UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Attribute-Based File Sharing System
Final Project Report

Project Supervisor:

Guomin Yang

Project Team:

Name	Student No.	Email Address
Anqi Gao	6308810	ag610@uowmail.edu.au
Miao Miao	5992436	mm736@uowmail.edu.au
Heng Zhang	6026461	hz546@uowmail.edu.au
Ke Xian	6109470	kx876@uowmail.edu.au

Nov 2020

Abstract

There are many file sharing systems in the market, such as Google Drive, One Drive. User can upload their files to the server and share these files or a folder to others by sending a link. In our project, we proposed a file sharing system using ciphertext policy attribute-based encryption (CP-ABE). In our system, each user is linked with a set of attributes that can describe the user identity, such as CSCI992 student, student in Wollongong campus. User can upload files to the server with a policy which define who can access the file. The file then will be encrypted and stored in the server. User can browse all the file stored in the server but can only access those files which their attributes can satisfied the policy define by the file uploader.

Table of Contents

Project Introduction and Proposal

1. Summary

The aim of this project is to create a file-sharing platform for users. The system is based on website. Considering the real business world, people are more willing to choose web-based application rather than installing an app in their PCs. In this system, only administrators can create user account, self-register is not allowed in the system. This is to ensure the validation of user information. Ciphertext policy attributed-based encryption technique is used in this project to ensure the security of data stored in our system. The key used to decrypt the file is based on the user's attributes. Thus, the attributes of user must be authenticated.

The project will run for nearly one year, the project management method we used is scrum. The project management tool we used for this project is called Taiga. These tools can help us to monitor the progress of the project and finish the project as planned.

The structure of this report is start with project introduction and proposal, followed by requirements analysis. Then we introduced the system design, including front-end design, database design and a brief back-end design. For back-end design, we don't have UML diagrams but a rough idea of what the function should do. The implementation of these functions is more depended on the preference of the programmer. After that, we have system implementation which indicated how the system builds up. The system test is discussed after system implementation. It shows how the functional requirements are tested, and the test result for each test case. In addition, we provide a case study for our system after system test. The workload for each team member is listed in the teamwork table and everyone has signed.

2. Project Description

In this project this system will be descripted as a file sharing platform which with an additional attribute-base access control. And the system has such key requirements:

a): Administrator can create users and provide each user with a secret key base on the user's attributes.

b): Each registered user can login the platform to upload files that will be encrypted based on an access policy specified the uploader.

c): Each registered user can only open files whose associated access policies are satisfied by the user's attributes.

For the development of this proposal, after analysed the project requirements and had meetings with the sponsor, we assign different parts to different members based on their own interests and workload from other subjects. Considering this project's workload and our group have two semesters to work on it. Thus, we design this project as a two semesters project about seven months, the details will be shown at the work plan part. The tasks are signed to different team members based on their own interests and abilities. The time period we estimate together, and we modify once based on advice from the sponsor. And the method and tools we choose are based on the sponsor suggestion and our previous experience.

3. Overall Purpose

The main purpose of this project is to create a file sharing system. Users can upload files to the server and download the file from the server. User should be able to see the basic information such as title and description of all the files stored on the server.

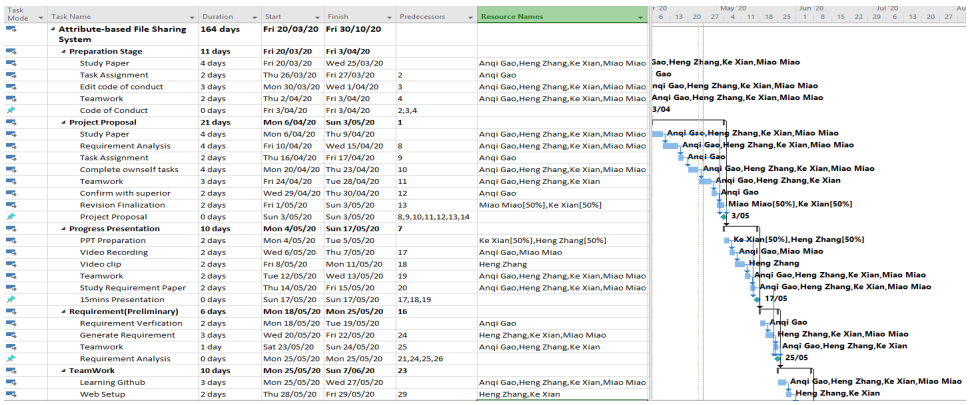
The file stored on the server should be encrypted. Users can only access the file if and only if their attributes satisfy the policy of the file. This is the way how we allow users to share files with a group of people who has a specific attribute.

Only administrator can create user accounts. In order to ensure the security of data, we need to make sure the attributes of each user are correct.

4. Project Work Plan

In this section, we would like to introduce work plan in more details. Based on the convenience of management, MS Project is the best tool to make project work plan. In order to complete project objectives, it is always essential to monitor the progress of the project. Agile project management help us to monitor our project from time to time. With the efforts and cooperation of team members, what needs to be done and by whom was arranged based on personal ability. The work was organised through weekly meeting to promote project development. Since our project has two semesters which lasted almost seven months. We divide the stages according to the assignments and the results need to be completed in each stage.

Now, we need to describe detailed work plan.

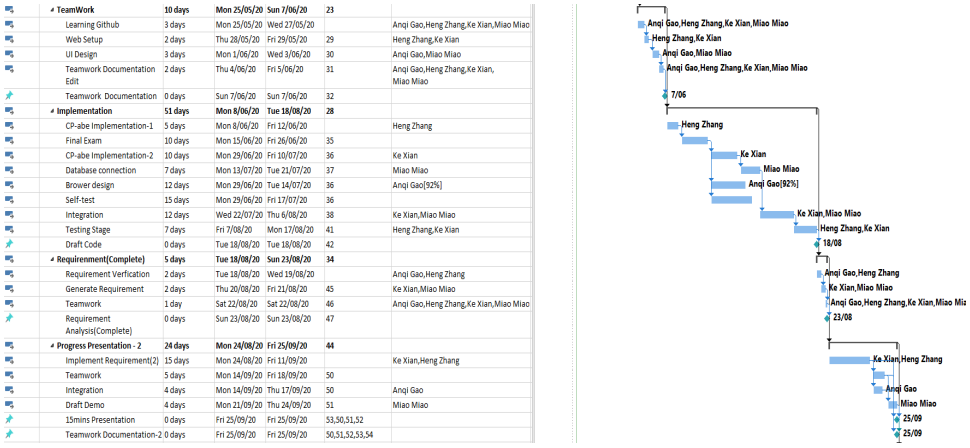


Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
	Attribute-based File Sharing System	**164 days**	**Fri 20/03/20**	**Fri 30/10/20**		
	Preparation Stage	**11 days**	**Fri 20/03/20**	**Fri 3/04/20**		
	Study Paper	4 days	Fri 20/03/20	Wed 25/03/20		Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Task Assignment	2 days	Thu 26/03/20	Fri 27/03/20	2	Anqi Gao
	Edit code of conduct	3 days	Mon 30/03/20	Wed 1/04/20	3	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Teamwork	2 days	Thu 2/04/20	Fri 3/04/20	4	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Code of Conduct	0 days	Fri 3/04/20	Fri 3/04/20	2,3,4	
	Project Proposal	**21 days**	**Mon 6/04/20**	**Sun 3/05/20**	**1**	
	Study Paper	4 days	Mon 6/04/20	Thu 9/04/20		Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Requirement Analysis	4 days	Fri 10/04/20	Wed 15/04/20	8	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Task Assignment	2 days	Thu 16/04/20	Fri 17/04/20	9	Anqi Gao
	Complete ownself tasks	4 days	Mon 20/04/20	Thu 23/04/20	10	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Teamwork	3 days	Fri 24/04/20	Tue 28/04/20	11	Anqi Gao,Heng Zhang,Ke Xian
	Confirm with superior	2 days	Wed 29/04/20	Thu 30/04/20	12	Anqi Gao
	Revision Finalization	2 days	Fri 1/05/20	Sun 3/05/20	13	Miao Miao[50%],Ke Xian[50%]
	Project Proposal	0 days	Sun 3/05/20	Sun 3/05/20	8,9,10,11,12,13,14	
	Progress Presentation	**10 days**	**Mon 4/05/20**	**Sun 17/05/20**	**7**	
	PPT Preparation	2 days	Mon 4/05/20	Tue 5/05/20		Ke Xian[50%],Heng Zhang[50%]
	Video Recording	2 days	Wed 6/05/20	Thu 7/05/20	17	Anqi Gao,Miao Miao
	Video clip	2 days	Fri 8/05/20	Mon 11/05/20	18	Heng Zhang
	Teamwork	2 days	Tue 12/05/20	Wed 13/05/20	19	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Study Requirement Paper	2 days	Thu 14/05/20	Fri 15/05/20	20	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	15mins Presentation	0 days	Sun 17/05/20	Sun 17/05/20	17,18,19	
	Requirement(Preliminary)	**6 days**	**Mon 18/05/20**	**Mon 25/05/20**	**16**	
	Requirement Verfication	2 days	Mon 18/05/20	Tue 19/05/20		Anqi Gao
	Generate Requirement	3 days	Wed 20/05/20	Fri 22/05/20	24	Heng Zhang,Ke Xian,Miao Miao
	Teamwork	1 day	Sat 23/05/20	Sun 24/05/20	25	Anqi Gao,Heng Zhang,Ke Xian
	Requirement Analysis	0 days	Mon 25/05/20	Mon 25/05/20	21,24,25,26	
	TeamWork	**10 days**	**Mon 25/05/20**	**Sun 7/06/20**	**23**	
	Learning Github	3 days	Mon 25/05/20	Wed 27/05/20		Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Web Setup	2 days	Thu 28/05/20	Fri 29/05/20	29	Heng Zhang,Ke Xian

Figure 4.1 Project work plan 1

As the figure shows, the total workdays are 164 days. Our project was started from March 18th. Preparation stage has produced code of conduct after 11 days' work. Except task assignment was done by Leader. The rest of tasks require the cooperation of all staff to complete. As of now, the preparation stage has ended. Project proposal is our current task, team leader plays the role of communication with the sponsor. After 21 days of hard work, the project proposal will submit on May 3rd. Our next task is to give a presentation on March 17th. Video recording and clip world take several days to accomplish it by Heng Zhang and Anqi, Miao Miao. PPT preparation will created by Ke Xian and Heng Zhang. 15 mins progress presentation is our next project objects, the main purpose of progress presentation is to monitor the whole project from time to time. Requirement analysis need to cooperate with sponsor, in the early stage of requirement analysis, we need to keep confirming with sponsor to get the exact demand. March

25th is the last day for us to correct it. In the end, preliminary requirement analysis is our deliverable.



	Task Name	Duration	Start	Finish	Predecessors	Resource Names
	TeamWork	10 days	Mon 25/05/20	Sun 7/06/20	23	
	Learning Github	3 days	Mon 25/05/20	Wed 27/05/20		Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Web Setup	2 days	Thu 28/05/20	Fri 29/05/20	29	Heng Zhang,Ke Xian
	UI Design	3 days	Mon 1/06/20	Wed 3/06/20	30	Anqi Gao,Miao Miao
	Teamwork Documentation Edit	2 days	Thu 4/06/20	Fri 5/06/20	31	Anqi Gao,Heng Zhang,Ke Xian, Miao Miao
	Teamwork Documentation	0 days	Sun 7/06/20	Sun 7/06/20	32	
	Implementation	51 days	Mon 8/06/20	Tue 18/08/20	28	
	CP-abe Implementation-1	5 days	Mon 8/06/20	Fri 12/06/20		Heng Zhang
	Final Exam	10 days	Mon 15/06/20	Fri 26/06/20	35	
	CP-abe Implementation-2	10 days	Mon 29/06/20	Fri 10/07/20	36	Ke Xian
	Database connection	7 days	Mon 13/07/20	Tue 21/07/20	37	Miao Miao
	Brower design	12 days	Mon 29/06/20	Tue 14/07/20	36	Anqi Gao[92%]
	Self-test	15 days	Mon 29/06/20	Fri 17/07/20	36	
	Integration	12 days	Wed 22/07/20	Thu 6/08/20	38	Ke Xian,Miao Miao
	Testing Stage	7 days	Fri 7/08/20	Mon 17/08/20	41	Heng Zhang,Ke Xian
	Draft Code	0 days	Tue 18/08/20	Tue 18/08/20	42	
	Requirenment(Complete)	5 days	Tue 18/08/20	Sun 23/08/20	34	
	Requirement Verfication	2 days	Tue 18/08/20	Wed 19/08/20		Anqi Gao,Heng Zhang
	Generate Requirement	2 days	Thu 20/08/20	Fri 21/08/20	45	Ke Xian,Miao Miao
	Teamwork	1 day	Sat 22/08/20	Sat 22/08/20	46	Anqi Gao,Heng Zhang,Ke Xian,Miao Miao
	Requirement Analysis(Complete)	0 days	Sun 23/08/20	Sun 23/08/20	47	
	Progress Presentation - 2	24 days	Mon 24/08/20	Fri 25/09/20	44	
	Implement Requirement(2)	15 days	Mon 24/08/20	Fri 11/09/20		Ke Xian,Heng Zhang
	Teamwork	5 days	Mon 14/09/20	Fri 18/09/20	50	
	Integration	4 days	Mon 14/09/20	Thu 17/09/20	50	Anqi Gao
	Draft Demo	4 days	Mon 21/09/20	Thu 24/09/20	51	Miao Miao
	15mins Presentation	0 days	Fri 25/09/20	Fri 25/09/20	53,50,51,52	
	Teamwork Documentation-2	0 days	Fri 25/09/20	Fri 25/09/20	50,51,52,53,54	

Figure 4.2 Project work plan

Teamwork is the last task of our first semester. The main purpose of teamwork is to write teamwork memo. Teamwork documentation need to be accomplished on June 7th. In terms of human resources arrangement, teamwork documentation should be coordinated by two people.

During winter holidays, the implementation of code needs to be on the agenda. Firstly, UI design and web setup are the first thing that needs to be done. After that, CP-abe implementation is the key point of our sharing system. Secondly, database system plays the role of connection between Browser and cp-abe algorithm. Finally, the last step of code implementation is integration. Our deliverable is the draft code of sharing system based on the file attribution. At the beginning of the second semester, complete requirement is our first task to accomplish. The complete requirements are designed based on the preliminary requirement; it will take us one week to finish the whole requirement. In the end of August, whole requirement will be submitted on time. Once we have done complete requirement, progression presentation need to be on the agenda, the second progress presentation should provide draft code to run it. At the end of task, teamwork documentation and 15 mins presentation are asked to finish on September 25th. Final project documentation and tradeshow demo are the last two tasks to be completed. Final documentation contains teamwork memo, evidence of relevance, it is required to have 80 pages. During the editing of final project documentation, it is essential to confirm with sponsor. All our team member would try our best to make final project documentation perfect. October 25th is the last day to submit final project documentation. After final project documentation has submitted on time, it shows that our project is at the end.

Tradeshow demo is the event of group presentation. From tradeshow PPT to demo the whole project within 2 hours, our core task is to guarantee the normal operation of the program during demo presentation. October 30th is the day to do tradeshow demo. Demo the project is the last task we need to submit. The whole project will be ended with the 2 hours presentation.

	Final Project Documentation	**22 days**	**Fri 25/09/20**	**Sun 25/10/20**	**49**	
	Team work memo	4 days	Fri 25/09/20	Wed 30/09/20		Anqi Gao,Heng Zhang
	Evidence of relevance	7 days	Thu 1/10/20	Fri 9/10/20	57	Ke Xian,Miao Miao
	Writing Paper	5 days	Mon 12/10/20	Fri 16/10/20	58	Heng Zhang
	Superior validation	3 days	Mon 19/10/20	Wed 21/10/20	59	Ke Xian
	Documentation Adjustmen	3 days	Thu 22/10/20	Sun 25/10/20	60	Miao Miao
	Final Project Documentation	0 days	Sun 25/10/20	Sun 25/10/20	57,58,59,60	
	TradeShow Demo	**4 days**	**Mon 26/10/20**	**Fri 30/10/20**	**56**	
	Teadeshow PPT	2 days	Mon 26/10/20	Tue 27/10/20		Anqi Gao[50%],Heng Zhang[50%]
	Presentation Training	2 days	Wed 28/10/20	Thu 29/10/20	64	Ke Xian,Miao Miao
	Demo the Project	0 days	Fri 30/10/20	Fri 30/10/20	65	

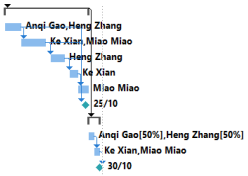


Figure 4.3 Project work plan

5. Human Resources

This section explains the whole plan of human resources management. It should provide a general description of what the plan includes and explain how the project manager and project team can use the plan to help us manage the project effectively.

Human resources management is an important part of file sharing system. The human resources management plan is a tool which will aid in the management of this project human resources events through the project.

The human resources management plan includes:

--- Role and responsibilities of team members through the project

--- Each team member detailed task assignment

Roles and responsibilities:

Roles and responsibilities of team members must be clearly defined in our project. Depending on the organization structure, project team members may display different professional skills and interested in the responsibilities for different function. Additionally, team members may have varying degree of authority and responsibilities.

For the attributed-based file sharing system, the following role and responsibilities have been established:

Sponsor: typically, responsible for confirming a series of key requirements, which can be summed up categories of vision, governance, and value realization. The sponsor provides guidance to the project leader and team member. The sponsor also plays the role of acceptance tester, which including check all kinds of deliverables and provide suggestions to correct it. The sponsor authorizes and approve all project expenditures.

Project leader: responsible for the overall success of the file sharing system. The leader is responsible for reporting project status in accordance with the teamwork documentation. The main task for project leader to be completed is task assignment based on personal ability. In addition, the leader is also responsible for communicating with sponsor to get requirement validation and schedule meeting regularly. The skills for the leader to be possessed include leadership, scheduling and effective communication. Professional project management tool - Agile should be a specialty.

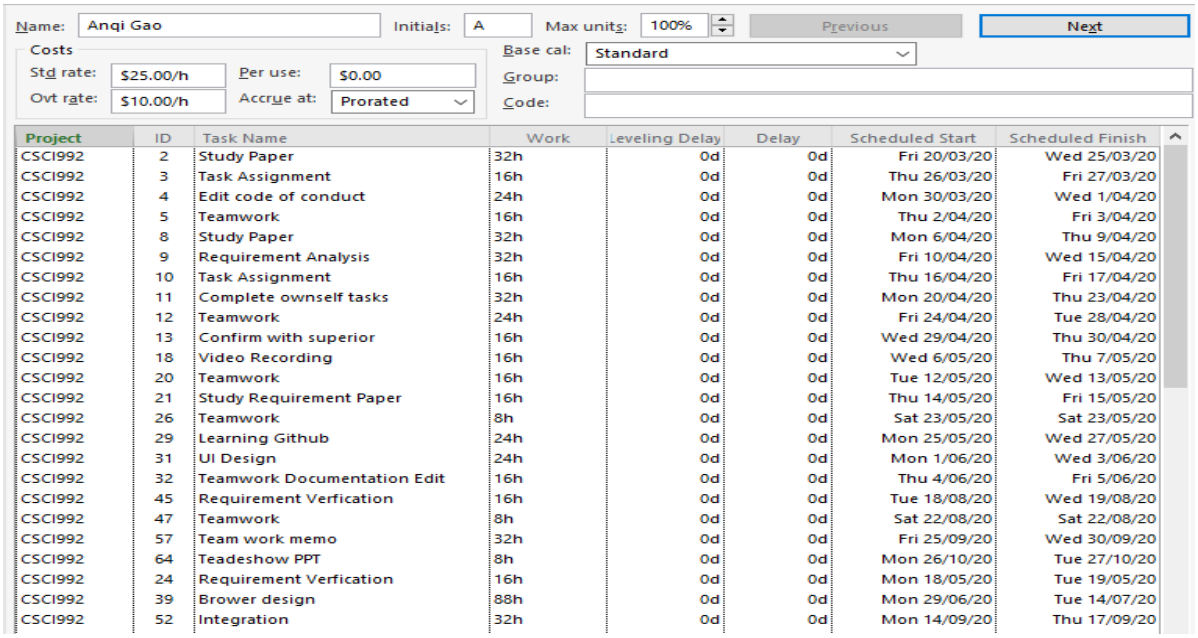
Programmer: responsible for gathering coding requirement for the file sharing system. Programmers are responsible all project design, coding, hence a good understand of programming languages (java, C++, etc.) is essential. Programmers will assist the implementation in order to monitor the software achievement through network infrastructure. The programmers are responsible for timely status reporting to the leader as required by the project work plan. Programmers should be good at java language and understand how to use GitHub, which is one of sharing platform tools, in addition, project management tool – Agile should be a daily tool. Moreover, SQL statement needs to use proficiently.

Tester: software test engineer should capable of designing test suites and should have the ability to understand usability issues. A tester is expected to have sound knowledge of software test design and test an execution methodology. It is essential for a tester to have great communication skills that can interact with the development term efficiently. Tester is responsible for designing test scenarios and conducting the test, and then submit questions to programmers in order to correct it. Based on the character of our project, tester and programmer would work simultaneously. Software test engineer should have the following skills: write cases proficiently according to requirement; understand code and know how to correct it; familiar with java language; learn from GitHub and share with team member via sharing platform; use Agile proficiently in order to manage project successfully.

Each team member’s detailed tasks/ assignments:

Based on the roles and responsibilities, we assigned everyone's task according to different demand.

Project leader & Programmer: Anqi



Name: Anqi Gao | Initials: A | Max units: 100% | Previous | Next

Costs — Std rate: $25.00/h | Per use: $0.00 | Ovt rate: $10.00/h | Accrue at: Prorated

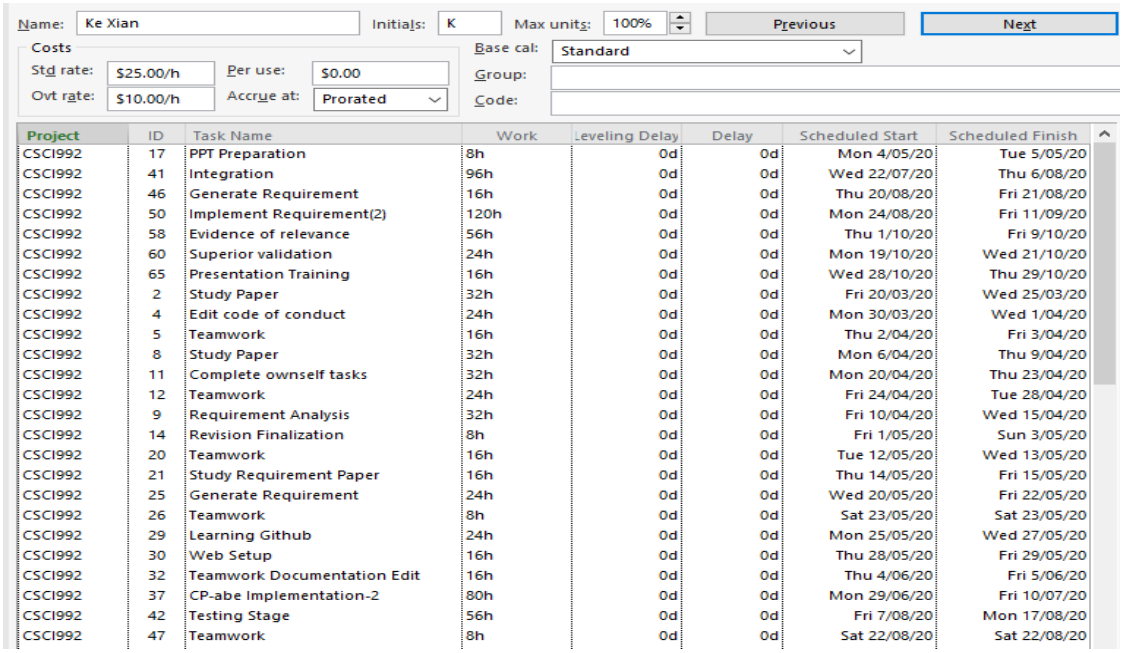
Base cal: Standard | Group: | Code:

Project	ID	Task Name	Work	Leveling Delay	Delay	Scheduled Start	Scheduled Finish
CSCI992	2	Study Paper	32h	0d	0d	Fri 20/03/20	Wed 25/03/20
CSCI992	3	Task Assignment	16h	0d	0d	Thu 26/03/20	Fri 27/03/20
CSCI992	4	Edit code of conduct	24h	0d	0d	Mon 30/03/20	Wed 1/04/20
CSCI992	5	Teamwork	16h	0d	0d	Thu 2/04/20	Fri 3/04/20
CSCI992	8	Study Paper	32h	0d	0d	Mon 6/04/20	Thu 9/04/20
CSCI992	9	Requirement Analysis	32h	0d	0d	Fri 10/04/20	Wed 15/04/20
CSCI992	10	Task Assignment	16h	0d	0d	Thu 16/04/20	Fri 17/04/20
CSCI992	11	Complete ownself tasks	32h	0d	0d	Mon 20/04/20	Thu 23/04/20
CSCI992	12	Teamwork	24h	0d	0d	Fri 24/04/20	Tue 28/04/20
CSCI992	13	Confirm with superior	16h	0d	0d	Wed 29/04/20	Thu 30/04/20
CSCI992	18	Video Recording	16h	0d	0d	Wed 6/05/20	Thu 7/05/20
CSCI992	20	Teamwork	16h	0d	0d	Tue 12/05/20	Wed 13/05/20
CSCI992	21	Study Requirement Paper	16h	0d	0d	Thu 14/05/20	Fri 15/05/20
CSCI992	26	Teamwork	8h	0d	0d	Sat 23/05/20	Sat 23/05/20
CSCI992	29	Learning Github	24h	0d	0d	Mon 25/05/20	Wed 27/05/20
CSCI992	31	UI Design	24h	0d	0d	Mon 1/06/20	Wed 3/06/20
CSCI992	32	Teamwork Documentation Edit	16h	0d	0d	Thu 4/06/20	Fri 5/06/20
CSCI992	45	Requirement Verfication	16h	0d	0d	Tue 18/08/20	Wed 19/08/20
CSCI992	47	Teamwork	8h	0d	0d	Sat 22/08/20	Sat 22/08/20
CSCI992	57	Team work memo	32h	0d	0d	Fri 25/09/20	Wed 30/09/20
CSCI992	64	Teadeshow PPT	8h	0d	0d	Mon 26/10/20	Tue 27/10/20
CSCI992	24	Requirement Verfication	16h	0d	0d	Mon 18/05/20	Tue 19/05/20
CSCI992	39	Brower design	88h	0d	0d	Mon 29/06/20	Tue 14/07/20
CSCI992	52	Integration	32h	0d	0d	Mon 14/09/20	Thu 17/09/20

Figure 5.1 Human resources - Leader

According to the resource sheet, the basic task for leader is to arrange task and confirm with sponsor. In addition, UI design is the basis for implementing interface design via code implementation, it is second important task for leader to focus.

Programmer & Tester: Ke Xian



Name: Ke Xian | Initials: K | Max units: 100% | Previous | Next

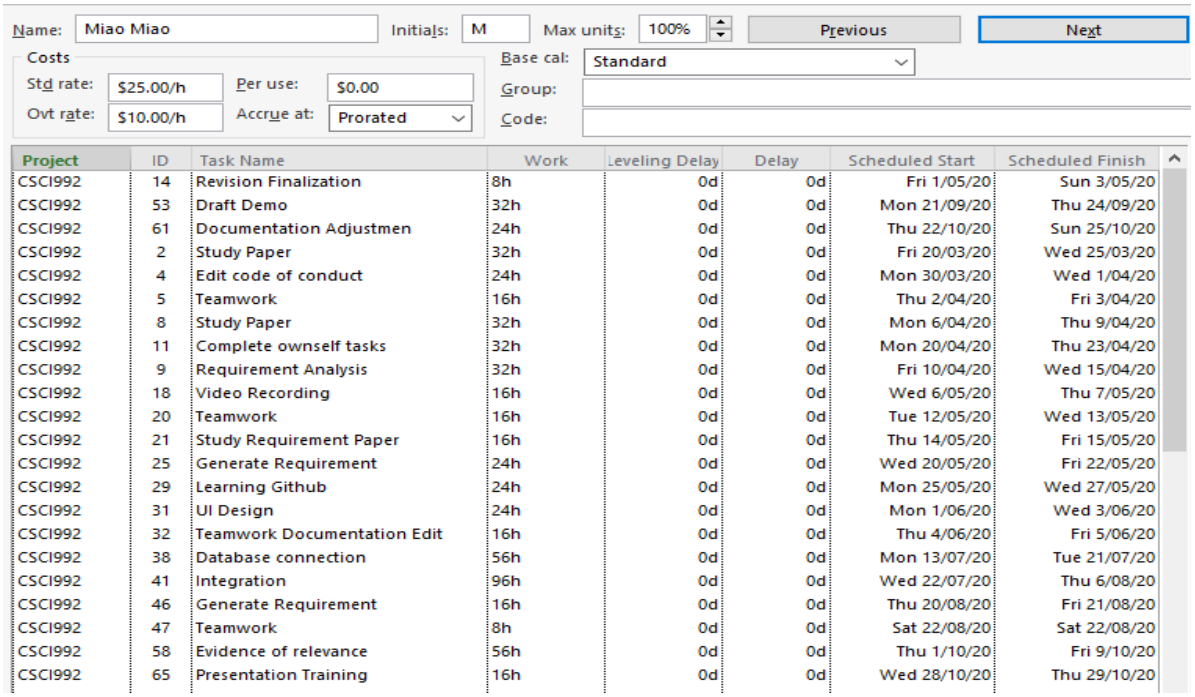
Costs — Std rate: $25.00/h | Per use: $0.00 | Ovt rate: $10.00/h | Accrue at: Prorated

Base cal: Standard | Group: | Code:

Project	ID	Task Name	Work	Leveling Delay	Delay	Scheduled Start	Scheduled Finish
CSCI992	17	PPT Preparation	8h	0d	0d	Mon 4/05/20	Tue 5/05/20
CSCI992	41	Integration	96h	0d	0d	Wed 22/07/20	Thu 6/08/20
CSCI992	46	Generate Requirement	16h	0d	0d	Thu 20/08/20	Fri 21/08/20
CSCI992	50	Implement Requirement(2)	120h	0d	0d	Mon 24/08/20	Fri 11/09/20
CSCI992	58	Evidence of relevance	56h	0d	0d	Thu 1/10/20	Fri 9/10/20
CSCI992	60	Superior validation	24h	0d	0d	Mon 19/10/20	Wed 21/10/20
CSCI992	65	Presentation Training	16h	0d	0d	Wed 28/10/20	Thu 29/10/20
CSCI992	2	Study Paper	32h	0d	0d	Fri 20/03/20	Wed 25/03/20
CSCI992	4	Edit code of conduct	24h	0d	0d	Mon 30/03/20	Wed 1/04/20
CSCI992	5	Teamwork	16h	0d	0d	Thu 2/04/20	Fri 3/04/20
CSCI992	8	Study Paper	32h	0d	0d	Mon 6/04/20	Thu 9/04/20
CSCI992	11	Complete ownself tasks	32h	0d	0d	Mon 20/04/20	Thu 23/04/20
CSCI992	12	Teamwork	24h	0d	0d	Fri 24/04/20	Tue 28/04/20
CSCI992	9	Requirement Analysis	32h	0d	0d	Fri 10/04/20	Wed 15/04/20
CSCI992	14	Revision Finalization	8h	0d	0d	Fri 1/05/20	Sun 3/05/20
CSCI992	20	Teamwork	16h	0d	0d	Tue 12/05/20	Wed 13/05/20
CSCI992	21	Study Requirement Paper	16h	0d	0d	Thu 14/05/20	Fri 15/05/20
CSCI992	25	Generate Requirement	24h	0d	0d	Wed 20/05/20	Fri 22/05/20
CSCI992	26	Teamwork	8h	0d	0d	Sat 23/05/20	Sat 23/05/20
CSCI992	29	Learning Github	24h	0d	0d	Mon 25/05/20	Wed 27/05/20
CSCI992	30	Web Setup	16h	0d	0d	Thu 28/05/20	Fri 29/05/20
CSCI992	32	Teamwork Documentation Edit	16h	0d	0d	Thu 4/06/20	Fri 5/06/20
CSCI992	37	CP-abe Implementation-2	80h	0d	0d	Mon 29/06/20	Fri 10/07/20
CSCI992	42	Testing Stage	56h	0d	0d	Fri 7/08/20	Mon 17/08/20
CSCI992	47	Teamwork	8h	0d	0d	Sat 22/08/20	Sat 22/08/20

Figure 5.2 Human Resources – Team member

Programmer & Tester: Miao Miao



Name: Miao Miao | Initials: M | Max units: 100% | Previous | Next

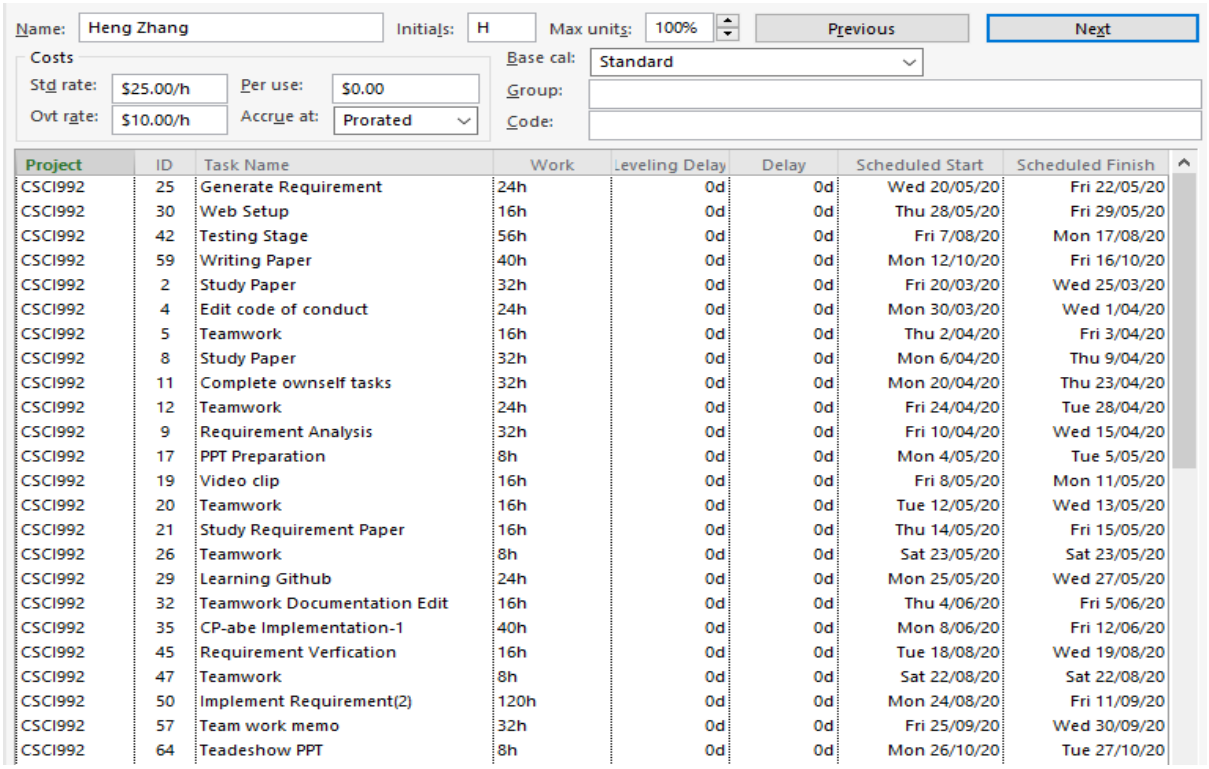
Costs — Std rate: $25.00/h | Per use: $0.00 | Ovt rate: $10.00/h | Accrue at: Prorated

Base cal: Standard | Group: | Code:

Project	ID	Task Name	Work	Leveling Delay	Delay	Scheduled Start	Scheduled Finish
CSCI992	14	Revision Finalization	8h	0d	0d	Fri 1/05/20	Sun 3/05/20
CSCI992	53	Draft Demo	32h	0d	0d	Mon 21/09/20	Thu 24/09/20
CSCI992	61	Documentation Adjustmen	24h	0d	0d	Thu 22/10/20	Sun 25/10/20
CSCI992	2	Study Paper	32h	0d	0d	Fri 20/03/20	Wed 25/03/20
CSCI992	4	Edit code of conduct	24h	0d	0d	Mon 30/03/20	Wed 1/04/20
CSCI992	5	Teamwork	16h	0d	0d	Thu 2/04/20	Fri 3/04/20
CSCI992	8	Study Paper	32h	0d	0d	Mon 6/04/20	Thu 9/04/20
CSCI992	11	Complete ownself tasks	32h	0d	0d	Mon 20/04/20	Thu 23/04/20
CSCI992	9	Requirement Analysis	32h	0d	0d	Fri 10/04/20	Wed 15/04/20
CSCI992	18	Video Recording	16h	0d	0d	Wed 6/05/20	Thu 7/05/20
CSCI992	20	Teamwork	16h	0d	0d	Tue 12/05/20	Wed 13/05/20
CSCI992	21	Study Requirement Paper	16h	0d	0d	Thu 14/05/20	Fri 15/05/20
CSCI992	25	Generate Requirement	24h	0d	0d	Wed 20/05/20	Fri 22/05/20
CSCI992	29	Learning Github	24h	0d	0d	Mon 25/05/20	Wed 27/05/20
CSCI992	31	UI Design	24h	0d	0d	Mon 1/06/20	Wed 3/06/20
CSCI992	32	Teamwork Documentation Edit	16h	0d	0d	Thu 4/06/20	Fri 5/06/20
CSCI992	38	Database connection	56h	0d	0d	Mon 13/07/20	Tue 21/07/20
CSCI992	41	Integration	96h	0d	0d	Wed 22/07/20	Thu 6/08/20
CSCI992	46	Generate Requirement	16h	0d	0d	Thu 20/08/20	Fri 21/08/20
CSCI992	47	Teamwork	8h	0d	0d	Sat 22/08/20	Sat 22/08/20
CSCI992	58	Evidence of relevance	56h	0d	0d	Thu 1/10/20	Fri 9/10/20
CSCI992	65	Presentation Training	16h	0d	0d	Wed 28/10/20	Thu 29/10/20

Figure 5.3 Human Resources – Team member

Programmer & Tester: Heng Zhang



Name: Heng Zhang Initials: H Max units: 100% Previous Next

Costs
Std rate: $25.00/h Per use: $0.00
Ovt rate: $10.00/h Accrue at: Prorated

Base cal: Standard
Group:
Code:

Project	ID	Task Name	Work	Leveling Delay	Delay	Scheduled Start	Scheduled Finish
CSCI992	25	Generate Requirement	24h	0d	0d	Wed 20/05/20	Fri 22/05/20
CSCI992	30	Web Setup	16h	0d	0d	Thu 28/05/20	Fri 29/05/20
CSCI992	42	Testing Stage	56h	0d	0d	Fri 7/08/20	Mon 17/08/20
CSCI992	59	Writing Paper	40h	0d	0d	Mon 12/10/20	Fri 16/10/20
CSCI992	2	Study Paper	32h	0d	0d	Fri 20/03/20	Wed 25/03/20
CSCI992	4	Edit code of conduct	24h	0d	0d	Mon 30/03/20	Wed 1/04/20
CSCI992	5	Teamwork	16h	0d	0d	Thu 2/04/20	Fri 3/04/20
CSCI992	8	Study Paper	32h	0d	0d	Mon 6/04/20	Thu 9/04/20
CSCI992	11	Complete ownself tasks	32h	0d	0d	Mon 20/04/20	Thu 23/04/20
CSCI992	12	Teamwork	24h	0d	0d	Fri 24/04/20	Tue 28/04/20
CSCI992	9	Requirement Analysis	32h	0d	0d	Fri 10/04/20	Wed 15/04/20
CSCI992	17	PPT Preparation	8h	0d	0d	Mon 4/05/20	Tue 5/05/20
CSCI992	19	Video clip	16h	0d	0d	Fri 8/05/20	Mon 11/05/20
CSCI992	20	Teamwork	16h	0d	0d	Tue 12/05/20	Wed 13/05/20
CSCI992	21	Study Requirement Paper	16h	0d	0d	Thu 14/05/20	Fri 15/05/20
CSCI992	26	Teamwork	8h	0d	0d	Sat 23/05/20	Sat 23/05/20
CSCI992	29	Learning Github	24h	0d	0d	Mon 25/05/20	Wed 27/05/20
CSCI992	32	Teamwork Documentation Edit	16h	0d	0d	Thu 4/06/20	Fri 5/06/20
CSCI992	35	CP-abe Implementation-1	40h	0d	0d	Mon 8/06/20	Fri 12/06/20
CSCI992	45	Requirement Verfication	16h	0d	0d	Tue 18/08/20	Wed 19/08/20
CSCI992	47	Teamwork	8h	0d	0d	Sat 22/08/20	Sat 22/08/20
CSCI992	50	Implement Requirement(2)	120h	0d	0d	Mon 24/08/20	Fri 11/09/20
CSCI992	57	Team work memo	32h	0d	0d	Fri 25/09/20	Wed 30/09/20
CSCI992	64	Teadeshow PPT	8h	0d	0d	Mon 26/10/20	Tue 27/10/20

Figure 5.4 Human Resources – Team member

6. Project Monitoring

Our team is using agile project management with Scrum framework for managing and monitoring our project.

Scrum is one of the agile methods designed to guide teams to iteratively and incrementally deliver products. It is often referred to as an "agile project management framework" and its focus is on the process of using experience, which enables teams to respond to changes quickly, efficiently, and effectively (Schwaber, 2004). Traditional project management methods have fixed requirements to control time and cost. In contrast, Scrum fixes time and cost to control demand. This is done with time frames, collaboration ceremonies, prioritized product to-do lists, and frequent feedback cycles (Sliger, 2011). Figure 6.1 shows the basic Scrum framework.

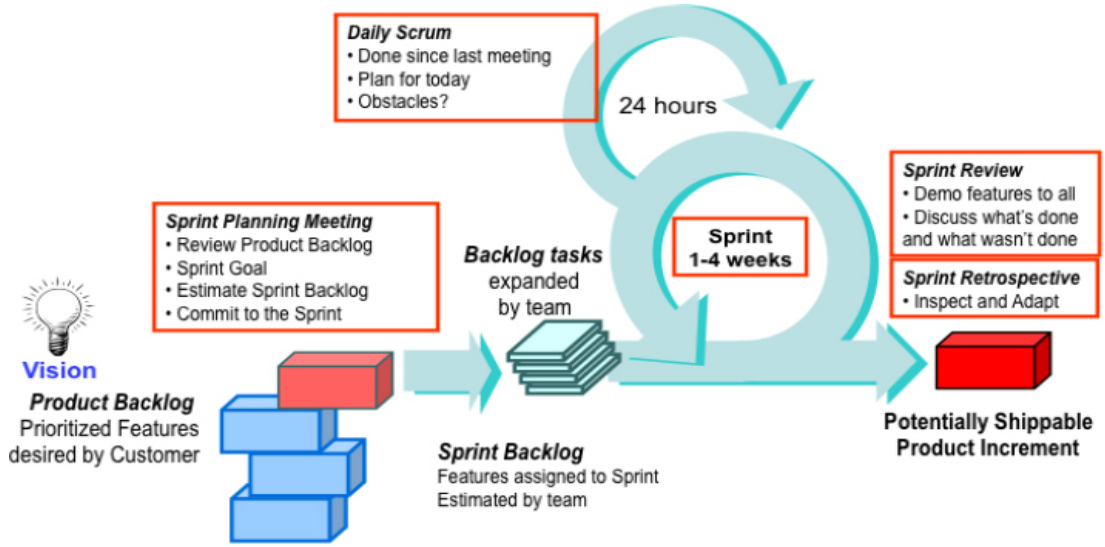


Figure 6.1 The Original Scrum Framework (sliger, 2011)

The project begins with a clear vision provided by the sponsor and our team members, and a set of product features in order of importance. These features are part of the product backlog. A time box commonly referred to as an iteration or sprint, is the set amount of time that the team has to complete the features selected. Sprints are generally from one to four weeks in length, and that length is maintained throughout the life of the project so as to establish a cadence. The team selects items from the product backlog that it believes can be completed in the sprint and creates a sprint backlog consisting of the features and tasks as part of the sprint-planning meeting.

The sprint-planning meeting is held on the first day of every sprint. The sponsor, and entire team are all in attendance. First of all, we are following project plan. The sponsor presents the set of features he would like to see completed in the sprint. Work estimates are reviewed to see if the team has the time to complete all the features requested in the sprint. If so, the team commits to the sprint. If not, the lower priority features go back into the product backlog, until the workload for the sprint is small enough to obtain the team's commitment. Once the sprint planning meeting is complete and the team has made a commitment, the team begins to track its progress using highly visible information radiators.

Once the team has committed to a sprint backlog, the task work begins. During this time in the sprint, the team is protected from interruptions and allowed to focus on meeting the sprint goal. No changes to the sprint backlog are allowed; however, the product backlog can be changed in preparation for the next sprint. During the sprint, the team checks in daily with each other in the form of a 15-minute meeting known as a scrum. Each team member states what they did

yesterday, what they plan to do today, and what is getting in their way. Due to the Covid-19, we are not going to have face-to-face meeting at present, we form a group chat, and send message to the group every day. At the end of the sprint, the team demos the work they have completed to the sponsor and gathers feedback that will affect what they work on in the next sprint. At the end of the sprint, we also invite sponsor to a sprint review meeting where the features that were completed in the sprint are demoed and feedback is requested.

In order to work more efficient, we use project management platform called “Taiga”. Taiga is a free and open-source project management system for start-ups, agile developers, and designers.

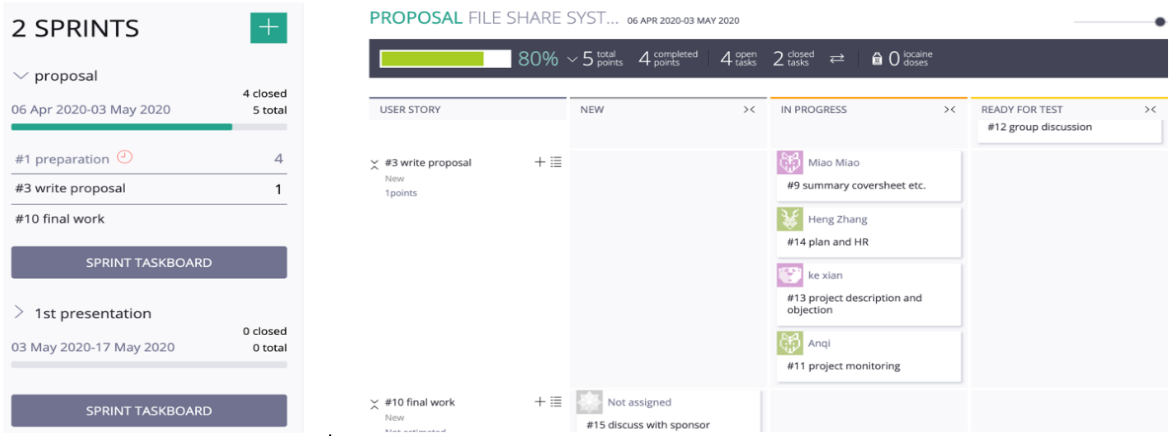


Figure 6.2a sprints | 6.2b sprint task board

Figure 6.2 a sprints and 6.2b sprint task board

As Figure 6.2a shows, we currently have 2 sprints, one is for now, another is for the next sprint, and we have not decided what tasks need to be there yet. In Figure 6.2b, it shows the task board for this sprint. And tasks were assigned to each team member, for those not assigned, actually those were assigned to everyone. By using Taiga, we can easily manage our project, have a clear view of what we need to do and who take the responsibility for each task.

7. Proposed Budget

Our team will cost estimation done by measuring the resource usage (salary of employees), the duration of each task, equipment fee and software service fee. The project manager will

reasonably track project costs, such as viewing potential issues related to project costs, allowing the team to gather appropriate data to generate the necessary metrics and reports.

Our initial budget was $28,000. Our team decided to use a local database to store user information instead of paying to outsource one. This would help save on server costs. Most of the expense for this project is estimated to be the human resource as we will not be outsourcing any software or hardware to store user data. Of course, in the project implementation process, if there are additional needs, we will make appropriate adjustments to the project budget.

Human resource	Rate/hr	Total amount of hrs	Total cost
Anqi Gao	$25	250*	$6,250
Miao Miao	$25	250*	$6,250
Ke Xian	$25	250*	$6,250
Heng Zhang	$25	250*	$6,250
Software Support (github, taiga etc.)			$1,000
Others			$2000
Total	-	1,000	$28,000

*Total working hours is estimate from project plan and add extra hours if we need to work overload.

Project Requirement and Domain Analysis

1. Product constraints

1.1 The purpose of the product

The main purpose of this project is to develop a file sharing system for UOW students and staff. The staff has administrator permission which refers to the teachers of the school of computing and IT. The students play the role of registered users, referring to the undergraduate and graduate students of school of computing and IT.

The system is convenient and reliable.

The system only allows UOW staff in IT support center to create user accounts, which can protect the security of data to a certain extent.

The system allows registered users to share files with a group of people without creating a group. In order to do that, we add attributes to each user which describe the user's identity. When the registered user uploads a file, the registered user can describe who can access the file by selecting attributes. Then the file is encrypted based on the access policy specified the uploader, and upload to the cloud server.

When the registered user downloads the file, the registered user can decrypt the file correctly if and only if the associated access policies are satisfied by the registered user’s attributes. This guarantees the confidentiality of the data.

1.2 The client and other stakeholder

The clients in this product includes: UOW teachers, UOW students

The stakeholders in this product includes: Project group member, Project supervisor, Project source supplier UOW students and UOW staff.

The actual differences among the clients are explained in the next section.

1.3 Users of the product

In this project, the file sharing system mainly designed for UOW. Thus, the users of this project are UOW students and UOW staff. There are two types of users that interact with the web portal: registered users of the system and administrator of the system.

The registered user can only log into the system with account information provided by administrator. This means that users can not register by themselves. The registered user can view files shared by other registered users after successful decryption. The registered user can upload encrypted files as well.

The administrators are managing the whole system. The administrators can manage the information for each registered user as well as the attributes of each registered users, modify users' attributes. The following figure shows the use case diagram of different types of users.

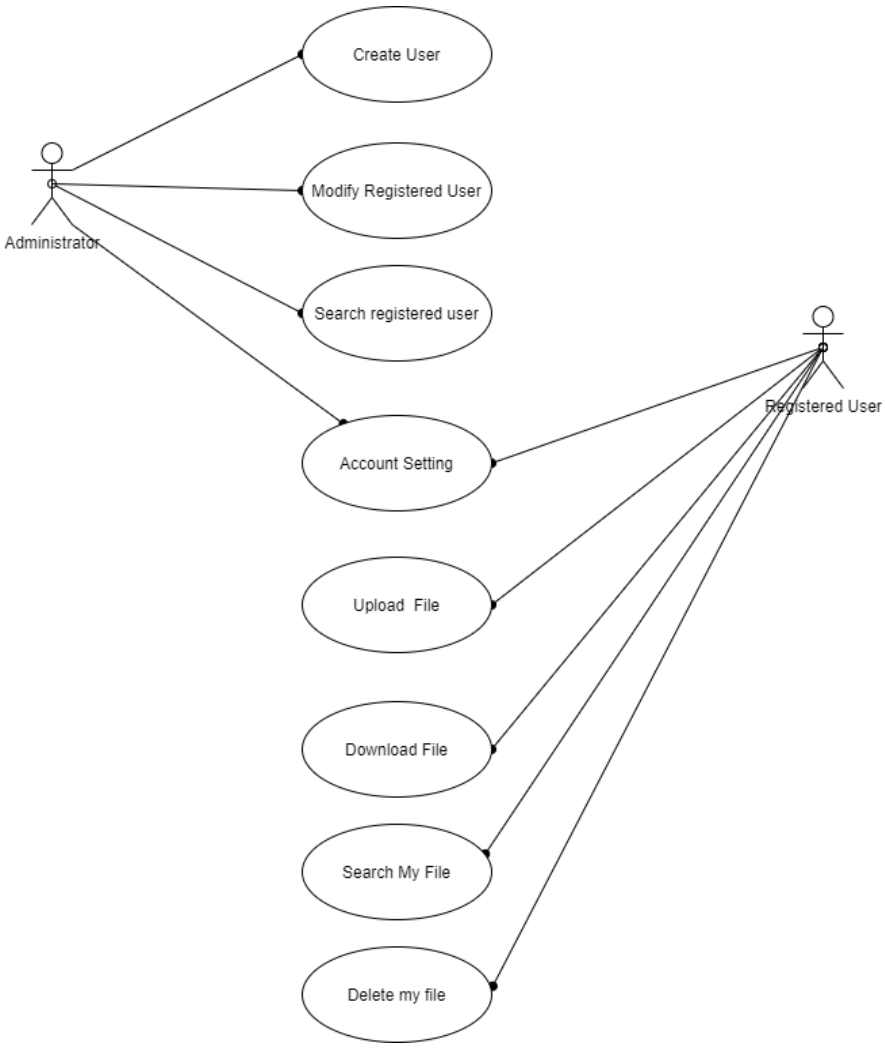


Figure 1.3.1 Use case diagram

1.4 Requirements constraints

Internet is the most important constraint for the system, since the system fetches data from the database over the Internet.

As for operating system, the system can support different systems such as Windows 8 or higher, Mac OS Catalina or higher. The system also supports Linux and Unix system.

Since encryption and decryption require high computation, the device should have enough computing power.

The system supports various versions of browsers, such as Firefox 79.0 or higher, Google Chrome 83.0.4103.116 or higher, Internet Explorer 11 or higher. But in order to get a better user experience, latest version of the browser is recommended.

1.5 Naming conventions and definitions

MTBF: mean time between failure

MTTR: mean time to restoration

FR: functional requirement

NFR: non-functional requirement

1.6 Assumptions

This requirement analysis is design based on our knowledge and imaginations. We assume our team members have enough knowledge to develop the system.

Considering about security, we assume the encryption algorithms and tools we used are secure. We assume the libraries we used work properly without bugs. We assume administrators are trusted.

2. Functional requirements

2.1 The scope of the product

This system is a file sharing system especially for UOW students and UOW staff. The system can support at least 5,000 visitors. The system can support maximum 100 users at the same time. There is no limitation of how many files the user can upload or download, but the maximum size of a single file is 200MB.

2.2 Functional and data requirements

2.2.1 Software requirements specification

ID	FR-01
TITLE	Log in
LEVEL	MUST
DESC	Administrator and registered users MUST log into the system using matched username/user ID and password. Users use correct username/user ID and password MUST be accepted.

ID	FR-02
TITLE	Forget password
LEVEL	MUST/SHOULD
DESC	If administrator/registered user forgot his/her password, the system requires administrator/registered user to provide username and email address/phone number. If the username and email address/phone number matches, the system should send a tempore password to the user via email/message.

ID	FR-03
TITLE	Force user (registered user/administrator) to change default password
LEVEL	MUST
DESC	Once the account created, users can log into the system with a default password. The system MUST ask the registered user/administrator to change the password if the user has a default password.

ID	FR-04
TITLE	Change password
LEVEL	MUST
DESC	The system allows users (registered user/administrator) to change their own password after log in. The system MUST ask for both old and new password. The system MUST ask user to confirm the new password.

ID	FR-05
TITLE	Text fields for password
LEVEL	MUST
DESC	The text fields for password MUST NOT allow copy and paste and the password MUST be hidden by dots.

ID	FR-06
TITLE	Password complexity check
LEVEL	MUST
DESC	If the new password is too simple, the password MUST not be accepted. The password MUST meet eight characters, including letters and numbers.

ID	FR-07
TITLE	Forget password
LEVEL	MAY
DESC	If the user (registered user/administrator) forgets the password, the system MAY send an email to the email address stored in the database with a temporary password.

ID	FR-08
TITLE	Automatic log out
LEVEL	SHOULD
DESC	If the user (registered user/administrator) has not taken any actions in 5 minutes, the register user/administrator SHOULD log out automatically.

ID	FR-09
TITLE	Manage registered user account
LEVEL	MUST
DESC	The system only allows administrators to create/delete registered user account.

ID	FR-10

TITLE	Registered user information check
LEVEL	MUST
DESC	When creating new account, each data fields MUST meet the requirements in **data requirements** (2.2.2).

ID	FR-11
TITLE	Modify registered user's attributes
LEVEL	MUST
DESC	The system only allows administrator to modify registered user's attributes, including adding/dropping attributes. Registered users cannot modify their attributes by themselves.

ID	FR-12
TITLE	Modify attributes
LEVEL	MAY
DESC	The system only allows administrator to modify attributes, including adding/dropping/renaming the category of attribute and attribute itself.

ID	FR-13
TITLE	Registered user filter
LEVEL	SHOULD
DESC	The system SHOULD provide filters which allow administrator to select a group of registered users by attributes or a user by username or ID.

ID	FR-14
TITLE	Upload File
LEVEL	MUST
DESC	The registered user can upload files to the system and select the attributes of who can access the files.

ID	FR-15
TITLE	Delete File

LEVEL	MUST
DESC	The file uploader can delete files from the system. The registered user can delete uploaded files through multiple choices and single choice.

ID	FR-16
TITLE	File usage
LEVEL	MAY
DESC	The registered user MAY known how many times the file the registered user uploaded has been downloaded.

ID	FR-17
TITLE	File list for registered user
LEVEL	SHOULD
DESC	Registered users can see all the files, but the file names for those are not able for the registered user to decrypt (i.e. the registered user's attributes satisfy the requirements) should be in grey.

ID	FR-18
TITLE	Download File
LEVEL	MUST
DESC	Registered users can download the file from system and can see the title and the robust description of the file selected for download.

ID	FR-19
TITLE	Search file
LEVEL	SHOULD
DESC	Registered users can search for files on the system based on full title or keywords as well as keywords of file description. To facilitate the registered user to download and delete operations.

2.2.2 Data requirements

ID	FR-19
TITLE	Users account

LEVEL	MUST
DESC	Administrators and users are stored in the same table with different types. Each record has name, id, password, email address and user type.

ID	FR-20
TITLE	Constraints on id and name
LEVEL	MUST
DESC	The id and name of each user MUST be unique.

ID	FR-21
TITLE	Constraints on email address
LEVEL	MUST
DESC	The email address of each user MUST contains '@', 'uow' and 'edu'.

ID	FR-22
TITLE	Constraints on user type
LEVEL	MUST
DESC	The type of each user MUST be 'registered' or 'admin'

ID	FR-23
TITLE	Attributes categorises
LEVEL	MUST
DESC	The categorises include but not limited to role (tutor etc.), department (SCIT etc.), subject (CSCI992 etc.)

ID	FR-24
TITLE	Attributes
LEVEL	MUST
DESC	Each attribute should fall into one category.

ID	FR-25

TITLE	Registered user associate with attribute
LEVEL	MUST
DESC	Each registered user's attributes MUST store in our database with proper format.

3. Non-functional requirements

3.1 Appearance requirements

ID	NFR-01
TITLE	Colour and logo
LEVEL	SHOULD
DESC	The product SHALL use the color and logo of UOW.

ID	NFR-02
TITLE	Look requirement
LEVEL	SHOULD
DESC	The product SHALL appear simple to use and intuitive, even to the first-time users.

ID	NFR-03
TITLE	Feel requirement
LEVEL	SHOULD
DESC	The product SHALL appear authoritative.

3.2 Usability requirements

ID	NFR- 04
TITLE	Usability requirements - Server access

LEVEL	MUST
DESC	Make sure visitors SHOULD load the site at any time, MUST NOT get error during load. There is MUST NO dead link on the site

ID	NFR-05
TITLE	Usability requirements - Website design
LEVEL	MUST
DESC	Keep the interface as simple as possible to ensure users have a better experience. Focus on what is important, guide visitors to do what you want them to do. MUST unify the interface style of the entire website.

ID	NFR-06
TITLE	Usability requirement - Resolution
LEVEL	SHOULD
DESC	The product SHALL adapt to any screen resolution.

ID	NFR-07
TITLE	Usability requirements - Feedback
LEVEL	SHOULD
DESC	When users visit the page, SHOULD offer an indication of success or failure of their actions.

ID	NFR-08
TITLE	Usability requirements - Credibility
LEVEL	SHOULD
DESC	SHOULD offer a clear “About Us” page together with your contact details. Ensure that the content of the website is true and accurate

ID	NFR-09
TITLE	Usability requirements - Help page
LEVEL	SHOULD

DESC	SHOULD offer a clear help page to provide how to use the system and frequently asked questions.

3.3 Performance requirements

ID	NFR-10
TITLE	Performance requirements - First paint time
LEVEL	MUST
DESC	The time that the user starts from opening the page to seeing the contents of the page is REQUIRED less than 3s.

ID	NFR-11
TITLE	Performance requirements - First screen time
LEVEL	MUST
DESC	The time it takes for everything to appear on the first screen of the user's browser is REQUIRED less than 5s.

ID	NFR-12
TITLE	Performance requirements – Respond rate
LEVEL	MUST
DESC	95% of the operations carried out in the system MUST respond within 5 seconds.

ID	NFR-13
TITLE	Performance requirements - Respond time
LEVEL	MUST
DESC	The maximum response time of the page MUST NOT exceed 30s.

ID	NFR-14
TITLE	Performance requirements - Capability 01
LEVEL	MUST
DESC	The system MUST be capable of supporting at least 5,000 visitors.

ID	NFR-15
TITLE	Performance requirements - Capability 02
LEVEL	SHOULD
DESC	The system SHOULD be able to support 100 concurrent users.

3.4 Operational requirements

ID	NFR-16
TITLE	Operational requirements - System fluency
LEVEL	MUST
DESC	The system MUST be smooth.

ID	NFR-17
TITLE	Operational requirements - Caton
LEVEL	SHOULD
DESC	The system SHOULD reduce the Caton of page conversion.

ID	NFR-18
TITLE	Operational requirements - Learning cost
LEVEL	MUST
DESC	The interface MUST be simple to reduce the users' learning cost.

ID	NFR-19
TITLE	Operational requirements - Operations
LEVEL	MUST
DESC	The system flow MUST satisfy the normal operational process, avoiding the misunderstanding operation process.

ID	NFR-20
TITLE	Operational requirements - Description
LEVEL	MUST

DESC	The descriptive words in the interface MUST be clearly and easy to understand.

3.5 Maintainability and portability requirements

ID	NFR-21
TITLE	Maintainability requirements - MTBF
LEVEL	MUST
DESC	The MTBF of the system MUST be 167 hours in 7*24 hours.

ID	NFR-22
TITLE	Maintainability requirements - MTTR
LEVEL	MUST
DESC	The MTTR of this system MUST in one hour.

ID	NFR-23
TITLE	Maintainability requirements - Maintenance frequency
LEVEL	MUST
DESC	The staff MUST maintain the system and server at least once a week.

ID	NFR-24
TITLE	Maintainability requirements – Support
LEVEL	MUST
DESC	Maintenance staff MUST be proficient in system maintenance, server maintenance and database maintenance.

ID	NFR-25
TITLE	Maintainability requirements - Maintenance time
LEVEL	SHOULD
DESC	Each time of maintenance SHOULD be finished in one hour and the maintenance staff's reaction time SHOULD in half an hour.

ID	NFR-26

TITLE	Portability requirements - Browser
LEVEL	MUST
DESC	The system MUST fits the popular browsers on the software store, such as, Safari, Google Chrome and Firefox and MUST adaptation the browsers' different editions.

ID	NFR-27
TITLE	Portability requirements - OS
LEVEL	MUST
DESC	The file upload system MUST be fit the mainstream operating systems, such as, Mac OS, Windows.

3.6 Security requirements

ID	NFR-28
TITLE	Security requirements – Attribute matching
LEVEL	MUST
DESC	User decrypt ciphertext with attribute-based private key based on access control policies, the result of user-matched attributes MUST be successful

ID	NFR-29
TITLE	Security requirements – Ciphertext storage
LEVEL	MUST
DESC	The ciphertext generated based on attributes MUST be stored on the server, the attributes cannot be fixed

ID	NFR-30
TITLE	Security requirements – Login function
LEVEL	MUST
DESC	The system MUST allow administrators and registered users to log in, and non-registered users MUST have no permissions.

ID	NFR-31
TITLE	Security requirements – Login function

LEVEL	MUST
DESC	The system MUST allow administrators and registered users to log in, and non-registered users MUST have no permissions.

ID	NFR-32
TITLE	Security requirements – File storage
LEVEL	SHOULD
DESC	For files stored in the server, encrypting data at rest SHOULD greatly reduce the chance of hacker attacking the server to obtain files.

ID	NFR-33
TITLE	Security requirements – Data integrity
LEVEL	SHOULD
DESC	Data integrity is important to the user, the system SHOULD prevent any unauthorized changes to the file.

ID	NFR-34
TITLE	Security requirements – Event log
LEVEL	SHOULD
DESC	Recording activities in the event log SHOULD effectively locate the cause of data breach or system crash.

ID	NFR-35
TITLE	Security requirements – Password Strength
LEVEL	MUST
DESC	Users use strong passwords can effectively reduce the risk involved in passwords. The password MUST contain both letters and numbers and MUST be case sensitive. The password MUST be at least 8 characters and 16 maximums.

ID	NFR-36
TITLE	Security requirements – Large File Transfer
LEVEL	SHOULD
DESC	If users share large files via the Internet, our system SHOULD support the transfer of large files.

ID	NFR-37
TITLE	Security requirements – Backup
LEVEL	SHOULD
DESC	Our system SHOULD support backing up the database to ensure that it can be retrieved when data is lost.

ID	NFR-38
TITLE	Security requirements – Rollback
LEVEL	MUST
DESC	When the program has an emergency, our system MUST support rollback to check the code to find the problem.

3.7 Cultural and political requirements

ID	NFR-39
TITLE	Cultural and political requirements – Privacy protection
LEVEL	MUST
DESC	The system MUST be obliged to protect the privacy of users.

ID	NFR-40
TITLE	Cultural and political requirements - Intellectual property
LEVEL	SHOULD
DESC	The system SHOULD accurately use of citation to ensure respect for the achievements of others

4. Diagrams

4.1 Data Flow

Figure 4.1.1 shows the data flow between administrators and registered user and the user database. All the user information is stored in user database. While creating new users, the user firstly needs to tell all the information to the administrator. The system is not involved in this step; thus, it is not in the diagram. The administrator types user's information to the system and assign attributes from attribute database to the user, then the user information and account details are stored in the database. The administrator can also modify user's information and modify attributes information. All the user can change their password after login to the system.

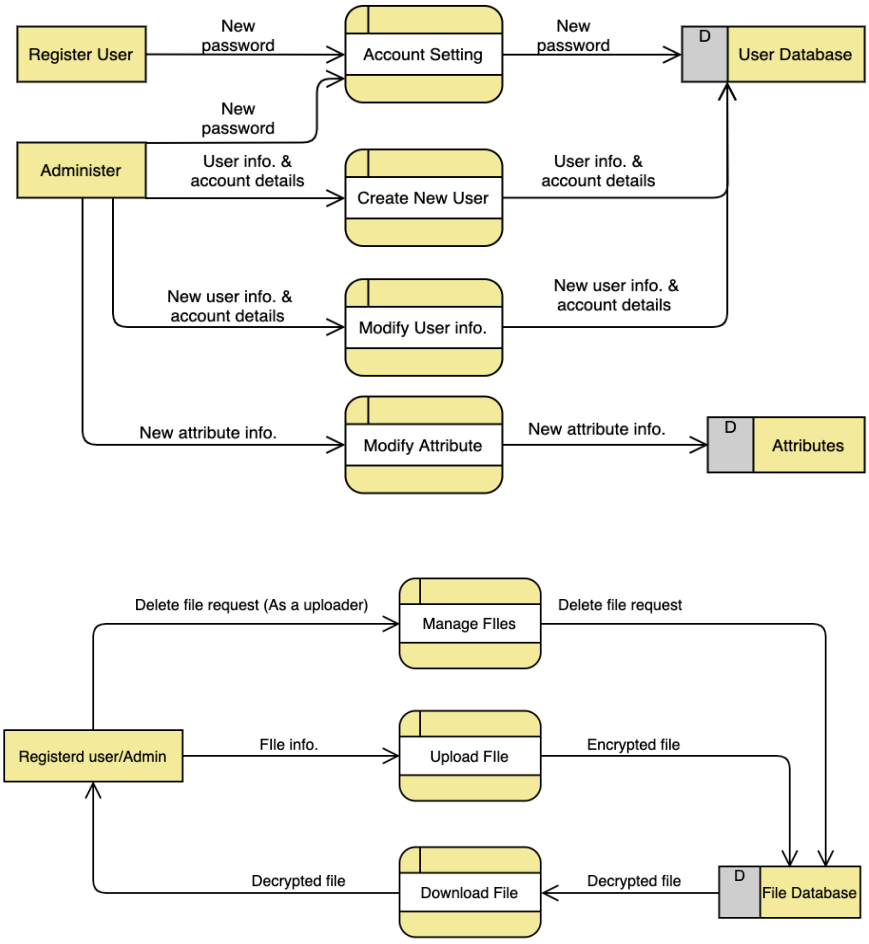


Figure 4.1.1 Data flow 1

Figure 4.1.2 shows the data flow between all the users and file storage. The registered user can upload files to the system. The user can select attributes to describe who can access

the files, those files will be encrypted based on the attributes and the ciphertext will stored in the database. While downloading the files, the registered user sends request to the sever, the server will check whether the registered user's attributes matches the requirements. If yes, the server decrypts the files based on the attributes and sends the plaintext back to the registered user. The registered user can delete files he/she upload.

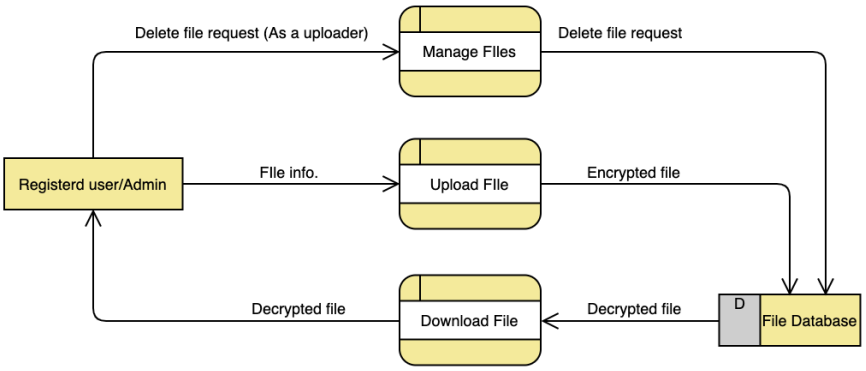


Figure 4.1.2 Data Flow 2

4.2 State UML

Figure 4.2.1 is a state diagram. It shows all the statements registered users and administrators can encounter on the file-sharing system. It also presents the relationship between these states.

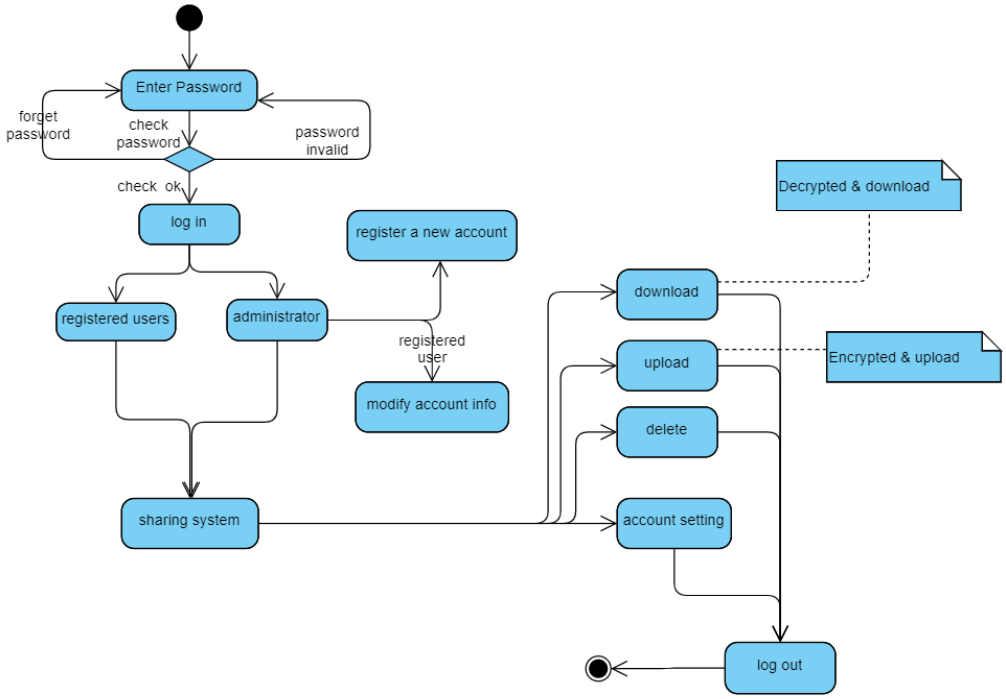


Figure 4.2.1 State diagram

System Design

1. UI Design

Figure 1.1 shows the login page of our system. As for the grey area, we plan to put images to make our website looks better. User can login to the system using username and password. There is a “About Us” button at the bottom, user can access out team information by clicking the button. The reason we provide our team information is we are developing a secure file share system; we want to convince users that our team is trustable.

We also provide a link for sending email with default password, which is “Forget password” next to the “Log in” button. After clicking the link, the user will navigate to forget password page. User is asked for username and email address, if that matches the record in database, we will reset the password to default for user.

Once user entered correct username and password, the user can log into the system by clicking the “Log in” button. After that, user will navigate to the main page.

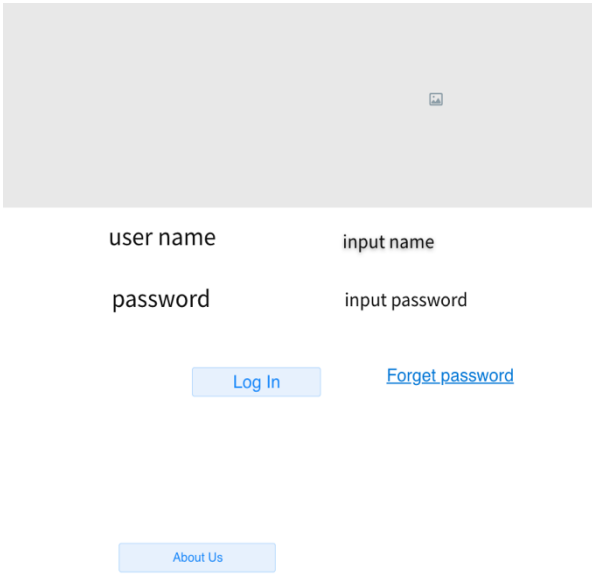


Figure 1.1 Login page

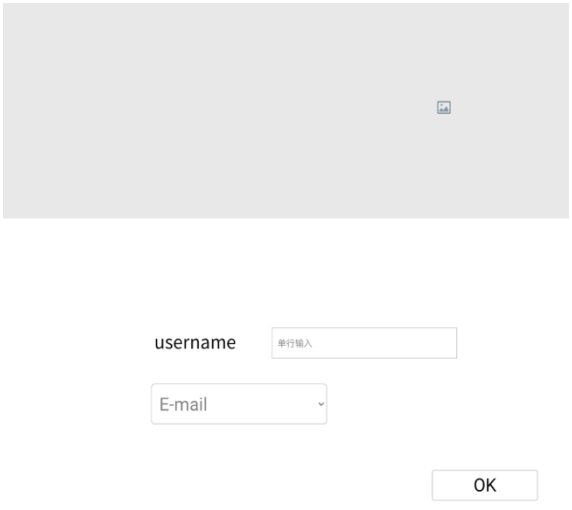


Figure 1.2 Forget password

There two types of user in our system. Firstly, we introduce the administrator main page. There are mainly two functions for administrator, user management and file management. On the left side, we make a menu list for administrator to choose from. The first one is “All users”, which

can list all the user information in the database as shown in Figure 1.3. Administrator can find user by typing user id or username in the search bar. We also provide filter to let administrator find a group of users with specified attribute. The table lists the user information. Administrator can select one of them and modify user information or delete the user form system by clicking the “modify” button in the top left corner. Administrator can register new user by clicking the “add new” button under the “modify” button.

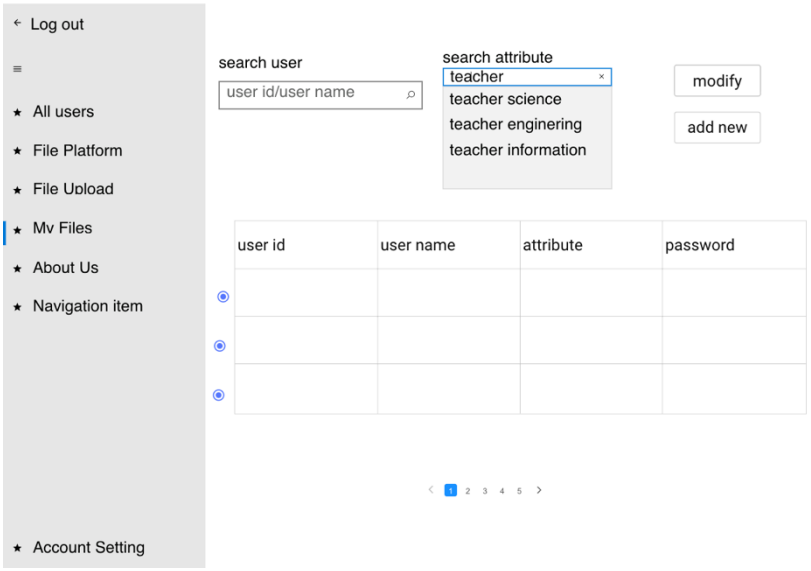


Figure 1.3 Main page of admin

The second function on menu bar is file platform. The menu bar remains on the left for administrator to easily change different functions. The file platform provides searching functions like user management. Files are listed five per page. Administrator can access the

file by click the title. In our system, administrator can access any file in the system. My file page is the same with file platform, but only lists the file uploaded by the administrator.

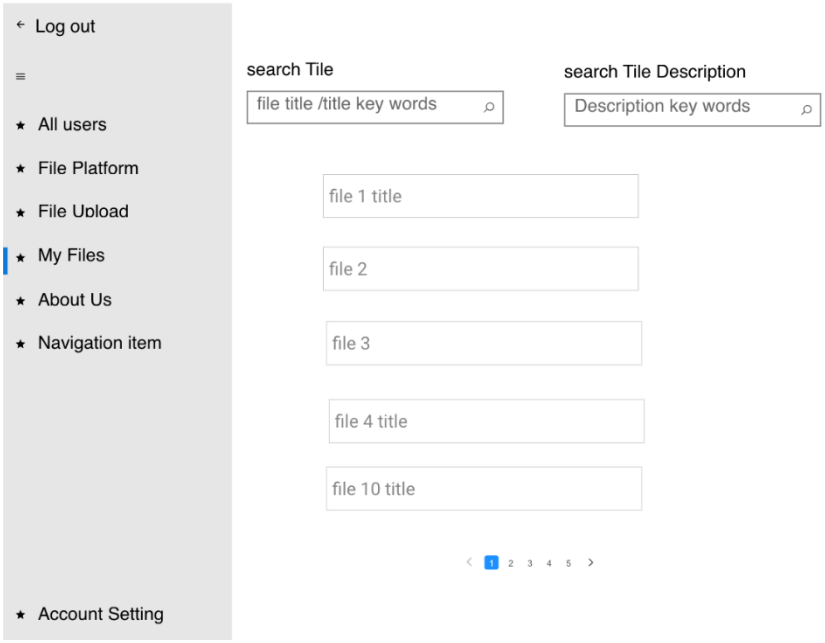


Figure 1.4 File platform (Admin)

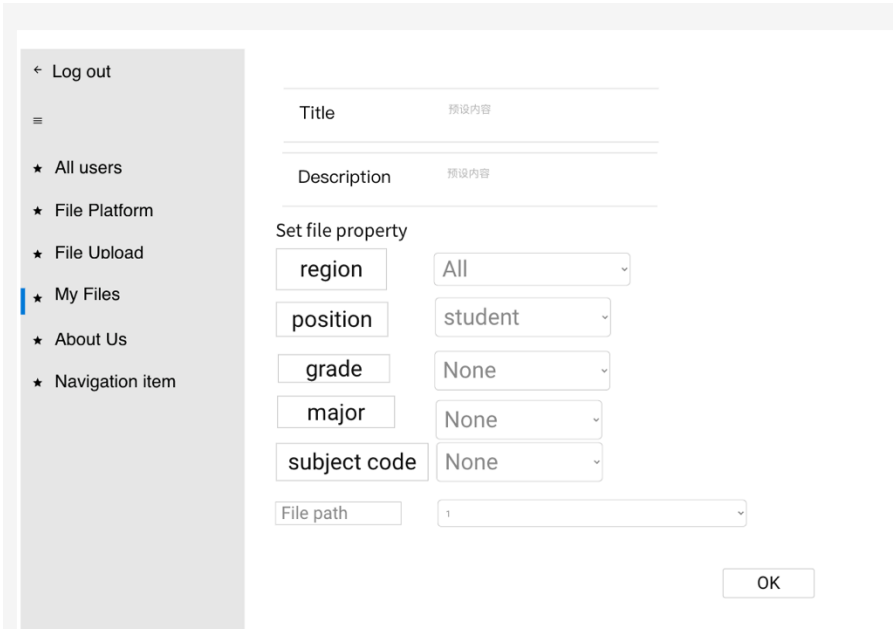


Figure 1.5 File upload (Admin)

The file upload page (Figure 1.5) allows administrator to upload file to the system. Administrator needs to type the title and description, select attributes to define who can access the file, and provide the file path.

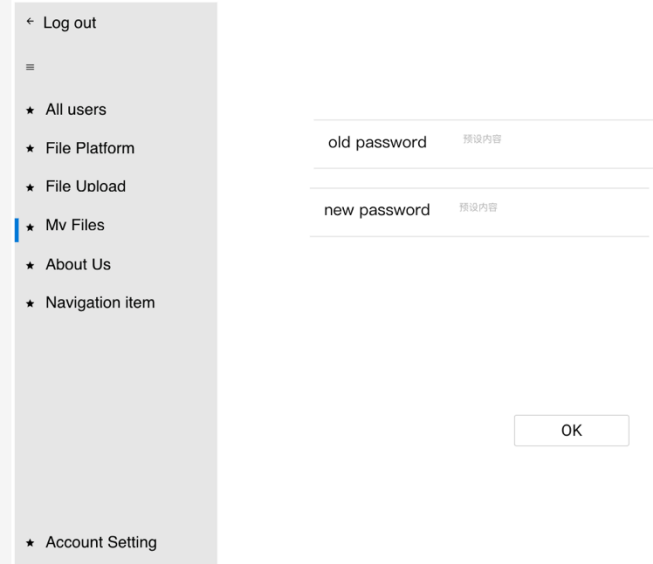


Figure 1.5 account setting page (Admin)

Administrator can change own password in “Account Setting”, which is located at the bottom of the menu bar. Administrator needs to provide correct old password and new password. Administrator can log out from the system clicking “Log out” on the top of the menu bar.

Another user type in our system is registered user. The design for registered user is same as administrator except the “All user” in the menu bar. To avoid repetition, the design for registered user is not introduced in this section.

2. Database Design

The information we need to store in our system is user information and file information. For user information, we decide to have four tables. The overview of database design is shown in Figure 2.1.

First, we need a table to store basic user information shown as the table below.

User_ID	Int (8)	PK
Username	Char (45)	
Password	Char (45)	
Type	Int(8)	
Email address	Char (45)	Contain '@'

The primary key is User_ID. The username and password are used for user to login. The user type is used for identifying registered user and administrator. The email address is used when user forget the password. The email address must contain '@'.

Then we need tables for category and attributes as shown below.

Category_ID	Int(8)	PK
Category_name	Char(45)	

Attribute_ID	Int(8)	PK
Category_ID	Int(8)	Foreign key from Category
Attribute _name	Char(45)	

These two have ID for primary key, and name to indicate the name of the variables. The reason we want to have category_ID in attribute table is because a student in CSCI992 might be a tutor in CSCI203. We link each attribute with a category. For example, in our system, the attribute of a user should be displayed as "CSCI992: student".

Then we need a table to link the user with its attribute.

ID	Int(8)	PK
User_ID	Int(8)	Foreign key from User
Attribute_ID	Int(8)	Foreign key from Attribute

We assign each link with an ID as primary key. The user can have multiple attributes; thus, we can have multiple records with the same User_ID but different Attribute_ID. We already link the attributes with categories, so we do not need it in the User_attribute table.

So far, we have done the user management part. Next, we need to store file information.

File_ID	Int(8)	PK
Title	Char(45)	
Description	Char(500)	
Uploader_ID	Int(8)	Foreign key from User
Path	Char(45)	
Policy	Char(45)	

In the above table, we list the file information stored in our database. Firstly, we have File_ID as primary key. The title and description are basic information of the file. The upload_ID is the file uploader's User_ID. The path indicated the path of the encrypted file stored in our system. The policy is a string that describe the user who can access the file, for example, "CSCI992: student Admin:Admin 1of2".

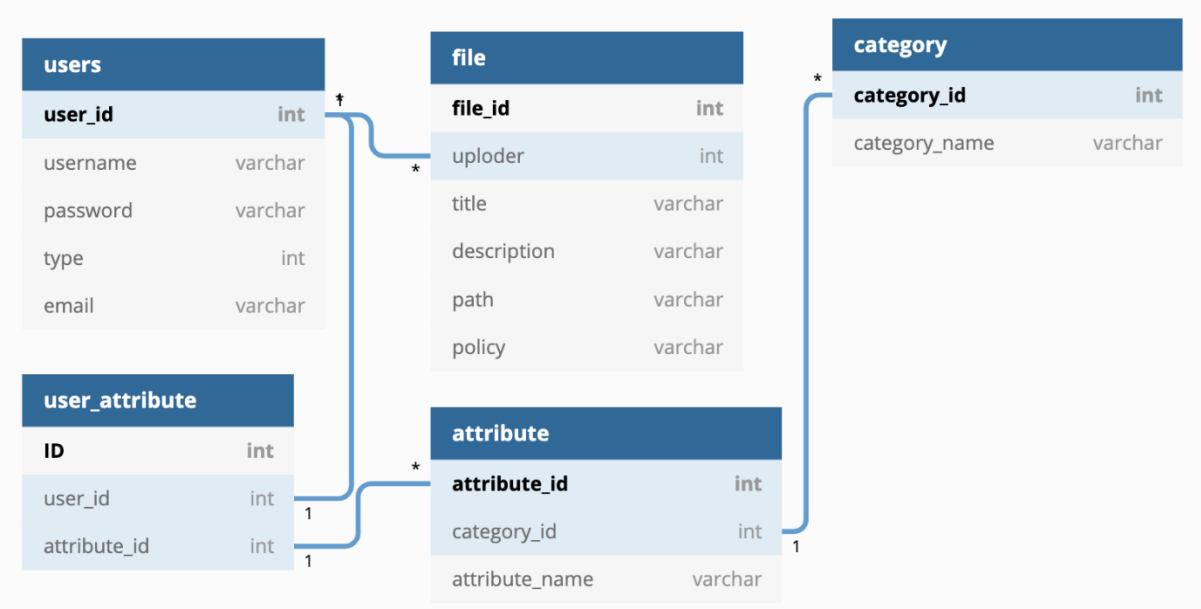


Figure 2.1 database schema

3. Back-end Design

3.1 User Login

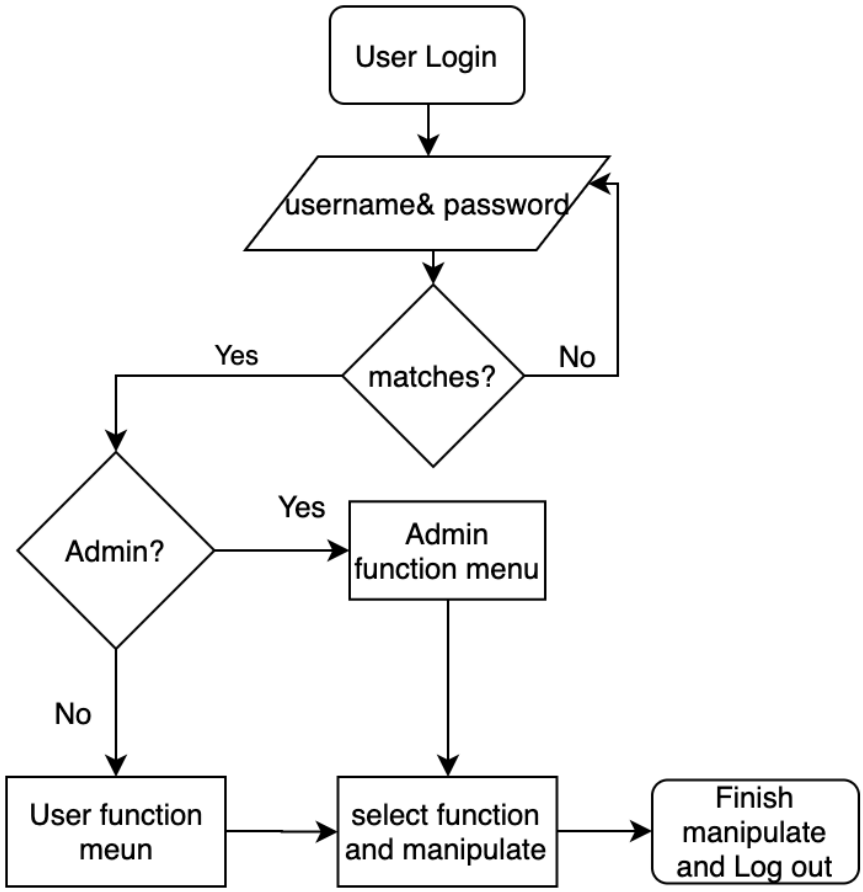


Figure 3.1 Flow Chart for system login

In our system, there are two types of user. Each type of user has different functions. First of all, the user needs to provide username and password to let the system verify the user is an authorised user. Once we got this information, we compare the username and password in our database. At the same time, we check the type of the user to decide which functions should be displayed on the menu bar.

3.2 User & File Management

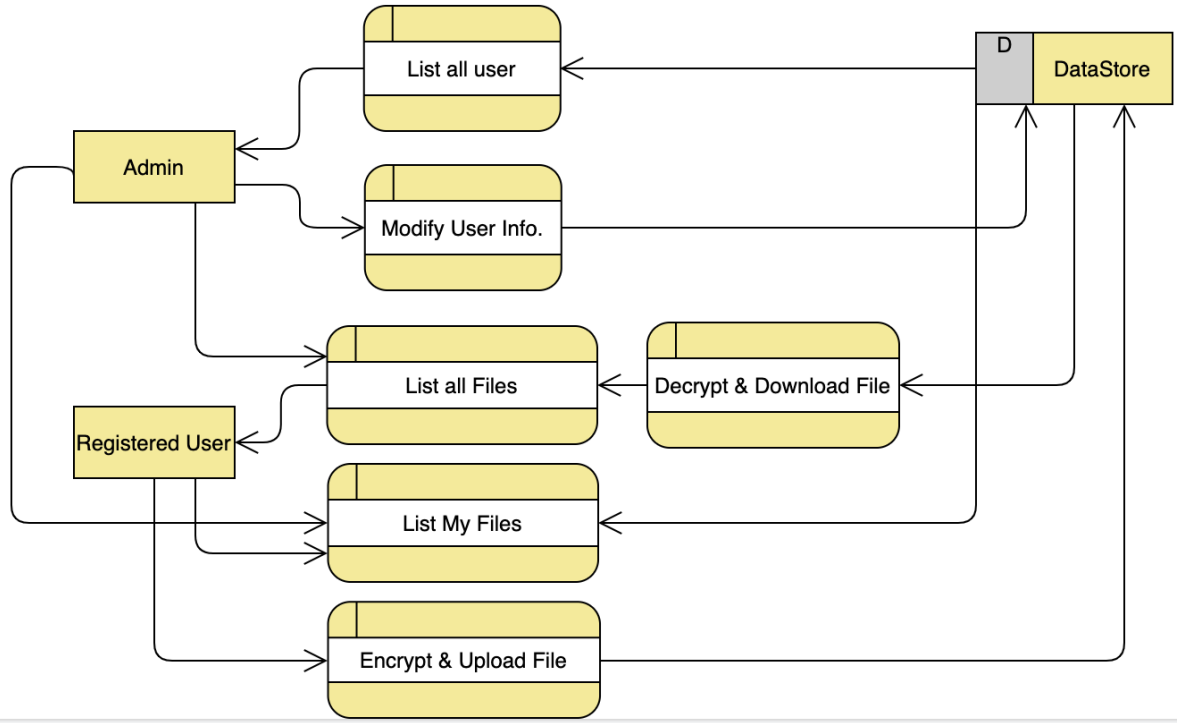


Figure 3.1 Data Flow for different functions

Figure 3.2 shows the data flow for different functions. The first one is “List all user” for administrators. This function will get all the user information stored in the database including the administrators, and we will display the information in a table.

The next function is “Modify user info”. This function will be done by select one user and click the “Modify” button. A new window will pop up to show the user information in different textfields. Each textfield is editable. The administrator can type new information and the function itself will take these as input then update the user information in the database.

The third function is “List all Files” for both administrators and registered users. This function is like “List all user”, it can get all information from the file_info table in the database and display in the website. For each file, we add a column to put a “Download” button for each file. The download function will be explained later. We also add a “Modify” button to each file to let administrators to change the file name, description and policy or delete the file form the database.

We also provide a “List my files” function for both administrators and registered users to view the files uploaded by themselves. We also add a “Modify” button to each file to let

administrators and registered users change the file name, description and policy or delete the file form the database.

We provide “Upload file” function for both administrators and registered users to share a file with other users in our system. There is no difference between administrators and registered users in this function, so we use the word user to represent both. The user first needs to type file information such as title, description and the file path. The file path we only consider physical path, files on the cloud storage is not supported. As for the policy, the system will list all the attributes in the database for user to select. The system does not allow the user to type policy. When uploading the file, the uploader is also saved in the database. Then the file will be encrypted and uploaded to the server. Note, for each file, we add an extra policy to let administrators access the file. The details about the encryption is discussed in section 3.3.

As we mentioned before, we have “Download file” function for administrators and registered users. Firstly, the system will get the private key of the user and try to use the key to decrypt the file. If the file can be decrypted successfully, the system will save the decrypted file and start to download the decrypted file. If the system fails to decrypt the file, it will return a warning saying that the user cannot access the file.

3.3 CP-ABE

The encryption method used for this project is ciphertext-policy attribute-based encryption, which is proposed by Bethencourt,J, et al. in 2007. In this section, four basic algorithms are introduced.

The first algorithm is “setup”, which generates a public key and a master secret key. The setup algorithm takes no input other than the implicit security parameter. It outputs the public parameters PK and a master key MK. The setup algorithm will choose a bilinear group G_0 of prime order p with generator g. Next it will choose two random exponents α, β $\in Z_p$. The public key is published as:

$$\mathrm{PK} = \mathbb{G}_0, g, h = g^{\beta}$$

The master key MK is (β , g^{α}).

The second algorithm is “keygen”, which generates a private key with a given set of attributes. The key generation algorithm will take as input a set of attributes S and output a key that identifies with that set. The algorithm first chooses a random r $\in Z_p$, and then random $r_j \in Z_p$.

for each attribute j $\in$ S. Then it computes the key as:

$$\text{SK} = \big(D = g^{(\alpha+r)/\beta}, \quad \forall j \in S: \quad D_j = g^r \cdot H(j)^{r_j}, D'_j = g^{r_j}\big).$$

The third algorithm is “cpabe-enc”, which encrypts a file according to a policy, which is an expression in terms of attributes. The encryption algorithm encrypts a message M under the tree access structure T. The algorithm first chooses a polynomial q_x for each node x (including the leaves) in the tree T. These polynomials are chosen in the following way in a top down manner, starting from the root node R. For each node x in the tree, set the degree dx of the polynomial q_x to be one less than the threshold value k_x of that node, that is, $d_x = k_x - 1$.

Starting with the root node R the algorithm chooses a random s $\in Z_p$ and sets $q_R(0)$ = s. Then, it chooses d_R other points of the polynomial q_R randomly to define it completely. For any other node x, it sets $q_x(0) = q_{parent(x)}$(index(x)) and chooses d_x other points randomly to completely define q_x.

Let, Y be the set of leaf nodes in T. The ciphertext is then constructed by giving the tree access structure T and computing

$$\text{CT} = \big(\mathcal{T},\ \tilde{C} = Me(g,g)^{\alpha s},\ C = h^s, \quad \forall y \in Y: \quad C_y = g^{q_y(0)}, C'_y = H(\mathsf{att}(y))^{q_y(0)}\big).$$

The fourth algorithm is “cpabe-dec”, which decrypts a file using a private key. The decryption algorithm takes as input the public parameters PK, a ciphertext CT, which contains an access policy A, and a private key SK, which is a private key for a set S of attributes. If the set S of attributes satisfies the access structure A then the algorithm will decrypt the ciphertext and return a message M.

We first define a recursive algorithm DecryptNode(CT, SK, x) that takes as input a ciphertext CT = (T , C,C, ˜ ∀y ∈ Y : Cy, C′ y), a private key SK, which is associated with a set S of

attributes, and a node x from T . If the node x is a leaf node then we let i = att(x) and define as follows: If i ∈ S, then

$$\begin{aligned}\text{DecryptNode}(\text{CT}, \text{SK}, x) &= \frac{e(D_i, C_x)}{e(D'_i, C'_x)} \\ &= \frac{e\left(g^r \cdot H(i)^{r_i}, h^{q_x(0)}\right)}{e(g^{r_i}, H(i)^{q_x(0)})} \\ &= e(g,g)^{rq_x(0)}.\end{aligned}$$

If i /∈ S, then we define DecryptNode (CT, SK, x) = ⊥.

We now consider the recursive case when x is a non-leaf node. The algorithm DecryptNode(CT, SK, x) then proceeds as follows: For all nodes z that are children of x, it calls DecryptNode(CT, SK, z) and stores the output as F_z. Let S_x be an arbitrary kx-sized set of child nodes z such that F_z is not equal to ⊥. If no such set exists, then the node was not satisfied and the function returns ⊥. Otherwise, we compute F_x as described below and return the result.

$$\begin{aligned}F_x &= \prod_{z \in S_x} F_z^{\Delta_{i,S'_x}(0)}, \qquad \text{where } {}^{i=\mathsf{index}(z)}_{S'_x=\{\mathsf{index}(z):z\in S_x\}} \\ &= \prod_{z \in S_x} \left(e(g,g)^{r \cdot q_z(0)}\right)^{\Delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{r \cdot q_{\mathsf{parent}(z)}(\mathsf{index}(z))}\right)^{\Delta_{i,S'_x}(0)} \text{ (by construction)} \\ &= \prod_{z \in S_x} e(g,g)^{r \cdot q_x(i) \cdot \Delta_{i,S'_x}(0)} \\ &= e(g,g)^{r \cdot q_x(0)} \qquad \text{(using polynomial interpolation)}\end{aligned}$$

Now that we have defined our function DecryptNode, we can define the decryption algorithm. The algorithm begins by simply calling the function on the root node R of the tree T. If the tree is satisfied by S we set A = DecryptNode (CT, SK, r) = e(g, g) rqR(0) = e(g, g) ^ rs. The algorithm now decrypts by computing

$$\tilde{C}/(e(C,D)/A) = \tilde{C}/\left(e\left(h^s, g^{(\alpha+r)/\beta}\right)/e(g,g)^{rs}\right) = M.$$

The Implementation of the Project

1. Front-End Implementation

1.1 User management

In this part we present the front end of this project. Figure 1.1.1 shows the login interface, this part uses a from panel to realize login verification, form parameters include “username” and “password” will pass to the back end by json and call the verification function.

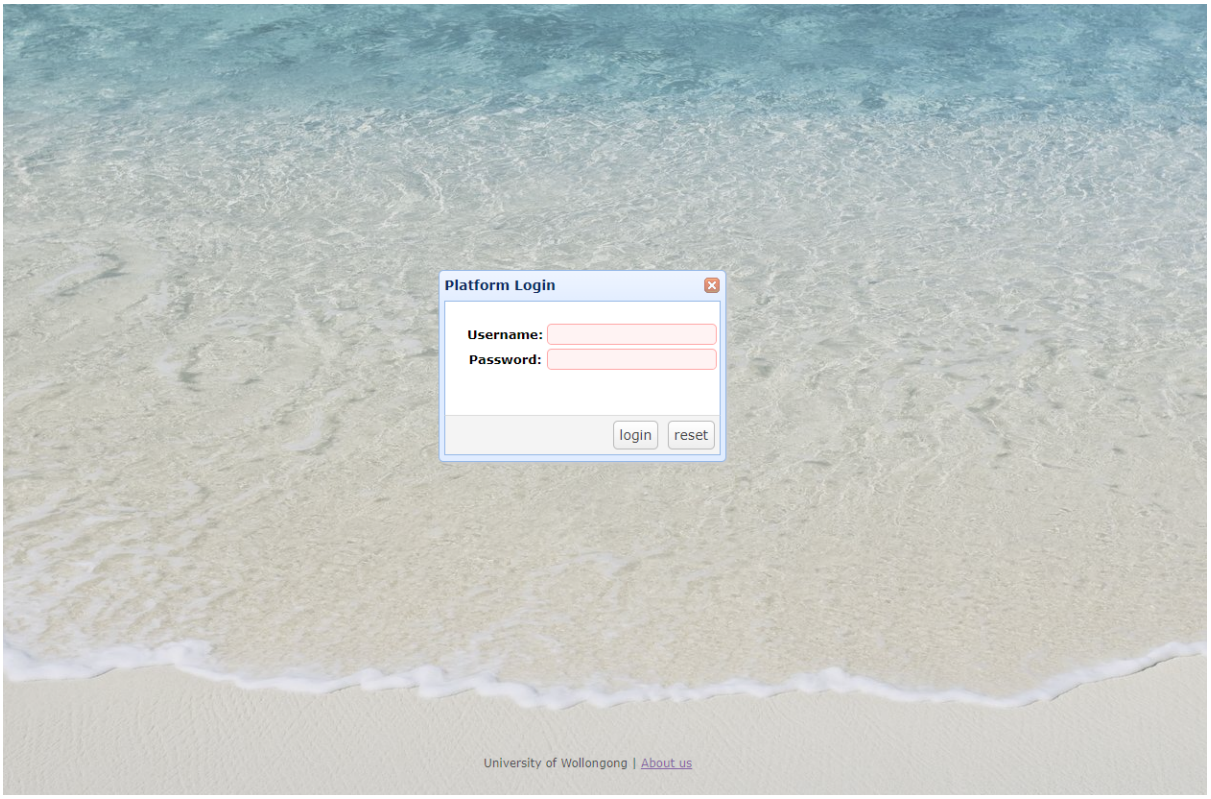


Figure 1.1.1 Login Interface

When verification succeeds, the webpage will jump to the index page. Only the administrator has the authority to manage the user information and file information. As shown in the figure 1.1.2 shows, the user information shows as a list, it contains some user information and operation buttons (add user, detail, delete). This authority is added in the database, there is a “power” table, the front-end using ajax to connect the “UserInfoController” to get the power list, thus only the administrator can visit the user management and file management panels.

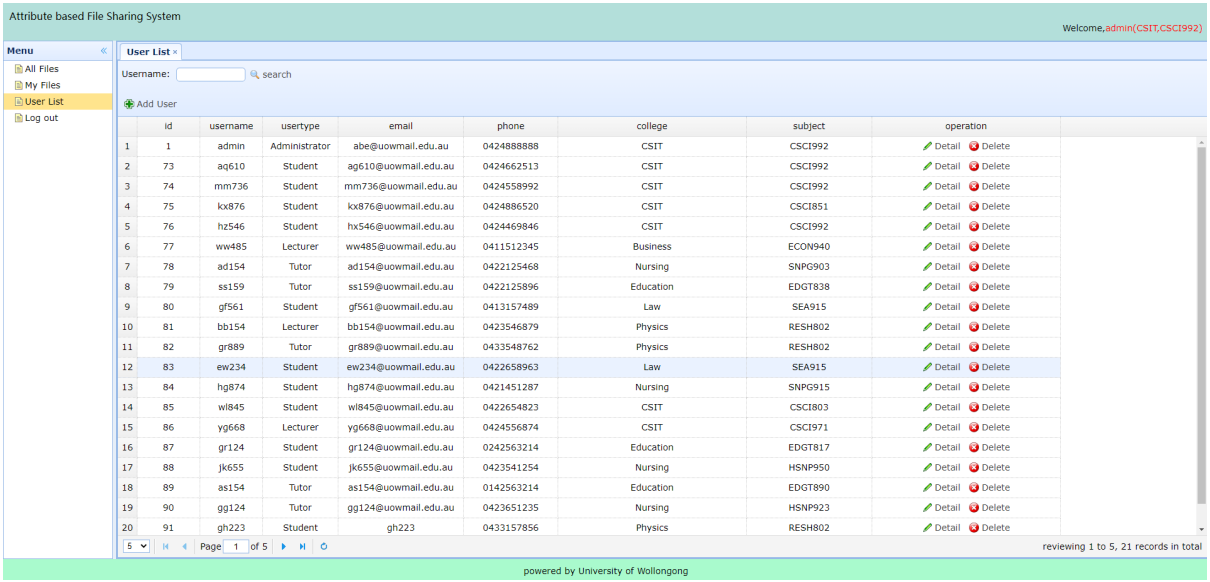


Figure 1.1.2 User Management

To get the user list, this project also uses ajax to get the back-end parameters from the database. The add user part, it designs a form panel, to submit the information to the back end. Figure 1.1.3 displays the add user panel, the user type, college and subject selections are all get from the database. When the information be added and click the save button, it will call the "saveUserInfo" function and pack the data send to the back end. Figure 1.1.4 shows this part code.

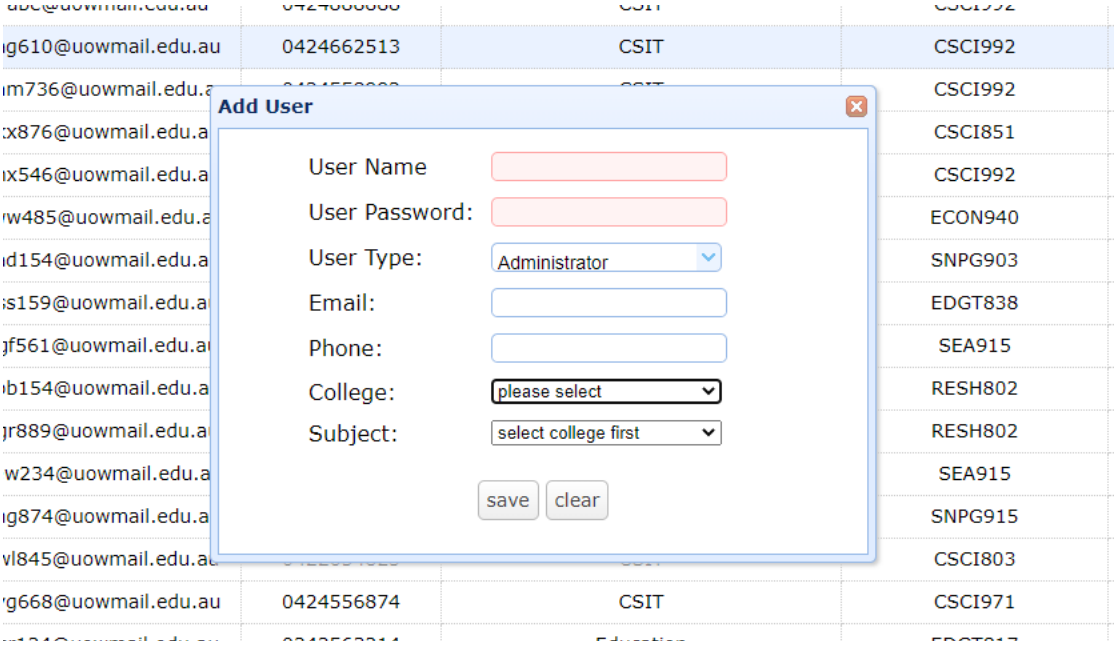


Figure 1.1.3 Add User

```
function saveUserInfo() {
    $("#ff_adduserinfo").form("submit", {
        url : 'userinfo/addUserinfo',
        success : function(result) {
            var result = eval('(' + result + ')');
            if (result.success == 'true') {
                $("#userListDg").datagrid("reload"); //refresh user list
                $('#ff_adduserinfo').form('clear'); //clear the user information form
                $("#dlg_addUser").dialog("close"); //close the user information panel
            }
            $.messager.show({
                title : "tips",
                msg : result.message
            });
        }
    });
}
```

Figure 1.1.4 Save User Information Code

For displaying the user detail, when clicking the "detail" button, it will call the "userInfoDetail" function and send the user id and row id at the same time. The user list will be saved at the web session, the specific user information will display by the row id. This detail panel also can be modified. The UI and code part will show in Figure 1.1.5 and Figure 1.1.6.

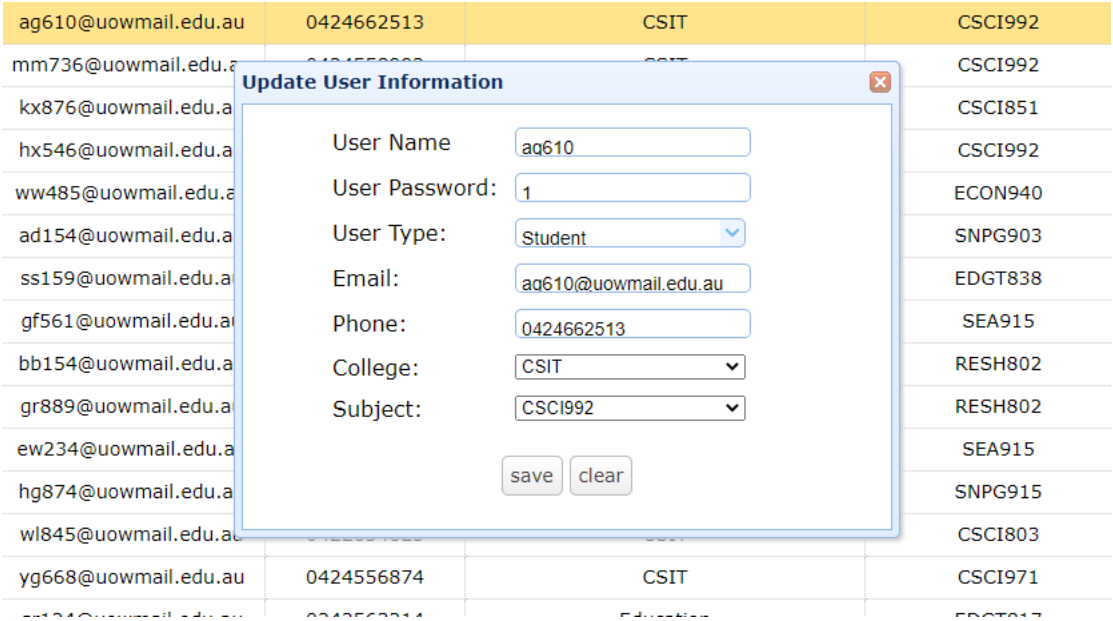


Figure 1.1.5 User Detail

```
// update user information
userInfoDetail :function(id,index){
    //get row data
    var row = $("#userListDg").datagrid('getData').rows[index];
    url = 'userinfo/updateUserinfo';
    document.getElementById("category").options.length=0;  //clear select
    document.getElementById("attribute").options.length=0;  //clear select
    $("#dlg_userinfodetail").dialog("open").dialog('setTitle',
    'Update User Information');
    $("#ff_userinfo").form("load", {
        "id"       : row.id,
        "userName" : row.userName,
        "password" : row.password,
        "userType" : row.userType,
        "email" : row.email,
        "phone" : row.phone,
        "category" : row.category_id,
        "attribute" : row.attribute_id,
    });
```

Figure 1.1.6 Update User Information Code

When deleting the user information, the “delete” button will pass the user id to the back-end and by the user id to delete the user information at the database. The code part shows in the Figure 1.1.7.

```
// delete user information by user id
deleteUserInfo : function (id) {
    $.messager.confirm('Confirm', 'Are you sure deleting this user infomation?', function(r) {
        if (r) {
            $.post('userinfo/deleteUserInfo', {id:id},function(result) {
                if (result.success == 'true') {
                    $("#userListDg").datagrid('reload');
                    $.messager.show({
                        title : 'Tips',
                        msg : result.message
                    });
                } else {
                    $.messager.show({
                        title : 'Tips',
                        msg : result.message
                    });
                }
            }, 'json');
        }
    });
    },
```

Figure 1.1.7 Delete User Information Code

1.2 File management

Figures 1.2.1 represents administrators can download, edit and delete any files in the system. Administrators have the highest authority to manage all the files on the sharing system. Figures 1.2.2 shows that normal users only can download files, they do not have the authority to edit or delete files that other people uploaded.

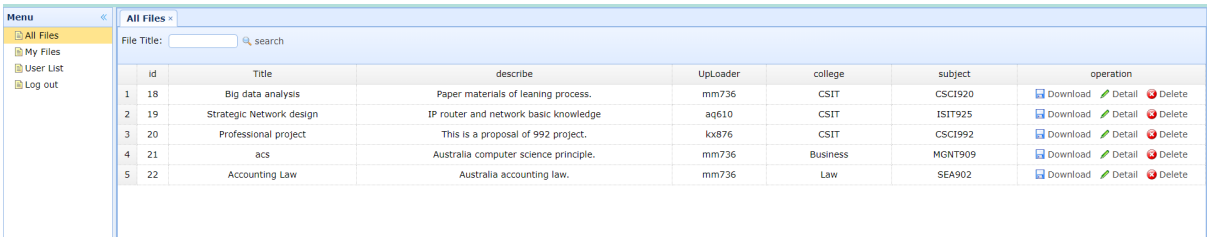


Figure 1.2.1 Admin all files pages

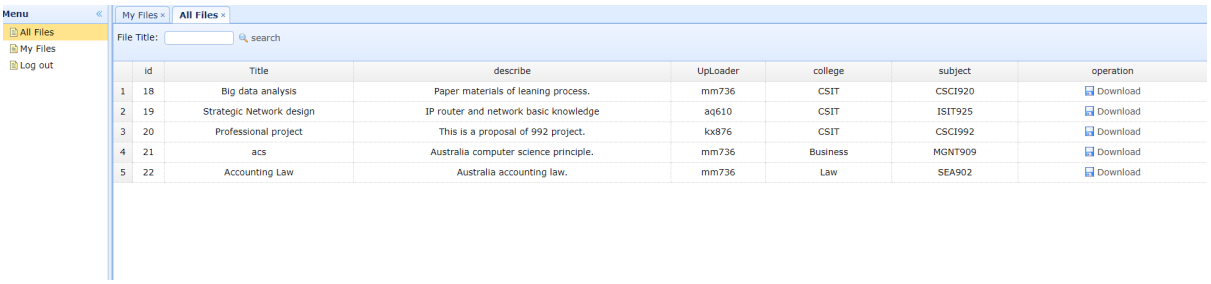


Figure 1.2.2 Normal users all files pages

Figure 1.2.3 shows web implementation of all files page. The login user is administrator when “userType” equals 1, otherwise the login user is student, teacher or university staff.

```
onLoadSuccess: function () {
    if (${sessionScope.user.userType} == "1") {
      $('.download').linkbutton({ text: 'Download', plain: true, iconCls: 'icon-save' });
      $('.detail').linkbutton({ text: 'Detail', plain: true, iconCls: 'icon-edit' }),
      $('.delete').linkbutton({ text: 'Delete', plain: true, iconCls: 'icon-no' });
    } else {
        $('.download').linkbutton({ text: 'Download', plain: true, iconCls: 'icon-save' });
    }
   },
```

Figure 1.2.3 UI code of all files page

The registered user can upload a new file to the sharing system and check all files which uploaded by themselves in my files page (Figure 1.2.4).

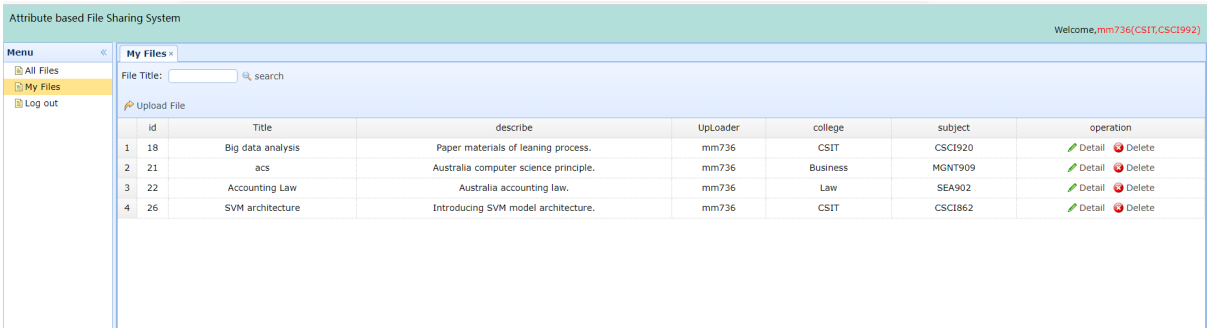


Figure 1.2.4 my files pages

As Figure 1.2.5 shows, add file page will pop out when click upload file button. The registered user can fill out all relative file information and choose uploaded file from local disk. This information was restructured as multipart form, and transfer to controller layer. If this file is empty, controller layer will respond “file is empty” notification as Figure 1.2.6. The code implementation in front-end as Figure 1.2.7 shows.

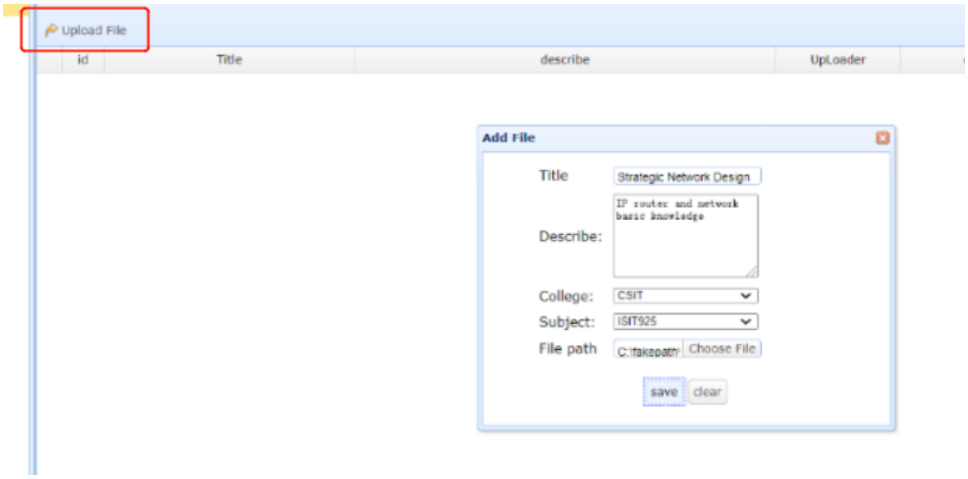


Figure 1.2.5 Upload file

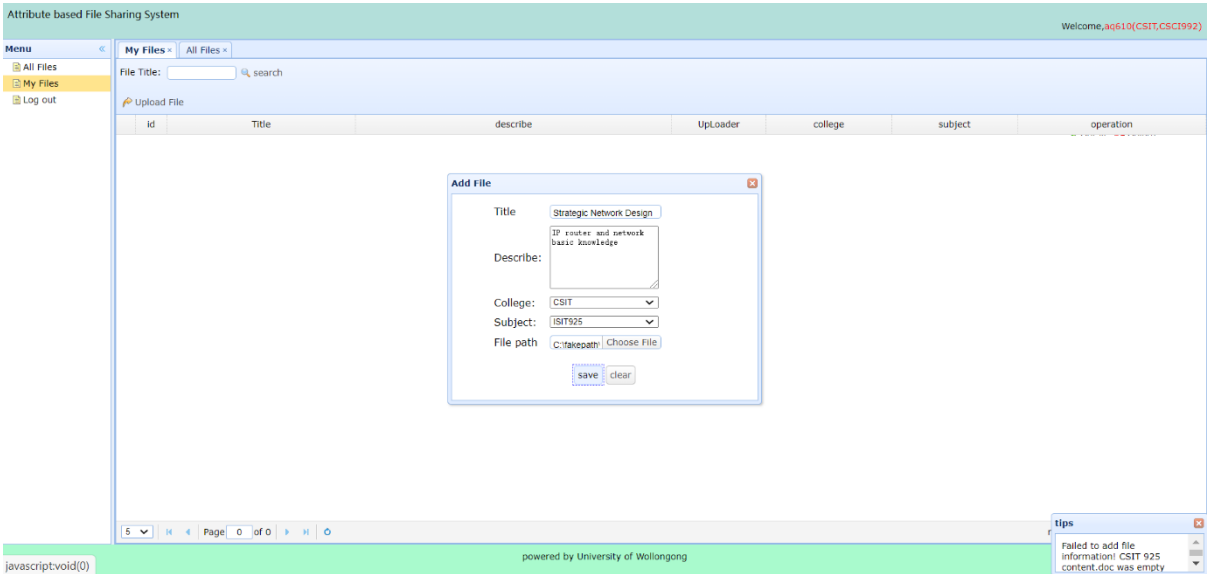


Figure 1.2.6 Upload empty file

```
<form id="ff_updatefileinfo" method="POST" action=""
    enctype="multipart/form-data">
<a href="javascript:void(0)" class="easyui-linkbutton"
   onclick="updateFileInfo();">save</a>

function updateFileInfo() {
    $("#ff_updatefileinfo").form("submit", {
        url : 'fileinfo/updateFileinfo', //????
        success : function(result) {
            var result = eval('(' + result + ')');
            if (result.success == 'true') {
                $("#myfileDg").datagrid("reload"); //refresh user list
                $('#ff_updatefileinfo').form('clear'); //clear the user information form
                $("#dlg_updateFile").dialog("close"); //close the user information panel
            }
            $.messager.show({
                title : "tips",
                msg : result.message
            });
        }
    });
}
```

Figure 1.2.7 Code implementation of upload file

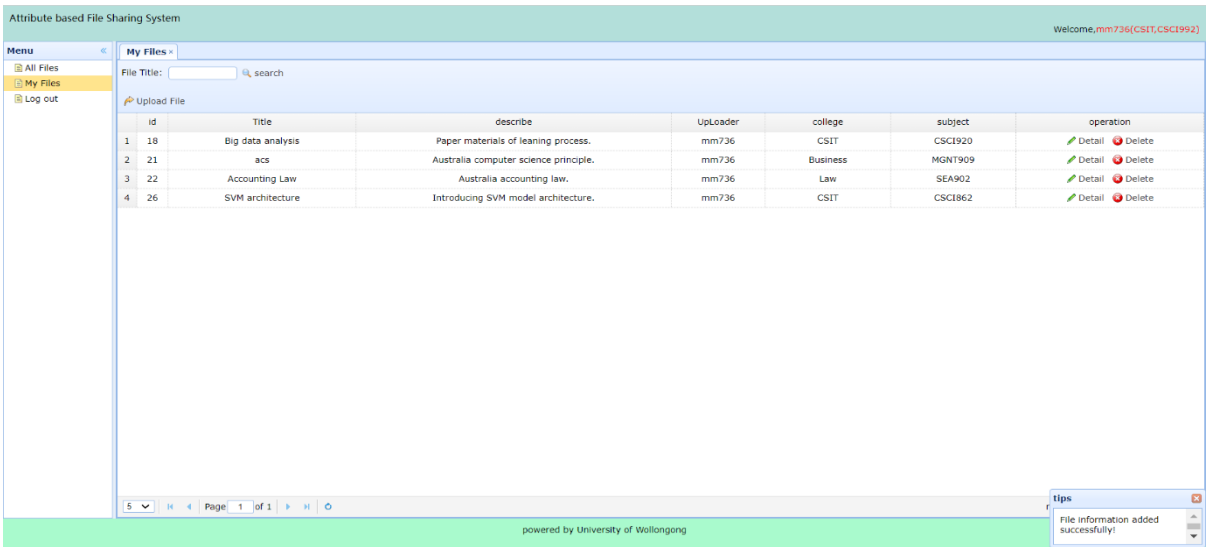


Figure 1.2.8 Upload file successful

The registered user can edit and delete files uploaded by themselves, shows as Figure 1.2.9 to Figure 1.2.12. A confirmation notification will pop out before you delete a file, which avoid of deleting useful file due to incorrect operation. The registered user only can edit file-related information but not upload a new file. You should click upload file button to upload a new file and set relative information if you want to modify the file.

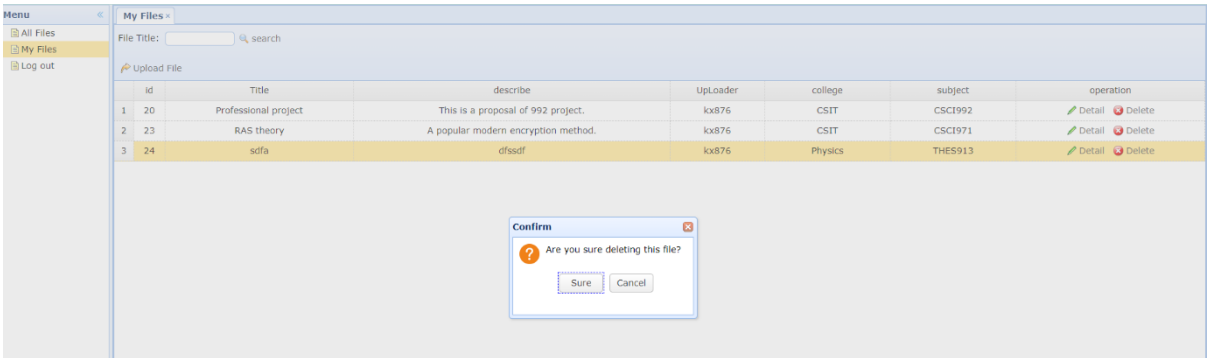


Figure 1.2.9 Delete file

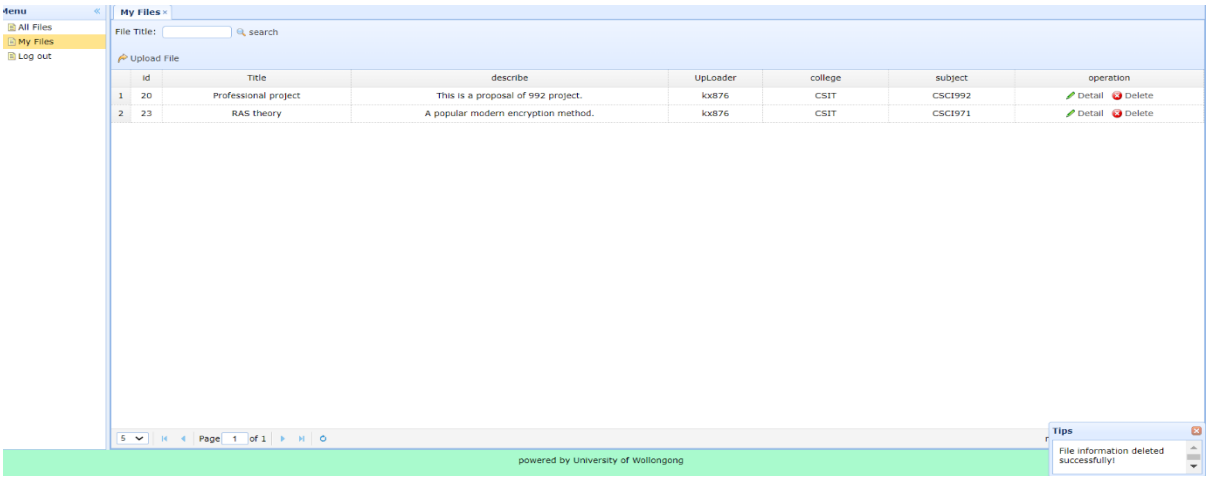


Figure 1.2.10 Delete file successful

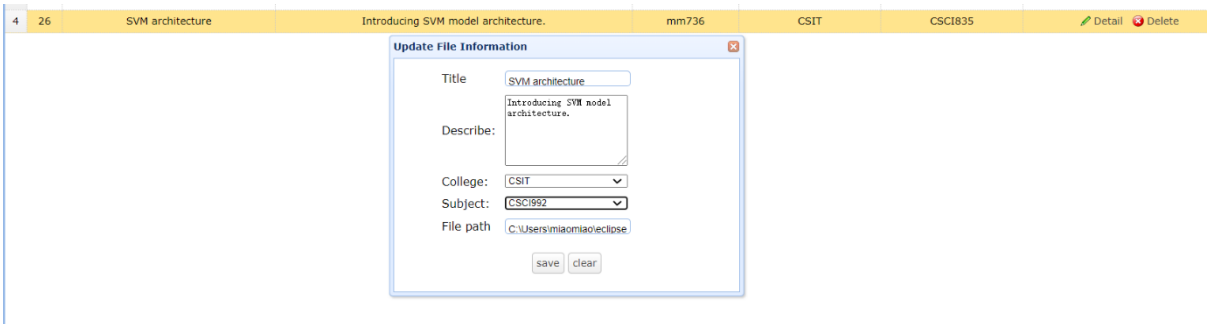


Figure 1.2.11 Edit file information

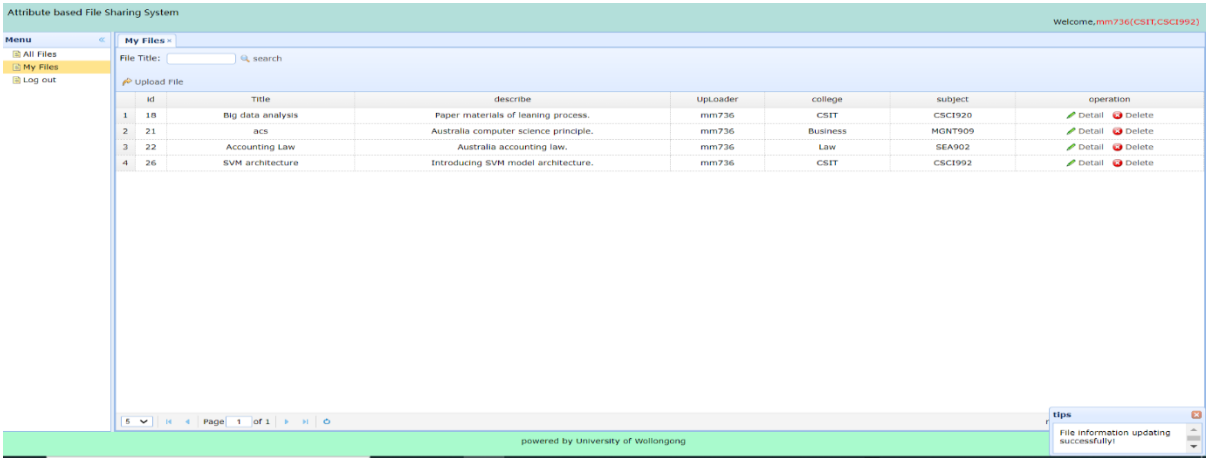


Figure 1.2.12 Edit file information

As Figure 1.2.13 shows, if attribute of the registered user is match with file's, the file will be downloaded to local disk. If the registered user's attribute is not match with file's, the file will not be download and will return a warning message (shown in Figure 1.2.14). When the registered user clicks the "Download" button, the front-end sent file id and user id to back-end, details show as Figure 1.2.15.

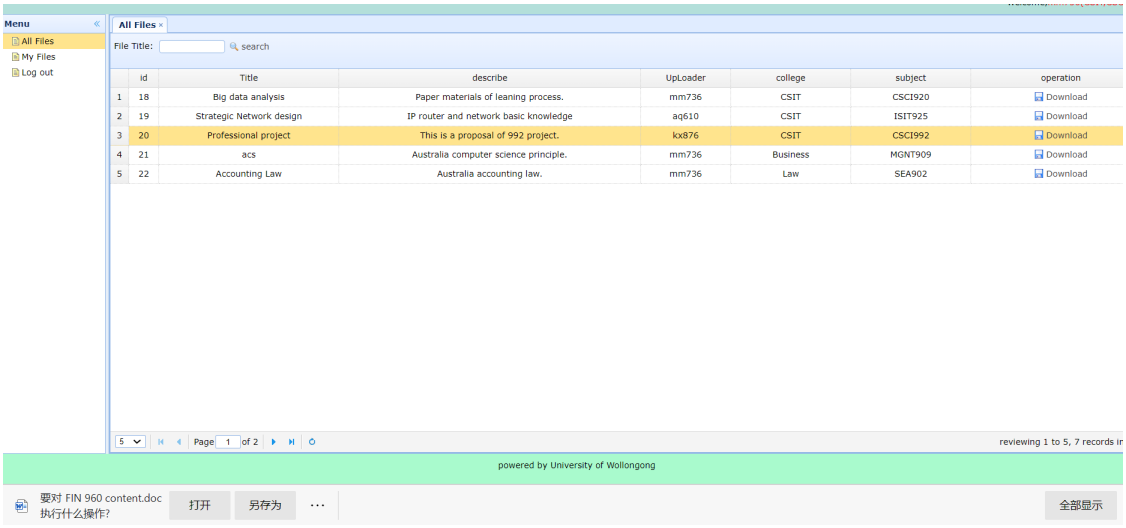


Figure 1.2.13 Download file

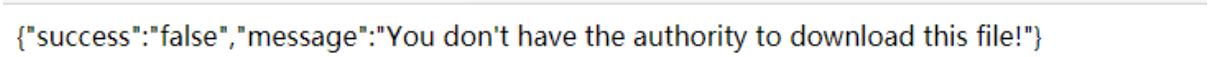


Figure 1.2.14 Download file failed

```
// download file
dloadFileInfo : function (id, userid)
{
    window.location.href = "fileinfo/downloadFileinfo?id=" +id + "&userid=" + userid;
    $.messager.show({
        title : 'Tips',
        msg : result.message
    })

},
```

Figure 1.2.15 Download file code implementation

2. Database Implementation

The “user_info”, “attribute”, “category” and “user_attribute” tables are implemented as what designed before. In this section, we only explain the changes we made.

We made some changes on “file_info” table. In “file_info”, we store the basic file information except the policy. Instead of having policy in the same table, we make another table called “file_attribute” to link the file with attribute like what we do for “user_attribute” table. This is to reduce redundancy in the database. The code for this part is shown below.

```
CREATE TABLE `file_info` (
  `id` int NOT NULL AUTO_INCREMENT,
  `file_title` varchar(45) CHARACTER SET gbk COLLATE gbk_chinese_ci DEFAULT NULL,
  `file_describe` varchar(45) CHARACTER SET gbk COLLATE gbk_chinese_ci DEFAULT NULL,
  `user_id` int DEFAULT NULL,
  `file_path` varchar(200) CHARACTER SET gbk COLLATE gbk_chinese_ci DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `idfile_UNIQUE` (`id`),
  KEY `user_idx` (`user_id`),
  CONSTRAINT `uploader` FOREIGN KEY (`user_id`) REFERENCES `user_info` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=18 DEFAULT CHARSET=gbk;

CREATE TABLE `file_att` (
  `file_id` int NOT NULL,
  `category_id` int DEFAULT NULL,
  `attribute_id` int DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

In order to implement different functions for different types of users, we decide to have two additional tables to store each function with different type of users. Firstly, we have a “functions” table to store all the functions we have in our system. The “url” stores the URL for different pages. In our system, some functions are displayed after certain function is chosen. For example, “download file” can be seen after user selected “list all file function”. This is done by “nodeorder”, “isleaf” and “parentid”. If the function is marked as “isleaf”, that means the function has no further functions provided. If the function is marked as “nodeorder = 2”, that means there is a previous function before it. The “parented” is the ID of the previous function. The code for functions table is shown below.

```
CREATE TABLE `functions` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(20) DEFAULT NULL,
  `parentid` int DEFAULT NULL,
  `url` varchar(50) DEFAULT NULL,
  `isleaf` bit(1) DEFAULT NULL,
  `nodeorder` int DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=gbk;
```

The "power" table link the functions with users. The code is shown below.

```
CREATE TABLE `powers` (
  `uid` int NOT NULL,
  `fid` int NOT NULL,
  PRIMARY KEY (`uid`,`fid`)
) ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

3. Back-end Implementation

3.1 User & file management

This project using Spring-boot framework to realize system coding. The coding part contains five layers (Controller, Service, Service Implement, Dao, Pojo). The "Pojo layer" declare the object so that the "Dao layer" can use these objects, the "Dao layer" operate the database, the "Service layer" declares the function interface and the "Controller layer" can directly call the service interface, the "Service Implement layer" realize the functions which declare at service layer, the "Controller layer" can call the service layer to realize some functions and it also can establish connection with front-end by the connection it can ensure data transfer between front-end and back-end. Next an example will use to display the whole back end realize process. This uses the delete user information as the example.

Firstly, declare the objects at "Pojo layer", this has to declare all the object which will use in the "Dao layer" and the object name should better correspond the field name in the database (Figure 3.1.1).

```java
package com.ecpbm.pojo;

import java.util.List;

public class UserInfo {
    private int id;
    private String userName;
    private String password;
    private int userType;
    private String email;
    private String phone;
    //Associated attributes
    private List<Functions> fs;
    private Integer category_id;
    private Integer attribute_id;
    private String category;
    private String attribute;

    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
```

Figure 3.1.1 Pojo Layer

Next, at the “Dao layer” declare a function and use the MySQL annotation to realize database operation. The function can be called at the service implement layer. At this function, the parameter “userid” be passed into it and use this parameter to delete the user which should be deleted at database (Figure 3.1.2).

```java
//delete user
@DeleteProvider(type = UserInfoDynaSqlProvider.class, method = "deleteUserInfo")
void deleteUserInfo(Integer id);
```

Figure 3.1.2 Dao Layer

Thirdly, at the “Service layer” declare the function as an interface so that the controller can call this function (Figure 3.1.3).

```java
//delete user
@DeleteProvider(type = UserInfoDynaSqlProvider.class, method = "deleteUserInfo")
void deleteUserInfo(Integer id);
```

Figure 3.1.3 Service Layer

Next, the “Service implement layer” inherit the service layer, thus the functions which declare at the service layer must be implemented at this implement layer. This layer can directly call the function realized at the dao layer (Figure 3.1.4).

```java
@Override
public void deleteUserInfo(Integer id) {
    // TODO Auto-generated method stub
    userInfoDao.deleteUserInfo(id);
}
```

Figure 3.1.4 Service Implement Layer

Finally, by preparing those interfaces, the controller can realize specific operation rely on the parameter from the front-end. The front-end using ajax to pass parameter to the controller,

controller rely on it to call functions for finishing this process and return status to the front-end (Figure 3.1.5).

```
//delete user info
@RequestMapping(value = "/deleteUserInfo", produces = "text/html;charset=UTF-8")
@ResponseBody
public String deleteUserInfo(@RequestParam(value = "id") String id) {
    String str = "";
    try {
        userInfoService.deleteUserInfo(Integer.parseInt(id));
        str = "{\"success\":\"true\",\"message\":\"User information deleted successfully!\"}";
    } catch (Exception e) {
        str = "{\"success\":\"false\",\"message\":\"Failed to delete user information!\"}";
    }
    return str;
}
```

Figure 3.1.5 Controller Layer

The whole system back-end realize by this process as the example shows.

3.2 File upload & download

In this web-basedproject, we use “HttpServletRequest” instance to manage file. Using multipart request upload local files to server. Front-end send a multipart request uploading the content of form, there is a request object that implements a “HttpServletRequest” interface in the controller instance.

In the controller instance, using “request.getSession().getServletContext().getRealPath()” get a directory path of servlet, using “java.io.File.TransferTo()” method to write the upload file to the specific file to the server. In the server, encrypting plain text file to cipher text file, and delete the plain text file. The details of implements back-end code shows as Figure3.2.1.

```
//add file info
@RequestMapping(value = "/addFileinfo", produces = "text/html;charset=UTF-8")
@ResponseBody
public String addFileInfo(FileInfo fi,
        HttpServletRequest request,
        @RequestParam(value="file") MultipartFile file) throws Exception {
    String str = "";
    if(!file.isEmpty())
    {
        try {
            String Cpath = request.getSession().getServletContext().getRealPath("/uploads/");      //cipher file
            String Ppath = request.getSession().getServletContext().getRealPath("/storeuploads/"); // plain file

            System.out.println(Ppath);

            String filename = file.getOriginalFilename();
            File filepath = new File(Ppath,filename);
            //if not exist then create
            if (!filepath.getParentFile().exists()) {
                filepath.getParentFile().mkdirs();
            }

            File encpath = new File(Cpath, filename);
            if(!encpath.getParentFile().exists())
            {
                encpath.getParentFile().mkdirs();
            }

            file.transferTo(filepath);

            String encfilename = Cpath + filename;
            String attributes = attributeService.getAttributeName(fi.getAttribute_id());
            String categorys = categoryService.getCategoryName(fi.getCategory_id());
            String policy = categorys + ":" + attributes;

            String inputfile = Ppath + filename;
            cpabeService.enc(pubfile, policy, inputfile, encfilename);

            System.out.println(encfilename);
            filepath.delete();
            fi.setFile_path(encfilename);
            fileInfoService.addFileInfo(fi);

            int file_id = fileInfoService.findFileIdByTitle(fi.getFile_title());
            fileInfoService.addFileAttributes(file_id, fi.getCategory_id(), fi.getAttribute_id());
            str = "{\"success\":\"true\",\"message\":\"File information added successfully!\"}";
        } catch (Exception e) {
            e.printStackTrace();
            str = "{\"success\":\"false\",\"message\":\"Failed to add file information! " + file.getOriginalFilename() +"too large!\"}";
        }
    }
    else {
        str = "{\"success\":\"false\",\"message\":\"Failed to add file information! " + file.getOriginalFilename() + " was empty\"}";
    }

    return str;
}
```

Figure 3.2.1 Upload file controller instance

In download controller instance, “HttpServletResponse” object represents response of server. Firstly, decrypting cipher text file to plain text file, and then Using “response.setContentType()” modifing response encoding method, “response.SetHeader()” setting a response header. Getting byte stream via “ServletOutputStream getOutputStream()” method, through the write(byte[] bytes) of the byte stream, bytes can be written into the response buffer, and then the byte content will be formed by the Tomcat server and returned to the browser. The details of implements back-end code shows as Figure 3.2.2.

```
//download file
@RequestMapping(value = "/downloadFileinfo", produces = "text/html;charset=UTF-8")
@ResponseBody
public String downloadFileinfo(@RequestParam(value = "id") String id,
                               @RequestParam(value = "userid") String userid,
        HttpServletRequest request, HttpServletResponse response) throws IOException, Exception {
    String str = "";
    String filePath = fileInfoService.findFilePathById(Integer.parseInt(id));
    // Get decrypted file path
    String path = request.getSession().getServletContext().getRealPath("/uploads/");    //cipher file
    // Decrypte ciphertext to plaintext
    String Dpath = request.getSession().getServletContext().getRealPath("/Decuploads/");// plain file
    File tmpFile = new File(filePath);
    String fileName = tmpFile.getName();
    File decpath = new File(Dpath, fileName);
    if(!decpath.getParentFile().exists())
    {
        decpath.getParentFile().mkdirs();
    }

    Integer categ_id = userInfoservice.findCategory(Integer.parseInt(userid));

    String attr_str = categoryService.getCategoryName(categ_id)+":"+attributeService.getAttributeName(userInfoservice.findAttribute(Integer.parseInt(userid))) ;
    byte[] prvkey = cpabeService.keygen(pubfile, mskfile, attr_str);
    String encFile = path + fileName;
    String decFile = Dpath + fileName;
    cpabeService.dec(pubfile, prvkey, encFile, decFile);

    File filepath = new File(decFile);
    if(filepath.getAbsoluteFile().exists())
    {
        response.setContentType("application/x-msdownload;charset=utf-8");
        response.setHeader("Content-Disposition", "attachment;filename=" + fileName/* "goals.docx" */ );
        response.setHeader("Content-Length",String.valueOf(filepath.length()));

        try {
            FileInputStream inputStream = new FileInputStream(filepath);
            ServletOutputStream outputStream = response.getOutputStream();
            int num = 0;
            byte[] b = new byte[1024];
            while ((num = inputStream.read(b)) != -1) {
                outputStream.write(b, 0, num);
            }
            inputStream.close();
            outputStream.close();
            outputStream.flush();
            filepath.delete();
            str = "{\"success\":\"true\",\"message\":\"Download file successfully!\"}";
        }
        catch (IOException e){
            e.printStackTrace();
            str = "{\"success\":\"false\",\"message\":\"Failed to download file!\"}";
        }
    }
    else {
        str = "{\"success\":\"false\",\"message\":\"You don't have the authority to download this file!\"}";
    }
    return str;
}
```

Figure 3.2.2 Download file controller instance

3.3 CP-ABE

In this project, we user jPBC library for the CP-ABE part. This library is developed by Angelo D.C. and Vincenzo I. in 2011. The version we used in this project is 1.2.0.

The prototypes of four basic functions mentioned in system design is shown below. We have introduced the mathematic principles in the design part, in this section, we will look into code to see how those algorithms are implemented.

```
public interface CpabeService {
        //generate public key and master key at the first time
        public void setup(String pubfile, String mskfile) throws IOException;

        //generate private key by att and return public key byte[]
        public byte[] keygen(String pubfile, String mskfile, String attr_str) throws IOException, NoSuchAlgorithmException;

        //encrypt the file
        public void enc(String pubfile, String policy, String inputfile,
                        String encfile) throws Exception;

        //decrypt the file
        public void dec(String pubfile, byte[] prvfile, String encfile,
                        String decfile) throws Exception;
}
```

The calculations are done within the Bswabe class. The implementation of how keys are generated is quite long, we put code in attached file (Bswabe.java), the details in not discussed is this section.

The setup algorithm will generate public key and master as shown in the code below. The public key and master key are stored in two files respectively, and then stored in the server. The code is shown below.

```
@Override
public void setup(String pubfile, String mskfile) throws IOException {
        // TODO Auto-generated method stub
        byte[] pub_byte, msk_byte;
        BswabePub pub = new BswabePub();
        BswabeMsk msk = new BswabeMsk();
        Bswabe.setup(pub, msk);

        /* store BswabePub into mskfile */
        pub_byte = SerializeUtils.serializeBswabePub(pub);
        commonService.spitFile(pubfile, pub_byte);

        /* store BswabeMsk into mskfile */
        msk_byte = SerializeUtils.serializeBswabeMsk(msk);
        commonService.spitFile(mskfile, msk_byte);

}
```

The keygen () method will read public key and master key from files, and generate private key based on the attributes. Those attributes are convert to a specific format using parseAttribure() method. We use the same method to deal with policy when we encrypt the file. This is to make sure the program can understand both attributes and policy. The driver code is shown below.

```java
@Override
public byte[] keygen(String pubfile, String mskfile, String attr_str) throws IOException, NoSuchAlgorithmException {
        // TODO Auto-generated method stub
        BswabePub pub;
        BswabeMsk msk;
        byte[] pub_byte, msk_byte, prv_byte;

        /* get BswabePub from pubfile */
        pub_byte = commonService.suckFile(pubfile);
        pub = SerializeUtils.unserializeBswabePub(pub_byte);

        /* get BswabeMsk from mskfile */
        msk_byte = commonService.suckFile(mskfile);
        msk = SerializeUtils.unserializeBswabeMsk(pub, msk_byte);

        /* store BswabePrv into prvfile */
        String[] attr_arr = langPolicyService.parseAttribute(attr_str);
        BswabePrv prv = Bswabe.keygen(pub, msk, attr_arr);

        /* store BswabePrv into prvfile */
        prv_byte = SerializeUtils.serializeBswabePrv(prv);
        return prv_byte;
}
```

Part of the code for encryption is shown below (left). Firstly, we get public key from the file, and take the policy string pass to the method. Then we encrypt the file based on the policy and store the ciphertext into a file. As for decryption, we get public key and private key, try to decrypt the file from “encfile” stored on the server as shown below (right). If the user’s attributes satisfied the policy, the user can decrypt the file, otherwise the system will return an error.

```java
/* get BswabePub from pubfile */
pub_byte = commonService.suckFile(pubfile);

pub = SerializeUtils.unserializeBswabePub(pub_byte);

keyCph = Bswabe.enc(pub, policy);
cph = keyCph.cph;
m = keyCph.key;
System.err.println("m = " + m.toString());

if (cph == null) {
        System.out.println("Error happed in enc");
        System.exit(0);
}

cphBuf = SerializeUtils.bswabeCphSerialize(cph);

/* read file to encrypted */
plt = commonService.suckFile(inputfile);
aesBuf = aesCoderService.encrypt(m.toBytes(), plt);

commonService.writeCpabeFile(encfile, cphBuf, aesBuf);
```

```java
/* get BswabePub from pubfile */
pub_byte = commonService.suckFile(pubfile);
pub = SerializeUtils.unserializeBswabePub(pub_byte);

/* read ciphertext */
tmp = commonService.readCpabeFile(encfile);
aesBuf = tmp[0];
cphBuf = tmp[1];
cph = SerializeUtils.bswabeCphUnserialize(pub, cphBuf);

/* get BswabePrv form prvfile */
prv_byte = prvfile;
prv = SerializeUtils.unserializeBswabePrv(pub, prv_byte);

BswabeElementBoolean beb = Bswabe.dec(pub, prv, cph);


if (beb.b) {
        System.err.println("e = " + beb.e.toString());
        plt = aesCoderService.decrypt(beb.e.toBytes(), aesBuf);
        commonService.spitFile(decfile, plt);
} else {
        System.out.println("function(dec) error!");
        //System.exit(0);
}
```

System Test

Test cases and test results should be in the same table, but in order to have a better format, we make two separate tables, the first one is to describe the test scenarios, and the second one is for test results.

1. Test cases

Test Case ID	Relevant Requirement ID	Test Scenario	Notes
Ecpbm_ST_ Login_01	FR-01	After the administrator and registered user enter the correct username and password on the login page, the login is successful.	Registered user is created by administrator.
Ecpbm_ST_ Login_02	FR-01	The administrator and registered users entered mismatched username or password information on the login page, the login is failed.	
Ecpbm_ST_ Login_03	FR - 02	The user can reset the password by providing the correct email address and phone number.	
Ecpbm_ST_ Login_04	FR - 03	If the user log into the system using default password, the system will redirect to change password page.	
Ecpbm_ST_ Login_05	FR - 04	User can change password after log in.	
Ecpbm_ST_ Login_06	FR - 05	The text fields for password cannot copy and paste and hidden by dots.	
Ecpbm_ST_ Login_07	FR - 06	If the new password is too simple, the password will not be accepted.	

Ecpbm_ST_Login_08	FR - 07	If user forgot the password, the system will send an email with a temporary password.	This requirement is NOT marked as MUST.
Ecpbm_ST_Logout_09	FR - 08	User will automatically log out after 5 minutes.	This requirement is NOT marked as MUST.
Ecpbm_ST_User_01	FR - 09	Once the administrator has successfully logged in, the user list will be displayed on the left menu and ready for admin to modify.	Only the administrator can operate the user management page.
Ecpbm_ST_User_02	FR - 09	User management page, the administrator can click the "Add User" button to add a new user. After successful addition, the new user will be displayed on the user management page.	
Ecpbm_ST_User_03	FR - 09	The administrator can click the "delete" button to delete the added user. After the deletion is successful, the user management page no longer displays the deleted user information.	
Ecpbm_ST_User_04	FR - 10	On the add user page, the user information can be saved successfully only when the username and password are not empty. Otherwise, the new user cannot be created.	
Ecpbm_ST_User_05	FR - 11	On the user management page, the administrator has the right to update user information when clicking the “detail” button.	Only under the authority of the administrator user, can “add user” and “update user information”.
Ecpbm_ST_User_06	FR - 11	After the registered user logs in successfully, there should be no “user list” menu on the left button.	

Ecpbm_ST_User_07	FR - 12	On the user management page, the administrator has the right to change the category of a user when clicking the "detail" button.	This requirement is NOT marked as MUST.
Ecpbm_ST_User_08	FR - 13	On user management page, administrators can query specific registered users by searching their "username".	
Ecpbm_ST_File_01	FR - 14	On "my file" page, registered users can click the "upload file" button to specify users of a specific "college" and "subject" to download the uploaded file.	Regardless of whether you are an administrator or a registered user, both have the authority to operate "All files" page and "my file" page.
Ecpbm_ST_File_02	FR - 15	On "my file" page, registered user has the right to delete the uploaded file via click "delete" button.	
Ecpbm_ST_File_03	FR - 15	On "All files" page, the administrator can click "delete" button to delete the specified files uploaded by registered user.	
Ecpbm_ST_File_04	FR – 16	The uploader can see how many times the file has been downloaded.	This requirement is NOT marked as MUST.
Ecpbm_ST_File_05	FR - 17	If the user's attributes do not satisfy the policy, the download button should be grey.	
Ecpbm_ST_File_06	FR – 18	On "All files" page, those users who are not authorized to decrypt the corresponding file will jump to the "You don't have the authority to download this file" prompt page when downloading.	
Ecpbm_ST_File_07	FR – 18	On "All files" page, those users who have permission to decrypt the corresponding file will download successfully when downloading.	

Ecpbm_ST_File_08	FR - 19	On “All files” page, the user can search for the corresponding file after entering the “title” in the “file title” input box.	
Ecpbm_ST_File_09	FR - 19	On “my files” page, the user can search for the corresponding uploaded file after entering the “title” in the “file title” input box.	
Ecpbm_ST_Data_01	FR – 19	In the database table user_Info, “userType” is divided into 1 and 2. 1 representatives administrator, 2 representatives registered user.	
Ecpbm_ST_Data_02	FR - 20	On the user management page, once a user with the same name is created, it cannot be saved. In addition, the id is unique in the database table user_info because ID is automatically generated.	Most test scenario occur in the database table user_info.
Ecpbm_ST_Data_03	FR - 21	The email address must contain ‘@’.	
Ecpbm_ST_Data_04	FR - 22	On the user management page, user type can drop down to select “administrator”, “Lecturer”,”Tutor” and “Student”.	
Ecpbm_ST_Data_05	FR - 23	On the user management page, college can drop down to select different college, such as Business, Nursing and so on.	
Ecpbm_ST_Data_06	FR - 24	On the user management page, when a different college is selected, the corresponding subject is displayed in the subject drop-down box.	
Ecpbm_ST_Data_07	FR - 25	In the database table user_Info, each user has id, username, password, usertype, email and phone.	

2. Test results

Test Case ID	Test Data	Expected Result	Test Result (screenshot is provided if applicable)
Ecpbm_ ST_Logi n_01	Username: admin Password:1	Administrator login in successfully.	Pass. Figure 2.1 shows the test result of administrator login.
Ecpbm_ ST_Logi n_01	Username: hz546 Password:1	Registered user login in successfully.	Pass. Figure 2.2 shows the test result of registered user login.
Ecpbm_ ST_Logi n_02	Username: admin Password:123	Administrator login in failed.	Pass. Figure 2.3 shows the test result of administrator login failed.
Ecpbm_ ST_Logi n_02	Username: hz546 Password: 123	Registered user login failed.	Pass. Figure 2.4 shows the test result of registered user login failed.
Ecpbm_ ST_Logi n_02	Username: Hz Password: 123	Unregistered user login failed.	Pass. Figure 2.5 shows the test result of unregistered user login failed.
Ecpbm_ ST_Logi n_02	Username: Password:	Prompt this field must not be blank.	Pass. Figure 2.6 shows the test result of prompt.
Ecpbm_ ST_Logi n_03	Username: mm Email:	User can reset the password to default.	Fail. This function is not implemented yet.
Ecpbm_ ST_Logi n_04	Username: Hz Password: 123	After login, the system will redirect to change password page.	Fail. This function is not implemented yet.

Ecpbm_ST_Login_05	Username: hz546 Password:1 New password:123	The password can change successfully.	Pass.
Ecpbm_ST_Login_06	Username: hz546 Password:123	The password cannot be copied or pasted and hidden by dots.	Partially Pass. The password field is hidden by dots. Figure 2.1 shows the result.
Ecpbm_ST_Login_07	Username: hz546 Password:123	The password is too simple, the system is not allowed.	Fail. This function is not implemented yet.
Ecpbm_ST_Login_08	Username: hz546 Password:123	If the user forget password, the system can send email with temporary password.	Fail. This function is not implemented yet.
Ecpbm_ST_Login_09	Username: hz546 Password:123	User should log out after 5 minutes.	Fail. This function is not implemented yet.
Ecpbm_ST_User_01	Username: admin Password:1	There is “user list” in the left “menu” bar	Pass. Figure 2.1 shows the test result of left menu bar after administrator login successfully.
Ecpbm_ST_User_02	Username: hz546 User Password:1 User Type: Lecturer Email: hx546@uowmail.edu.au; Phone: 0424469846 College: CSIT Subject: CSCI992	User successfully added and the new user displays on the user management page.	Pass. Figure 2.7 shows the test result of add user window. Figure 2.8 shows the test result of the new user displayed on the user management page.

Ecpbm_ ST_User _03	Username: hz546 User Password:1 User Type: Lecturer Email: hx546@uowmail. edu.au; Phone: 0424469846 College: CSIT Subject: CSCI992	Registered user “hz546” delete prompt “user information deleted successfully” and no longer displayed in the user management interface	Pass. Figure 2.9 shows the prompt when deleting user. Figure 2.10 shows the test result of deleted interface.
Ecpbm_ ST_User _04	Username: Password:	When the mouse moves to the username and password input boxes, it prompts "this filed must not be blank"	Pass. Figure 2.11 shows the test result of prompt when the username and password are empty.
Ecpbm_ ST_User _05	Username: hz546 User Password:1 User Type: Lecturer Email: hx546@uowmail. edu.au; Phone: 0424469846 College: CSIT Subject: CSCI851	Registered user "hz546" successfully updated user information, and prompted "user information updating successfully"	Pass. Figure 2.12 shows the window of update user information. Figure 2.13 shows the user management interface after updating user information successfully.
Ecpbm_ ST_User _06	Username: hz546 Password: 1	There is no “user list” in the left “menu” bar	Pass. Figure 2.2 shows the test result of left menu bar after registered user “hz546” login.

Ecpbm_ ST_User _07	Username: hz546 User Type: Lecturer New User Type: Student	After admin log in, the category of hz546 should change to student from lecturer.	Pass. Similar to change user attributes, as shown in Figure 2.12.
Ecpbm_ ST_User _08	Username: hz546	After the search, the user management page displays the user information with the username “hz546”.	Pass. Figure 2.14 shows the test result of its username “hz546”.
Ecpbm_ ST_File_ 01	Title: CSCI992 Description: CSCI992 meeting document College: CSIT Subject: CSCI992 File path: C:\Users\garyz\Desktop\ 992 - meeting.docx	The file is uploaded successfully, and the uploaded file is displayed in “My Files” page and “All Files” page.	Pass. Figure 2.15 shows the interface for adding files; Figure 2.16 shows my files and all files interface after adding files.
Ecpbm_ ST_File_ 02	Title: CSCI992	The file named CSCI992 was deleted successfully, and there is no record of the file named CSCI992 on my file page.	Pass. Figure 2.17 displays the pop-up box when deleting files and the page display after deleting files.
Ecpbm_ ST_File_ 03	Title: CSCI992	The file named CSCI992 was deleted successfully, and there is no record of the file named CSCI992 on all file page.	Pass. Figure 2.18 shows the “All files” page after deleting the file named “CSCI992”
Ecpbm_ ST_File_ 04	Username: mm736 Password: 1	User should able to see how many times the file she uploaded has been downloaded.	Fail. This function is not implemented yet.

Ecpbm_ ST_File_ 05	Username: mm736 Password: 1	The download buttons for files which user cannot access to should be in grey.	Fail. This function is not provided since we need to see the result if the user cannot decrypt the file.
Ecpbm_ ST_File_ 06	Username: mm736 Password: 1 College: CSIT Subject: CSCI851	The file named CSCI992 failed to download and jumped to the massage page prompt “"success": "false”, “message”: “You don't have the authority to download this file!"	Pass. The page prompt is {"success":"false","message":"You don't have the authority to download this file!"}
Ecpbm_ ST_File_ 07	Username: mm736 Password: 1 College: CSIT Subject: CSCI992	The classification attribute of the username “mm” is consistent with the file attribute classification of CSCI992, and the file is downloaded successfully.	Pass. Figure 2.19 shows the test result of “All files” page after “mm” downloading file named “CSCI992” successfully
Ecpbm_ ST_File_ 08	Username: admin Password: 1 File Title: CSCI992	After the search, the results displayed on the “all files” page are consistent with the input text.	Pass. Figure 2.20 shows the test result of “All files” page after “admin” searching file named “CSCI992”
Ecpbm_ ST_File_ 09	Username: zh Password: 1 File title: CSCI992	After the search, the results displayed on the “my files” page are consistent with the input text.	Pass. Figure 2.21 shows the test result of “my files” page after “hz546” searching file named “CSCI992”
Ecpbm_ ST_Data _01	Database username: root Password: 123456	The user_type in the database table user_info is divided into 1 and 2,3,4 representing admin, lecturer,	Pass. Figure 2.22 shows the the composition of the database table user_Info.

		tutor and student respectively.	
Ecpbm_ ST_Data _02	Database username: root Password: 123456	The user_id in the database table user_Info is unique.	Pass. Figure 2.22 shows the user_id in database table user_info.
Ecpbm_ ST_Data _03	Username: admin Password: 1 Email: 12345	The email address is invalid and should not be accepted.	Fail. This function is not implemented yet. Because we do not provide sending email function.
Ecpbm_ ST_Data _04	Username: admin Password: 1	The user_type drop-down has four types: administrator, lecturer, tutor and student.	Pass. Figure 2.23 shows four user types drop-down – administrator, lecturer, tutor and student.
Ecpbm_ ST_Data _05	Username: admin Password: 1	The college drop-down has different types, such as business, nursing and so on.	Pass. Figure 2.24 shows two categories drop-down – Business, nursing, CSIT, Education and so on.
Ecpbm_ ST_Data _06	Username: admin Password: 1	When a different “college” is selected, the corresponding subject is displayed in the “subject” drop-down box.	Pass. Figure 2.25 shows the corresponding subject when the corresponding college is selected.
Ecpbm_ ST_Data _07	Database username: root Password: 123456	The overall composition of user information is shown in the database table user_info.	Pass. Figure 2.22 shows the composition of user information in the database table user_info.

Figure 2.1 Administrator: admin

Figure 2.2 Registered user: hz546

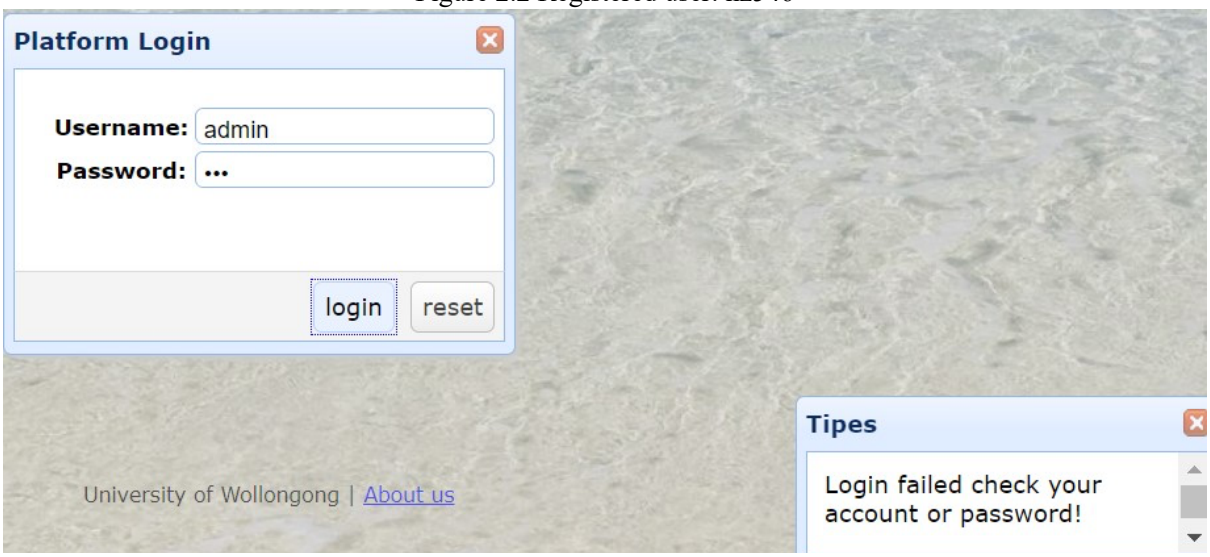


Figure 2.3 Administrator – admin/123

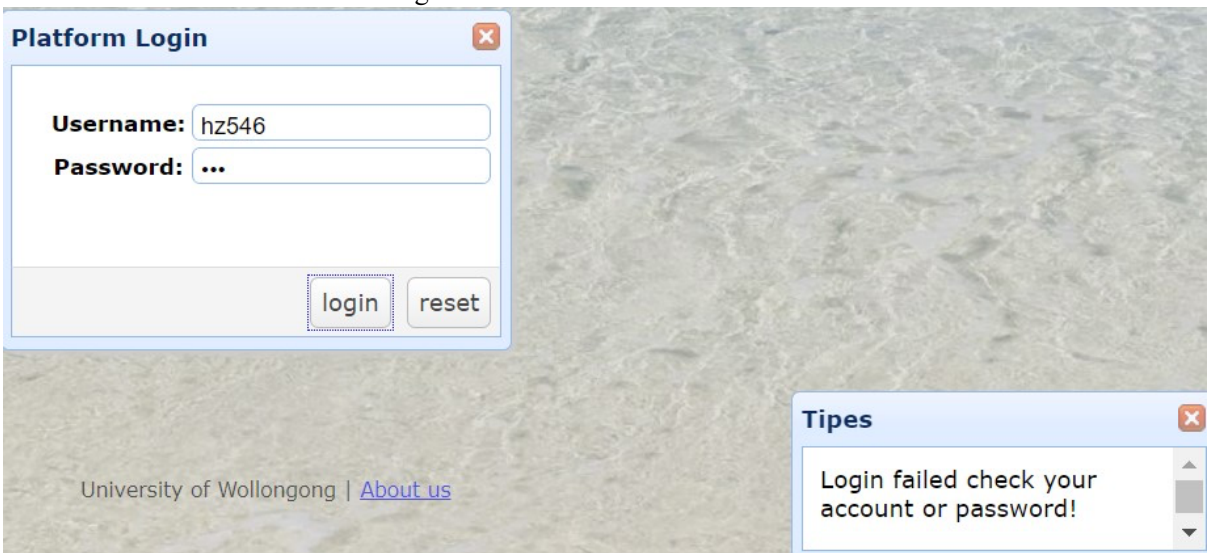


Figure 2.4 Registered user – hz546/123

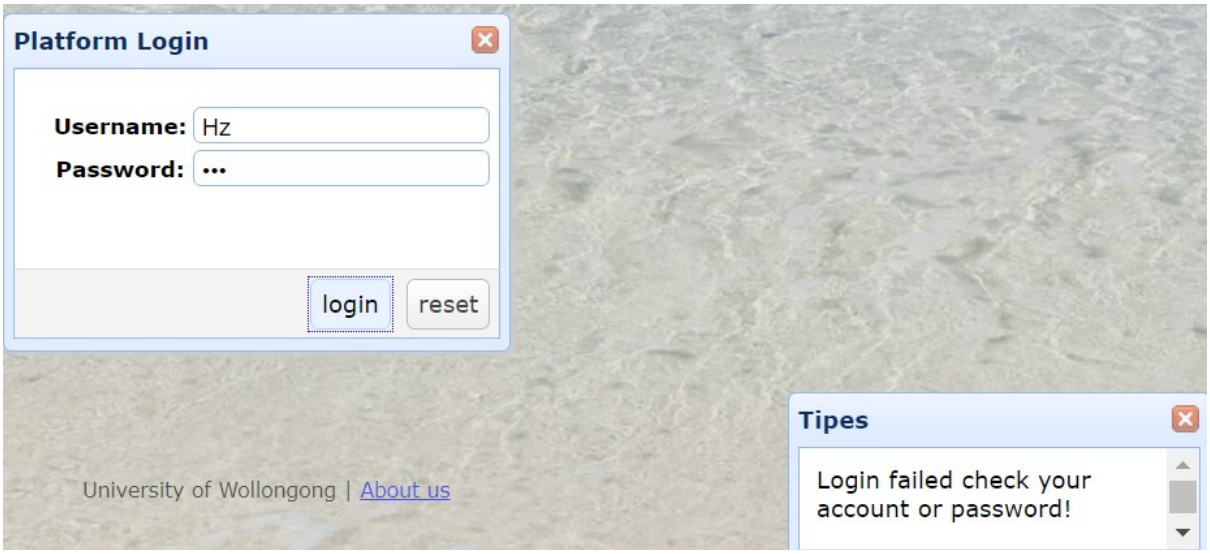


Figure 2.5 Unregistered user login – Hz/123

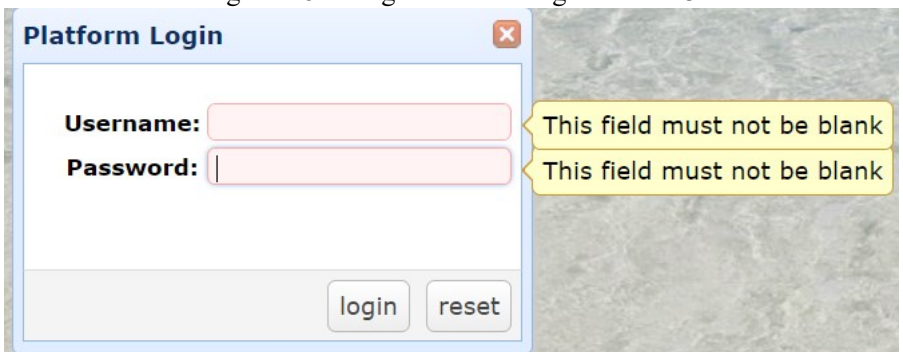


Figure 2.6 Prompt – This field must not be blank.

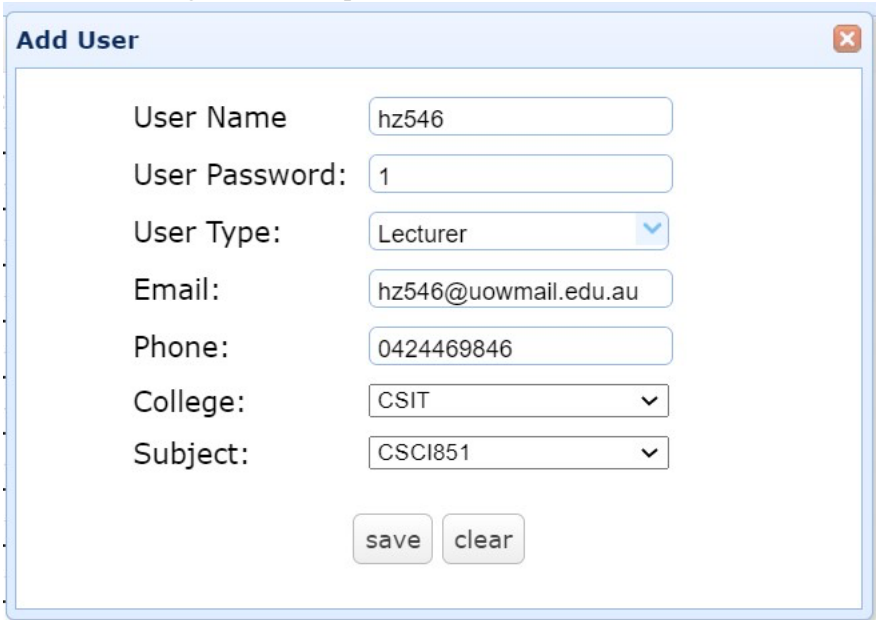


Figure 2.7 Add user window

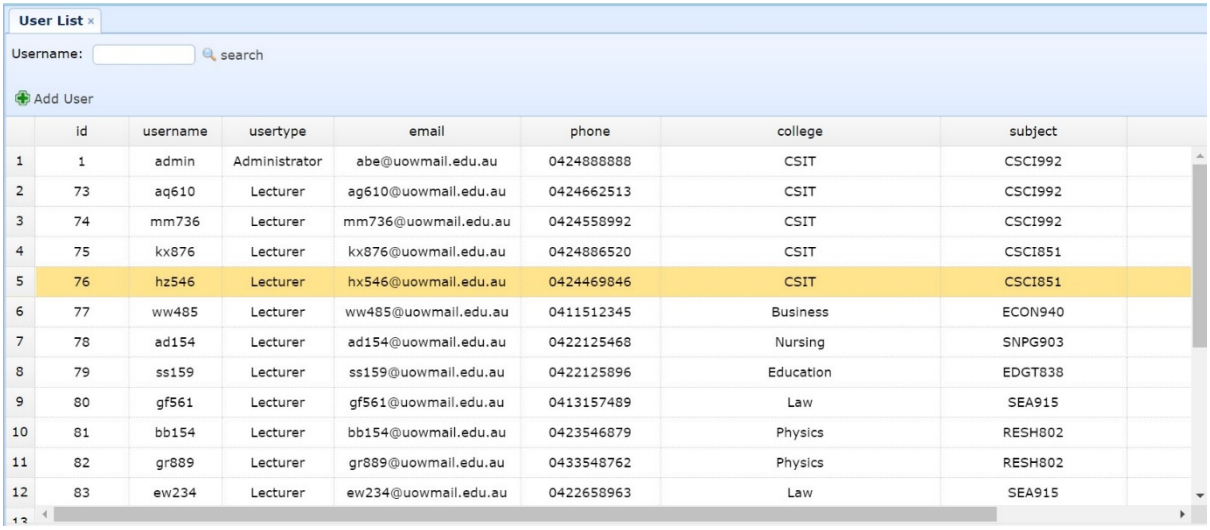


Figure 2.8 New user displayed on the user management page

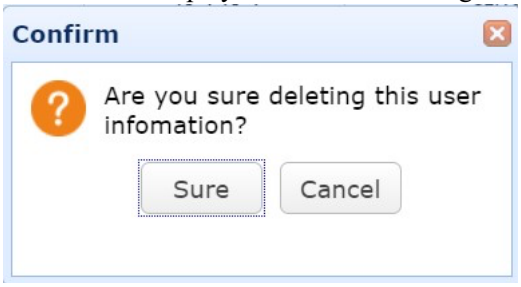


Figure 2.9 Delete user – prompt

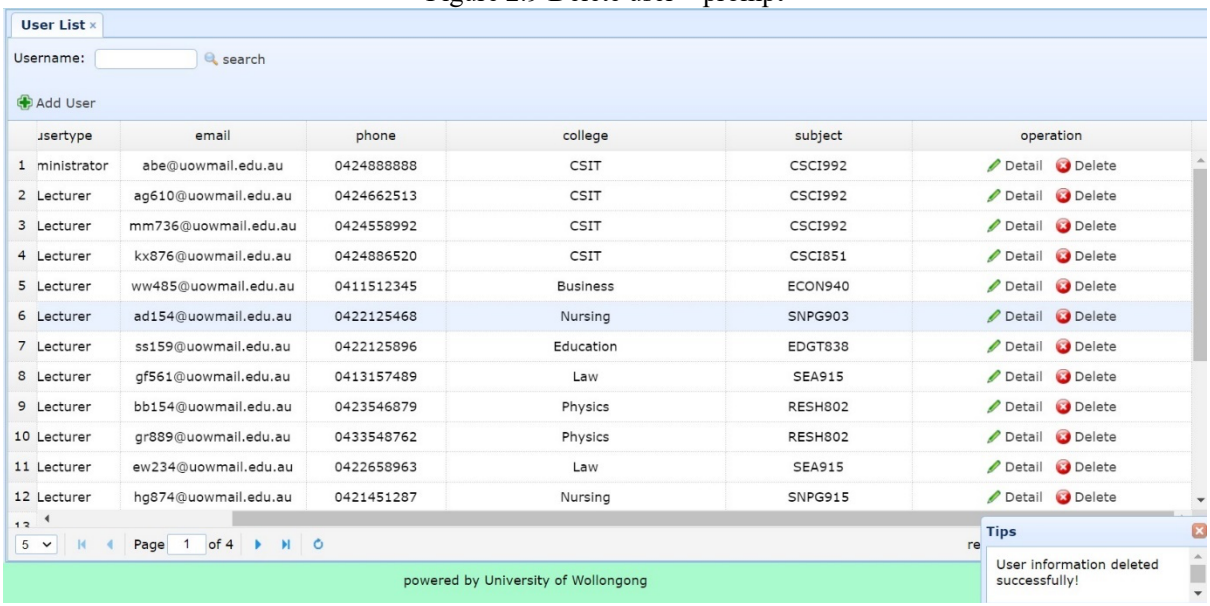


Figure 2.10 User management interface after deletion

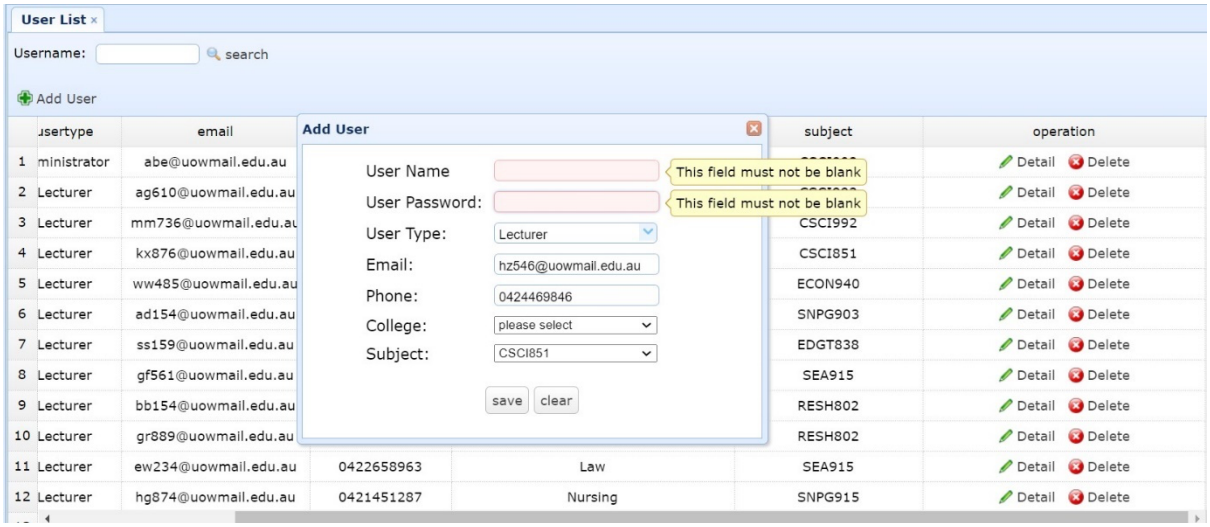


Figure 2.11 Prompt – Both username and password are empty

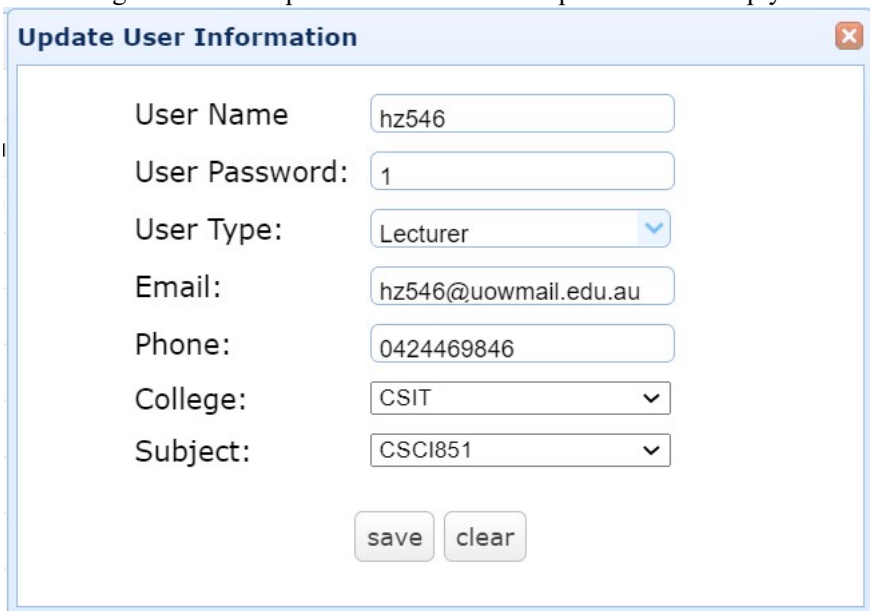


Figure 2.12 – Update user information window

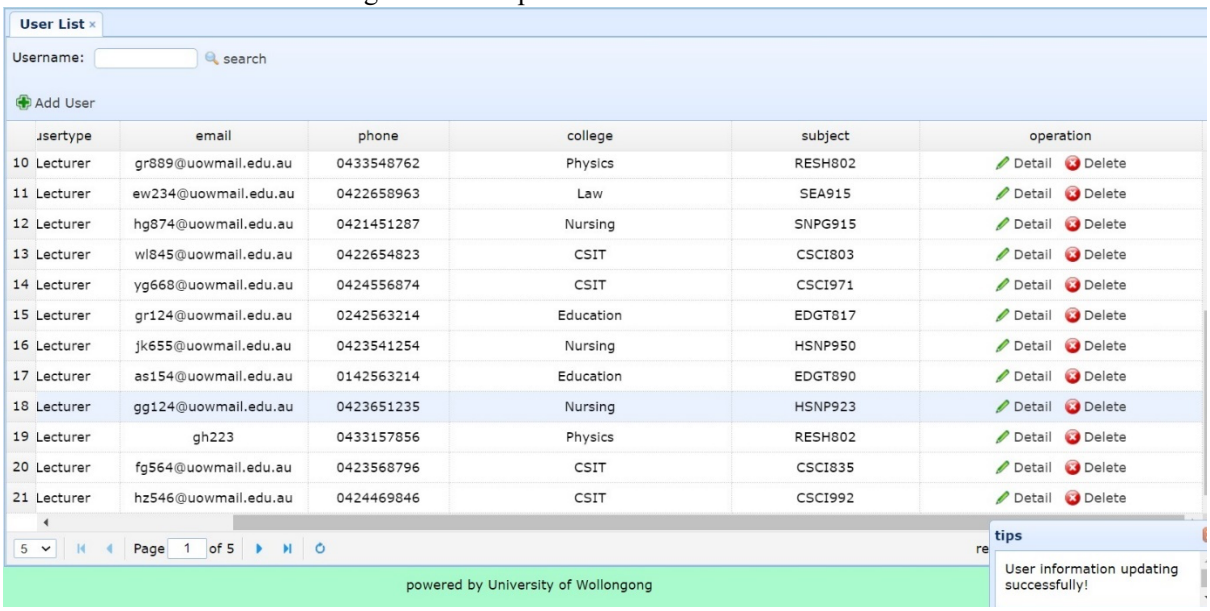


Figure 2.13 User list after update user information successfully

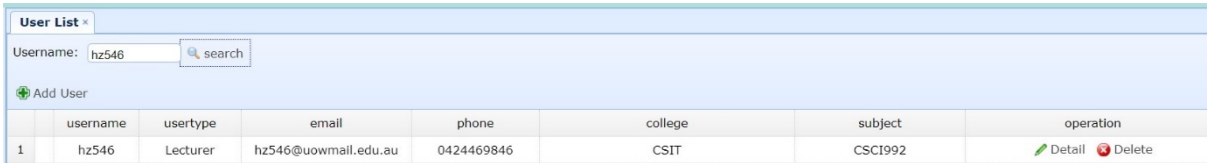


Figure 2.14 The test results page with the username “hz546”

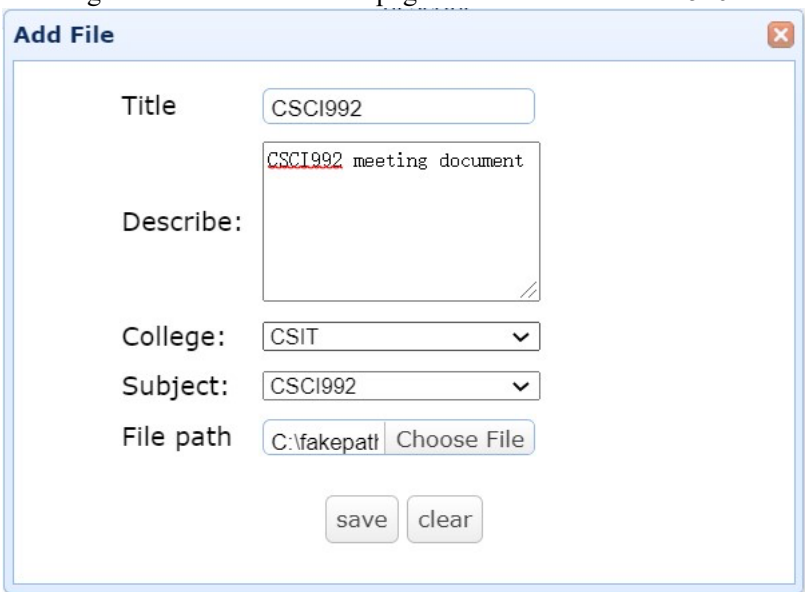


Figure 2.15 Add File interface

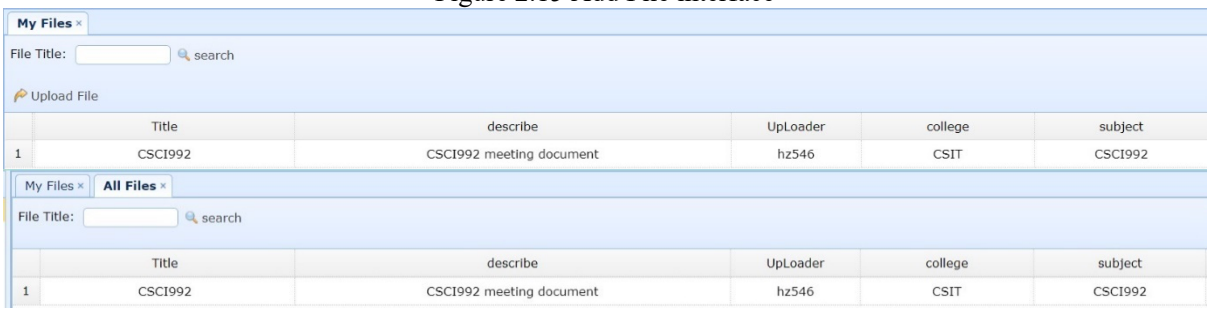
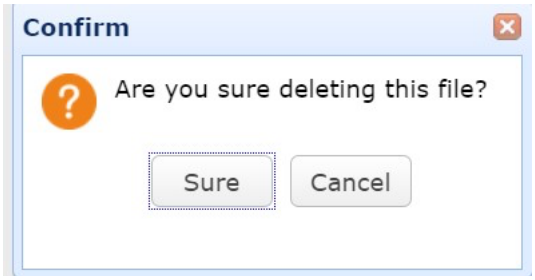


Figure 2.16 My files page and All files page



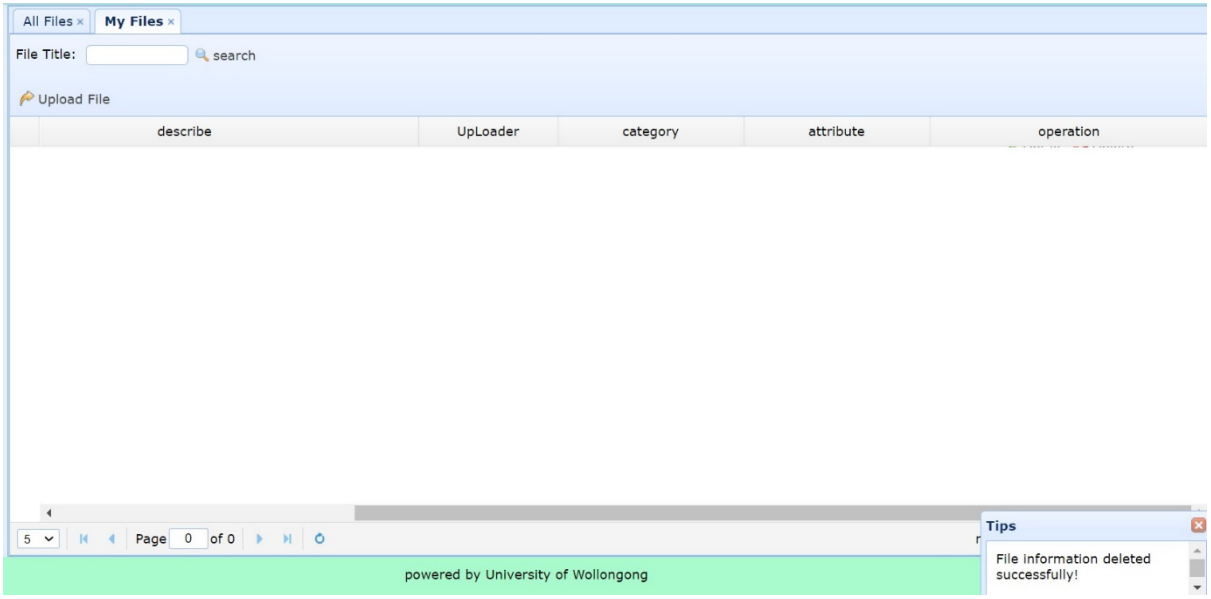


Figure 2.17 Delete the pop-up box and the deleted page

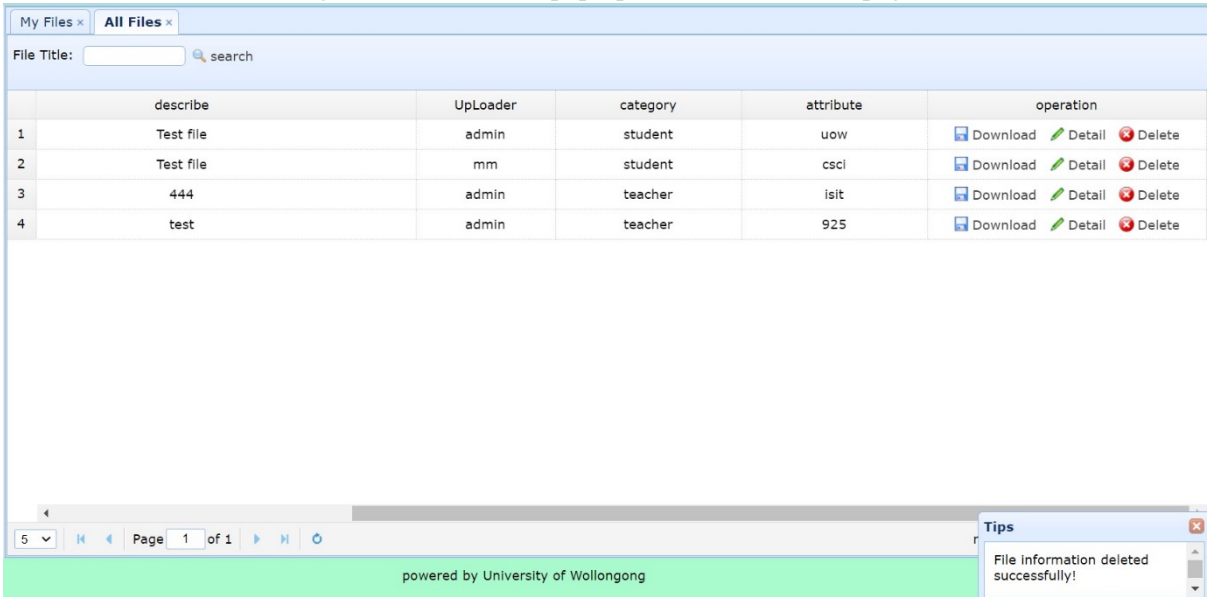


Figure 2.18 Delete the file named “CSCI992”

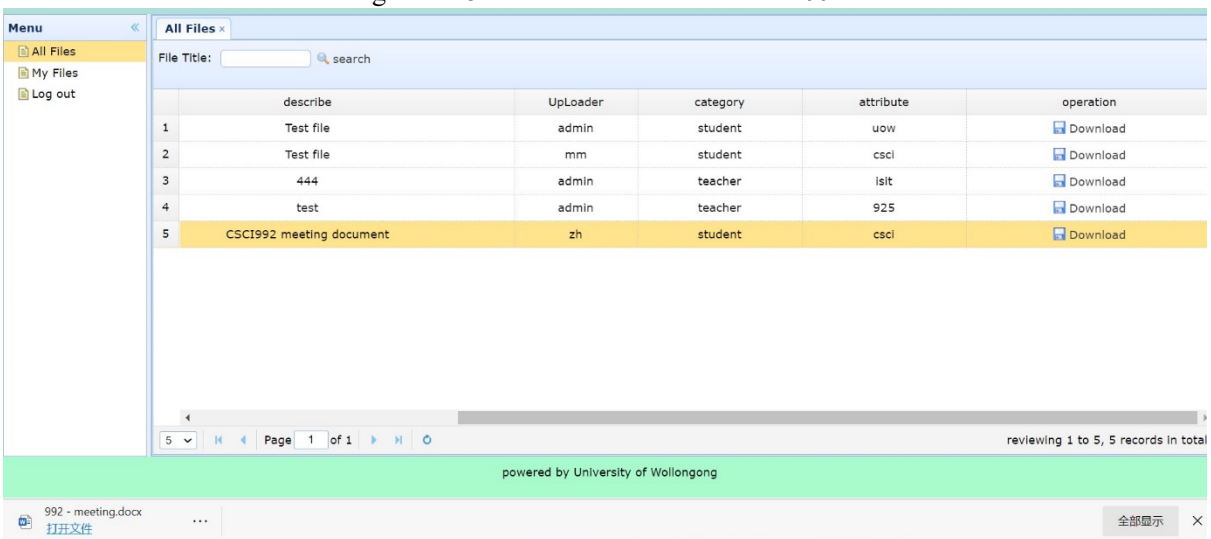


Figure 2.19 “mm736” user downloading successfully

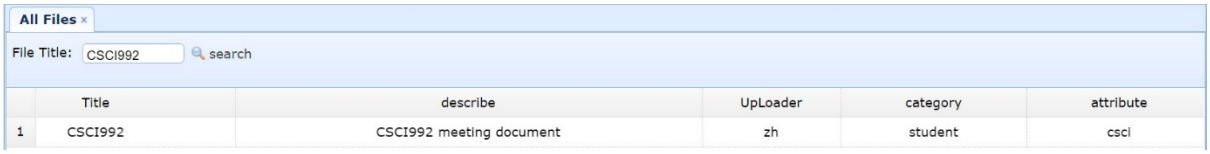


Figure 2.20 “All files” page searching result

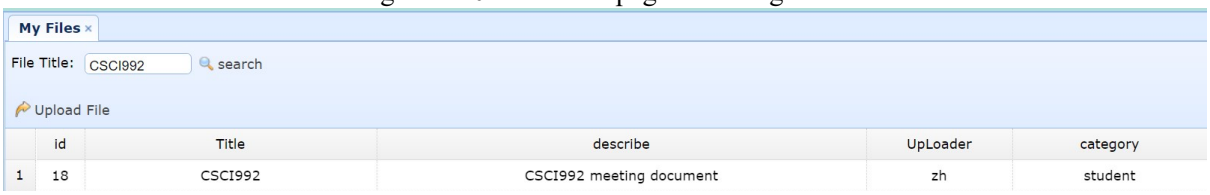


Figure 2.21 “my files” page searching result

id	userName	password	userType	email	phone
1	admin	1	1	abe@uowmail.edu.au	0424888888
73	aq610	1	2	aq610@uowmail.edu.au	0424662513
74	mm736	1	2	mm736@uowmail.edu.au	0424558992
75	kx876	1	2	kx876@uowmail.edu.au	0424886520
77	ww485	1	2	ww485@uowmail.edu.au	0411512345
78	ad154	1	2	ad154@uowmail.edu.au	0422125468
79	ss159	1	2	ss159@uowmail.edu.au	0422125896
80	gf561	1	2	gf561@uowmail.edu.au	0413157489
81	bb154	1	2	bb154@uowmail.edu.au	0423546879

Figure 2.22 Database table user-info

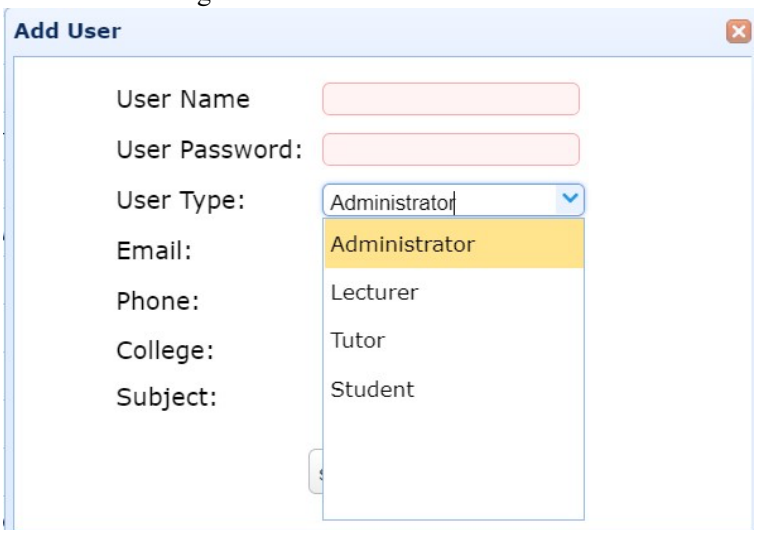


Figure 2.23 User-type

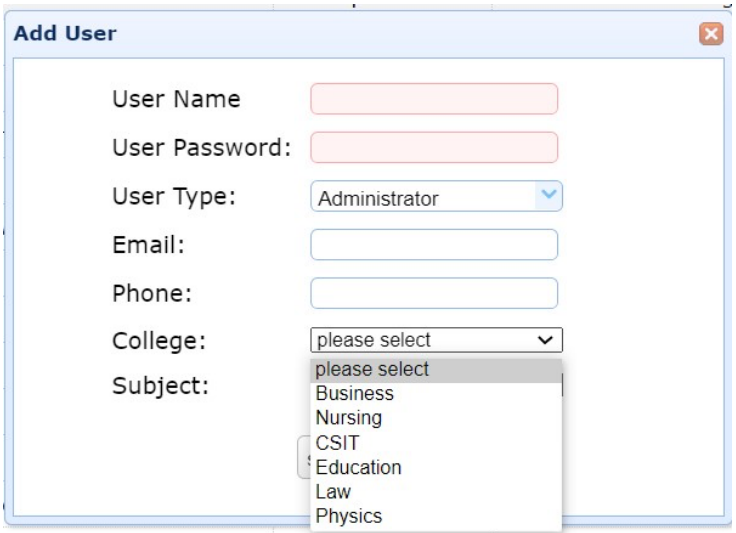


Figure 2.24 College

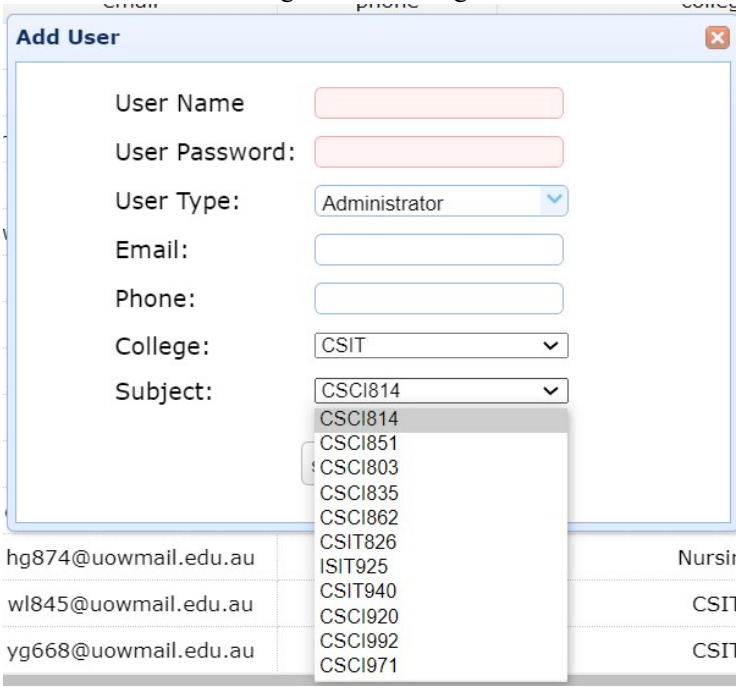


Figure 2.25 CSIT – subject

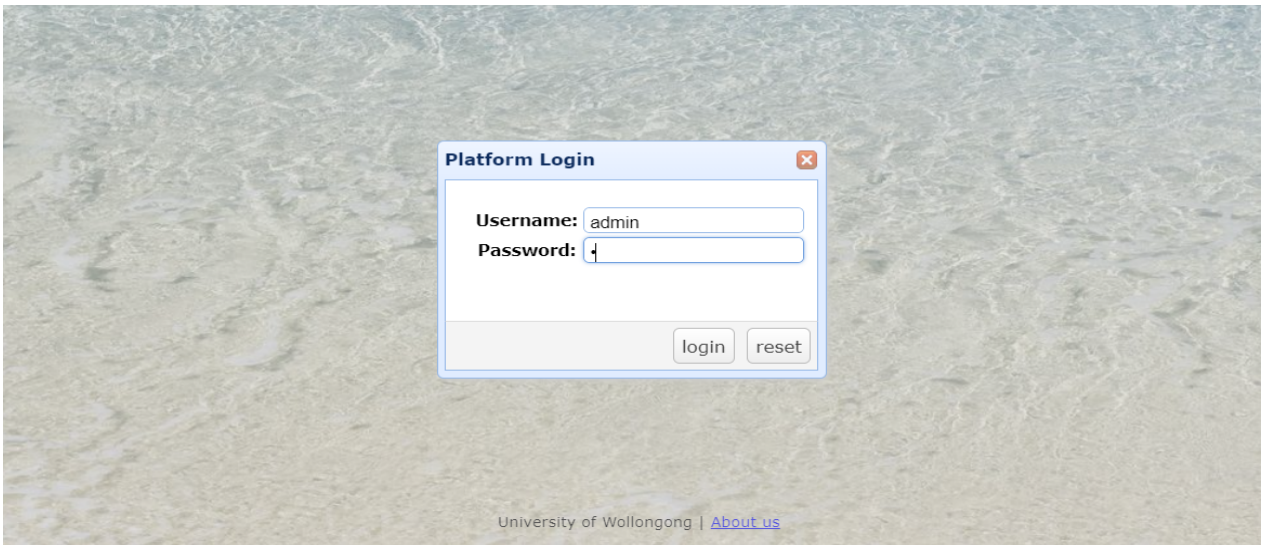
Project Timeline

Our project has divided two semesters to complete. For the first semester, the preliminary requirement has been submitted on 31st, May. The core task of our second semester is to design the whole requirement and then implement requirement through Java language. In our project, there are ten tasks to be completed. At the end of the first semester, the first five tasks have been complete. For the second semester, we have finished task 6 and the rest tasks is planned to be finished before 1st November. In our project timeline table, subtasks are derived from the subheading of each task. At the beginning of November, we have completed the first nine tasks. This means that the final tradeshow demo will be held on the 12th. The following figures is to show timeline and task assignment.

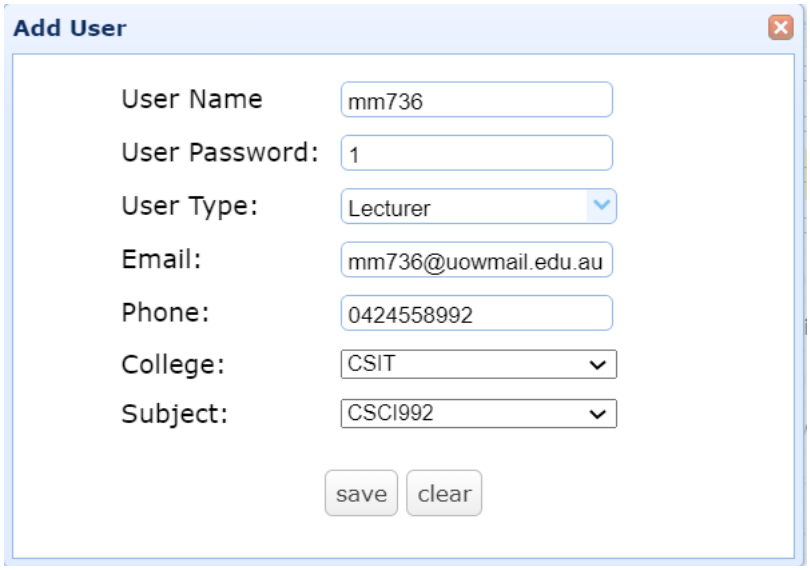
Task name	Stard date	End date	Assigned	Duration	Progress
	2020/3/18	2020/10/30	Group E	226	95%
Task 1 Code of Conduct	2020/3/18	2020/4/5	Group E	18	100%
Task 2 Project Proposal	2020/4/15	2020/5/3	Group E	18	100%
Task 3 Progress Presentation	2020/5/6	2020/5/17	Group E	11	100%
Task 4 Preliminary Requirement	2020/5/21	2020/5/31	Group E	10	100%
Task 5 Teamwork Document	2020/6/3	2020/6/12	Group E	9	100%
Task 6 Requirement(Complete)	2020/8/11	2020/8/23	Group E	32	100%
Subtask 6.1 User responsibilities	2020/8/11	2020/8/23	Heng Zhang	12	100%
Subtask 6.2 Data Flow	2020/8/11	2020/8/23	Anqi Gao	12	100%
Subtask 6.3 State UML	2020/8/11	2020/8/23	Miao Miao	12	100%
Subtask 6.4 Implement User Interface	2020/8/11	2020/8/23	Ke Xian	12	100%
Task 7 Progress Presentation 2	2020/8/24	2020/9/25	Group E	32	100%
Subtask 7.1 Overview Progress	2020/8/24	2020/9/25	Anqi Gao	32	100%
Subtask 7.2 UI Design	2020/8/24	2020/9/25	Heng Zhang	32	100%
Subtask 7.3 Realized Function	2020/8/24	2020/9/25	Ke Xian	32	100%
Subtask 7.4 Funture Work	2020/8/24	2020/9/25	Miao Miao	32	100%
Task 8 Teamwork Documentation 2	2020/9/26	2020/10/11	Group E	15	100%
Subtask 8.1 Key Areas	2020/9/26	2020/10/11	Anqi Gao	15	100%
Subtask 8.2 Key Responsibilities	2020/9/26	2020/10/11	Ke Xian	15	100%
Subtask 8.3 Meeting Minutes	2020/9/26	2020/10/11	Miao Miao	15	100%
Subtask 8.4 Project Timelines	2020/9/26	2020/10/11	Heng Zhang	15	100%
Task 9 Final Report Documentation	2020/10/12	2020/11/1	Group E	20	100%
Subtask 9.1 System Design	2020/10/12	2020/11/1	Anqi Gao	20	100%
Subtask 9.2 Front-End Implementation	2020/10/12	2020/11/1	Ke Xian	20	100%
Subtask 9.3 Back-End Implementation	2020/10/12	2020/11/1	Miao Miao	20	100%
Subtask 9.4 System Test	2020/10/12	2020/11/1	Heng Zhang	20	100%
Task 10 Tradeshow Demo	2020/11/2	2020/11/12	Group E	10	0%

Case Study

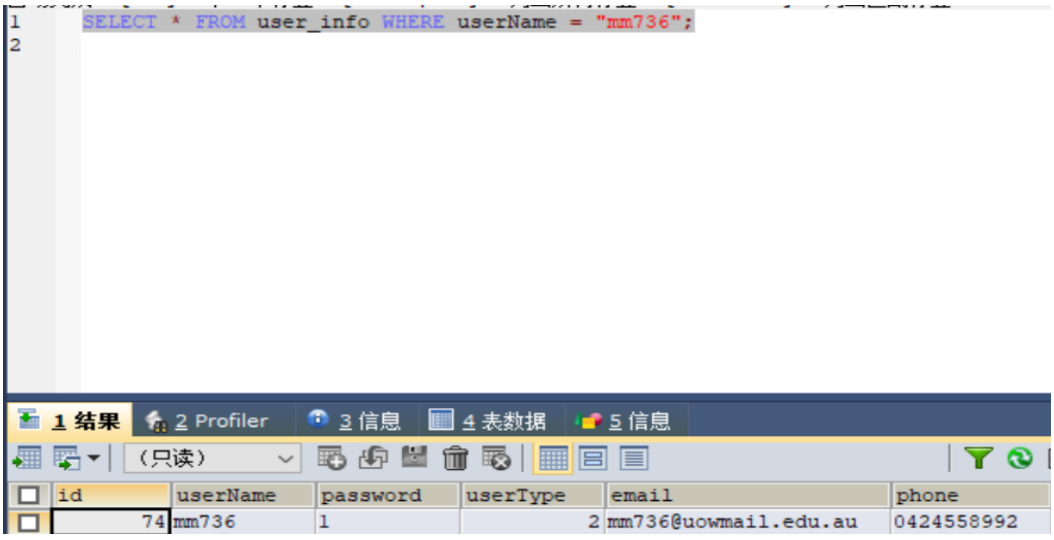
For the whole project, we have four different kinds of user type. The first one is administrator, the main responsibility for the administrator is to do user management and file management. The main responsibility of other users which refer to Lecturer, Tutor and student, is to upload files and download corresponding uploaded files according to different file attributes. The first step for attribute file system is to create new user based on the authorities of administrator. As far as we know, the username and password for administrator is admin and 1 respectively. The first step is to login the system. username: admin; password: 1



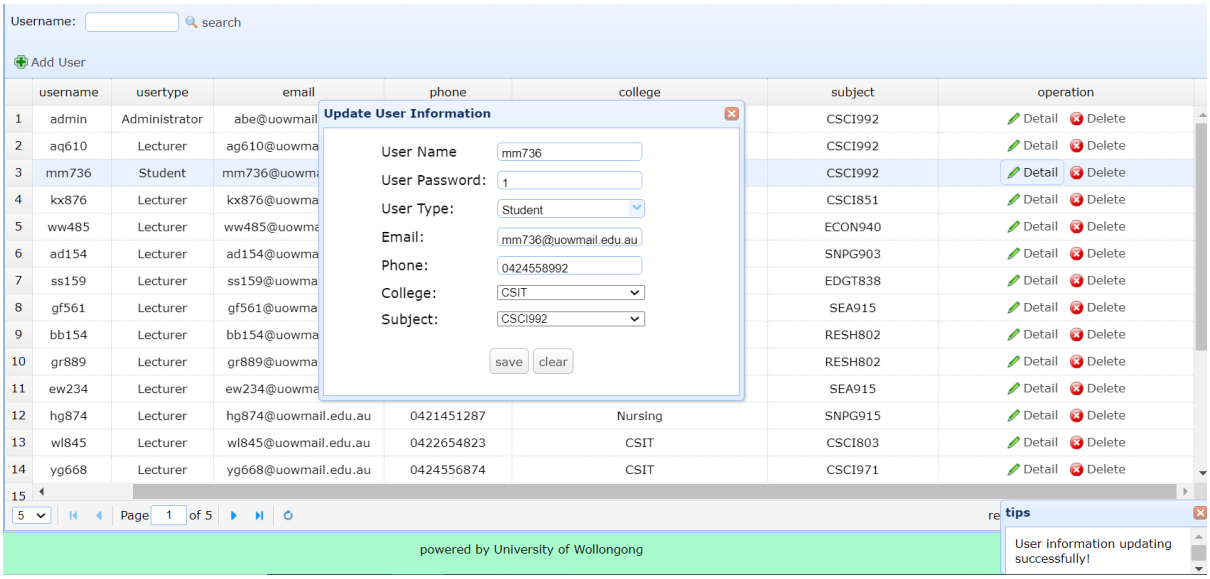
Until we successfully log in to the system, enter the User_list page. And the click “Add user”, The “add user” pop-up window is displayed as follows, and we fill in the information as shown in the figure below.



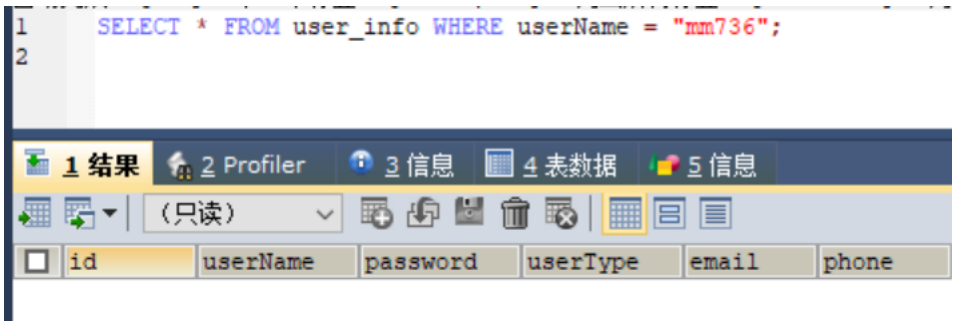
After clicking save, the user information mm736 information is stored in the database.



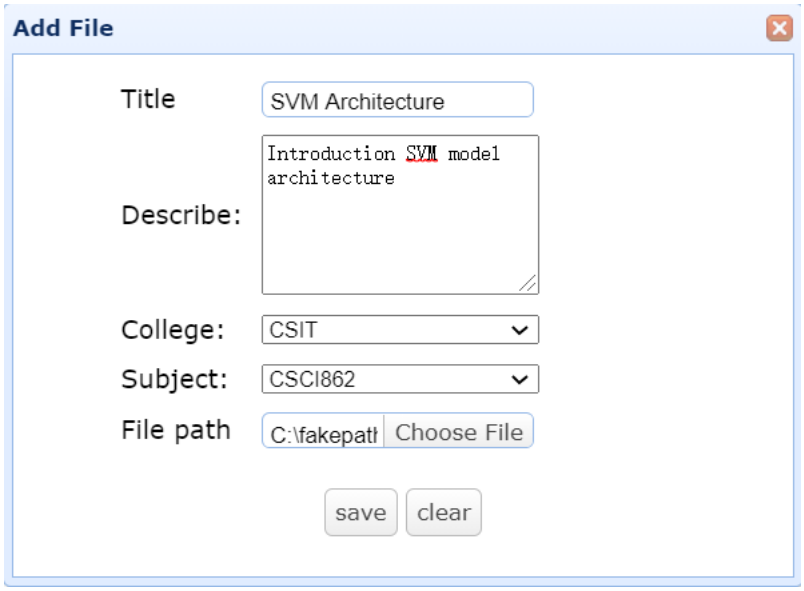
Once the username "mm736" is successfully created, the administrator can delete and modify user information.

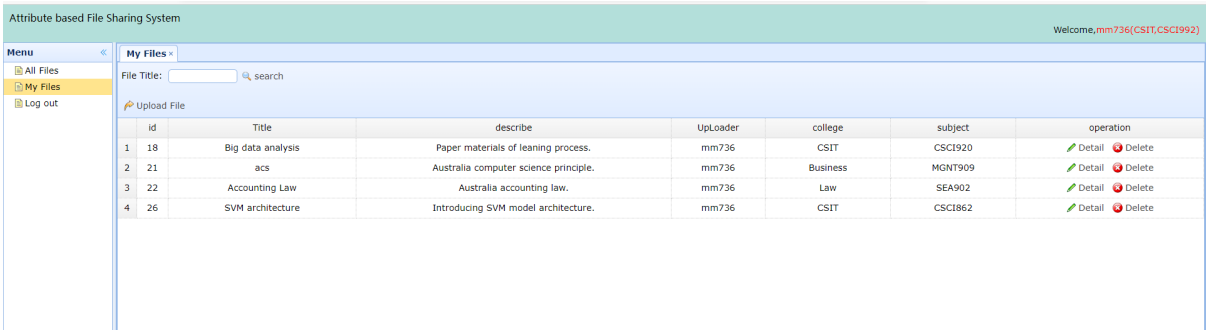


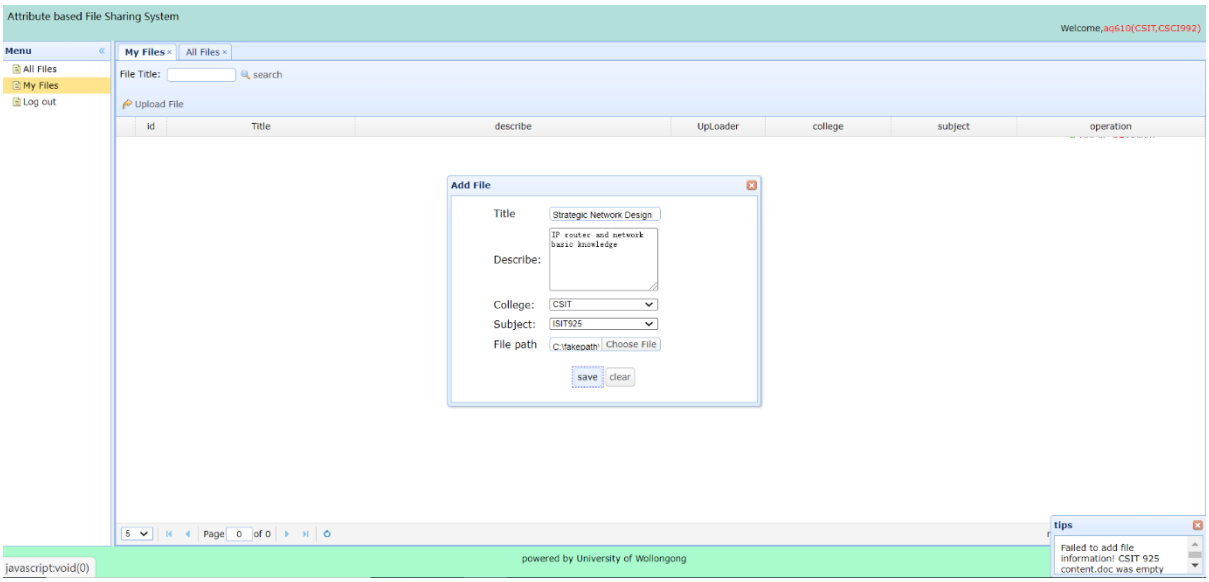
When you click on “Detail”, a window for updating user information will pop up. Once you have updated the user information, after saving successfully, the user information is updated. The user type in the user information is changed from “lecturer” to “student”. When you want to delete the user, click “delete” to delete the user. The deleted users will not be displayed in the database.



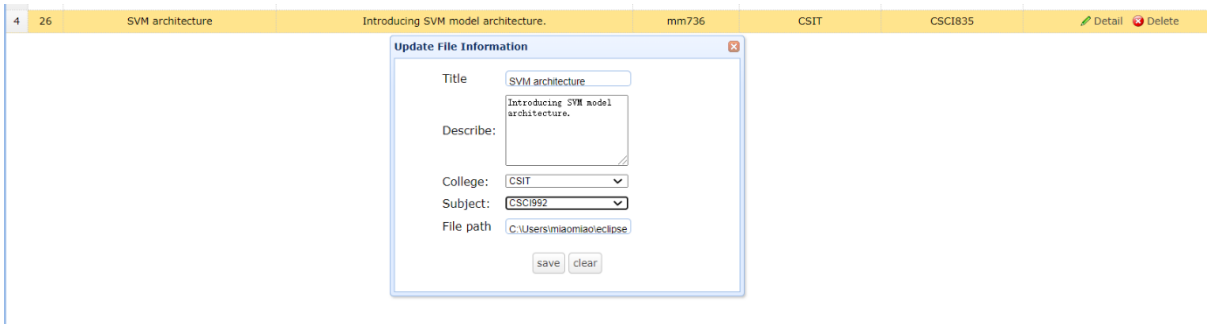
Created users can log in to the file sharing system to share files. when Upload files in “My Files”, you can update uploaded file information and delete files under “My Files”. In addition, When the uploaded file is empty, the page prompts that adding failed and the file is empty.



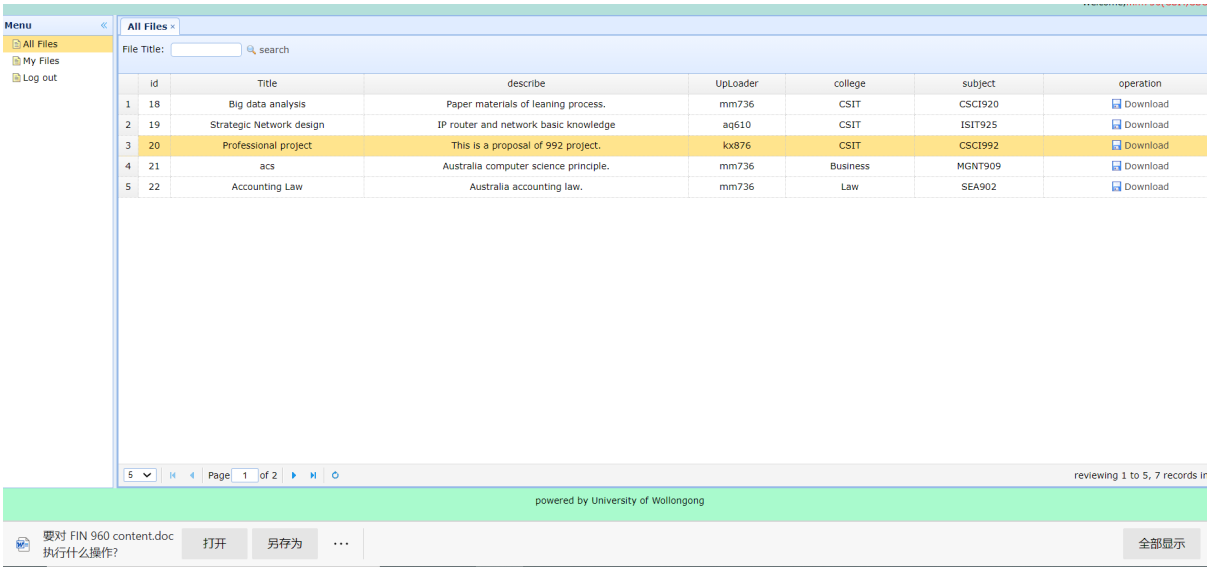




After uploading the file, User “mm736” can modify and delete uploaded files. Besides the uploaded file is displayed on the “All files” page.



On the "All Files" page, users of the same college and subject can download the files of the same college subject. This means that the subject and subject are also CSIT, and users of CSCI992 can decrypt and download files uploaded by "mm736".



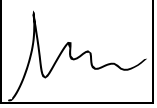
Otherwise, users cannot decrypt files that are not the same college and subject. The page will prompt that the download failed, and there is no permission to download the file.

Teamwork Document

The whole project is done by four of us. For assignment 1 to 7, we assign tasks to different people, every finished their own work. For assignment 8, Ke Xian and Miao Miao are busy with coding, the paperwork is done by Anqi Gao and Heng Zhang. The project management is done by Anqi Gao as the team leader. The system design is done by the whole team. We had several meetings to discuss the design and the database design is recorded by Anqi Gao, the UI design is recorded by Miao Miao. The implementation of database is done by Anqi Gao, but Ke Xian modify the database while implementing the system.

As for coding, the work is not divided equally. This is based on the coding ability of each team member, not due to someone has low participant. In order to save time, we assign the coding tasks to those people who are confident to finish the work on time. The front-end and back-end for user and file management is done by Ke Xian. The file upload and download functions are implemented by Miao Miao. The CP-ABE part is implemented and tested separately by Anqi Gao, the integration is done by Ke Xian and Miao Miao. The system test is done by Heng Zhang. Every time we had new features, Heng tests the whole system and gives us feedback. Additionally, Heng helps others to do the tasks when the other team member has no time to finish it. Those tasks are usually simple and small but annoying and time consuming. Those tasks are not listed in the table, but it does counts.

The total contribution is summarized in the table below:

Name	Contribution for Paperwork	Contribution for implementation	Signature
Anqi Gao	Assignment 1-8: 25% Final report: 30%	Database, CP-ABE	
Heng Zhang	Assignment 1-7: 25% Assignment 8: 75% Final report: 40%	System test	
Ke Xian	Assignment 1-7: 25% Final report: 15%	Database, Front-end, Back-end for user/file management	
Miao Miao	Assignment 1-7: 25% Final report: 15%	UI Design, Back-end for file upload and download	

Conclusion

The whole project after two semesters has finished successfully. In this project, we started from writing code of conduct, then step by step until we finished the coding and the final report. In the early stage, we don't know each other, till now we can cooperate together, learn the concept of cypher security etc., we have gained a lot. This project is a challenge for us, because of the COVID-19, we cannot communicate face to face as usual, moreover, we need to consider the time zone effect. This makes the discussion within the group tied up. Fortunately, each of our team members contributed to the project. Anqi Gao, as the team leader, in the process of communicating with the supervisor, she played a role as a communication bridge. Miao Miao is good at expressing her opinions in each group meeting, which provides direction for the development of the project. Ke Xian is a leader in technology and plays an important role in the development of the entire project. During the entire project process, Heng Zhang reduced the probability of project errors and enabled the entire project to be completed on schedule. So, in the process of completing the project, we did encounter many problems, such as the change of requirements, the actual difficulty of the requirements, but with the concerted efforts of the entire team, the problem was solved perfectly. In addition, every team member clearly knows the overall process of the project and what needs to do in different stages.

In conclusion, this project not only allowed us to learn more about the file sharing mechanism, but also allowed us to have a clearer plan for the overall project process.

Acknowledgement

From selecting our preferred topic to implementing the whole system, it took nearly ten months to finish our graduation project. During this period, we have read related papers, similar work done by others and online information based on spring framework and encryption. we would like to express our sincere gratitude and deepest thanks to these authors.

Thanks to Guomin Yang for took time out of his busy schedule to conduct detailed analysis and guidance on the problems we encountered and deepened our understanding of the subject of attribute-based encryption.

Reference

[1] Bethencourt, J, et al., (2007). Ciphertext-Policy Attribute-Based Encryption. 2007 IEEE Symposium on Security and Privacy (SP ’07), Berkeley, France. ff10.1109/SP.2007.11ff. ffhal-01788815

[2] Sliger, M. (2011). Agile project management with Scrum. Paper presented at PMI® Global Congress 2011—North America, Dallas, TX. Newtown Square, PA: Project Management Institute.

[3] Schwaber, K. (2004). Agile project management with Scrum. Redmond, WA: Microsoft Press.

[4] Schwaber, K., & Beedle, M. (2001). Agile software development using Scrum. Upper Saddle River, NJ: Prentice Hall.