COMP4801 Final Year Project [2023/24]

Interim Report

Re-design of HKU's Course Enrollment System with More Features

Group member: Zhang Zhijie (UID: 3033093416)

Xiong Hongruifeng (UID: 3035771921)

Date of submission: 21/01/2024

Abstract

The current HKU course enrollment system has taken a lot of criticism from many HKU students because of its complicated procedures and user interface. Meanwhile, students should have access to course selection resources ahead of time to plan for upcoming semesters. This paper introduces a new course enrollment system to streamline course selection operations and integrate the course information into a single platform. It applied the single-page application structure to reduce the switching time between web pages. The front-end system was the starting point of the whole development process. A part of the skeleton for the course "shopping cart" and the class timetable has been established. Until now, most of the finished work is on the front-end part. In the future, we will dedicate much more effort to constructing the back-end system, such as implementing a high-concurrent server. The possibility of applying any machine-learning model to the course recommendation mechanism will also be discussed.

Table of Contents

1. Introduction

1.1 Background

Course selection and enrollment are subjects of significant interest and discussion among students before the commencement of each semester. For most traditional primary or secondary schools, the teaching team would determine the course timetable at the beginning of each semester, and each student in a class would share the same schedule. In contrast to primary or high school curricula, the university curriculum usually provides many course choices. Although most faculties have compulsory course requirements for their students, they also allow students to choose a significant number of electives so that students still have a highly flexible course arrangement. Meanwhile, students must avoid the time clashes between courses during course selection. The same course may be split into two or more sub-classes each semester. Some courses cannot co-exist with mutually exclusive courses and require students to finish prerequisites. Students must guarantee their course combinations are valid before sending them to the faculty for approval.

Students need resources to determine their course selection in advance to plan for their studies in the following semester. It is necessary to refer to the course information, such as course time, content, teacher, etc., allowing students to arrange their schedules to manage their time wisely. There are some online resources for students' references. They can first find each course's details on the respective faculty's website. The school course selection system also records the complete list of the courses to be selected and provides some basic information about them and attached course website links.

1.2 Motivation- An analysis of the school system

It is necessary to analyze the problems in the existing system to achieve the objective. Many students complain about the cumbersome course selection procedures in the school system. The first step of selection is finding the entry of the course in the system database. Figure 1 shows a part of the user interface of the course list in HKU's system. The table is divided by the components of the graduation requirements. Each section has a sub-list of courses that are further categorized based on their respective fields. The user can browse the course code and description by clicking the expansion arrow. Meanwhile, it also can display the grade

obtained (if the student has already finished the course) and the status of the courses (Taken, In progress, Planned). However, it does not provide a search function, so students must look for a course by browsing the whole list and checking each entry. This is a low-efficiency process.

Figure 1. A screenshot of the course list in HKU's course selection system

Source: [1]

After finding the desired course, students need to add the course to the "course shopping cart," which stores the courses to be sent for requests. This process has several steps. Clicking on the course name will navigate the user to the subclass selection page. After determining the subclass selection, users will be led to the class time information page. This is the last step of cart adding. It requires several web page jumps to add a course to the cart. Moreover, the same process will be applied to each class individually. When users send the courses to the faculty for approval, a similar series of actions is needed. This control flow significantly affects the user experience since it decreases the efficiency of the selection process.

1.3 Objective

We will design a system that integrates the enrollment functions and information display to improve the experience of collecting course information and the student enrollment process. It includes a login system, a course Selection Platform, and a course Information System. This system has a new UI design. Some auxiliary functions, such as keyword searching and filtering, will be added to the course list. This project aims to redesign the shopping cart,

enhancing the visual rendering for a smoother user experience. Its purpose is to address the issue of low-efficiency control flow while adding courses to the cart and transmitting the final decision to the server. The new control flow can reduce the click time in the actions. Our system will also include a recommender system function for students. This system can provide suggestions for students to choose courses based on their previous courses and their majors.

1.4 Outline

We have discussed the background and the motivation of our project. The remaining paper will introduce the methods that have been or will be applied, including programming language, system structure, UI design, etc. Then, we will present the completed work on the front-end part of our system, a course list page, and a calendar page. The final two parts cover the next step plan for the backend part and end the paper with thoughts on the possible application of artificial intelligence technologies to our system.

2. Methodology

This section introduces the development tools utilized (programming languages, framework, IDE, etc.). It then illustrates the system architecture to demonstrate how the front-end part and back-end will be connected. The user interface design will also be discussed, including the control flow, logic, and comparison with the school's system. The final part will briefly introduce methods related to the Database, user authentication, and testing.

2.1 Development tools

This project is a web application. The programming languages for the front-end and back-end components are different. Most of the code will be written in Java code in the back end. Java is an object-orientated language that breaks the whole system into separate modules. It also builds cross-platform applications since Java runs on the JVM, which can be installed on different operating systems. Another feature of Java is safety since it is type-enforcing, and JVM provides garbage collections for memory recovery.

Three languages, HTML, CSS, and JavaScript, will be utilized in the front-end part. HTML defines the visual skeleton of a webpage. CSS describes the presentation of the elements in

the HTML. The JavaScript files regulate the behaviors of the elements. However, this project will not choose the original version of JavaScript. Instead, a variant called TypeScript will be used. It can be regarded as JavaScript with additional syntaxes. For example, the editor will prompt the user to provide type information for the parameters in the function definition, avoiding possible crashing during runtime.

We utilized Vue 3 as the primary framework for constructing the web pages for the front-end development. This framework is built upon the foundations of HTML, CSS, and JavaScript. Vue's essential features include declarative rendering, achieved through a template syntax that describes HTML output based on JavaScript state, and reactivity, involving automatic tracking of JavaScript state changes and efficient updates of the DOM in response to those changes [2]. Employing Vue 3 and external UI components like Vue-Cal and Ant Design, we aim to create a visually appealing and user-friendly web application that provides students with efficient access to course information and management.

The back-end part of the development will be based on the SpringBoot framework. It provides a significant number of third-party libraries for Java development. Many libraries are “out-of-the-box” since they do not need the user’s configuration. By leveraging Spring Boot, we can shift focus from infrastructure concerns to developing the core business features of the application [3].

We will utilize different IDEs in front-end and back-end development. VS code supports a variety of programming languages and contains abundant helper extensions. The front-end development requires the cooperation of multiple programming languages. Therefore, VS code integrates the development environments into a single platform. The front-end project's components and code files can be managed in a sample place. The back-end IDE, IntelliJ IDEA, is dedicated to the Java and Kotlin languages. It also includes support for Java-related frameworks, such as Maven and SpringBoot.

2.2 System architecture

The front-end part defines the appearance of the application and the interactions between the user and the application. It includes two web pages. The first view is the Login interface, and the other is the enrollment system page. Each page includes several function components. For

example, the student's class time and the course list are the components of the course selection interface. View and component files all have the ".vue" suffix. Each vue file has a template area, a script area, and a style definition area. They are written in HTML, TypeScript, and CSS, respectively, corresponding to the functions of these three areas.

The back-end system will contain two function modules. The first will be communicating with the clients and databases. The requests from the client carry different commands, and the back-end system may apply relative operations to the database, like creating or deleting entries. The second is the concurrency management module. It allows the system to process requests from more than one client simultaneously.

API connects the front-end system and back-end system. The payload is usually a JSON string. API documentation is required to regulate the structure of the data transmitted and the response of the opposite party. It describes the expected behavior of the second party and its software in response to a specific request from the first party [4].

2.3 User interface

Our system is a single-page application (SPA) to solve the problem of frequent webpage jumps in the school's existing system. The only jumping is from the Login page to the course enrollment page. Figure 2 demonstrates the layout of the home page. The sider is used to switch the content. Content is the place of the course list. SPA enables users to interact with websites without reloading entire pages from the server to improve performance and a more interactive browsing experience [5].

The user interface will consist of three main views. The initial page presented to users will be the login page. Once logged in, users will be directed to the system's main page. The navigation bar will be positioned on the left side of the page, while the main content will be displayed on the right. Students can switch between the course calendar and course list views through buttons located in the navigation bar. This switching mechanism will be implemented using Vue's router mechanism.

To create other webpage components, such as layouts, buttons, and tables, we will import Ant Design. This UI library offers a wide range of pre-built components that will enhance the

overall aesthetics and functionality of the application. In the course list views, students will be able to browse through the course information and details, and they will have the option to add or remove courses from their shopping cart. We will import Vue-Cal to implement the class timetable. This component will visually represent the course calendar using color blocks to indicate the status of each course. Clicking on a block will trigger a floating window that allows students to view the details of the selected course.

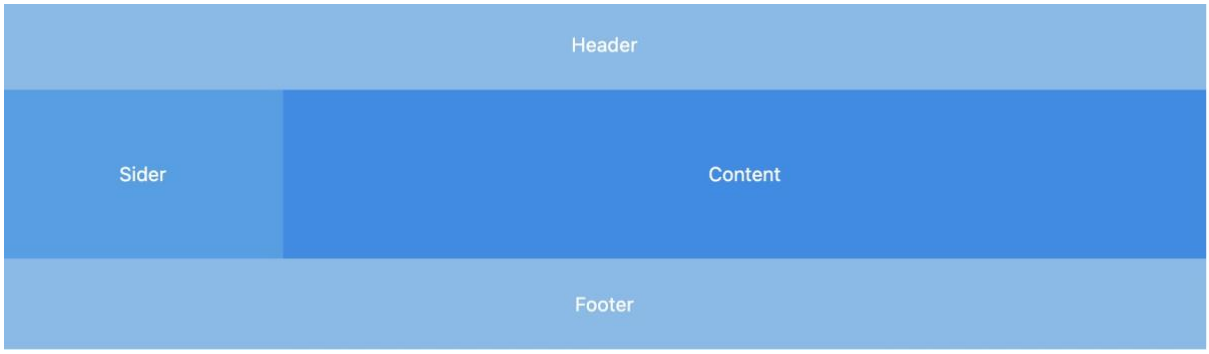

Figure 2. The layout utilized in the course enrollment page.

Source: [6]

2.4 High concurrency-capable back-end server

The back-end server receives a request from the front-end and communicates with the database. The two major requests are data request, which asks the server to fetch the data, and enrollment request, which asks the server to add the user to the corresponding courses' appending list. To cut down the waiting time of the user, we can reduce the workload of each request or handle multiple requests at the same time.

The data can be classified into static data and dynamic data. Static data like the course information and the identity information will not change during the whole enrollment process. The dynamic data like the enrollment records will be modified by the requests. For the data request, a last modified time will be maintained. If the data is not modified since the last request, the request will be skipped to reduce the server workload.

The speed of the database is one of the bottlenecks of the backend performance. The distributed database is implemented (See Figure 3). The data is evenly distributed at multiple sub-databases so that multiple requests can be processed in parallel in sense of internet traffic. A static mapping table is maintained to guide each request to the sub-database containing the

data it acquires. To optimize performance, the frequently accessed data should be stored in different sub-databases as possible.

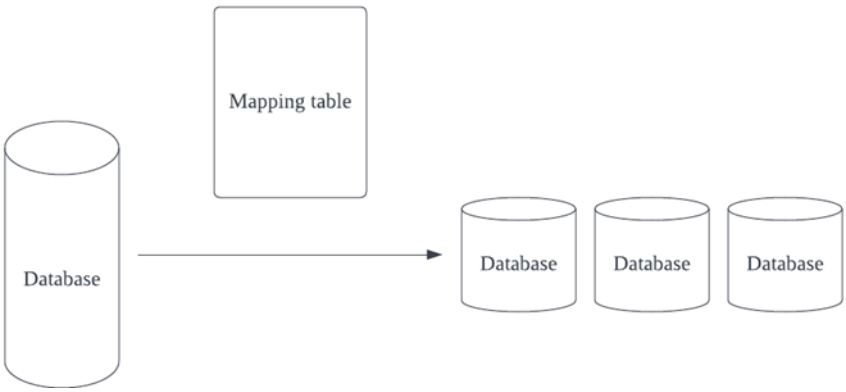

Figure 3 The distributed database

A multi-process workflow is designed to process the enrollment request (See Figure 3). The checking is done by the front-end. For each request, the server allocates a process to read the list of courses received and forward the request to the handlers according to the mapping table.

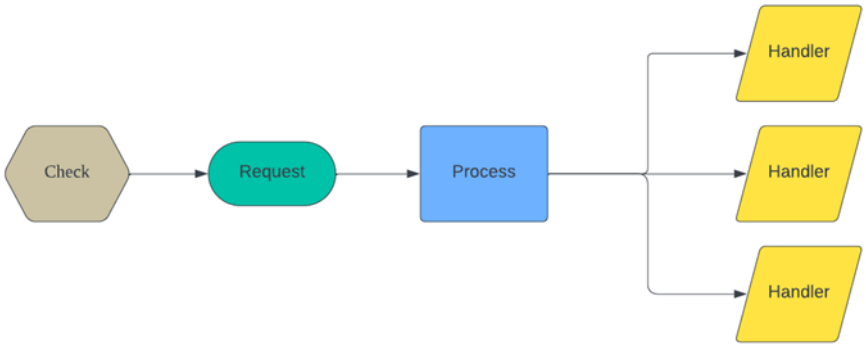

Figure 4 Workflow of handling enrollment request

For the static data, since the data is read only, multiple requests can be handled at the same time without worrying about consistency. For the dynamic data, to ensure consistency, only the requests for different courses can be handled at the same time. Each each course only corresponds to one handler, but a handler can correspond to multiple courses. Multiple handlers work consistently in each sub-database.

To test the response time, a stress test, which uses the K6 framework to create a thousand dummy users accessing the back-end server at the same time, is conducted to estimate the server's response time. It is suggested that presence of feedback prolongs Web users' tolerable

waiting time and the tolerable waiting time for information retrieval is approximately 2 seconds [7], so the actual response time will be compared with it.

2.5 Recommendation System

In addition to addressing waiting times and concurrency, our refined system will introduce an integrated recommendation model to predict and recommend courses to individual students based on their preferences, past selections, and academic objectives.

The recommendation model consists of a large language model (LLM) and a linear discriminant analysis model (LDA) (See Figure 4). The features are extracted by LLM from the course title and description to produce a correlation table, indicating the similarity between courses. Combined with the course feedback and the user's enrollment history, these features can be used to predict the courses the user prefers. An assumption is made that if a student achieved a good grade in a course, he or she will get good grades in similar courses.

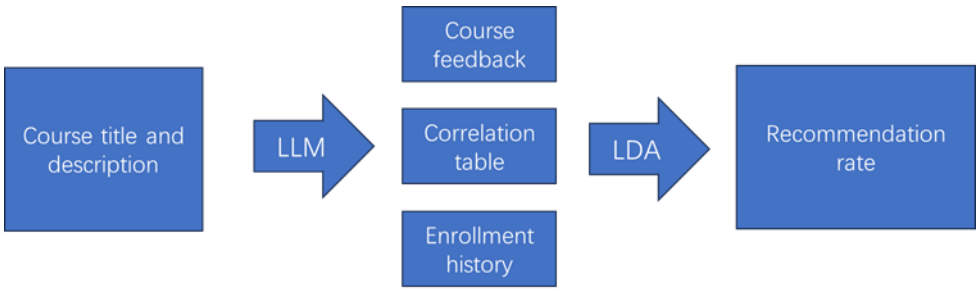

Figure 5 Overview structure of the recommendation model

However, due to the course data like academic history involving in privacy, the data required for training the model is lacking. To solve the problem, the movie data, which is accessible as a MovieLen public dataset online, is used to train the models first, then apply transfer learning to it. The movie and course data have many similarities. Considering that they both have title, description, comments, and there is enrollment history for courses and watching history for movie, it is reasonable to do transfer-learning on movie data.

3. Work accomplished to date

This section introduces the early-stage results of this project. The achieved process is all on the Homepage component of the front-end part. Firstly, the effects will be presented and illustrated. The following content will describe the difficulties met and the relative solution.

3.1 Presentation of the finished work

Figure 6 is the homepage snapshot showing the class timetable. The left sidebar is the navigation for switching two different contents. However, the “Class Timetable” was not selected. That is because when the users enter the home page from the Login page, the default content will be the class timetable. Some of the class blocks in the table are yellow.
This color means the course has already been added to the “shopping cart” and is waiting to be sent to the back-end server. The other course blocks are in green, meaning the students successfully enrolled in these courses. The course length determines the height of each block. A button is put above the calendar for simultaneously enrolling in all the courses in yellow. Meanwhile, users can left-click the blocks to check the details of the courses in a pop-up window (Figure 7). If users want to delete or drop the courses, they can right-click the blocks to finish the process.

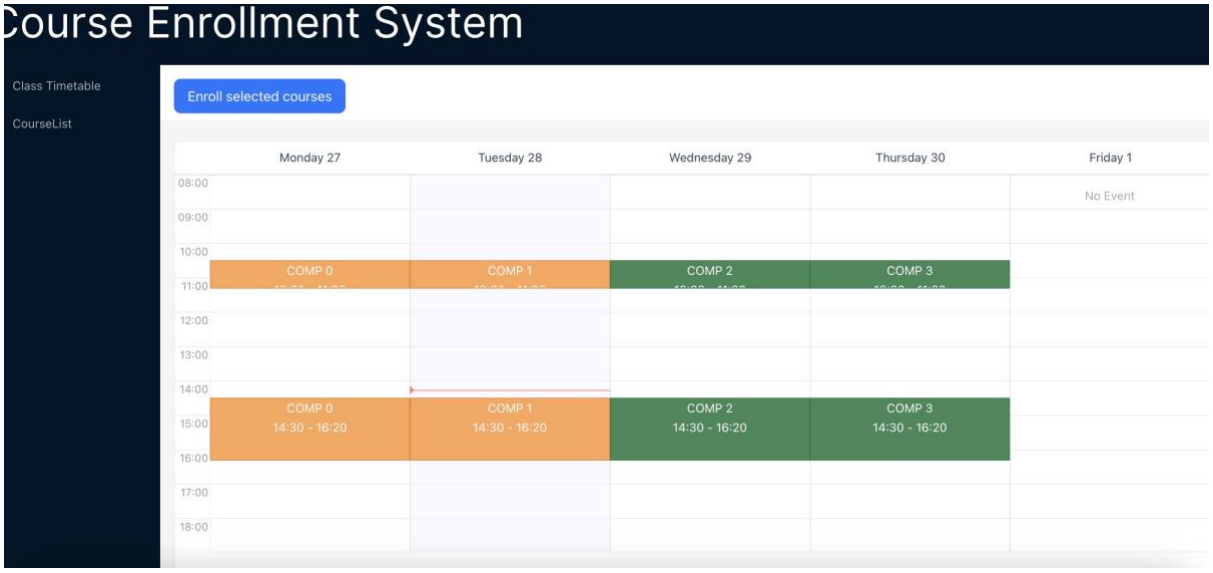

Figure 6 A screenshot of the homepage.

Figure 8 demonstrates the functions implemented to date on the course list. The “Course Name” in the table's header includes a search button. The user can type the keyword in a

search field after clicking the button. The matched phrases will be highlighted in each entry if the search result is not empty.

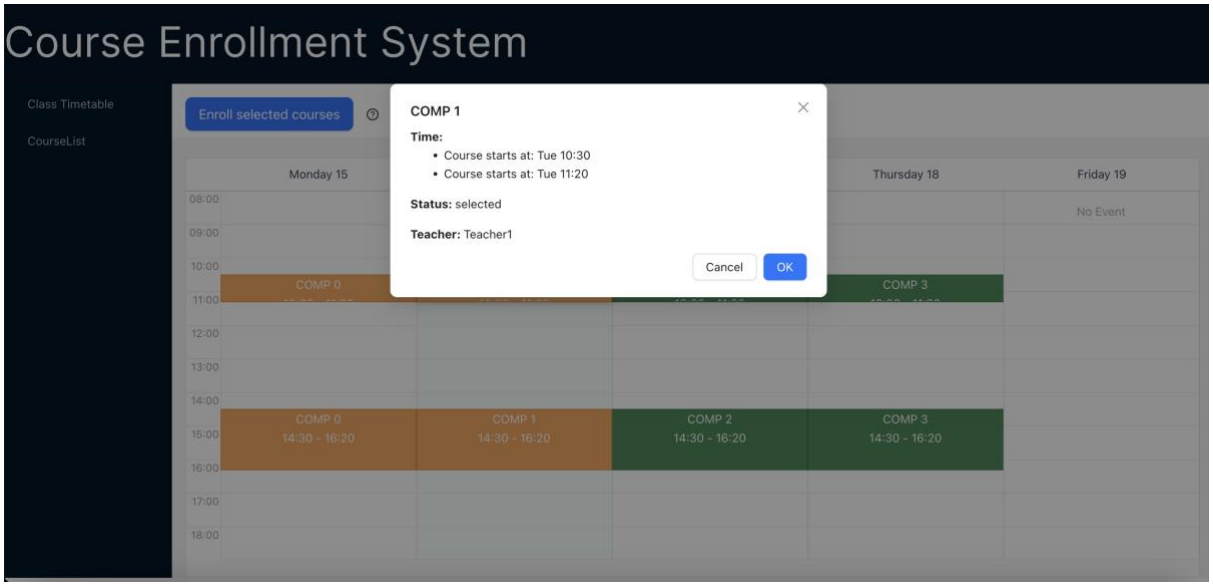

Figure 7 Left-Click a block to check the details

The right-most column is the available action on each entry. If the course has not been added to the class timetable or finished, the action will be "add." The system will not allow the student to choose the courses if he/she does not finish the prerequisite courses or there is a time clash on the timetable. The action option will change to "delete" if a course is put on the calendar (temporary list), such as the course of the first entry in Figure 8. A "drop" option is provided for the enrolled courses. Students cannot take any action on the course they have already finished. When browsing the course list, students can click the "Give me some suggestions" button above the table for course selection advice.

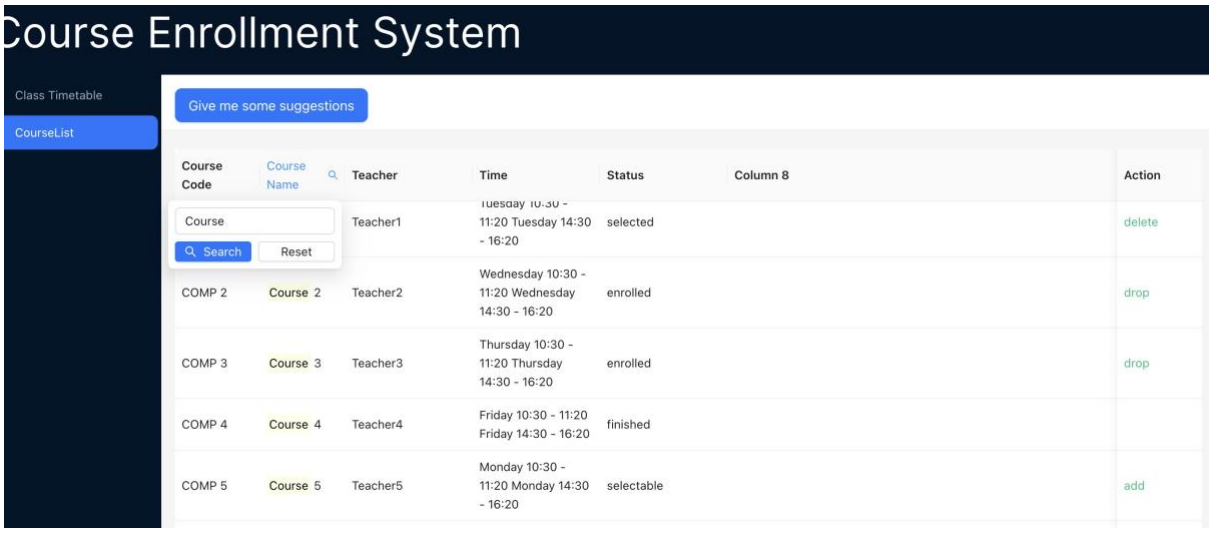

Figure 8. A screenshot of the webpage showing the course list.

3.2 Difficulties encountered and mitigations

Since an external calendar component implements the class timetable, the data loaded into the class timetable should include the absolute time for each period. However, the raw data from the back-end server only contains the weekdays and timeslots for each course. To solve this problem, a transformation process is required to calculate the date of the nearest specific weekday. For example, if today is 23/10/2023(Monday), to correctly load the time of a period on Tuesday to the class timetable, a transformer function will add “24/10/2023” in the time data of that period.

4. Next stage plan

Attention has been given to the fact that many front-end functions have not yet been implemented. Regarding the class timetable, there should be the ability to label courses with more different colors. For example, the courses already sent to the server should be colored differently as a new status. When clicking on the calendar blocks, a prompt window should appear to show more details of the respective courses, not only the status and the teacher name. Additionally, a filter will be added for the course code column in the course list, as the code prefix represents the teaching department of the respective course. Furthermore, add/drop functions will be provided for enrolled courses, and implementing this function will depend on communication with the back-end server.

Milestone	Proposed Due date
Basic front-end functions	01/01/2024
Basic back-end functions	01/01/2024(behind schedule)
Stress test on high-concurrency mechanism	01/04/2024
Testing and implementation of recommendation algorithm	

Table 1 The proposed schedule

The proposed completion date for the essential functions and connection between the front and back end is 01/01/2024. However, our group is still working on implementing the basic back-end functions, so we are behind schedule. Therefore, the back-end system and database construction will be the primary focus in the coming period. The following task will be to optimize the system, and a stress test will evaluate the system's performance under heavy

network traffic. Finally, the design of the recommender algorithm for course selection suggestions is expected to be finished by the same time.

5. Conclusion

We have introduced the course list and course calendar function on our website. Blocks are used to show the time and length of the courses, and different colors indicate different statuses. The search function and recommendation system entries are on the course list interface.

Currently, our work has made it possible to simplify the course selection process. In the school's system, students often need to check each course group individually. This search method can be inefficient, especially when there are many options. However, our system's user interface can significantly reduce webpage switching times during the course selection. This is due to the design of a single-page application module.
Course list and timetable are the two central parts of the front-end system, and most front-end functions have been completed. Most remaining tasks are within the back-end part. Some new functions in the front end rely on the cooperation between the front and back-end systems. Since the back-end system has not been built up and the back-end and front-end are not connected, several buttons, such as enrollment and recommendation, are currently unavailable. Hence, the next immediate step is the implementation of the back-end part.

There are still some limitations to our system. The first problem is that our system does not have a reviewing feature. When searching for their preferred courses, students don't rely solely on information provided on the department website. They also consider feedback from former students, which significantly influences their decision-making process. The second limitation is that we lack data on students' course preferences for model training. A possible solution is for the review content to be used as data for machine learning algorithms in the future. For instance, advanced natural language processing (NLP) technologies can extract critical features from the feedback words, enabling the development of a robust recommendation model by analyzing and understanding the nuances and sentiments expressed in the reviews. It helps empower the system to make more accurate and personalized recommendations to students seeking their ideal educational path.

References

[1] HKU Portal, https://hkuportal.hku.hk/ (accessed Oct. 15, 2023).

[2] "What is Vue?," Introduction | Vue.js, https://vuejs.org/guide/introduction.html#what-is-vue (accessed Oct. 16, 2023).

[3] "Getting started: Building an application with Spring Boot," Getting Started | Building an Application with Spring Boot, https://spring.io/guides/gs/spring-boot/ (accessed Oct. 16, 2023).

[4] A. Fitzgerald, "What is API documentation? [+ how to write and read it]," HubSpot Blog, https://blog.hubspot.com/website/api-documentation (accessed Oct. 18, 2023).

[5] MozDevNet, "Spa (single-page application) - MDN web docs glossary: Definitions of web-related terms: MDN," MDN Web Docs Glossary: Definitions of Web-related terms | MDN, https://developer.mozilla.org/en-US/docs/Glossary/SPA (accessed Oct. 17, 2023).

[6] Ant Design Vue - enterprise-class based on ant design and vue.js, https://antdv.com/components/layout (accessed Oct. 17, 2023).

[7] F. F.-H. Nah, "A study on tolerable waiting time: how long are Web users willing to wait?," Behaviour & Information Technology, vol. 23, no. 3, pp. 153–163, May 2004, doi: 10.1080/01449290410001669914.