香港都會大學
HONG KONG METROPOLITAN UNIVERSITY

ELEC S411F (2025/26)

Electronic and Computer Engineering Project

Interim Report

FYP Matching System

Project Code:	L12
Student Number	13700797
Student Name	Chong Ka Hei
Supervisor:	Dr. Bell Liu
Submission Date:	3-10-2025

Declaration of Originality

I, Chong Ka Hei, declare that this report and the work reported herein was composed by and originated primarily from me. While generative AI tools may have been used to assist in certain aspects of this work, their contributions were limited to supportive roles and were under my direct supervision and control. I take full responsibility for the originality, accuracy, and integrity of the content presented in this report.

This report has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others, as well as any outputs from generative AI tools, has been appropriately acknowledged in the text, and a list of references is provided in the reference section.

Please tick the relevant boxes to indicate the aspects of your work that were composed or inspired by generative AI tools:

	Checklist of AI Usage	*Please Specify*
☐	Brainstorming ideas	
☐	Literature review assistance	
☐	Refining text	
☐	Generating code	
☐	Creating visual content	
☐	Data analysis	
☐	Other	

By completing this checklist, I confirm that all AI-assisted contributions were used ethically and responsibly, and I take full accountability for the final content of this report.

Note: Students are reminded to keep original copies and/or drafts of their work prior to incorporating inputs from generative AI tools to demonstrate the evolution of their ideas and ensure transparency in the creative process. Failure to provide original copies and/or drafts upon request may result in marks deduction and the work being flagged as a potential plagiarism case.

30-9-2025

TABLE OF CONTENTS

1. Introduction 3

1.1 Aims and Objectives 4

1.1.1 Revised Objectives Based on Implementation Progress 4

1.2 Organization of the Report 7

2. Literature Review 8

2.1 Technology Stack Selection and Justification 8

2.2 Analysis of Existing and Alternative Allocation Systems 9

3. Methodology 12

3.1 System interface design and workflow 12

3.2 System Workflow Summary 19

3.3 Design and Implementation Updates 20

3.3.1 Tech Stack Improvements 20

3.3.2 User Interface Updates 21

3.3.2.1 Drag-and-Drop for Ranking Choices 21

3.3.2.2 Project Filtering 21

3.3.3 Matching Algorithm Enhancements 22

4. Experiment 23

4.1 System Implementation Overview 23

5. RESULTS AND DISCUSSION 26

5.1 Project Status and Achieved Milestones 26

6. CONCLUSION AND FURTHER WORK 27

6.1 Conclusion 27

6.2 Data Display and Basic Analysis 27

6.3 Notification and Announcement Center 27

6.4 Improvements to Teacher and Admin Portals 28

6.5 Usability Improvements and User Feedback 28

6.6 Basic Testing and Deployment Preparation 29

References 30

1. INTRODUCTION

The Final Year Project (FYP) is a cornerstone of the Electronic and Computer Engineering program in Hong Kong Metropolitan University. It is a two-term, 10-credit course designed to equip students with the ability to carry out an effective research or development project on their own. Even in other university or other subject, students are also required to finish their FYP to conclude what they learnt in their main subject. A crucial and often complicated aspect of this exercise is the allocation of projects to supervisors and students. Before student start to process their work, they should first propose or choose a related topic of their main subject for their FYP. It is a three-stage system: Proposal, Matching, and Clearing. While it functions, this process heavily employs manual coordination, dispersed communication (e.g. using email and forms), and lacks a central decision-making platform with transparency in real-time.

This manual approach presents several significant challenges that hinder efficiency, fairness, and the overall user experience for all stakeholders involved:

1. Lack of Transparency: Students have limited visibility into which projects are popular or which supervisors are in high demand before making their selections. Most of the student have no experience about their FYP topic and may not be familiar with the professors proposing the topics. Consequently, they are often forced to make this crucial choice based solely on a brief project description, without contextual data that could inform a better decision.

2. Inefficiency for Administrators: Inefficiency for Administrators: Managing student preferences, calculating matches based on GPA and stated interests, and processing the clearing stage for unmatched students manually is naturally time-consuming and prone to human mistake. Even if certain components of matching are mechanized under the existing system, administrators are still faced with the inconvenience of handling special cases, exceptions, and the overall coordination, which continues to be a burdensome administrative task.

3. Limited Visibility for Supervisors: The "big picture" is not easily visible to teachers. They lack insight into how their projects are relatively more or less popular than others, which would provide valuable feedback to influence subsequent project

design. Another significantly reported issue is the "silo effect"; when supervisors submit official and present their project titles, they lack a systematic way of seeing what other faculty members are proposing. This have the tendency to lead to imbalance project difficulty and scope, and in some instances even replication or duplication of project ideas, ultimately reducing the diversity and balance of available alternatives for students.

4. Delayed and Opaque Communication: Students and teachers may not be aware of their assignment status or the overall progress of the allocation process until the final announcement. Students and teachers are kept in the dark regarding their assignment status and the overall progress until the final announcement. This prolonged uncertainty prevents effective planning, fosters anxiety, and eliminates any opportunity for proactive issue resolution, ultimately concentrating pressure and potential disputes at the single point of the final result release.

1.1 AIMS AND OBJECTIVES

The overarching aim of this project is to design, develop, and deploy a comprehensive, web-based FYP Matching System. This system will serve as a centralized digital platform to automate, streamline, and bring transparency to the entire Final Year Project allocation process. It seeks to replace the current fragmented and manual approach, thereby enhancing efficiency, fairness, and the overall experience for students, academic supervisors, and administrative staff. It would be a well-developed FYP matching system that improve the efficiency of the matching period.

1.1.1 REVISED OBJECTIVES BASED ON IMPLEMENTATION PROGRESS

While the overarching aim of this project remains to design, develop, and deploy a comprehensive web-based FYP Matching System, the implementation progress to date allows the original objectives to be refined and reported with clearer status indicators.

With that aim in mind, The project is organized into the following specific, measurable objectives:

1. **To Develop a Centralized and Role-Based Web Application**

   To create a single, unified web portal accessible to all stakeholders. The system will feature three distinct user interfaces tailored to the needs of Students, Supervisors, and Administrators, ensuring that everyone can easily find what they need for their job.

2. **To Implement a Robust and Transparent Matching Algorithm**

   To engineer a core matching algorithm that automates project allocation. It mainly looks at the project choices students list in order of preference. If more than one student wants the same project, GPA will be used to decide who gets it. This keeps things objective and fair for everyone.

3. **To Enhance Decision-Making with Real-Time Data Analytics**

   To integrate data visualization features that provide real-time insights. Students will be able to see how popular projects are (like how many times a project has been picked) to help them choose wisely. Supervisors will get a bird's-eye view of all the available projects. This will lead to better teamwork and make sure all projects are well-designed.

4. **To Streamline Administrative Workflow and Management**

   To build a solid admin system that handles key tasks automatically. It takes care of user accounts, manages project deadlines, starts the matching process, keeps an eye on what happens, and deals with students who don’t get matched. This cuts down on manual work and mistakes.

5. **To Ensure System Reliability and Usability**

   To ensure the final product is not only functional but also robust and user-friendly. That is, rigorous testing to provide system stability at maximum load (e.g., deadline time) and designing an easy-to-use interface [9] that requires minimal training for all user types, thereby encouraging high levels of adoption.

   Table 1 summarizes the current status of each original objective:

Objective	Original Status	Current Status	Remarks
Develop a Centralized and Role-Based Web Application	Planned	Implemented	A role-based web portal has been implemented using React 18 and Vite, with distinct interfaces for Students, Supervisors, and Administrators.
Implement a Robust and Transparent Matching Algorithm	Planned	Implemented	An enhanced Gale–Shapley-based matching algorithm has been developed, incorporating GPA-based tie-breaking and project capacity constraints for fair allocation.
Enhance Decision-Making with Real-Time Data Analytics	Planned	Partially Implemented	Real-time project popularity counters have been implemented in the Student portal, while full statistical charts and dashboards remain under development.
Streamline Administrative Workflow and Management	Planned	Partially Implemented	Core administrative features such as project review and manual allocation are functional; deadline

			management and notification mechanisms are still pending.
Ensure System Reliability and Usability	Planned	Partially Implemented	Basic functional testing has been carried out; comprehensive load testing, security testing, and user acceptance testing have yet to be completed.

Table 1

1.2 ORGANIZATION OF THE REPORT

This paper is organized into a series of key chapters that organize the background, growth, and implementation plan of the proposed FYP Matching System step by step. Section 1 (Introduction) has established the background, reasons, basic objectives, and targets of the project. Following this, Section 2 (Literature Review) offers a critical analysis of existing systems and describes the rationale behind the adoption of the proposed technology stack, specifically MongoDB, Node.js, Express.js, and React, over alternatives. Section 3 (Methodology) describes the overall system architecture, role-based interfaces, and workflow. Section 4 (Experiment) outlines the current implementation of the prototype system. Section 5 (Results and Discussion) summarizes the achieved milestones and reflects on the current status. Section 6 (Conclusion and Further Work) concludes the report and outlines the planned enhancements and future work. This structure is meant to impose a concise and coherent sequence, drawing the reader from an understanding of the problem to an appreciation of the proposed solution and its proposed realization.

2. LITERATURE REVIEW

The development of a modern web application like the FYP Matching System requires a properly chosen set of technologies for the database, the backend, and the frontend. The selected stack, the MERN stack (MongoDB, Express.js, React, Node.js) [3, 6], was chosen after considering alternatives on the basis of suitability to the project requirements of flexibility, performance, and efficiency of development.

2.1 TECHNOLOGY STACK SELECTION AND JUSTIFICATION

2.1.1 MongoDB: The NoSQL Database

MongoDB is a document database used to build highly available and scalable internet applications. [7] With its flexible schema approach, it’s popular with development teams using agile methodologies. Offering drivers for all major programming languages, MongoDB allows you to immediately start building your application without spending time configuring a database. Why MongoDB is being used but not other database system? This document model works great for the FYP system's data [13]. A Project document can hold all the details like title, description, and supervisor. A Student document can list their project preferences. The schema is flexible, so it’s easy to make adjustments as build. Plus, its JSON structure fits right in with the JavaScript backend and frontend, which makes data exchange simple.

2.1.2 Node.js & Express.js: The Backend Foundation

Node.js is a cross-platform, open-source JavaScript runtime environment that allows developers to execute JavaScript code outside of a web browser. [8] Traditionally, JavaScript was limited to client-side web development within browsers, but Node.js extends its capabilities to server-side and network applications. Express.js (or simply Express) is the most popular Node.js web application framework, designed for building web applications and APIs [2]. It's often called the de facto standard server framework for Node.js. It also provides a thin layer of fundamental web application features without obscuring Node.js features. Node.js and Ecprewss.js over options like Django or Spring Boot mainly because it lets us stick to one language, JavaScript, for everything [1]. Both the front end and back end. This makes development smoother, since developers don't have to switch between languages as much, and

it's easier to share code. Node.js also handles a lot of connections at once really well, which is super important when everyone's trying to submit preferences at the same time.

2.1.3 React: The Frontend Library

React (also known as React. js or ReactJS) is a free and open-source front-end JavaScript library [10] that aims to make building user interfaces based on components more "seamless". It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. React is picked over other options like Vue.js and Angular because its reusable components let us build interfaces for Students, Teachers, and Admins that are consistent but still unique. [14] Moreover, React's Virtual DOM makes sure the user experience is quick and fluid, which is key for showing stuff that changes a lot, like live popularity numbers. Also, React is easier to learn and more flexible than a complete framework like Angular.

2.2 Analysis of Existing and Alternative Allocation Systems

This section analyzes the current HKMU system and compares it to other common automated allocation systems. By checking out these options, a better picture can be created and explained why I designed the new system the way I did. The goal is to keep what's good about the current way of doing things, fix what's not working, and add in good stuff from other systems that have already been proven to work well.

2.2.1 The Current HKMU System: A Batch-Processing "Black Box"

The existing allocation process at HKMU is formally structured into three distinct phases, as stated in the student handbook. However, from a user experience and technological perspective, each phase is characterized by a lack of integration and transparency.

1. Proposal Phase：In this initial stage, supervisors propose their project titles. However，they don't have a central place to see what other faculty members are working on. Because of this, projects might not be consistent in difficulty and might even cover the same ground.
2. Matching Phase：This is where the automated part really kicks in, but it's also the least clear. Students look at a project list and rank their choices using a simple form, like a Google Form. Once the time is up, the system uses a secret matching process. It looks

at student preferences and GPAs, but no one knows exactly how it works or gets updates as it goes. Neither students nor supervisors can see what's happening.

3. Clearing Phase：For students who are not matched with any of their preferred projects in the Matching Phase, the system enters a clearing phase. But, this part needs help from the administrator to put students into available spots. It's not always clear how this works, and students don't get to see what's happening, which can be stressful.

Advantage：The three-phase setup makes the admin side straightforward. Additionally, the automation in the Matching Phase makes things easier than doing it all by hand.

Disadvantage：The biggest problem with the system is that it isn't clear or connected. Each of the three parts “Proposals”, “Matching”, and “Clearing” works on its own. Because people have to use different tools for each part, it's hard to plan ahead. This also makes everyone worried and unsure if they can trust the system.

2.2.2 Considered and Rejected: The First-Come-First-Served (FCFS) Model

Automating a first-come, first-served system to make things better, but I decided against it. An FCFS system gives feedback right away and is easy to set up, but it's not really fair. It puts too much importance on connection speed instead of things like a student's abilities or if they're a good fit for a project. Since this idea is fair in school, It was a good choice for the system.

2.2.3 Considered and Adapted: Principles from Stable Matching Algorithms

Using standard matching methods from computer science, like the Gale-Shapley algorithm [4], which is developed by David Gale and Lloyd Shapley, finds a stable matching between two groups by having one group propose to the other based on preference lists. But these methods are too complicated and not easy for people to understand. Plus, they need everyone to submit complete lists of what they want, which is not doable here.

So, my system doesn't use a complex matching method directly. Instead, it takes the main idea "respecting ranked choices" from stable matching theory [11] and makes it simpler. The system uses a clear, rule-based method that focuses on the order of preferences and uses GPA to break

ties. This way, it's fair and easy for everyone to understand, balancing correctness with being user-friendly.

2.2.4 Comparative Analysis and Positioning of the Proposed Solution

After reviewing the current system and other options, a clear design goals is set for the FYP Matching System. It needs to keep the current HKMU system's batch-processing speed after the deadline, but also make it much easier to understand and use, since it's not very clear or connected right now. Here's a quick comparison of what I am aiming for.

Aspect	Current HKMU System	Proposed FYP Matching System
Transparency	**"Black-box" model.** Undisclosed matching logic.	**"Glass-box" model.** Clear, rules-based algorithm (Preferences + GPA).
Supervisor Insight	**Siloed view.** No visibility into other projects.	**Holistic dashboard.** Ability to view and compare all projects.
Student Decision-Making	**Blind submission.** No data on popularity or competition.	**Informed strategy.** Real-time popularity statistics for projects.
User Experience	**Fragmented workflow.** Relies on separate, generic tools.	**Unified platform.** A single, integrated web application for all tasks.
Communication	Prolonged uncertainty. Status unknown until final release.	**Process visibility.** Clear indications of the current phase and status.

Table 2

To summarize, the FYP Matching System aims to fix the key problems with the current HKMU system. It brings an innovative method that emphasizes transparency, gives users more control, and focuses on their needs. At the same time, it keeps the current system's fair, batch-processing method and avoids the downsides of simpler or overly complicated models like FCFS and pure stable matching.

3. METHODOLOGY

This section describes the conceptual and procedural aspects of the system, its basic needs which were the subject of the earlier chapter and the system interface sketches implemented here. The HKMU FYP Matching System of this study features a role-based interface design with three separate user portals, following a clear three-tier architecture [5]. The implementation of each interface is carefully planned to address the requirements of the various user roles of the Final Year Project environment.

3.1 SYSTEM INTERFACE DESIGN AND WORKFLOW

3.1.1 Authentication Interface

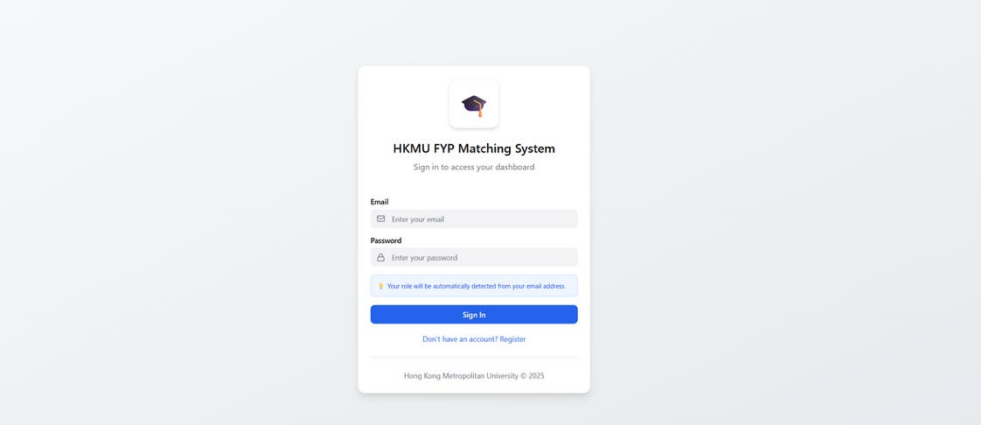

Figure 1 (login page)

In figure 1, the system has a unified access point for students, teachers, and administrator. When you login, there are spots for your email and password to show you what to type. One neat thing it does is figure out your character by looking at your email. Then, it sends you to the right place after you log in. There is also a sign-up link for students and teachers who don't have their account yet.

3.1.2 Student Portal Interface

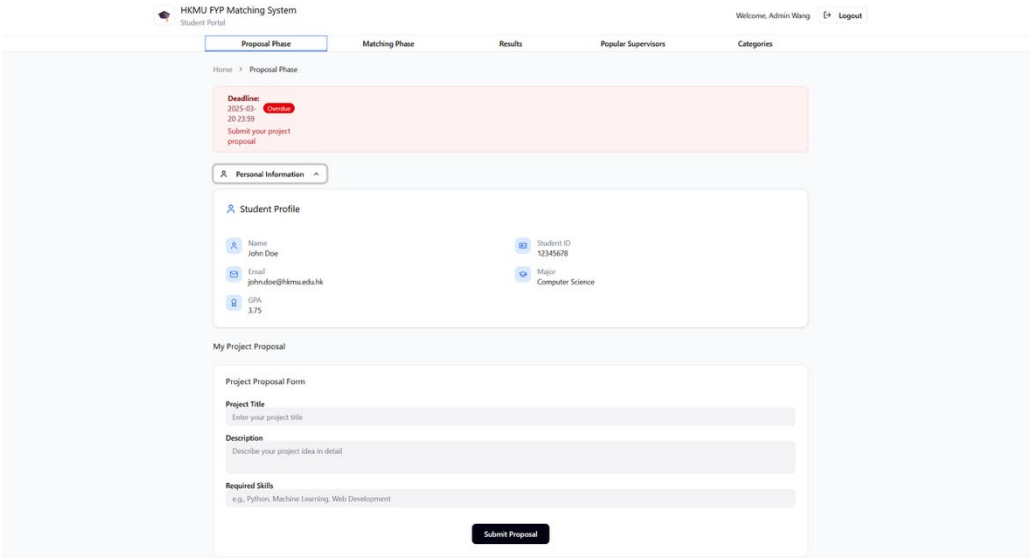

Figure 2 (student account, propose phase)

In figure 2, students can find a detailed proposal form where they need to fill in their project title, description, and the tech skills needed. The system will show their profile information, like name, email, GPA, student ID, and major. The form is easy to use with examples and straightforward ways to submit.

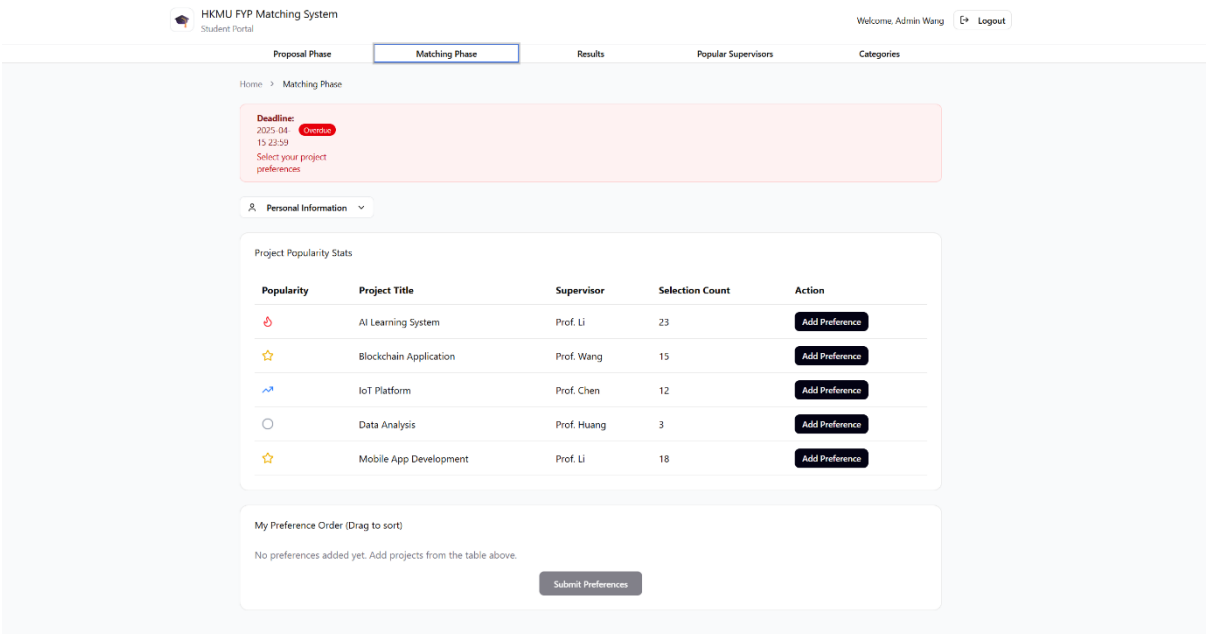

Figure 3 (student account, matching phase)

In figure 3, students can check out projects through a live interface that shows how popular each one is at that moment. The table lists project names, supervisors, how many students have picked them, and an Add Preference button. Students can rank projects by dragging and dropping them into their preferred order, and the system shows which ones they have already chosen.

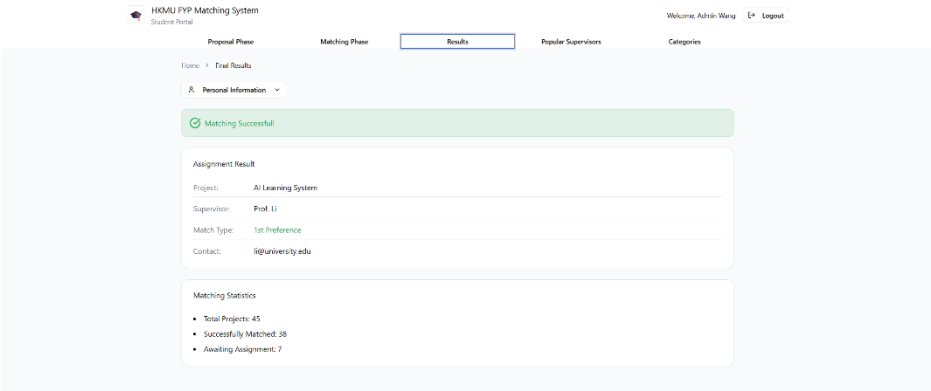

Figure 4 (student account, result)

After the matching process, it turns to result part just shown as figure 4. Students get assignment results that show their project, supervisor, match type ((like 1st Preference) and contact details. The system also shows matching stats (for example, 38 of 45 projects were a successful match) and links to helpful resources, such as popular supervisors and category stats.

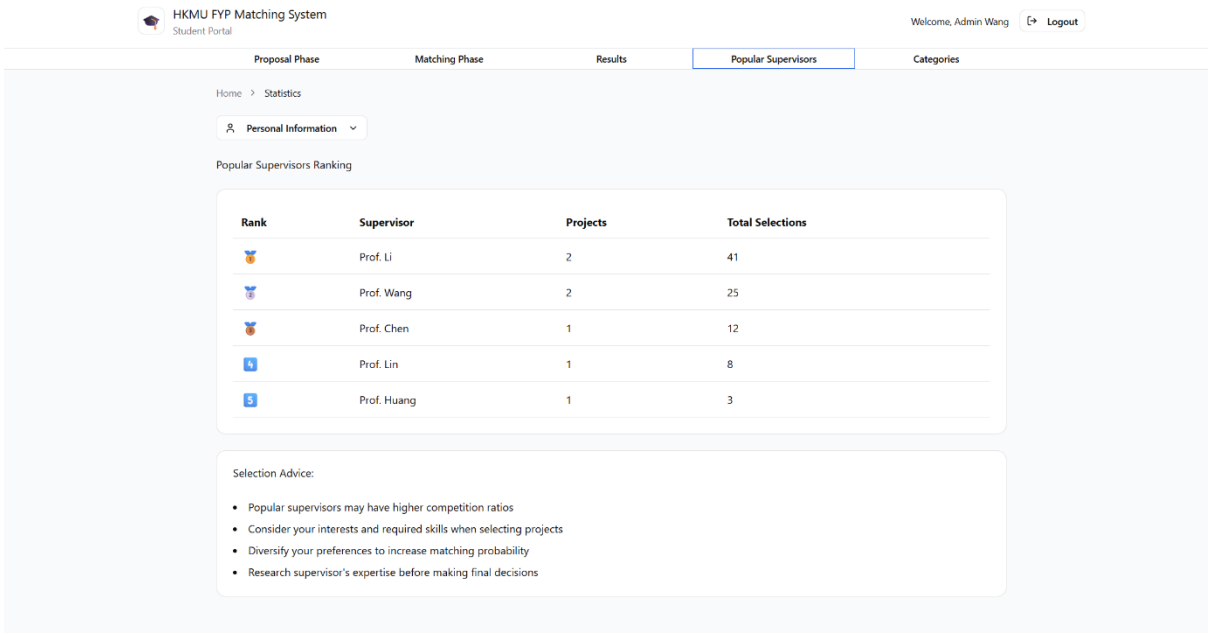

Figure 5 (student account, popular supervisors)

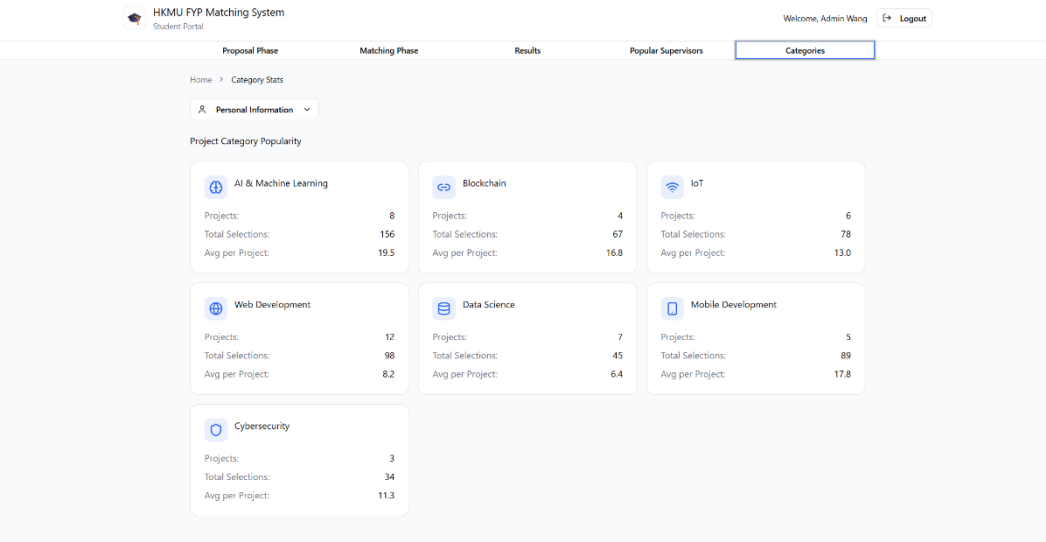

Figure 6 (student account, categories)

Figure 5 and 6 show how students can view supervisor rankings by project count and selections, plus category popularity stats. This information, along with selection tips, helps students pick projects that match their interests and have a decent chance of getting chosen.

3.1.3 （Application Management Interface）

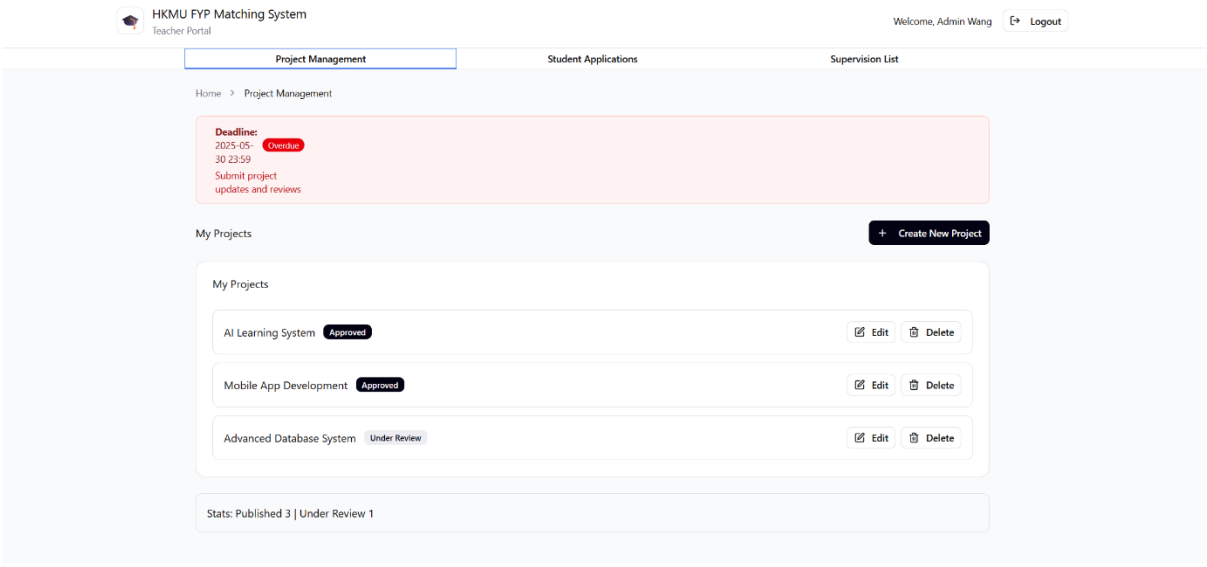

Figure 7 (teacher account, project management)

Supervisors see a dashboard with their projects and applicant lists as shown as figure 7. They can view student applicants, along with their GPAs and preference rankings, which helps them

make good selection choices. Each project section organizes applicant info clearly using visual layout.

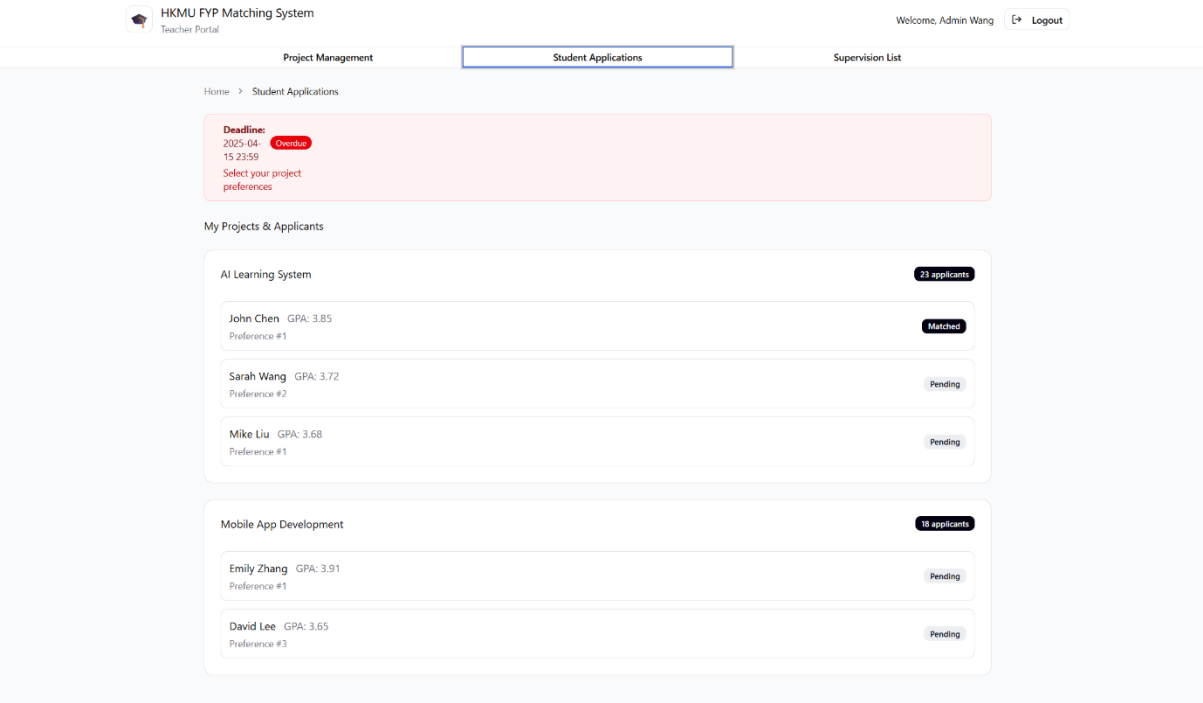

Figure 8 (teacher account, student application)

Teachers are able to see their assigned students, along with their GPA, department, and contact info as shown as figure 8. The system has tools to manage schedules and set up meetings, which helps with project guidance during the school year.

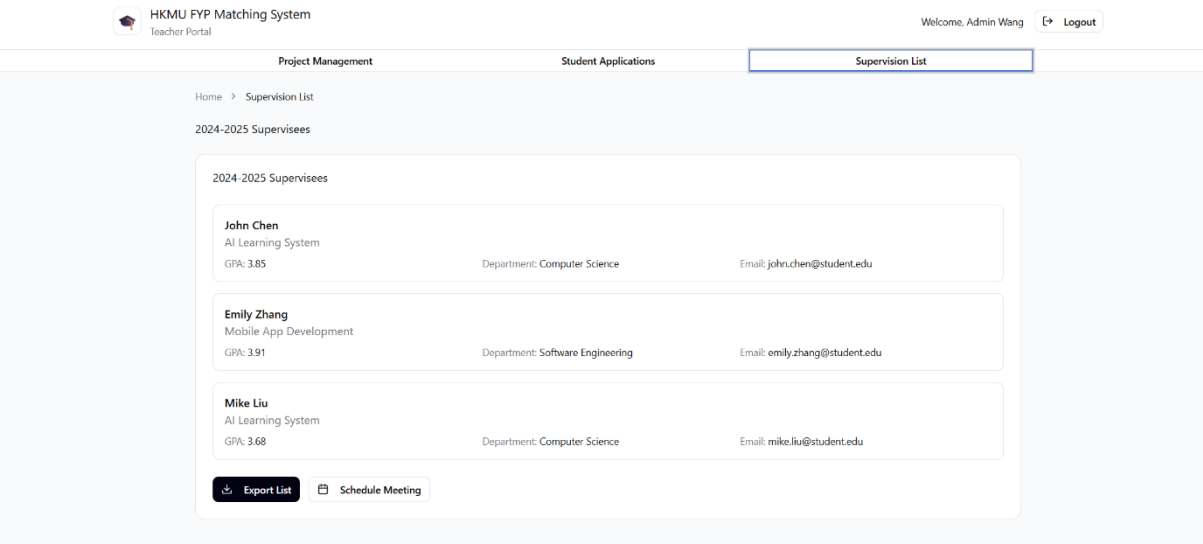

Figure 9 (teacher account, supervision list)

The Supervision List page shows the final list of students each teacher is assigned to for the school year (like 2024-2025) as shown as figure 9. You'll see a simple list with each student's full details: name, project title, GPA, department, and email. It's the go-to spot for managing supervisions and usually includes handy tools like a Schedule Meeting button to help teachers keep in touch and track how their students are doing on projects.

3.1.4 Admin Portal Interfaces

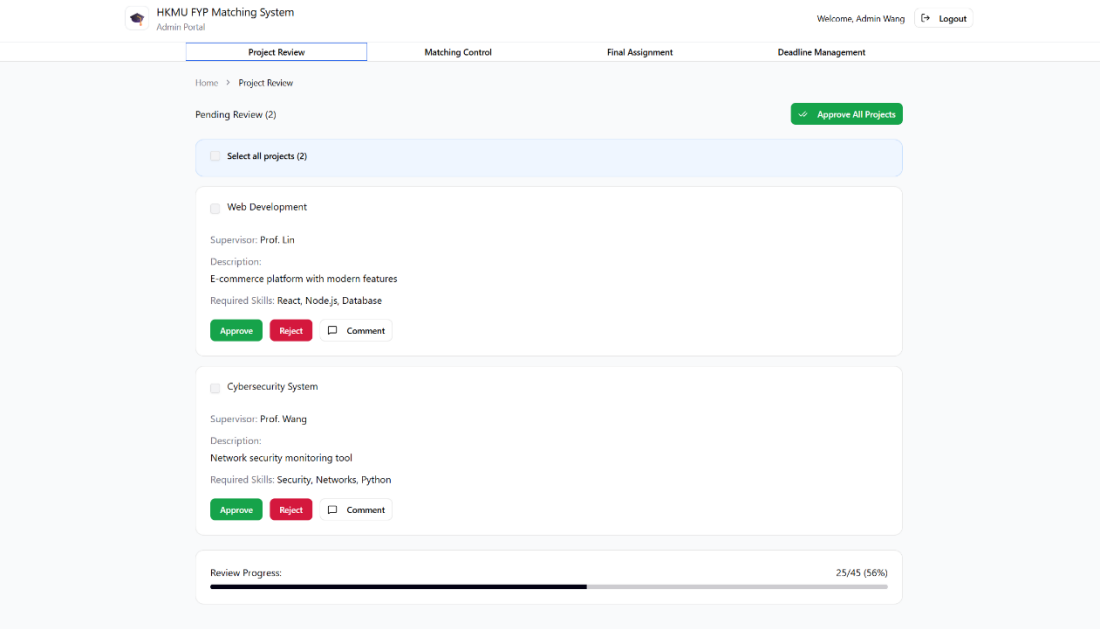

Figure 10 (admin account, project review)

Admins can see projects that need approval on the review dashboard as shown as figure 10. For each project, they'll find details, required skills, and options to approve, reject, or add comments. A progress bar shows how the review process is going, like 25 out of 45 projects reviewed, 56% done.

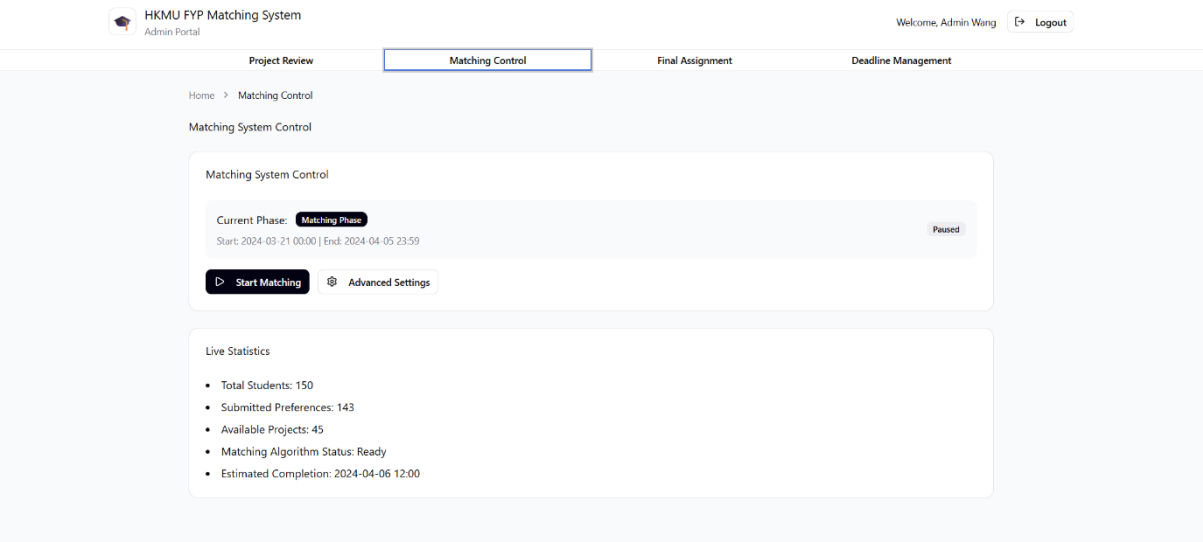

Figure 11 (admin account, matching control)

The admin matching dashboard features phase management with start/end timestamps, live statistics (e.g., 150 students, 143 preferences submitted), and algorithm control mechanisms as shown as figure 11. The interface includes advanced settings and estimated completion forecasting for comprehensive matching oversight.

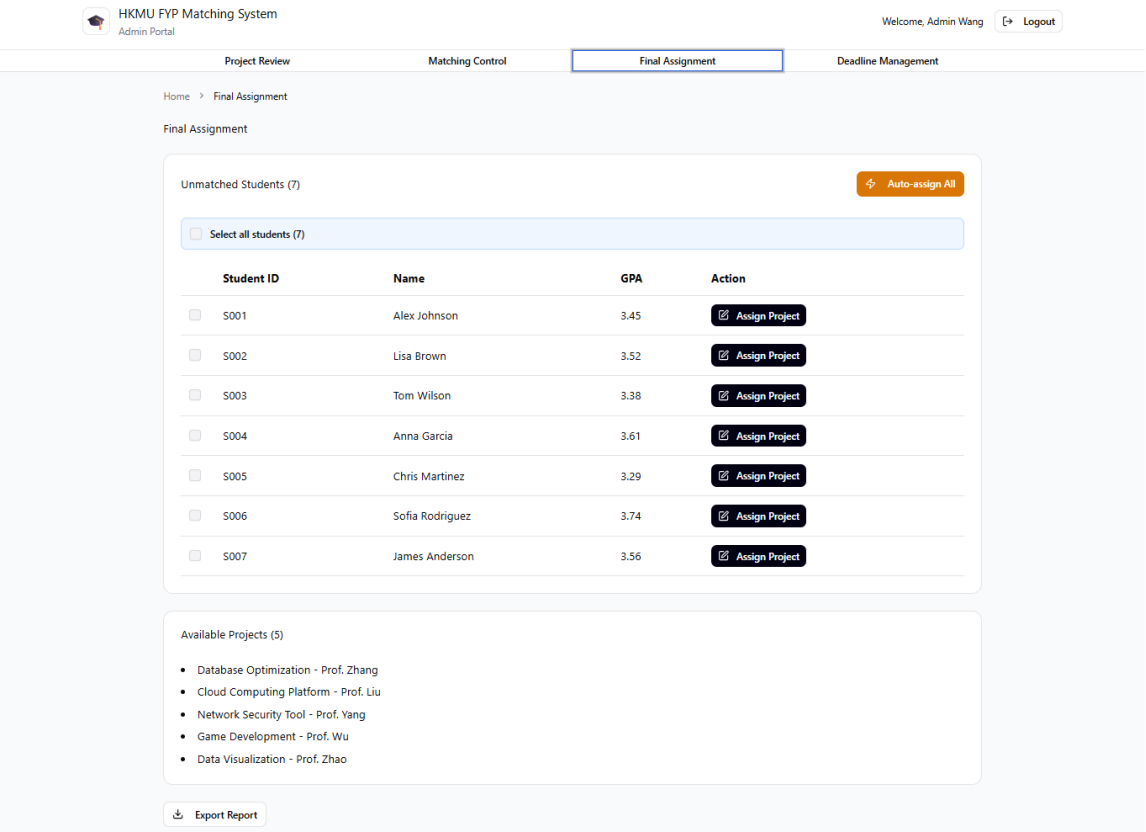

Figure 12 (admin account, final assignment)

If there are students who don't get matched automatically, administrators can easily put them into open projects by hand. The system shows a list of students who still need a project (like 7 students, for example), along with their GPA and the projects that are still available as shown as figure 12. This helps make sure everyone gets a fair placement, even if the automatic system doesn't work for them.

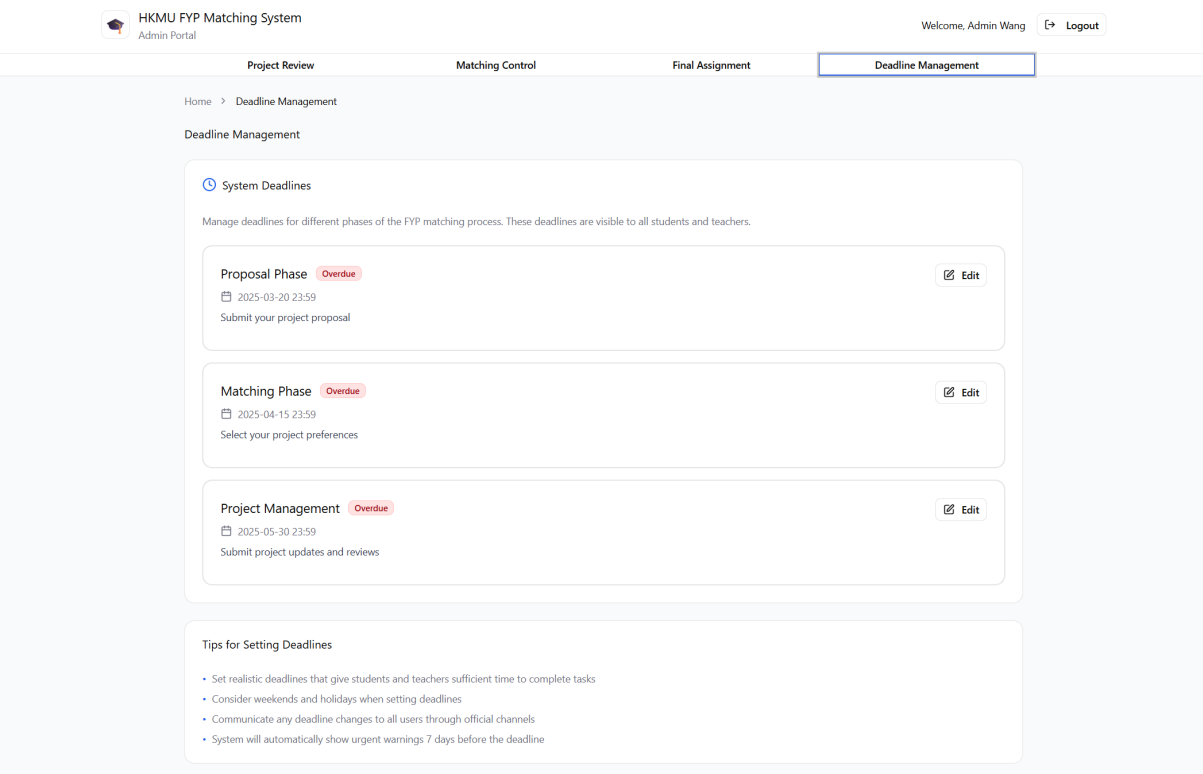

Figure 13 (admin account, deadline management)

Admins can control deadlines across the whole system, setting and changing important phase timelines with automated alerts as shown as figure 13. The interface handles the proposal, matching, and project management stages, keeping academic calendars and communication guidelines in mind.

3.2 System Workflow Summary

Phase 1: Proposal and Review

Students submit project proposals, and administrators check and sign off on project lists. The system keeps the review process in order and makes sure progress is tracked and quality is maintained.

Phase 2: Preference-Based Matching

Students look through the projects, check out what's popular, and then rank what they want to work on. The system then uses a smart algorithm to match students to projects, looking at their preferences, grades, and how many spots are open in each project.

Phase 3: Results and Management

Students see their approved matches, along with supervisor contact information, and teachers get their student lists. Administrators handle special situations by assigning matches manually and keeping projects on track using deadlines.

The interface design works well for everyone involved because it gives them the right tools for their roles, keeps things organized, and is designed to be easy to use [9]. It mixes automated tasks with admin controls. Therefore, work is shared fairly, but there remains flexibility to handle special cases.

3.3 Design and Implementation Updates

This subsection summarizes the main design refinements and implementation decisions made after the initial proposal. It focuses on how the actual system differs from, or extends beyond, the original plan.

3.3.1 Tech Stack Improvements

In Section 2 mentioned, why the MERN stack has been chosen. Since then, some specific improvements to make development easier, keep things organized, and boost how well it runs.

Table X shows a comparison between the original high-level design and the implemented solution:

Component	Original Design	Implemented Solution
Frontend framework	React (generic)	React 18 with Vite 5, providing a faster development server, optimized builds, and a modern tooling ecosystem.

Styling approach	Inline styles / global CSS	Modular CSS with per-component stylesheets, improving maintainability and reducing style conflicts across pages.
State management	Not explicitly specified	React Hooks (including useState, useEffect, and useMemo) for reactive state management and performance optimisation.
Routing	Not explicitly specified	React Router DOM v6 with role-based protected routes for Students, Supervisors, and Administrators.

Table 3

These improvements keep the original three-tier setup but make the code easier to change and add to.

3.3.2 User Interface Updates

The user interfaces are close to the original designs shown in Section 3.1, but several usability updates have been introduced to make the system easier to use.

3.3.2.1 Drag-and-Drop for Ranking Choices

The first design showed ranking project options as a simple list. Now, students can rank their choices by dragging and dropping them in the order they want. This change makes ranking simpler and helps to avoid mistakes that could happen when students reorder with buttons.

3.3.2.2 Project Filtering

The project search tool is improved to help students pick the right projects. Here's what's new, find projects by searching keywords in the title, description, or supervisor's name. Filter by skills needed, using tags pulled from listed required skills for each project. Choose projects by supervisor using a dropdown list. And, Sort projects by popularity, title, or supervisor's name. These updates give students more info and options when choosing their FYP projects.

3.3.3 MATCHING ALGORITHM ENHANCEMENTS

The first plan was to use a basic, straightforward method based on matching ideas. This would mainly use student choices and grades. Now, this idea is a Gale-Shapley-style system but with extra limits and ways to settle ties.

Here's what the updated algorithm does:

- GPA tie-breaker: If more than one student wants the same project, pick the one with the better GPA. It's a clear and simple way to decide.
- Project capacity limits: Set a limit on how many students can work on each project. This keeps things manageable.
- Preference limits: Students can list up to 10 project choices, ranked in order. This keeps the system from getting bogged down and helps students focus.
- Popularity tracking: keep tracking of how many students want each project, updating the counts as students make their choices.

This information shows up in the Student portal. These updates make the whole process fairer and easier to understand, while still sticking to the original plan.

3.3.4 Data Export Functionality

Some CSV export options are being added on the server. This makes it easier to keep records, create reports, and do other analysis. The data can export out of the system.

4. EXPERIMENT

4.1 SYSTEM IMPLEMENTATION OVERVIEW

The current prototype has been implemented using React 18 with Vite on the frontend and Node.js/Express with MongoDB on the backend. The main setup, login system, and Student portal have been completed according to the original plan. Role-based portals for students, supervisors, and administrators have been set up and are running in a local development environment. The Teacher and Admin portals currently provide their main pages and layouts, with further backend integration planned.

4.2 MATCHING ALGORITHM IMPLEMENTATION

In the current implementation, the matching algorithm is based on a Gale–Shapley-style design with additional constraints.

4.3 PROTOTYPE SCREENSHOTS

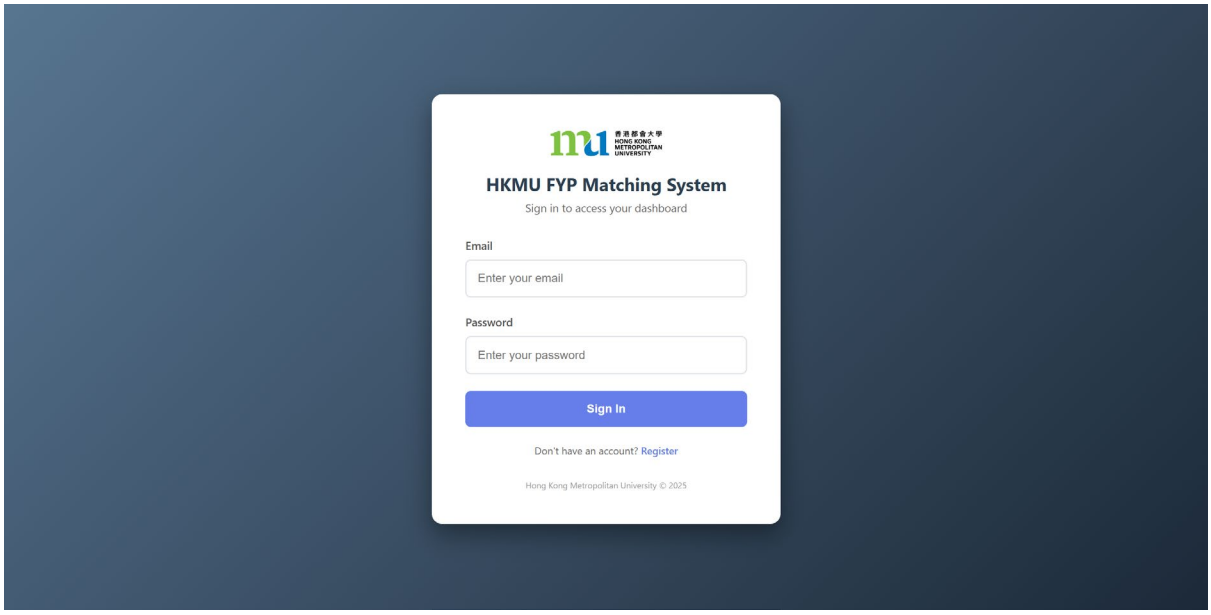

Figure 14 (Implemented login interface for all user roles.)

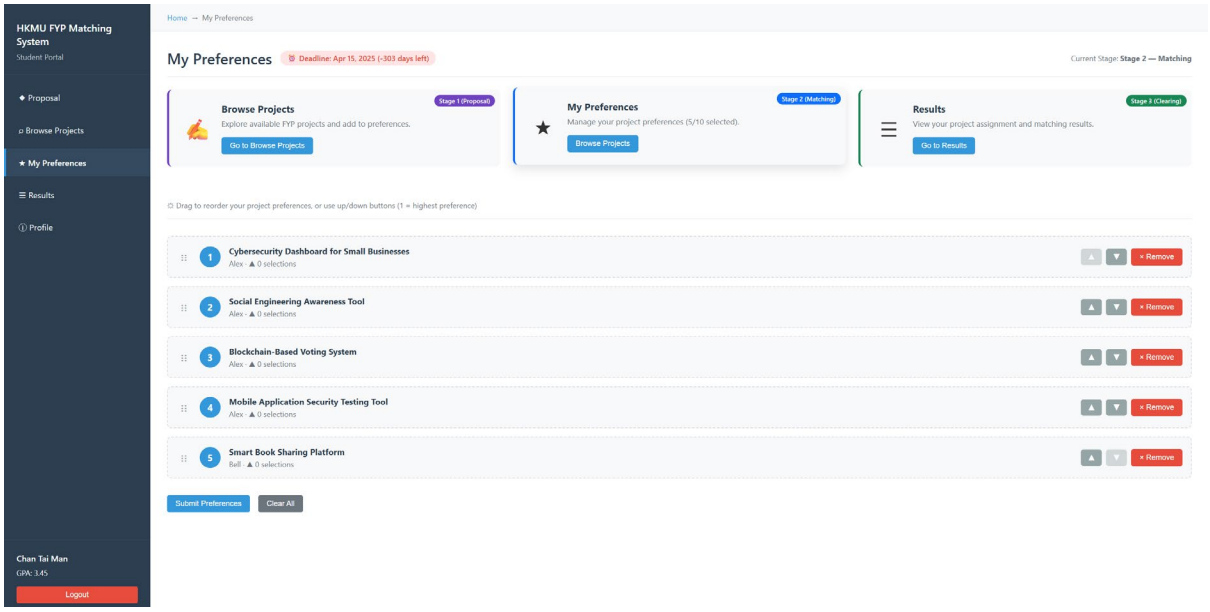

Figure 15 (Student portal with drag-and-drop interface for project preference ranking.)

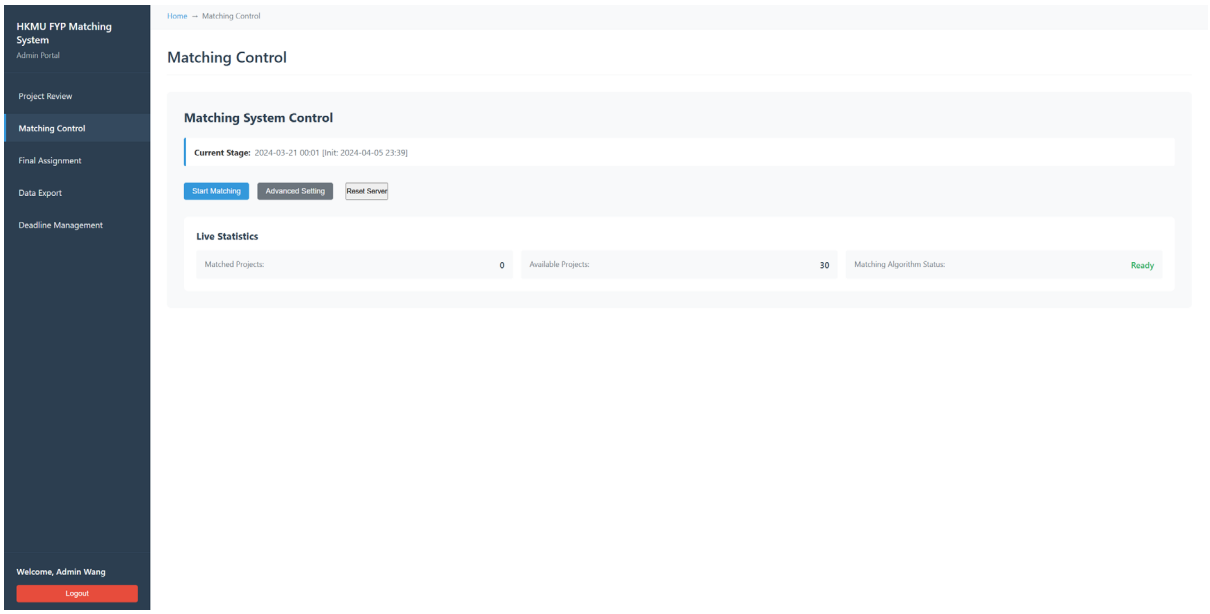

Figure 16 (Admin portal for project review and matching control.)

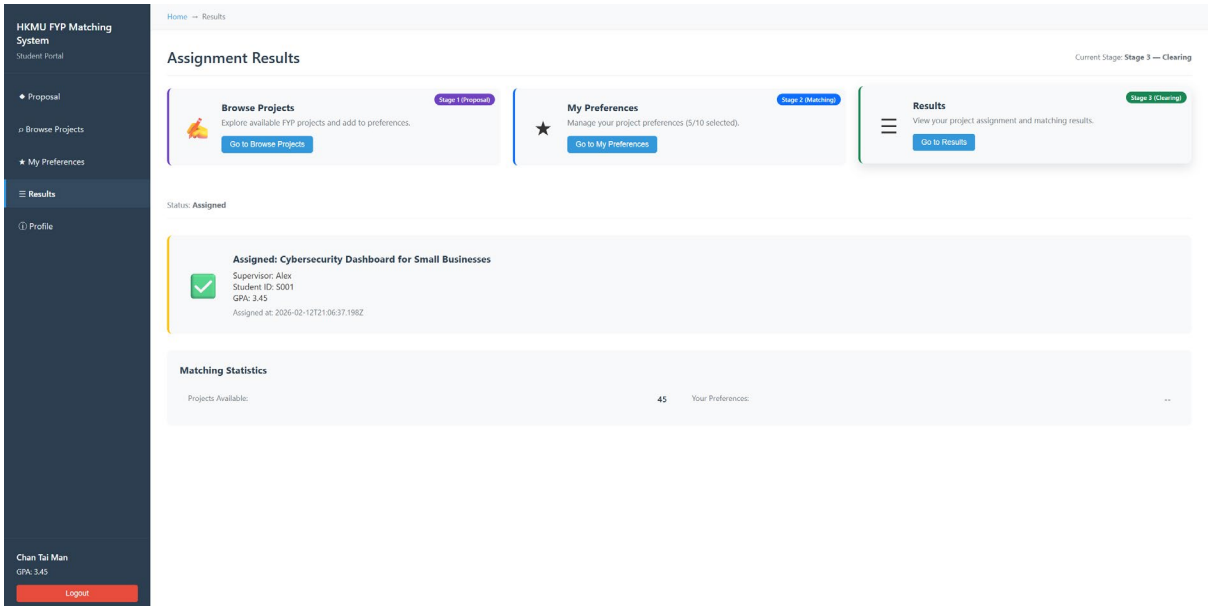

Figure 17 (Student view of final matching result and summary statistics.)

5. RESULTS AND DISCUSSION

5.1 PROJECT STATUS AND ACHIEVED MILESTONES

The project started with stages for system setup, portal design, algorithm development, and testing. As of this report, the initial setup, login mechanism, and basic Student portal have largely been completed according to schedule. The matching algorithm has also been implemented slightly earlier than planned. In contrast, the Teacher and Admin portals are still in progress and require additional backend work. Data visualization, systematic testing, and deployment preparation have not yet started and are planned for the next phase. Overall, the core functionality is in place, but further development is required before the system is ready for real-world use.

6. CONCLUSION AND FURTHER WORK

6.1 CONCLUSION

This report has described the motivation, design, and current implementation status of the FYP Matching System. The main goal of the project – to provide a clear, centralized online platform for Final Year Project allocation – remains unchanged. A role-based web application has been created with dedicated areas for students, supervisors, and administrative staff. The enhanced matching process can now assign students to projects based on their stated preferences, GPA, and project capacity limits. Although the system is not yet ready for deployment, the work completed so far shows that the planned design is technically feasible and consistent with the original objectives. The remaining work mainly focuses on improving usability, completing portal integration, and strengthening reliability, rather than changing the core architecture.

6.2 DATA DISPLAY AND BASIC ANALYSIS

Currently, the system offers simple numbers, like how popular a project is, to help students make choices. Next, a straightforward visual can give users a clearer idea of the project choices. Planned improvements include:

6. A small stats area showing the total number of students, projects, and students matched to projects.
7. Simple charts to show how popular projects are (like a bar chart showing how many students picked each project).
8. Simple summaries for administrators, like the number of students not matched and how full each project is.

These visuals will be added using simple chart tools that work with the current React setup, without changing how the system matches students to projects.

6.3 NOTIFICATION AND ANNOUNCEMENT CENTER

Inside the Application Right now, people have to refresh pages to get updates on project status. To reduce any confusion, a notification center should be added inside the application.
Planned improvements include:

- Each portal will have a notification area for system messages (like phase changes, project approvals, or updated deadlines).
- Each person will have a simple inbox showing read and unread messages.

- Icons will tell users when they have new notifications.

All notifications will be in the web application, so external emails won't be used. This keeps things simple.

6.4 Improvements to Teacher and Admin Portals

The existing Teacher and Admin portals show the main pages and layouts, but some processes are still basic or not fully finished. Completing and refining these key processes instead of introducing brand-new systems.

For the Teacher portal, planned improvements are:

- Finishing project management functions (creating, editing, and archiving projects on the web).
- Making the applicant view better so supervisors can easily see details such as GPA, department, and preferred order.
- Adding simple note or comment sections for supervisors to write remarks about their students or projects.

For the Admin portal, planned improvements are:

- Refining the project review process by adding status indicators (like pending, approved, or rejected).
- Creating an easier way to manually assign students who have not been matched.
- Making the deadline management easier to use, so it is simpler to set up and check phase start and end times.

These changes will improve the portals and make them more user-friendly, without adding much complexity to the system.

6.5 Usability Improvements and User Feedback

While the current interfaces work, not all usability aspects have been checked with actual users. Future work will involve small improvements based on user feedback.

Planned actions include:

- Hold user feedback sessions with a small group of students, supervisors, and administrators to find confusing workflows or missing details.
- Improve form layouts, labels, and error messages to avoid wrong input and make each step clearer.
- Add help (like tooltips or short explanations) on important pages like the preference submission and matching results screens.

These changes focus on making the system simpler to learn and use, mainly for new users.

6.6 Basic Testing and Deployment Preparation

So far, testing has mainly been carried out manually during development. To increase confidence in the stability of the system, a modest amount of structured testing and deployment preparation is planned.

Planned improvements include:

- Adding a few unit tests for important backend parts, like the matching system and key API points.
- Run simple load tests for common busy times. For example, when many students send in or change their choices close to the deadline.
- Get the system ready to launch on a test server, setting up the basics for a final version and environment settings.

These steps should make things more reliable without needing a fully automated or huge setup.

REFERENCES

[1] Abdullah, A. (2019). Benefits and Challenges of Node.js. GeeksforGeeks. https://www.geeksforgeeks.org/benefits-and-challenges-of-node-js/

[2] Express.js Foundation. (n.d.). Express - Node.js web application framework. Retrieved October 2, 2024, from https://expressjs.com/

[3] FreeCodeCamp. (2022). What is the MERN Stack? https://www.freecodecamp.org/news/what-is-the-mern-stack/

[4] Gale, D., & Shapley, L. S. (1962). College admissions and the stability of marriage. The American Mathematical Monthly, 69(1), 9–15. https://doi.org/10.1080/00029890.1962.11989815

[5] IBM. (n.d.). What is three-tier architecture? Retrieved October 2, 2024, from https://www.ibm.com/topics/three-tier-architecture

[6] MongoDB, Inc. (n.d.). MERN stack. Retrieved October 2, 2024, from https://www.mongodb.com/mern-stack

[7] MongoDB, Inc. (n.d.). Why use MongoDB? Retrieved October 2, 2024, from https://www.mongodb.com/resources/products/fundamentals/why-use-mongodb

[8] Node.js Foundation. (n.d.). About Node.js. Retrieved October 2, 2024, from https://nodejs.org/en/about

[9] Nielsen, J. (1994, April 24). 10 usability heuristics for user interface design. Nielsen Norman Group. https://www.nngroup.com/articles/ten-usability-heuristics/

[10] React. (n.d.). React – A JavaScript library for building user interfaces. Retrieved October 2, 2024, from https://react.dev/

[11] The Nobel Prize. (2012). Stable allocations and the practice of market design. https://www.nobelprize.org/uploads/2018/06/popular-economicsciences2012.pdf

[12] Vercel. (n.d.). What is the MERN stack? Retrieved October 2, 2024, from https://vercel.com/docs/beginner-symbols/what-is-the-mern-stack

[13] W3Schools. (n.d.). What is MongoDB? Retrieved October 2, 2024, from https://www.w3schools.com/mongodb/mongodb_intro.php

[14] Wikipedia contributors. (2024, September 20). React (software). In Wikipedia. https://en.wikipedia.org/wiki/React_(software)