# PROJECT PROPOSAL

## Course Management Bot


UNSW
AUSTRALIA

COMP9323 17S2

Lei Zhao, z5047143@student.unsw.edu.au
Yijun Fang, z5061743@student.unsw.edu.au
Minhui Su, z5032791@student.unsw.edu.au
Yuanjie Yang, z5032760@student.unsw.edu.au

The aim of this report is to provide an brief introduction to our project, a course management bot with respect to the researches and investigations we have made so far for. Detailed information will include feature analysis and our design of how the system will be implemented.

# Part1.

## a) Requirements and features analysis:

Our project, which is a course management bot, aim to offer the following functionalities:

Lectures and Mentors will be able to:

- check/upload lecture materials and other resources
- send email to students interactively or with template of different purposes
- check/set/delete reminders for himself or students for upcoming events
- check/set/modify assignment information and mark assignments with the bot interactively

Students will be able to:

- check lecture materials and other resources
- send emails to lecturer/mentors interactively
- view/delete/set reminders for user himself for upcoming events
- check/submit assignment

## b) Use-case

### Feature: Sending E-mail

- Use case 1: Send email interactively

| | |
|---|---|
| User | Teacher and student |
| Workflow | - User sends a request to send email<br>- Bot asks who the receiver(s) is/are<br>- User replies receiver email address<br>- Bot asks what the subject is<br>- User replies subject's content<br>- Bot asks what the body is<br>- User replies body's content |
| Precondition | User needs to login |
| Result | Email sent |

- Use case 2: Send email with template

| | |
|---|---|
| User | Teacher |
| Workflow | - User sends a request to send email to somebody with condition(ie. sick)<br>- Bot gives a corresponding email templates for user to confirm |

| | |
|---|---|
| | - User confirms and sends or cancels |
| Precondition | User needs to login |
| Result | Email sent |

## Feature: Class resource

- Use case1: check class resource

| | |
|---|---|
| User | Teacher or student |
| Workflow | - User sends a request to check resources of class with the key word(e.g. Resources name)<br>- Bot lists all matched resources |
| Precondition | User needs to login |
| Result | Get required resources |

- Use case2: upload class resource

| | |
|---|---|
| User | Teacher |
| Workflow | - User sends a request to upload resources<br>- Bot asks the destination<br>- User uploads the file<br>- Bot confirms with user if the destination is correct<br>- User confirms or cancel |
| Precondition | User needs to login |
| Result | Resource is uploaded |
| Postcondition | Resource file is saved and path to it is inserted into database |

## Feature: Reminder

- Use case 1: List reminders

| | |
|---|---|
| User | Teacher and student |
| Workflow | - User sends a request to check the reminder with keywords (e.g. Assignment 1)<br>- Bot replies with the list with reminders |
| Precondition | User needs to login |
| Result | Reminder information is returned to the user |

- Use case 2: delete reminders

| | |
|---|---|
| User | Teacher and student |
| Workflow | - User sends a request to check reminder<br>- Bot replies with the list with reminders<br>- User can delete one of them. |
| Precondition | User needs to login<br>At least one reminder |
| Result | Reminder info is returned to the user |
| Postcondition | Information of this reminder is removed from the database |

- Use case 3: Set reminder

| | |
|---|---|
| User | Teacher and student |
| Workflow | - User sends a request to set reminder<br>- Bot asks user to input reminding time<br>- User sends the reminding time<br>- Bot asks user to input reminder content<br>- (email sent part) |
| Precondition | User needs to login |
| Result | Reminder set |
| Postcondition | The detailed information of reminder is inserted into the database |

- Use case 4: Send emails to remind students who have not submitted the assignment

| | |
|---|---|
| User | Teacher |
| Workflow | - User sends a request to send email to remind students who hasn't submitted the assignment.<br>- Bot replies "which assignment?"<br>- User enters the assignment name. |
| Precondition | The user needs to log in . |
| Result | Emails are sent to those who have not submitted the specific assignment. |

## Feature: Assignment

- Use case 1: check assignments resources

| | |
|---|---|
| User | Teacher and student |

| | |
|---|---|
| Workflow | - User sends a request to check assignment with keywords like the assignment title<br>- Bot replies with the list of matched assignments resources. |
| Precondition | User needs to login |
| Result | Users get required assignment resources |

- Use case 2: Set an assignment

| | |
|---|---|
| User | Teacher |
| Workflow | - User sends a request to set assignment with the title<br>- Bot asks the title of the assignment only if title not given last before<br>- User replies title<br>- Bot asks due date of the assignment<br>- User sets the due date of the assignment<br>- Bot asks the user to upload specification<br>- User uploads description for the assignment<br>- Bot ask what are the files student can upload including file name and type.<br>- User reply a set of file names with suffix<br>- Bot will preview all the information //confirms with the user for all assignment information<br>- User confirms information or cancel<br>- If user confirms, bot will ask if send email to students.<br>- User can reply yes to inform students, no for not.<br>- The email will be sent to students with all the information about the assignment.<br>- The conversation ends. |
| Precondition | User needs to login |
| Result | Due date and specification of the assignment are set and could emails to all students to inform them. |
| Postcondition | The detailed information of assignment is inserted into the database |

- Use case 3: Modify assignment information

| | |
|---|---|
| User | Teacher |
| Workflow | - User sends a request to modify certain assignment<br>- User replies due date or specification<br>- Bot asks for new information<br>- User updates information<br>- Bot confirms with user<br>- User confirms information |

| | |
|---|---|
| Precondition | User needs to login |
| Result | Due date and specification of the assignment are modified and emails will be sent to all students to inform them. |
| Postcondition | The detailed information of assignment in the database is modified |

- Use case 4: Mark assignment

| | |
|---|---|
| User | Teacher |
| Workflow | - User sends a request to mark certain assignment<br>- Bot returns an unmarked assignment and ask for result mark<br>- User inputs the mark for that assignment<br>- Bot confirms with user<br>- User confirms information |
| Precondition | User needs to login |
| Result | Marks of the assignment are set and emails will be sent to related students. |
| Postcondition | The mark of assignment is inserted into the database |

- Use case 5: Submit assignment

| | |
|---|---|
| User | Student |
| Workflow | - User sends a request to submit certain assignment<br>- User upload a file<br>- Bot confirms with user<br>- User confirms information |
| Precondition | User needs to login |
| Result | Mark of the assignment is set and emails will be sent to related students. |
| Postcondition | Assignment file is saved and path to it is inserted into database |

## Feature: Invalid input handling

- Use case 1: User types the wrong information and the bot cannot understand

| | |
|---|---|
| User | Teacher and student |
| Workflow | - User sends wrong information<br>- The bot goes back to the beginning with the word”Hello” |

| | |
|---|---|
| Precondition | User needs to login |
| Result | The conversation continues in the previous state |

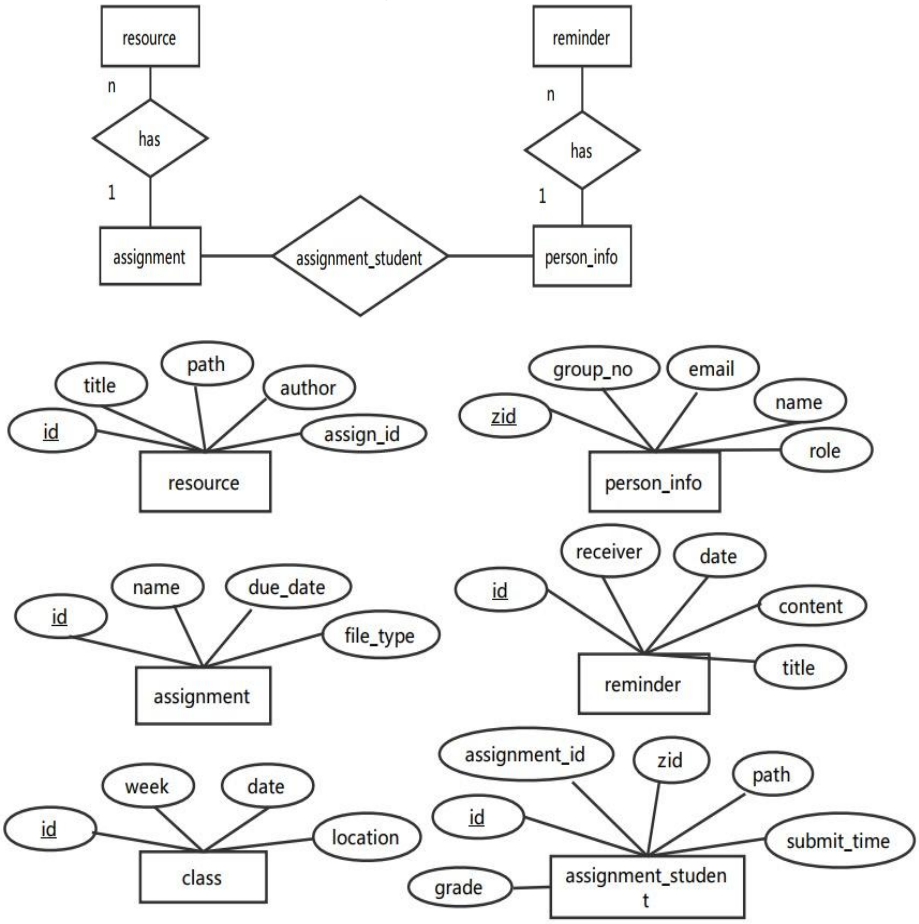
- Use case 2: User types “exit” or leaves the page without finishing the conversation

| | |
|---|---|
| User | Teacher and student |
| Workflow | - The conversation finishes and no results will be returned |
| Precondition | User needs to login |
| Result | The conversation goes back to the beginning state |

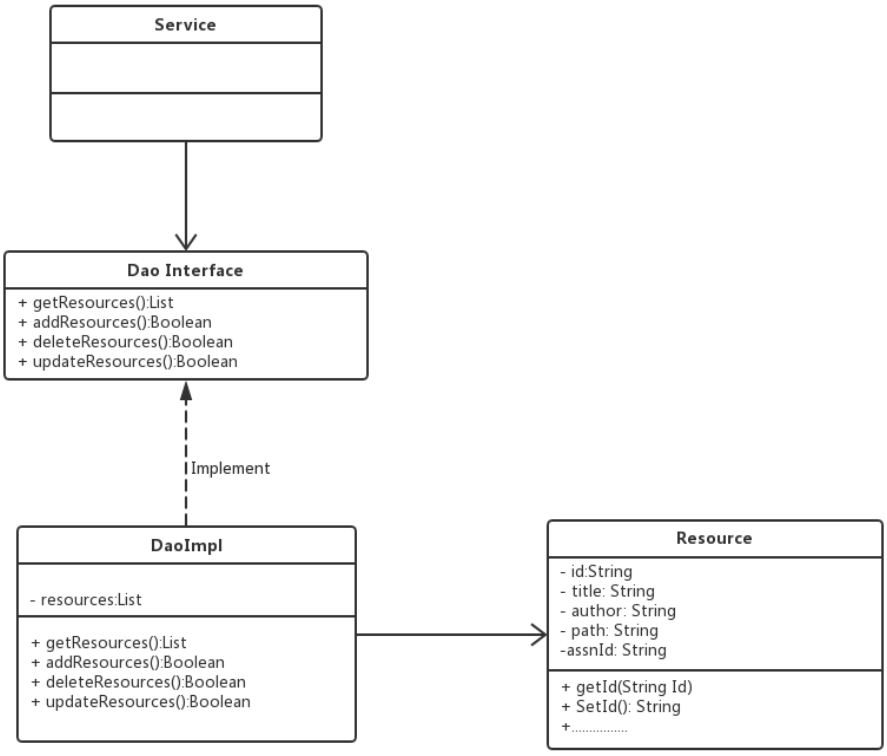
# Part 2. System Design

In terms of the front-end, third party instant message software such as Telegram, Facebook Messages and Slack will be used to interact with the user. For the backend, Java Spring will be used to receive request from the user and send requests to the API.AI server to retrieve the intent and parameters. After that, we will use these intent to call the responding functions with parameters and exchange data with database. For the database part, MySQL will be used. In order to interact with the database, we will use the Java MyBatis library.

For the database part, the following is the ER model



The following is the UML diagram:




A simple system architectural diagram can be found below: