THE UNIVERSITY *of* ADELAIDE

Curriculum mapping tool

Team-07 Ruwen Yi Yuhao Wang Ying Liao Yingyu Wei
Supervisor: Claudia Szabo

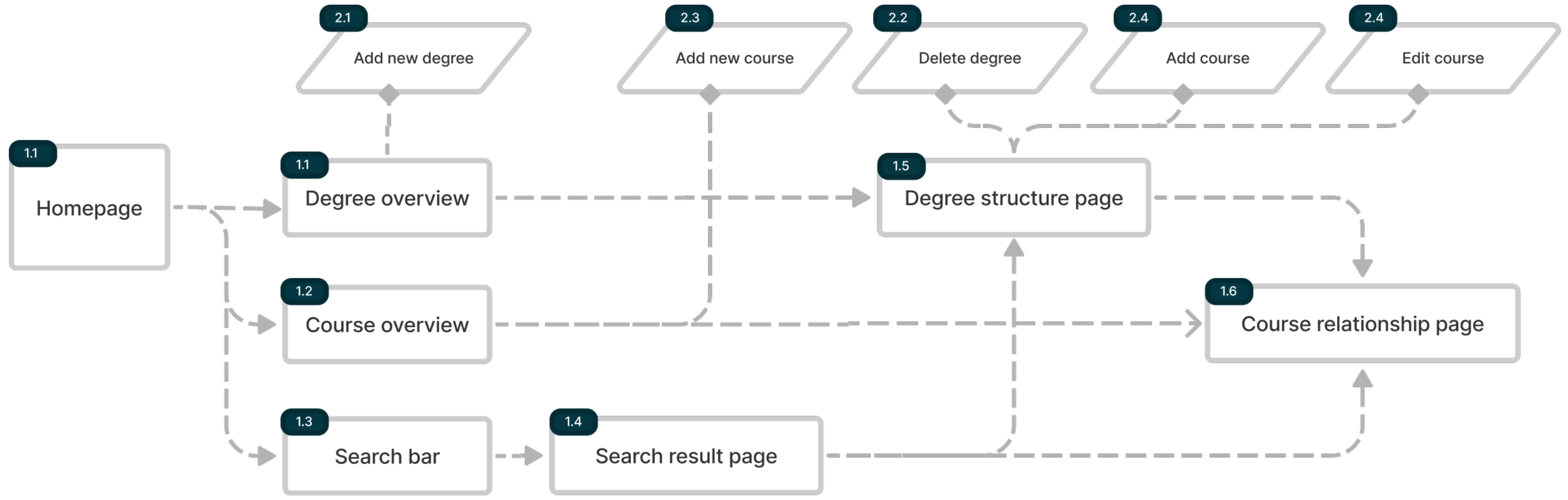
Introduction

It is difficult to coordinate hundreds of courses across multiple degrees, especially when the surrounding information is dispersed over multiple websites. **Streamlining** this process is this project's main objective.

The **data-set** consists of 7-degrees in the Faculty of Computer Science, spanning over 208 courses of the University of Adelaide.

Our project aims to develop a web-application for the supervisor. This will allow for: the presenting of degrees, the visualising of course connections, the identification of course prerequisites and the incompatibles, and the outlining of course structures.

User flow



Homepage

Degree overview - 7 degrees →Add degree
Course overview - 208 courses → Add new course
Search bar - Allow for the specific search of degrees and/or courses.

Degree structure page

Show courses in the stream or major of degree.
→ Delete stream or degree
→ Add, edit, delete course

Course relationship page

Display core information related to this course.

The web application consists of three main modules. Each of which supports the client's workflow and the decision-making process. Each module is integrated with the use of different functionalities; and they are all interconnected by a system of buttons and hyperlinks.

Achievements

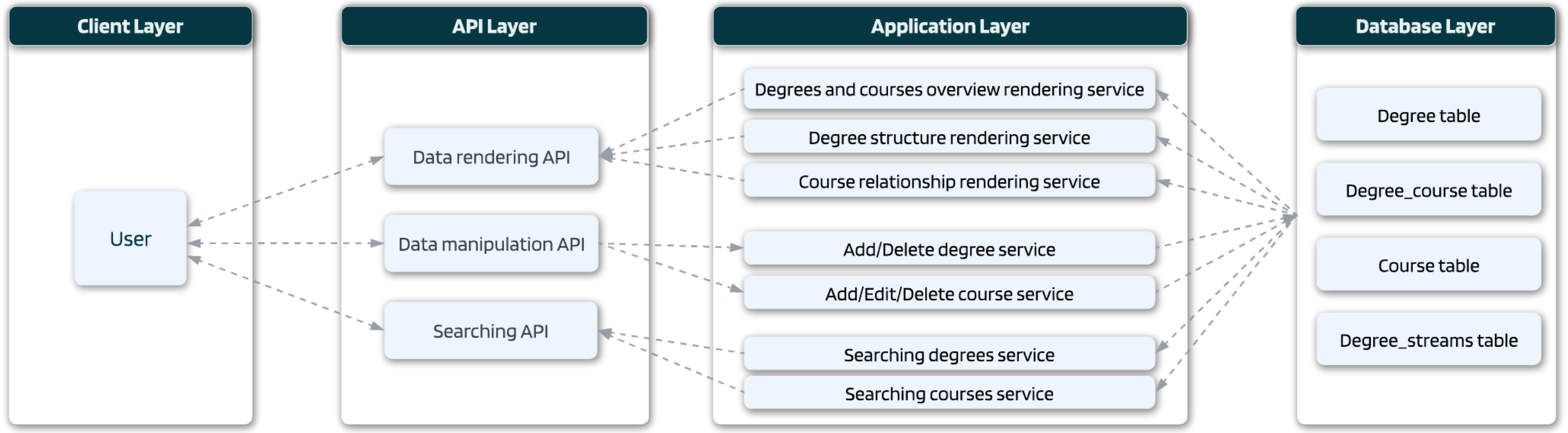
Milestone 1

- Design software architecture, user interface, and database.
- Construct the skeleton of the project using node.js express framework
- Build a database for the collected degree data and course data from the university system.
- Render the degree overview page on the server side with the use of ejs, and then send this to the client side via an API.
- Set up links to degree structure page

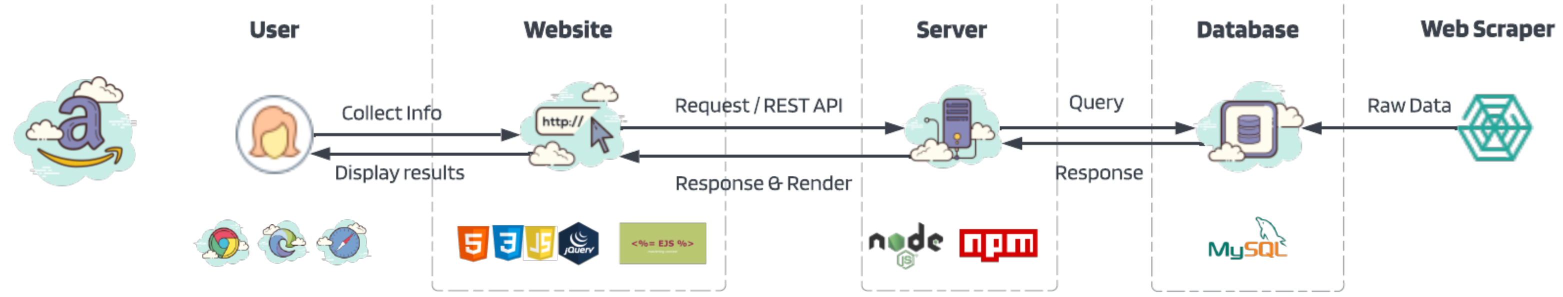
Milestone 2

- Render the course overview page, degree structure page, course relationship page on the server side with ejs and then send this over to the client side via an API.
- Implement searching function to request data via APIs and inject on the webpages dynamically
- Add, edit, delete courses
- Add, delete degrees

Software architecture


Client Layer
User
API Layer
Data rendering API
Data manipulation API
Searching API
Application Layer
Degrees and courses overview rendering service
Degree structure rendering service
Course relationship rendering service
Add/Delete degree service
Add/Edit/Delete course service
Searching degrees service
Searching courses service
Database Layer
Degree table
Degree_course table
Course table
Degree_streams table

Tools and data flow



Future work

· **Improve loading speed to allow for faster response and reload times.** Especially in circumstance which involve a larger amount of data.

· **Improve website security and performance.** We will set up firewall, use TLS encryption and access control.

· **Expand product market and customer base** to a wider range of factualities and universities.

· **Implement the course evaluation functionality,** so that students can give feedback on courses.

Conclusion

In general, we have fulfilled all the needs of our customers. Clients can quickly access any requested information through our website, and adapt it will be adapted to their needs. The amount of data is limited at the moment. However, in the future development we plan to optimise both the performance and the security of the application.