PROJECT PROPOSAL

Unit Set Builder

Group16
Zhanerken Azimbayev (23733728)
Qianqian Li (23770748)
Ankita Narvekar (23829237)
Yuqiao Qian (24129392)
Siting Xiang (23737625)
Siyuan Zhou (23861658)

Aim

The aim of this project is to develop a user-friendly web application called "Unit Set Builder" for the UWA. This application will manage various types of unit sets associated with courses or majors, replacing some functionality of the existing "Caidi" system. The focus is on an improved UI/UX for complex scenarios, with an API developed to output unit set data in a JSON format for use in various UWA platforms.

Background

The UWA uses a system called "Caidi" to manage unit sets for its courses and majors. The existing system has limitations in user interface, which the client wants to address. The new system must be user-friendly for academic staff and accessible to students without an IT background.

User Stories

- As a teaching staff member, I want to create, view and manage unit sets with nested groups so that I can accurately represent the structure of courses and majors, including bridging, core, and optional units.
- As a teaching staff member, I want an intuitive and responsive UI/UX that simplifies the process of building and modifying unit sets, reducing the time and effort required.
- As a developer, I need an API that outputs unit set data in a JSON format so that I can easily integrate it with other university systems.

Software Requirements

User Interface / Experience (UI / UX)

The UI will be clean, intuitive, and accessible, catering to both academic staff inputting data and students accessing it, regardless of their IT knowledge. Users will be able to create and modify nested groups within unit sets, with the interface designed to handle complex scenarios seamlessly.

Backend and Database

The backend will feature secure management of unit set data, with user authentication and role-based access control to protect sensitive information. A robust database system will be implemented to efficiently store and retrieve complex nested data structures, ensuring high performance even with large datasets.

API Development

An API will be developed to output unit set data in JSON format, ensuring compatibility with existing UWA platforms like the Handbook and study planners. The API will be well-documented to facilitate easy integration with other university systems.

Performance and Optimisation

The application will be optimised for fast loading times and responsiveness, even when handling large datasets or complex queries.

Deployment and Maintenance

The system will be deployed seamlessly with minimal disruption to users, ensuring a smooth transition from the existing setup. Comprehensive documentation and training materials will be provided to facilitate adoption for both administrators and end-users.

Minimum Viable Product (MVP)

The MVP will deliver a functional version addressing the core needs of target users, focusing on essential features. This approach allows early user feedback, enabling iterative improvements and reducing the risk of unnecessary features.

Task 1: Better UI/UX Design

Description: Users of "Caidi" are having a bad user experience due to its outdated design. By completing this function, user should have a convenience, user-friendly approach to create, manage unit set. For Figma prototype of this process, please refer to Appendix [4].

Rationale: User must scroll up and down when the unit set is very long, which reduced efficiency and provided users a bad experience.

Task 2: Restructure Database

Description: With existing database design and data flow, it is hard to optimise user experience by using our plan. We decided to separate the "Specialisation" out from unit set. The Entity-Relationship Diagram (ERD) for the database structure is provided in Appendix [5] for further reference.

Rationale: User will have to create multiple same specialisations for different unit sets due to current database structure does not support reuse of specialisation.

Task 3: Output Unit Set

Description: The system should have ability to output the unit set information in organised PDF format, and return unit set data to UWA handbook, Study planner etc.

Technology used for the project

The MVP of the project excludes certain features, planned for future iterations.

The project uses Bootstrap for the frontend, ensuring a responsive interface. Flask is chosen for the backend due to its lightweight nature, ease of use, and the team's familiarity, allowing rapid development. Django, though considered for its flexibility, was not selected due to its steep learning curve and longer development time.

SQLite3 is the chosen relational database for managing complex relationships and ensuring strong data integrity. Non-relational databases were excluded as they lack the necessary integrity constraints and advanced querying capabilities.

Risk Management

Risk Matrix

Risk	Likelihood (1-5)	Impact (1-5)	Priority (Likelihood * Impact)
Misunderstanding Project Scope	3	5	15
Scope Creep	4	4	16
Challenges in Back-End System Integration	4	5	20
Identified Skills Gaps in the Team	3	3	9
Unforeseen Force Majeure Events	2	5	10

Risk Mitigation Strategies

- **Understanding Scope**: Ensure all stakeholders have a clear and common understanding of the project scope. Conduct regular scope reviews and communicate any changes promptly to all team members.

- **Scope Creep**: Regularly review project scope and maintain clear documentation of all agreed-upon features and deliverables. Implement change control procedures to manage and assess any proposed changes.
- **Back-End Integration Issues**: Conduct early-stage integration testing and maintain clear documentation for API and database interactions. Engage in regular code reviews and pair programming to catch potential issues early.
- **Skills Gaps:** Provide targeted training sessions for team members, especially in areas like security and integration. Consider external consultation if needed.
- **Force Majeure:** Create a contingency plan with alternative timelines and resource allocations to minimise disruption. Ensure regular backups of all project data and code.

Preliminary Security Threat Modelling

Threat Analysis (STRIDE)

- **Spoofing**: User authentication will be secured to prevent unauthorised access.
- **Tampering**: Data integrity will be ensured through input validation and secure data transmission.
- **Repudiation**: Logging mechanisms will be implemented to track actions within the system, providing accountability.
- **Information Disclosure**: Sensitive information will be encrypted both in transit and at rest to protect against unauthorised access.
- **Denial of Service**: Rate limiting and monitoring will be used to protect the system from denial-of-service attacks.
- **Elevation of Privilege**: Role-based access control will ensure that users can only perform actions appropriate to their permissions.

Mitigation Strategies

Regular security reviews and audits will be conducted to identify and mitigate potential threats, while continuous monitoring and logging will be implemented to detect and respond to security incidents.

Project Plan

The project's timeline are detailed in the Gantt chart, which can be accessed in Appendix [6].

- **Milestones**: The next stages will include the completion of the MVP, further user testing, and refinement based on feedback.
- **Task Breakdown**: Tasks will be divided among team members, with clear deadlines and responsibilities assigned.
- **Documentation**: All meetings with the client have been documented, with feedback integrated into the development process.
- **Review and Approval**: The client has been engaged in reviewing the MVP and other project artifacts, with their feedback guiding further development.

Conclusion

This project proposal outlines the foundation for developing the “Unit Set Builder” application. It reflects careful consideration of client requirements, user needs, and technical challenges. The next steps will focus on finalising the MVP, refining the user experience, and ensuring that the system is robust, secure, and ready for deployment.

Appendix

[1] Team members:

- Zhanerken Azimbayev (23733728)
- Qianqian Li (23770748)
- Ankita Narvekar (23829237)
- Yuqiao Qian (24129392)
- Siting Xiang (23737625)
- Siyuan Zhou (23861658)

[2] The source code is available in the GitHub repository.

[3] For team communication and collaboration, please refer to our Teams channel.

[4] The UI design prototype exported from Figma is available in PDF format.

[5] The Entity-Relationship Diagram (ERD) can be accessed via this link.

[6] The Gantt chart illustrating the project timeline can be accessed here.