment has an ID |
| comment\_association\_ID | Shown which association the comment belongs to. |  | comment | The comment belongs to a society. |
| comment\_event\_ID | Shown which event the comment belongs to. |  | comment | The comment is comment a event. |
| comment\_user\_ID | Shown who made the comment. |  | comment | Every comment has a owner. |
| comment\_display\_name | Whether the comment anonymity. | Anonymity | comment | If comment is in anonymity the value is 0. If it is not the value is 1. |
| comment\_author\_IP | The real place of the user. | Ip | comment | Every comment sends by a real computer. And the computer has an Ip. |
| comment\_date | Shown when the comment was sent. | Date | comment | Shown when the comment was sent. |
| comment\_content | The content of the comment. | Comment, text | comment | Every comment needs to have a message |
| msg\_ID | Uniquely identifiers the messages. | Messages | message | Every message needs to have an ID |
| chatroom\_ID | Identify which chartroom the message belongs to. |  | message | Evert message belongs to a chartroom. |
| msg\_senderID | Identify the sender | Sender | message | Every message has sender. |
| msg\_sendername | The name of the sender | Name | message | Every sender has a name. |
| msg\_date | The date of the message send. | Date. | message | Every message has a send time. |
| msg\_content | The content of the message. | Content,text | message | Every message has a message text. |
| msg\_agent | The agent of the sender | agent | message | The agent contains information of the browser and system version . |
| manager\_ID | Uniquely identifiers the manger. | mangerID | manager | Every society has manager. |
| manager\_society | The society of the manager. |  | manager | Every society has manager. |
| manager\_level | The member in the society has the level | Level | manager | The level is member, manager, proprieter. |

Sequence diagrams

Chapter 7

Evaluation Design

This section of the report will set out of the evaluation design for the website based on the F.U.R.P.S. module. It will focus on five main aspects: functionality, usability, reliability, performance, supportability.

**7.1 Functional Testing**

*Test the performs, generality and security of the website. Determine whether the systems meet all functional requirements and ensures the requirements are properly satisfies by the website.*

Black box testing and equivalence partitioning technique will be applied in this part and is based on the functional requirement of the system without viewing the internal structure testing. Firstly, the test case will be divided into valid and invalid partitions, and the representative values from each partition will be selected from each partition as test data. Then the internal structure and implementation of the item being tested will not be considered. If all the inputs have the expected corresponding output results, the system passes functional testing.

Following are some main functions that will be tested: Log-in function, search function, chatroom function, Al recommender system, calendar reminder system.

Here are several general steps that tests will implement:

o List a collection of system inputs data based on the functional requirements.

o Determine expected outputs according to the functional requirements.

o Feed the system test cases and record the actual outputs.

o Compare the actual outputs with the expected results.

**Log-in testing**

During the black box test, the tester will try 100 times to log in by using the same ID and different password. This part will be divided into two stages, registration part and log-in part. In the registration stage, the website will make user to confirm its password that should contain at least one up case character, longer than 8 numbers and containing both characters and numbers. In the log-in part, if the user enters his password, the system then compare the entered password with his stored password on the database. If the password is not correct, the system will reject the user to enter. Here are some expected results that system might satisfy as in the following **tables.** After 100 tests, analysis the result by using the cross validation methods, if 99.9% of the test shows that the log-in function runs well, then this is good.

|  |  |  |  |
| --- | --- | --- | --- |
| Test number | User Id | Entered Password | Expected result |
| Test 1 | A0001 | Cr2376446 | true |
| Test 2 | A0001 | cr2376446 | false |
| Test 3 | A0001 | Cr237644 | false |
| Test 4 | A0001 | cr237644 | false |
| Test 5 | A0001 | Cr2367446 | false |

**Search function**

The search function allows users to search the societies, events, university departments, and other users. In the black box test, 100 tests will be take placed by comparing searched different information, and the information stored in the databases. According to a different type of users, the obtaining searched result might also be different, due to the secret level of Databases. For example, the manager of the student societies could have the right to manage its members and also in the search function, and they could gain more information than general users. This part of testing, the primary user will be divided into the manager and general uses. In the managers part, the users could gain the history of its social data, such as the number of subscribing, the number of members in history period, the active level of the society, etc. While as for general users, they could only gain information about societies, university department offices, events, and other users. Here is some conventional example of the search function that satisfies the result of the website in the following table. Using the cross-validation methods, if 99.9% of the test shows that the search function runs well, then this is good.

|  |  |  |  |
| --- | --- | --- | --- |
| Test number | Test user | Search question | Expected result |
| Test 1 | manager | History number of members | true |
| Test 2 | manager | History number of active level | true |
| Test 3 | General user | History number of members | false |

**Chatroom function**

Users could communicate with different other users in its social group chatroom. This chatroom will appear on the homepage of each society. It will store the time and communication words of users in the databases. During the black box test, the tester will have two main functions, and the first one is evaluating the response time of the chatroom function. 100 test will be implemented to test the chatroom response time, if 99.9% of the test shows that the chatroom function response within 3 seconds, then this is good. Here is some conventional example of the chatroom function that satisfies the result of the website in the following table.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Number | Test total time | Test number of times | Results(total time/number of time) |
| Test 1 | 20s | 20 | 1 |
| Test 2 | 44s | 40 | 1.1 |
| Test 3 | 80s | 60 | 1.5 |

The other central part will be checking the connection between the chatroom and its database. By using the SQL language, the tester will try to get the information that users communicated in the chat room, such as time and communication words. Then compare the information from the database and the information in the chatroom. Using the analysing tools and the cross-validation methods, if the content matches the same, then it shows that the connection between the chatroom and databases is good, if 99.9% of the test shows that the search function runs well, then this is good.

**Al recommender system**

The recommender system is based on the database of users’ previous participated or searched or rated activities to suggest some societies or events to users, which they might be interested in. During the black box test, 100 tests will be implemented to check the recommended function. The test might be divided into two main stages. The first stage is before the real black test begins: the tester is supposed to check the accuracy of the recommender system model which is based on 10000 datasets of the sample users’ information and trained by ML algorithms. The expected accuracy of the model could be at least 80%. Otherwise, the recommender system is not useful for suggesting. Secondly, in the black testing stage, 10 different volunteer tester will use 10 different user Id, and after logging in 10 times, joining 10events(which means joined one event during each log in time). After 100 tests, analysis the result by using the cross-validation methods, if 99.9% of the test shows that each tester join the suggestion events is above 7 times, the system will then is quite good. Here are some expected results that system might satisfy as in the following tables.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Number | Al Algorithm | Model Accuracy | Evaluation |
| Test 1 | K-nn | 83.94% | true |
| Test 2 | LSTM | 79.90% | false |
| Test 3 | Decision-tree | 75.68% | false |

**7.2 Performance Testing**

This part will evaluate the performance of the website, which includes the size of the window in the different operation system and also the hardware Evaluating how well the website will perform under the expected workload.

o Determine the time from when a user enters a request until the first character of the

response is received

o Determine maximum number of users load the software website simultaneously can

handle

o Determine if the website is stable and the request can be handling correctly under varying

loads

A specific workload will be simulated by automated clients performing user actions at random. The system should be capable of allowing these clients to communicate with one another in a limit time.

**7.3 Usability Testing**

Ensure the system is ease of use to the most of the first-time-use users under the situation which without consulting the manual.

o Users should be easy to learn the basic functions like searching societies (or activity), posting personal information, registering for events with the minimal frustration after a 3 minutes for being use familiar with.

o Ensure the users could get their expected outcomes in a suitable limited time. Take an example, the users are able to quickly find the related society activities holds recently by searching specific keywords in search bar.

After the training, giving five random system instructions to test the users and record their correctness rate and the time to finish each instruction. If more than 95% first-time- use users do not make exceed one errors on average and the average time is in an acceptable manner, the system passes the testing.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Number | Test topic | Test samples | Estimated average completion time/result |
| Test 1 | Find the searching bar. | 100 users | 7s |
| Test 2 | Search and calculate the time until appearances of results. | 100 users | 10s |
| Test 3 | Look for their interested club page without searching. | 100 users | 30s |
| Test 4 | Enter their individual home page. | 100 users | 20s |
| Test 5 | Successfully sign up for an activity. | 100 users | 50s |
| Test 6 | Find the individual schedule bar without searching | 100 users | 25s |
| Test 7 | Search results are related. | 100 users | More than 60% |

**7.4 Recoverability Testing**

*Testing whether the operations can be continued after a disaster or after the integrity of the system has been lost. The purpose of recoverability testing is to verify the system’s ability to recover from varying points of failure.*

1.Design specific system interfaces and clear relationships and design high quality use case diagram.

2.Generate code with understandable format and enough notations.

o Provide some unexpected situations which might possibly cause problems to test.

The system should be able to keep running after receiving unexpected data.

o Terminate the network connection while the website is running and analyse the system’s ability to continue receiving data from the point at which the network

connection is broken

o Restart the system while the website has a definite number of sessions are

uploading and check whether the system is able to recover all the information or not

The testing bed should be as close to the actual conditions as possible. The places where the failure happens, corresponding testing results and the time recovery from failure should be recorded. Each test should be run at least three times and the system is not perfect at recovering from the crashes if any test fails once or more times on one specific failures. After restoration, the basic function including searching, uploading, editing and deleting record should be retest again in case the restoration test damage.

|  |  |
| --- | --- |
| Checkpoint | Expected result |
| Multiple registration/log in/out for 100 times. | The system could jump from the log in page and process successfully. |
| Internet is broken while sending a message in chat room. | The previous record will be store in database. |
| Internet is broken before information submission(registration, personal/club information, activity information, user comments). | Previous stored data will be been affected. |
| Establish a large number of concurrent connections with server to test whether system have enough concurrent ability. | System is not crash and data can be updated timely. |

**7.5 Security Testing**

*The system should ensure the security of users’ personal information and prevention the misuse or attack of the system. Experiments could be implemented from following aspects*:

o Test the aspect of authentication by inputting different pairs of username and password. The system should only allow correct pairs of username and passwords and reject continuing user’s operations under wrong pairs account information.

o Detect sensitive information. Cookies should not store any sensitive information likes password, username etc and all sensitive information should be encrypted. Once the risk of account leakage is detecting, the system should remind users.

o Test the website on professional testing software.

The system should be adjusted immediately if any test fails in security testing period. It is important to keep the system itself and vital information safe.

|  |  |  |
| --- | --- | --- |
| Test Number | Test topic | Estimated time and result |
| Test 1 | Enter 1’ and 1=2 union select 1,2 in searching bar. | false |
| Test 2 | Use security tool such as web security scanner to detect the cyber leaks such as XSS, SQL injection, OPEN redirect, PHP File Include | No necessary security leaks are detected. |

**7.6 Database Evaluation design**

About the relation database the evaluate criteria can define in four parts.

**Attribute semantics**

Whenever an attribute is grouped into a relational pattern, we suppose an attribute which belongs to a relation must have a connection in the real world. The semantics of a relation need clearly explain. Design a relation pattern need to explain quickly, clearly and intuitionistic. If a relation pattern corresponds to a related entity, the semantics is clear. Otherwise, one relation pattern corresponds to many relation entities, and the semantics is not precise. In conclusion, the semantics can explain, and the database system is successful.

**Reduce redundant information in tuples**.

Bigger target is making the storage space minimises occupied by the primary relationship. The method of grouping attributes into a relational pattern has a significant impact on storage space. The primary relationship model is designed to have no insertion, deletion, and modification exceptions in the relationship. If there is an exception, be annotated to ensure that the program that updates the database can operate correctly.

**Reduce NULL values in tuples.**

Basic above, if many attributes do not apply to all relation to the tuples, there will be many NULL in the tuples. In this way the storage space will be waste. So if the NULL is inescapable then should make sure they are suitable for special situations, not for most tuples in relationships.

**The possibility of generating a parasitic tuple is not allowed.**

The relationship model should be designed so that they can link equivalence conditions on the attributes of the most (original code, outer code) pair, and this connection guarantees that no spurious element ancestors will be generated.

There are three people involved in the database design so the evaluation all responsible for them. However, change the work of every people. Each person does not evaluate the work completes by himself.

In conclusion, the result we expect first is that it can not only comprehensively and accurately evaluate the performance of the database system, but also can not fall into the slump of database performance factors. Second is the database performance evaluation system should provide a general direction for performance problem analysis and resolution.

Chapter 8

Business Rules

The project Moments can be defined as a social platform. After some adjustment, it aims to be an online community, for people to find friends, make friends, enjoy their hobbies and share the experience and exciting moments with each other.

At the very beginning, it will be a free website. We welcome more users to visit this website, to build their societies or clubs and share their moments. Meanwhile, we will improve site performance and collect data and feedback from users. We are trying to make the website more interesting, more usable and more attractive as we hope to build such a fun online world to serve people. Furthermore, there are several goals we want to achieve, and one is information integration; the second is bridge the gap between people. We are exploring how to do so.

If the website succeed to achieve these goals and there are some active users, we will consider to make the website more commercial, to keep running and developing this website.

1. Firstly, there may be a membership system. If a user charges money, he will unlock more functions on this website, or there will be some changes on his picture frame or personal homepage. However, it is not necessary to do so because all essential and core functions are free to use to everyone.

2. Secondly, the advertisement will be another income source. We may select some acceptable advertisements to show and we will try to make them not that conspicuous, or we may design them to be funnier and suitable for this website.

3. Thirdly, the website may allow some original works or bring in some virtual product consumption, such as online games, live streaming and some videos. The creator of the work will get most of the payment, and the website gets some profit sharing.

Chapter 9

Review Plan

**9.1 Completed Tasks**

In February and March, we settled the product requirements and the design of the whole product from database transaction, data to the user interface design. The work have been completed in two months consisting of:

1. Requirement analysis (Finish Date: 2/15/2019)

a. Project background research

b. Website functions analysis

c. Produce corresponding requirement documentation

1. Project design (Finish Date: 3/14/2019)

a. Design research

b. Interface design

c. Database design

d. Draw related design diagrams and chart

e. Produce corresponding design documentation

1. Web pages design & implementation

a. Design the major web pages settle the contents and layout

b. Implementation the webpage design with HTML, CSS, JavaScript

1. Website database construction
2. Website spread cartoon video (Finish Date: 2/14/2019)

**9.2 Deliverable**

The deliverable we have produced so far consisting of:

1. Requirement Documentation (Finish Date: 2/15/2019)

(Including: User views/ requirements, initial use case diagram, system boundary diagram, transaction requirements, system requirements, project background research etc.)

1. Design Documentation (Finish Date: 3/14/2019)

(Including: Data dictionary, global logical data model, logical and physical table structures, business rules, transaction matrix, functional descriptions, pseudo code for key algorithms, use case diagram and descriptions, user interface design, Gantt chart etc.)

**9.3 Future Plan**

In the coming time, we will aim to turning ideas and designs into reality with programming. The main tasks consists of:

Website user interfaces implementation

a. Related language: HTML, CSS, JavaScript

b. Expected deliverables: homepage, society webpage, department webpage, user personal page, chat room, event pages and the consistent jumping between the pages

c. Expected deadline (for the initial version): 3/20/2019

d. Formal deadline(for the final version)： 4/29/2019

**a.Structure database**

a. Related language: SQL, PHP

b. Expected deliverables: the interaction between the website and

c. Expected deadline (for the initial version): 4/1/2019

d. Formal deadline(for the final version)： 4/29/2019

**b.Desirable functions**

a. Related technique: artificial intelligence

b. Expected deliverables: recommendation system, hot search system, location, language transfer, voting system and message reminder

c. Expected deadline (for initial version): 4/13/2019

d. Formal deadline: 4/29/2019

**c.Testing**

a. Testing with always interact the programming. The testing includes five aspects: functional testing, performance testing, usability testing, recoverability testing and security testing

b. Expected deliverables: an evaluation report according to the product testing

c. Formal deadline: 4/29/2019

**d.Demo documentation & system**

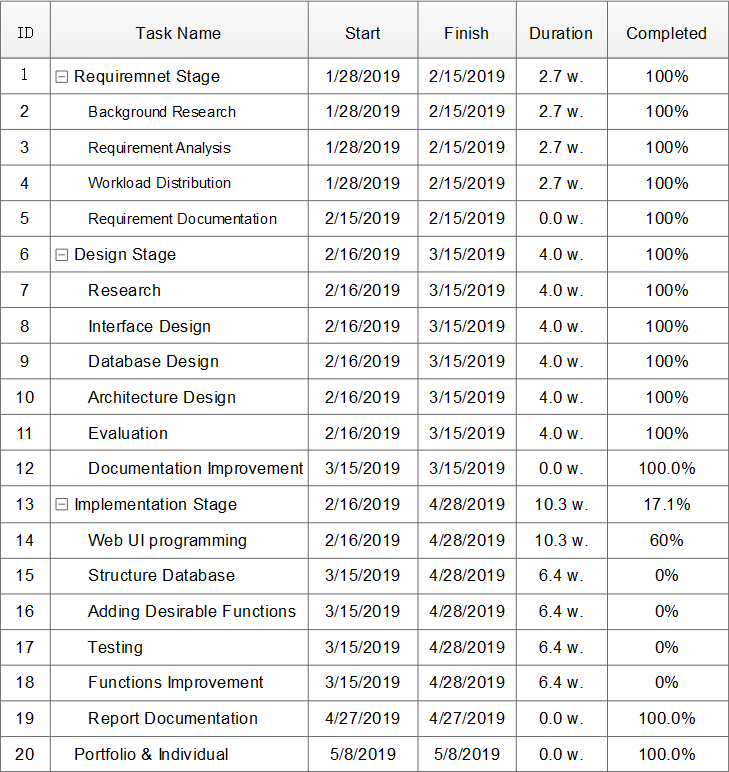
a. Expected deliverables: Specification documentation, user manuals and fully executable version of the system

b. Deadline: 4/29/2019

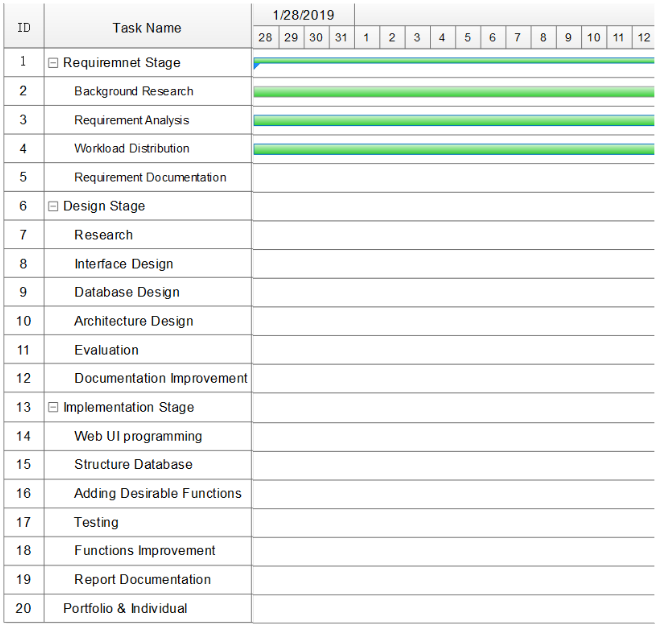
**9.4 Gantt Charts**

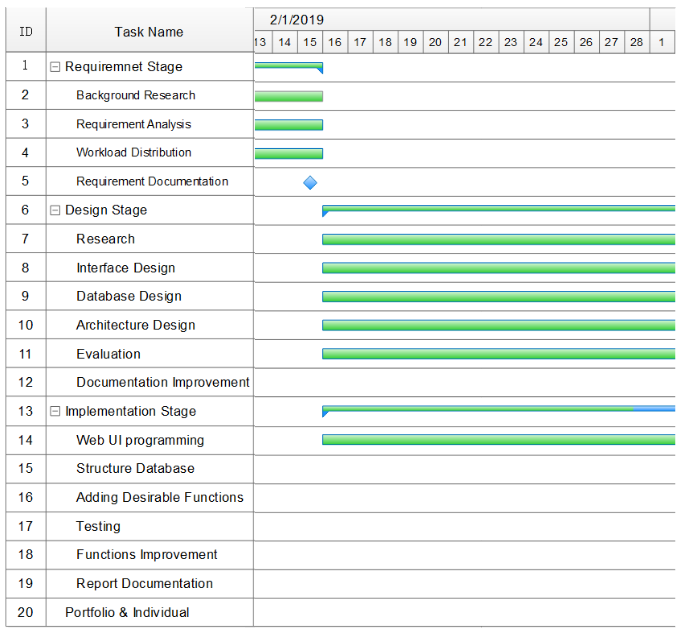
The following are the Gantt Charts of the project. Gantt Charts displays the project schedule information by listing project activities, the corresponding start and finish time. Compared to the Gantt Charts which are produced in the requirement stage, the charts at the end of the design stage have some changes according to the requirements of the project. The tasks are divided more detailed, and the finish processes have some changes.

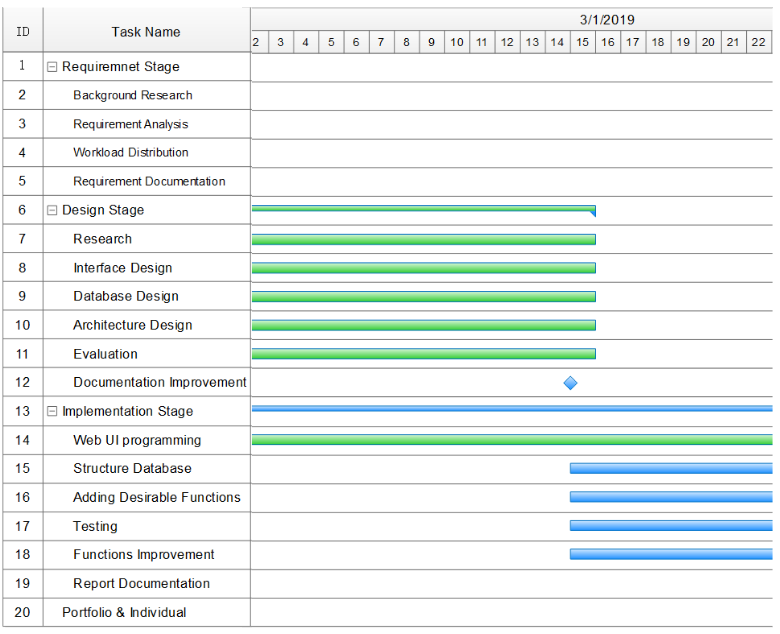
The first table shows 4 main periods and each period have some subtasks, the corresponding start and finish time and the process rates. The next 5 charts show the tasks have been split up into the months. The green parts in the graphs indicate the processes have already completed and the blue parts show the uncompleted tasks. Each diamond is a milestone that indicates the finish of a primary task.

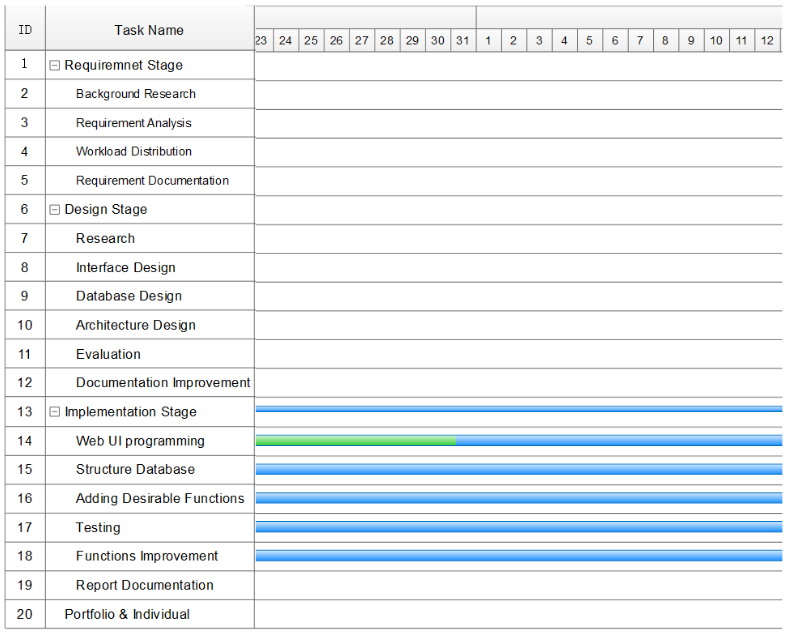


­









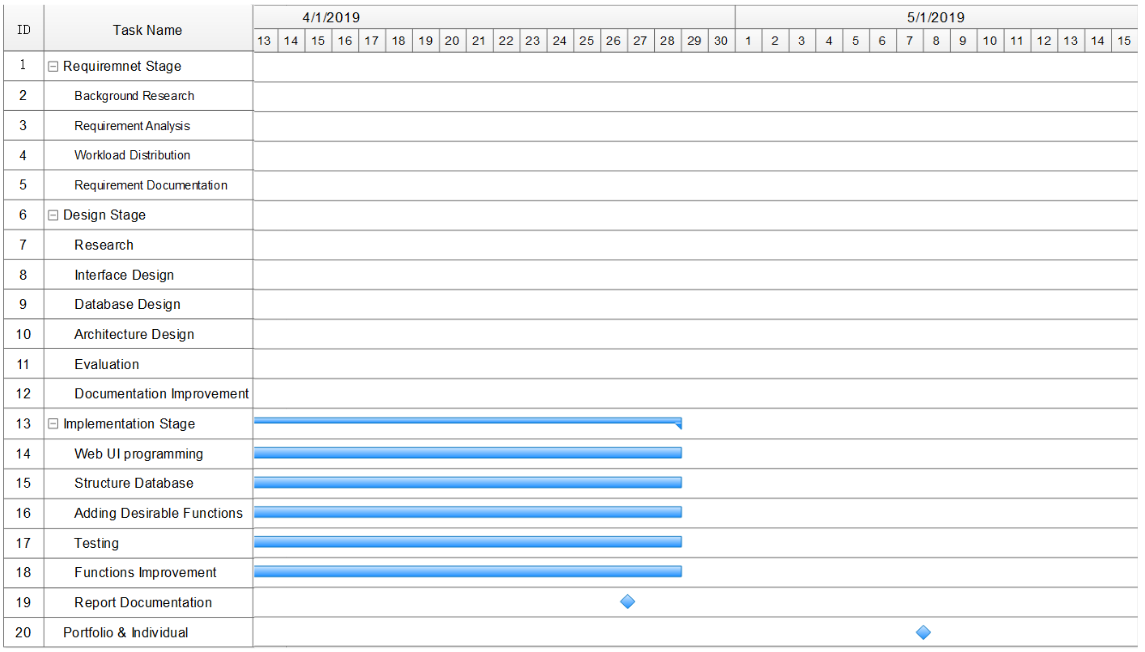


Figure21: Gantt Chart

Chapter 10

Bibliography

1.U.lrike,B.Vandenbosch,”Information Overload in a Groupware Environment: Now You See It, Now You Don't,”Journal of Organizational Computing and Electronic Commerce ,Vol.8,pp127-148,June 09,2010.[Online].Available:

<https://www.tandfonline.com/doi/abs/10.1207/s15327744joce0802_3>

2.“About us,”[liverpool.ac.uk,2019.[online].Availabel](http://liverpool.ac.uk):

<https://www.liverpool.ac.uk/about/>

3.“Societies,”[liverpoolguild.org,2019.[online].Available](http://liverpoolguild.org):

<https://www.liverpoolguild.org/main-menu/societies>

4. J.D.Foubert and L.A.Urbanski, “Effects of Involvement in Clubs and Organizations on the Psychosocial Development of First-Year and Senior College Students,”NASPA Journal,Feb 10,2006.[Online].Available:

<https://www.tandfonline.com/doi/pdf/10.2202/1949-6605.1576>

5. Cooper, D. L., Healy, M. A., & Simpson, J. (1994). Student development through involvement: Specific changes over time. Journal of College Student Development, 35, 98–102.

6.D.Mwiti, “How to build a Simple Recommender System in Python,” Towards Data Science, Oct 3, 2018.[Online].Available:

<https://towardsdatascience.com/how-to-build-a-simple-recommender-system-in-python-375093c3fb7d>

7.T.Selker and J.Goler,“The SAVE System — Secure Architecture for Voting Electronically,” BT Technology Journal, Vol 22, pp 89-95, Oct, 2004.[Online].Available:

<https://link.springer.com/article/10.1023/B%3ABTTJ.0000047587.09395.e5>

8.N.M.Salish and M.Z.Norsid,“A DESIGN OF VOTING SYSTEM BY USING AN ANDROID APPS FOR FISHERMAN,” ARPN Journal of Engineering and Applied Sciences, Vol 12,Mar, 2017. [Online].Available:

<https://www.researchgate.net/publication/315665861_A_DESIGN_OF_VOTING_SYSTEM_BY_USING_AN_ANDROID_APPS_FOR_FISHERMAN>

9.R.M.Rawi,“Project Risk Identification For New Project Manager,” [PMtimes.com](http://PMtimes.com), 2013.[Online].Available:

<https://www.projecttimes.com/articles/project-risk-identification-for-new-project-manager.html>

10.T.M.Connolly and C.E.Begg,“Database Solutions,” Addison Wesley, Nov, 2007.[Online].Available: <https://www-dawsonera-com.liverpool.idm.oclc.org/abstract/9781405890342>

11.M.T.Goodrich and R. Tamassia,“Algorithm Design,” John Wiley & Sons,2011.

12.P.Melville and V.Sindhwani, “Recommender Systems,” IBM T. J. Watson Research Center,2010.

1. E.Rosenthal, “Intro to Recommender Systems: Collaborative Filtering,”,Nov, 2015.[Online].Available:

https://www.ethanrosenthal.com/2015/11/02/intro-to-collaborative-filtering/

Chapter 11

Appendix

