# 9900 Project Proposal

## Team CZHX

**DevOps Developer**

Tianpeng Chen

z5176343

z5176343@ad.unsw.edu.au

**Frontend Developer**

Yifan Zhao

z5127440

z5127440@ad.unsw.edu.au

**Backend Developer**

Wenxiao Xu

z5142629

z5142629@ad.unsw.edu.au

**Scrum Master & Backend Developer**

Minjie Huang

z5129528

z5129528@ad.unsw.edu.au

## Statement of the Problem

Currently, UNSW is using not only one-course information viewing system but more than four different kinds of systems. For instance, handbook, WebCMS, Ed, CSE website and other systems. Moreover, there is a lack of connection between these courses including showing the prerequisite and the excluded classes. The handbook is used to check some necessary information like course capacity, the unit of credit, course outline, the fee, the belonged faculty and school. The WebCMS, Ed, and CSE website, on the other hand, provides some additional information such as previous assignments, the forum, the Q&A section, and other details about the course.

The existing approaches can easily cause confused CSE students as when they need to enrol a course; they have to find information from multiple sources. Sometimes the process can be frustrated and too complicated.

On the other hand, tutors and lecturers do not have enough support to manipulate the data online. For example, some simple questions would be asked frequently during the semester. However, tutors have to repeat the answer again and again, which causes a massive waste of time and effort.

As a world-leading university, we firmly believe that a better solution can be, and should be found to improve user experience in many ways.

## Aim

It is more than necessary to implement a new student-tutor support system to overcome the confusing about isolated courses information and simplify the flow of understanding the courses to enrol. Combined with a chatbot, it can deliver a much more user-friendly experience. This new system should be able to help both students and staff in a specific course in different aspects. Moreover, we would like to design a Portable application that can be used in not only one course but every CSE course. As we design the universal API, the target course can be freely adjusted by clients in the future.

# Epics

As we decided to focus on the first project(X-o-Bot), we came up with several personas and user stories to help the process of our design. After some brainstorm meetings, finally we create a list of epics:

---

## - Course information (For students)

A considerable part of our application will be delivering the course information. As entering the application, the course information should be able to display by chatting with a chatbot. It should be gathered and cleaned up from multiple data sources including handbook, WebCMS, and other potential sources on the Internet. Additionally, it should remain the aesthetic design while displaying the information.

This epic has an estimated difficulty score of 6/10. An estimated scope score of 8/10. With an estimate implementation time of 2 weeks.

---

## - Knowledge enquiry (For students)

The other main feature of the application should be a knowledge enquiry system which relies on a knowledge base. This knowledge base should already include numbers of relevant questions (e.g. "what is the meaning of master theorem?") and glossary(e.g."TF-IDF"). If the students start to ask questions, chatbot needs to understand what kind of problem are they asking and present the correct/ relevant answers ideally.

After the question is answered, students have a chance to give feedback on this answer. If the student provides positive feedback, the weight of this answer will be more likely to appear in the next similar situation; if not, this question will be sent to the database and waited for a manual review from tutors and lecturers.

This epic has an estimated difficulty score of 8/10. An estimated scope score of 8/10. With an estimate implementation time of 4 weeks.

- **Voice input (For students)**

  The traditional chatbot can be helpful. However, it would be even more useful with mobile voice input. With voice input, students can conveniently use the application anywhere. This should be seen as an extra feature as the main feature should be able to work without voice input.

  This epic has an estimated difficulty score of 4/10. An estimated scope score of 6/10. With an estimate implementation time of 2 weeks.

- **Admin analysis&statistics dashboard (For staff)**

  The dashboard is a function for staff. This epic will significantly support tutors and lecturers by relieving them from answering simple and repeatedly-asked questions and giving them real insight into students' feedback. Only the enquires labelled with negative feedback will be shown in an unsolved question queue, giving staff the ability to focus on the actual valuable questions.

  Moreover, the statistic function can be incredibly helpful in terms of collecting common questions and popular difficulties. With the help of the new system, lecturers and tutors can check the statistics every week, and improve the lectures and tutorials dynamically.

  This epic has an estimated difficulty score of 7/10. An estimated scope score of 4/10. With an estimate implementation time of 3 weeks.

## Final Epic Selection

We shall try to implement and deliver all the epics mentioned above, according to the importance of each epic. However, epics that related to chatbot are more critical in this project. As a result, we would like to achieve those epics primarily.

## Design Standards

While we focus on delivering the features we mention, we should also notice that a splendid UI can contribute equally to the application. That is why we would like to make sure that when users launch our app, it should bring out the natural, user-friendly, minimalist interface. In this project, we would use Ant Design as it is not only easy to use but also convenient to set up the universal rules and style. In other words, it can offer the team working in the same design principles.

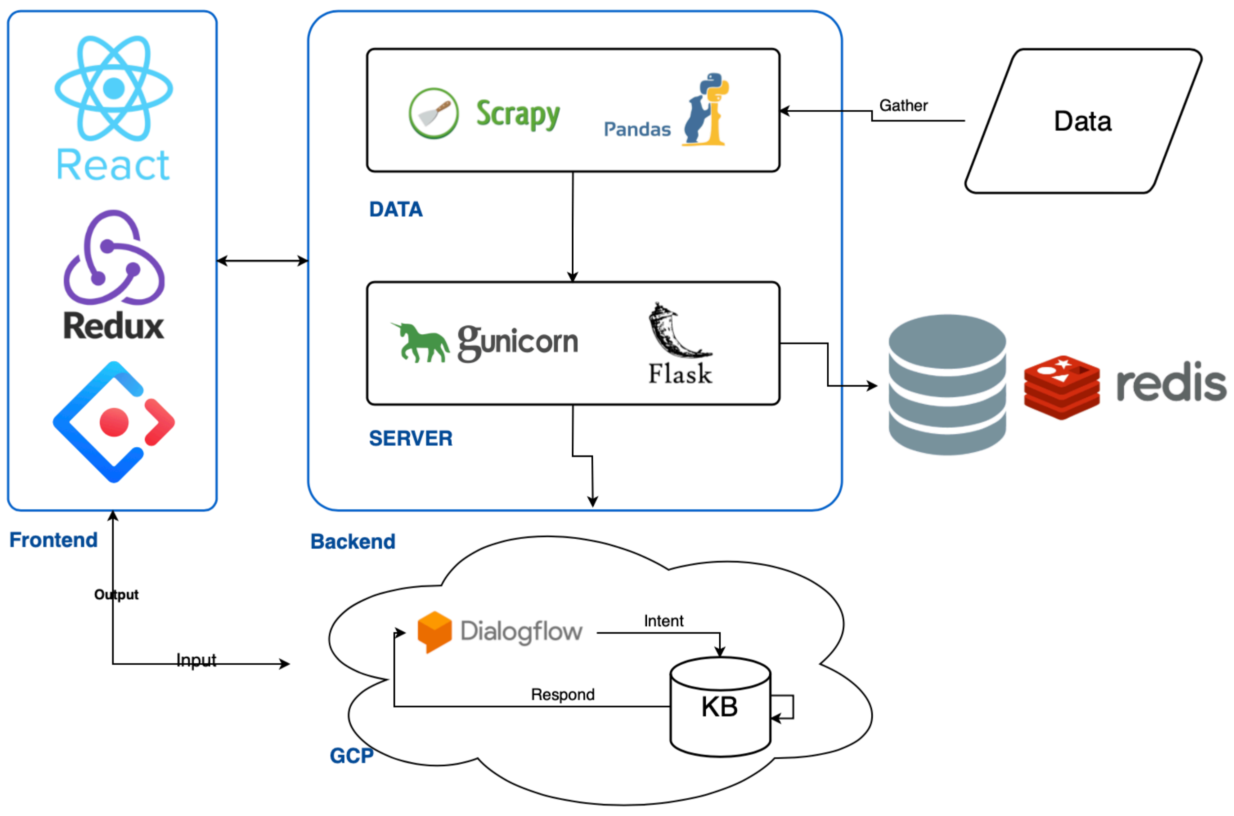
## Project Deliverables and Beneficiaries

At the end of the project, we hope to have a new web-based application, which integrates courses information from multiple resources and combines with a chatbot that can provide reliable feedback and answers to the online queries of students. Besides, this application should carry on the data from individual sources.

The new application should keep all of the current functionality from handbook and WebCMS, with a fantastic UI and more arranged information structure. Accessing this application should be able to help students grant information on CSE courses and relevant knowledge.

# Method and Technical Approach

## Architecture Design



## Technical Stack

*Our choice of techs was firmly based on our team member's experience and strength.*

- BACKEND - Python (Flask)

*With the majority of our members being proficient with Python and web service design, we chose Python as the primary development language for the backend and data gathering.*

- FRONTEND - Javascript (NodeJS, ReactJS, Ant Design)

*JavaScript is the most popular programming language for the frontend. With modern JS frameworks like ReactJS and UI libraries, it's much easier to build a user-friendly frontend.*

- MIDDLEWARE- Google DialogFlow

*Being the leading natural language conversation solution, Google DialogFlow supports both Python and JavaScript to integrate their API into our product. In this project, Google DialogFlow plays a crucial role as both the conversation builder and the knowledge base maintainer. Conversation Builder is the core module supporting our chatbot to chat as humans, while the knowledge base determines how to respond to a specific question.*

- DATABASE - Redis

*Redis is a common key-value in-memory database, which in our context can be utilised as a cache/storage to our knowledge base, as well as session storage to help us identify each user.*

## Other Technique and Practice

*DevOps has been a hot topic in these days, which suits perfectly with Agile development.*

- CircleCI Pipeline

*CircleCI is a CI/CD provider with the capability of being integrated to our Github repository, making every commit correct and deployed onto our Platform-as-a-service (PaaS) provider.*

- Pivotal Web Service

*Pivotal Web Service provides a cloud runtime for common web apps which is just fine for holding our frontend and backend server with its free trial.*

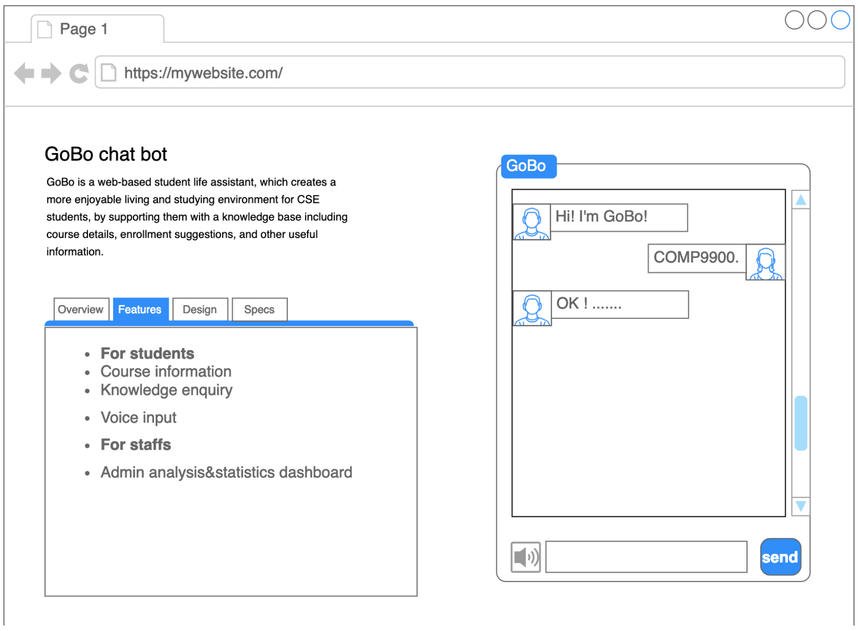
- Cloud Foundry

*Cloud Foundry offers a command-line-interface (CLI) utility making it much easier to deploy our software onto the cloud server.*

- Test-Driven-Development (Preferably Frontend)

*Test-Driven-Development is an evolutionary approach that enables developers to take small steps when coding, which can be combined with Agile development properly not only to enhance the code quality but to relief developers from building tedious documentation.*

# Product prototype

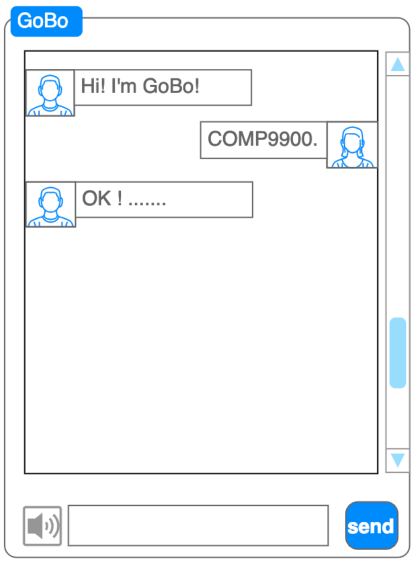


GoBo chat bot

GoBo is a web-based student life assistant, which creates a more enjoyable living and studying environment for CSE students, by supporting them with a knowledge base including course details, enrollment suggestions, and other useful information.

- **For students**
- Course information
- Knowledge enquiry
- Voice input
- **For staffs**
- Admin analysis&statistics dashboard

Desktop Layout



Mobile Layout

## Stand-up/Sprint Schedule

We plan to spend eight weeks on implementing this project. Before the actual implementation, we will spend the first three weeks on setting up the moving target, the technical direction, and the working environment. From the end of the third week, we will structure both frontend and backend at the same time, by two groups which contains two members in each of the group. The agile development will last 3-4 weeks before we wrap up everything we have, which will spend one more week. Lastly, we will work together for any incident that may appear during the developing period.

The last week will be used to ensure project running efficiency and aesthetics.

**Stand-up:**

- Saturday (Online)
- Monday (In person)
- Wednesday (In person)

**Sprints:**

- Week3-4
- Week5-6
- Week7-8
- Week9-10

**Weekly meeting time:**

Wednesday (PM 3:00 - 5:00)

Saturday (PM 3:00 - 5:00)

## Measurement

With the new system, there should be a huge reduce on courses query and server stress.