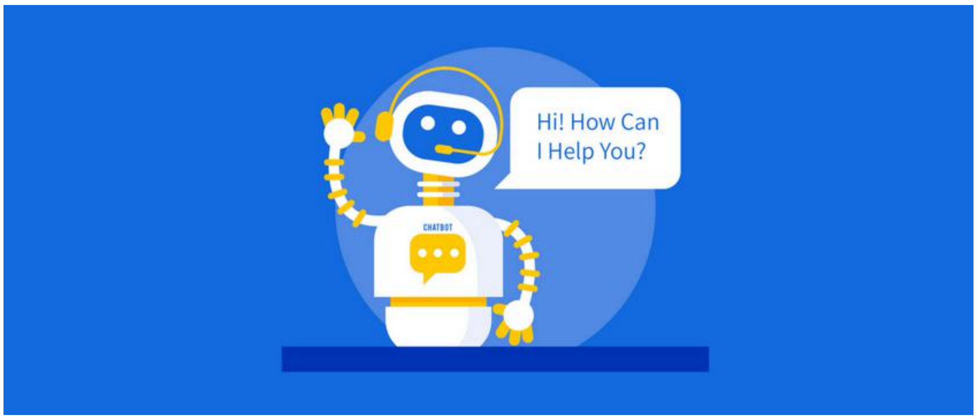
Report

COMP9900, 2019s3

X-o-Bot: A dialogic framework for Conversational agents



Team: Salted Fish
Team member
Dong Zhu: z5165957
(Scrum Master &Full stack developer)
Email: z5165957@ad.unsw.edu.au

Jinpeng Jiao: z5143932
Full stack developer
Email: z5143932@ad.unsw.edu.au

Bohan Zhao: z5141730
Full stack developer
Email: z5141730@ad.unsw.edu.au

Yaqi Yang: z5143675
Full stack developer
Email: z5143675@ad.unsw.edu.au

contents

1. Introduction

1.1 Chatbot development history

The application of intelligent dialogue to the field of education was tried as early as 1970 by designing SCHOLAR tutors, but natural language processing became one of its biggest challenges at the time [1].

Earlier versions of AutoTutor responded to student questions with an estimate of what the system knew about the students. AutoTutor focuses on the subject matter of the conversation and the message passed between the tutor and the student.

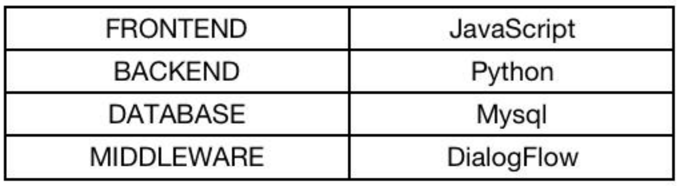
In the last 15 years, the application of conversational agent (CA) technology in automatic tuition has been extensively studied [2]. Due to the dynamic nature of CA's dialogue, it offers the flexibility of supporting students, using feedback to scaffold their process in a personalized way, specifically where it is needed. In dealing with some conversations of students, the conversation agent can learn and explain to the students during the conversation. At present, not only in the field of education, but also in many other fields, conversational agents have played a pivotal role and performed well.

1.2 The importance of conversational agents

Session agents have important applications in many areas.It is a tool to reduce the cost of customer service. For a long time, the customer service department is the cost centre of financial institutions. The larger the organization, the larger the number of customer service personnel required by the organization. The number of problems that customer service needs to deal with changes with the organization's marketing activities, so the number of customer service personnel required by the organization is also very flexible. How to quickly recruit and demobilize customer service poses a challenge. With the advent of conversational agent technology, intelligent agent can not only replace the manual customer service to response, but also handle large-volume transactions anytime, anywhere.

In the field of education, the way students ask teachers about their problems is e-mail and website generally, but when the lecturer is busy or forgets to process the e-mail, the reply time of the mail is often quite long. And if student asks a question similar to what the previous student posted, the time to wait for a response is not worth it. Conversational agent can solve this problem well. For example, the course assistant can help student to search the Q&A database of the course. If there is a known question, it will return the answer directly. In addition, the course assistants provide great convenience in course information inquiries.

1.3 Technical Details



FRONTEND	JavaScript
BACKEND	Python
DATABASE	Mysql
MIDDLEWARE	DialogFlow

Figure 1. Technical method form

In Figure 1 we present the components and the tools for each part of the code. For the development or front-end and back-end, we decided to compile in JavaScript and Python, respectively. JavaScript is the most popular front-end programming language, and unlike other programming languages, Python performs very well in deep learning.
For the database, we chose the most common language MySQL.
For the agent development we used Dialogflow. Dialogfow is a Google service that runs on the Google Cloud Platform; it is Built on Google infrastructure and incorporates Google's machine learning expertise and products its natural language processing (NLP) is very efficient.

1.4 Course assistant compared to single course website

According to our random survey of several students, students in the course website in addition to downloading the necessary courses’ materials, the most is to post questions on the course website and search for information related to the course.
Compared with the single course web page, the convenience of the course assistants we develop is not only reflected in the quick search and post the questions, but also in the query of the course information. The chat bot contains all the course information and course pages about the subject of Information Technology, which can replace the traditional clicks and dozens of clicks. In addition, teachers can view all unresolved questions and respond quickly on the course assistant client.
In addition, for a single course website, if the number of comments on the course website is very large, it is quite difficult to find the exact question from all the comments, especially if there is no search function. Therefore, for those students who need to frequently post questions and communicate frequently, the course assistant is undoubtedly a very good choice, simplifying the uploading problem and intelligentizing the search question. It is a real little assistant.

2. Requirements

2.1 Functional Requirements

A chatbot used to help students and students in the course and reduce the information used by students or teachers to query various course information.
The primary purpose is to provide information with the target and answer the user's questions. The primary purpose of this is to save the search time for students, lecturers, and tutors.

- Student:

  Students can use the system to view the necessary information about the course. The information comes from the school's handbook, but it can effectively save the students' search time. You can get most of the required information directly by asking the robot questions.
  Besides, after students choose their courses through the system, they can ask the robots to get answers from tutors or lecturers. If this is a new unanswered question, the robot can save the questions and wait for answers. After being answered, the student can view the answer.

- Staff:

  Staff can use the system to view the necessary information about the course. The information comes from the school's handbook, but it can effectively save the staff search time. You can get most of the required information directly by asking the robot questions so that the staff can understand the course arrangement more directly.
  Besides, after the staff selects the courses that the staff is responsible for teaching through the system, staff can ask the robot for a new unanswered question list, the staff can answer the question to the robot. The robot can save the problem, and reduce the staff between the multiple course websites and the time to switch between multiple problem pages.

2.2 Features

Our course assistant can answer these questions:
1: About the course information on the handbook:
    (1) The name of the course:
    For example: Tell me the name of COMP9900
    (2) Course details:
    For example: Tell me detail of COMP9517 and COMP9444
    (3) Semester of the course:
    For example: Which semester offers COMP9900
    (4) Lecturer of the course:
    For example: Can I know the professor of COMP9900 of term 1
    For example: Who is the teacher of comp6714
2: About the course recommendation:

(1) Courses can be recommended based on semester or knowledge:

For example: Could you please tell me some courses about deep learning?

For example: Could you please recommend some courses related to 3D games during T1?

3: About Q&A:

(1) Answer a question:

i. Start a question from ‘I want to ask a question’

ii. Or ask a question directly and ask questions based on the chatbot's response.

(2) Upload the question and wait for the teacher to answer

4: Uploading pictures

Ask a questions by uploading pictures.

5: Staff can look up question asked by student:

For example: Show me the question list

6: Choosing the questions to answer:

For example: Answer the question Q5678.

7: Answering the question:

For example: Type the answers to chatbot after choosing questions,the answers will saved to database and shown on student systems.

2.3 Interface

Interface is the interaction between data and users, and it is an indispensable part of every application. Students and teachers will interactions with the functionality of the interface to use the Chatbot application.

3. Implementation

3.1 Data

3.1.1 Data Preparation

Course information is one of the most important parts of the course assistant's data. In order to make the course information of our server more comprehensive and more specific, we have taken all the information technology courses from UNSW's handbook official website, including undergraduate and master's courses.

In addition, for each student, we have set up all students to enroll in 3 courses. In order to test, we selected COMP9321, COMP9021, COMP9900 three courses as a test course, and we crawled all the questions and answers about the relevant courses from the relevant course website, and finally imported these questions and answers into the database of the course assistant.

On the approximate matching of the statements, we selected a training set of statement relevance from Kaggle's dataset website [3]. This training set will be applied to the construction of the statement approximate matching model.

3.1.2 Data Processing

In the dataset of the course information, we included the course name, course ID and introduction obtained from the source code of the website using Python crawling technology. In addition, for the detailed page of each course introduction, we selected the course UOC, School, Faculty, and Offering_terms, stuff_names. If the student still needs to know the timetable for the course, our system can also provide a timetable link of the course, which the user can click on to access the timetable page of the course.

Moreover, for the introduction of the course, we extracted the keywords of the course. For example, in the course mainly deals with the knowledge of machine learning and artificial intelligence we will add the labels of machine learning and artificial intelligence to this course. When the user asks "Please tell me more about the artificial intelligence course", the course assistant can also query the course with the artificial intelligence tag from the known tag category to return the course information to the user.

For the question and answer data set, we first deleted the question and answer answered by the teacher. Secondly, for some questions on the course website, if the question is too long (more than 100 words), the length of the sentence used for question matching will be truncated, which means that only the first 100 words are matched.

Finally, process the problem dataset obtained from the Kaggle website. The first two columns of the dataset are selected as valid datasets. In the approximate matching of the sentences, we selected 5000 questions and processed the sentences into words, and used these words to construct the bag of words model (BOW) for statement processing.

3.1.3 Database

Our course assistant uses MySQL as our database, because MySQL is a database with excellent performance and stable and autonomous. Putting the extracted data into the database for

anytime-use. And we also can add, delete, and modify data at any time, which is convenient and quick.
MySQL is a high performance, stable, and autonomous database, so we use MySQL as a database. To meet the needs of multiple developers to use the database for development, we use the Amazon Relational Database Service (Amazon RDS) to build our database. We have established the following 10 forms to implement the functionality of our database(see Figure 2).

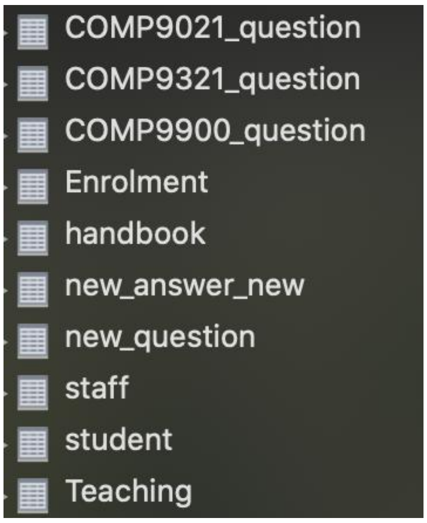

COMP9021_question
COMP9321_question
COMP9900_question
Enrolment
handbook
new_answer_new
new_question
staff
student
Teaching

Figure 2 database table list

The relationship between the database tables can be expressed using the following figure(see Figure 3).

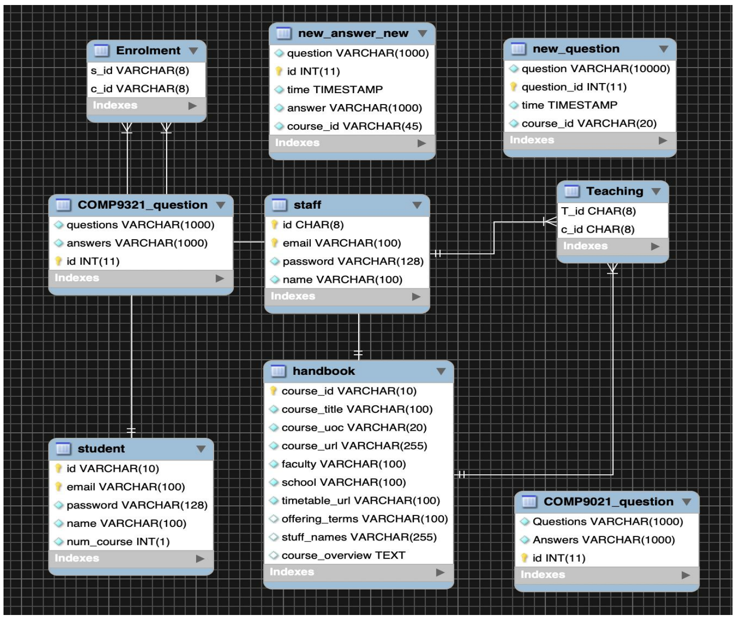


Figure 3 relationship between the database tables

The 'handbook' table stores data from the school handbook website, obtained through web crawlers. This table contains all the necessary information about the course. Various query functions for basic course information have been design in our assistant.
The 'Student' table and 'Staff' table record user information registered by Student and Staff, and user authentication.
The relationship between the student and the course is a many-to-many relationship like n:m, the

database table 'Enrollment' is dedicated to storing this relationship.
The relationship between the staff and the course is a many-to-many relationship like n:m, the database table 'Teaching' is dedicated to storing this relationship.
New_question and new_answer are used to store the student's new question and the teacher's new answer and include a time stamp.
The other three database tables starting with the course ID as the table name, are used to store the Q&A data compiled from the Q&A section of the course website.

3.2 System Architecture

3.2.1 System Architecture

Our chat bots are only deployed on the web, and any user can access it using a browser on any device on any platform(see Figure 4).
Before the user input reaches the backend, there will be a dialog stream for the first process, which will be passed to the backend and then processed by the backend. The backend includes a dialog flow module, information acquisition and query, and includes the ability to approximate the problem. After the problem is solved, the information is returned and stored for the next query or used by more users.
The database (MySql) is connected to the system's unified interface and built on AWS_RDS.

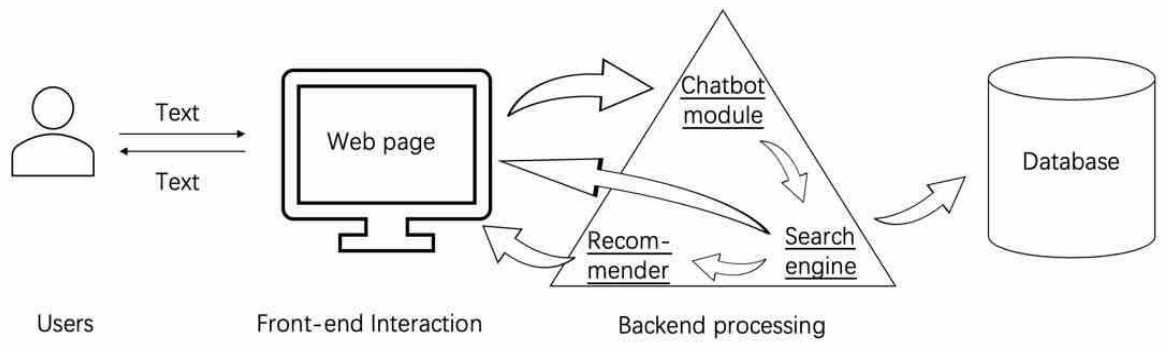


Figure 4 system Architecture

3.2.2 Work Flow Chart

The user has the registration, log in, and course selection switching functions. Each interface contains different functions. For the chat-bot, different interfaces can answer different questions. Figure 4 shows the work flow of our assistant:

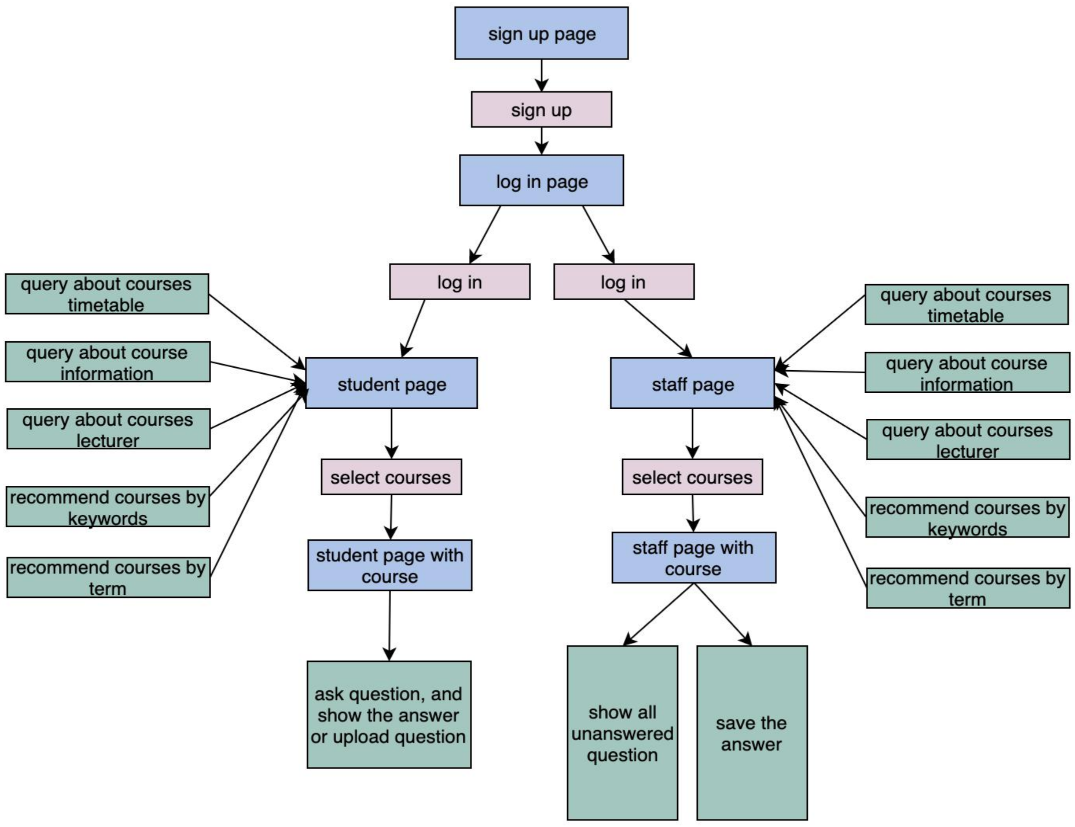

sign up page
sign up
log in page
log in
log in
student page
staff page
query about courses timetable
query about course information
query about courses lecturer
recommend courses by keywords
recommend courses by term
select courses
student page with course
staff page with course
ask question, and show the answer or upload question
show all unanswered question
save the answer

Figure 4 Work Flow

3.3 Function implementation

This part consists of all implementation details of the system. The course chatbot is divided into two parts, for student and staff, it will show below.

3.3.1 Function for student

(1) Image recognize:

Due to a large group of people prefer to upload images contained question, our team added this feature by using PIL module. It can recognize texts on pictures and use it as a query to our chatbot. And in order to satisfy this requirement, chatbot can recognize pictures that contains query words. For example, you can upload screenshot you caught from websites see the Figure 5.1.

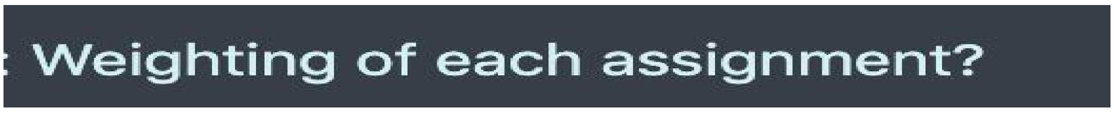
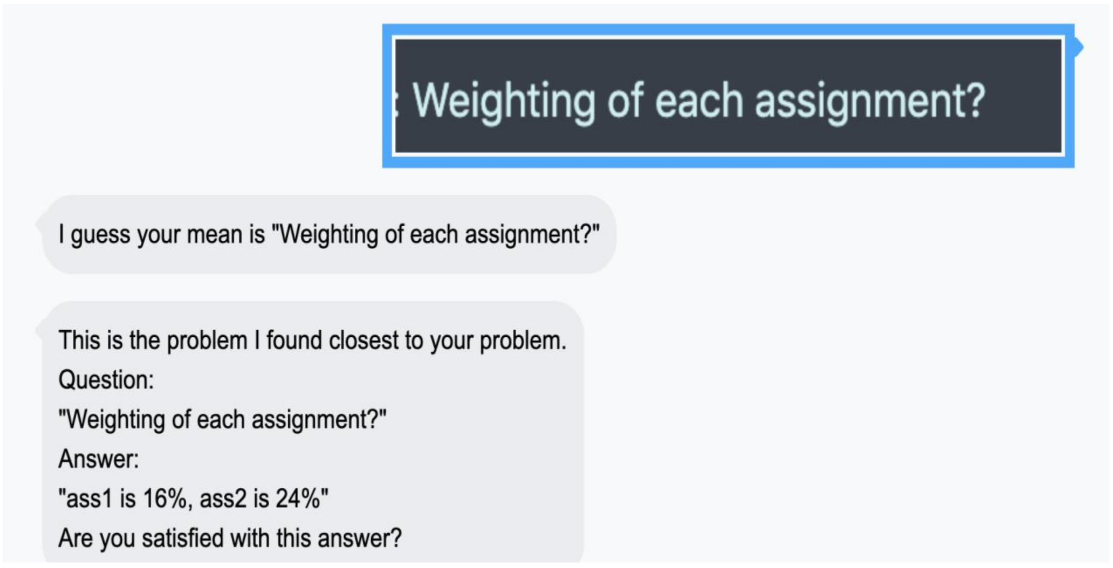

: Weighting of each assignment?

Figure 5.1 screenshot

And our chatbot can response like Figure 5.2 shows:

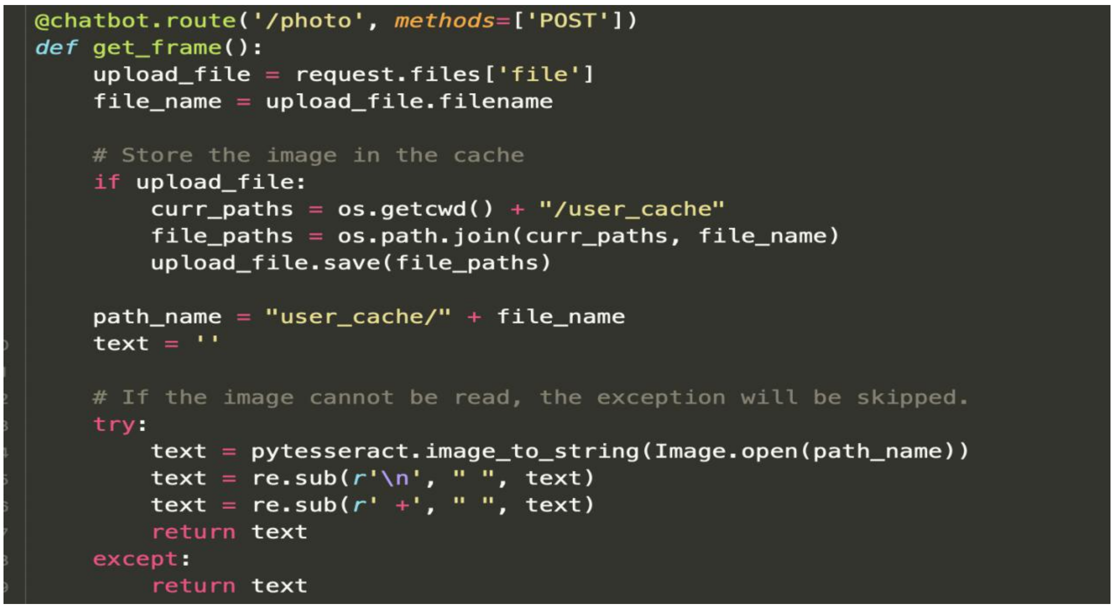

Weighting of each assignment?

I guess your mean is "Weighting of each assignment?"

This is the problem I found closest to your problem.
Question:
"Weighting of each assignment?"
Answer:
"ass1 is 16%, ass2 is 24%"
Are you satisfied with this answer?

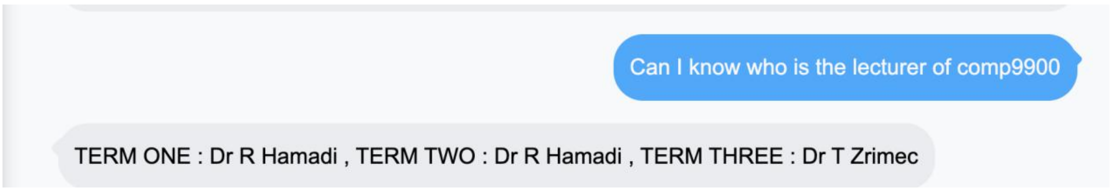
Figure 5.2 image recognize

And we implement PIL module to achieve this see the Figure 5.3.


```
@chatbot.route('/photo', methods=['POST'])
def get_frame():
    upload_file = request.files['file']
    file_name = upload_file.filename

    # Store the image in the cache
    if upload_file:
        curr_paths = os.getcwd() + "/user_cache"
        file_paths = os.path.join(curr_paths, file_name)
        upload_file.save(file_paths)

    path_name = "user_cache/" + file_name
    text = ''

    # If the image cannot be read, the exception will be skipped.
    try:
        text = pytesseract.image_to_string(Image.open(path_name))
        text = re.sub(r'\n', " ", text)
        text = re.sub(r' +', " ", text)
        return text
    except:
        return text
```
Figure 5.3 screenshot

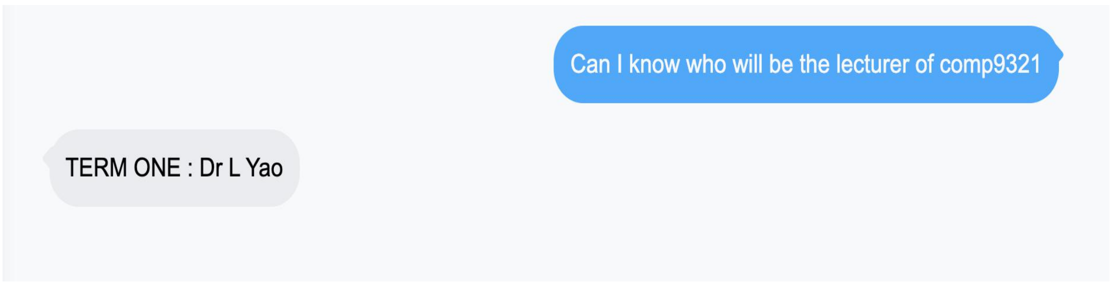
(2) Course information part:

(1) Course lecturer or course admin searching
Our chatbot can answer questions like: Can I know who is the lecturer of comp9900 see the figure 6.1.The system also allows to change the query,see the example in Figure 6.2.


Can I know who is the lecturer of comp9900

TERM ONE : Dr R Hamadi , TERM TWO : Dr R Hamadi , TERM THREE : Dr T Zrimec

Figure 6.1

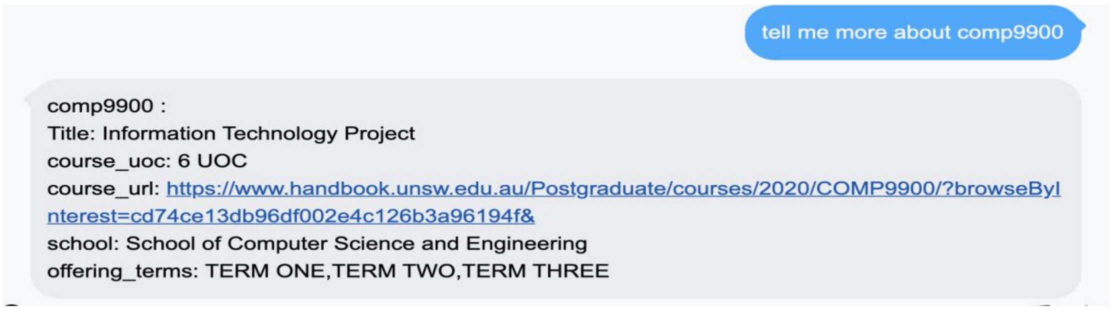

Can I know who will be the lecturer of comp9321

TERM ONE : Dr L Yao

Figure 6.2

(2)Course information searching

Our chatbot can return the whole course information for users, if user ask questions such like: tell me more about comp9900.It will show out the detail of the course you searched see the Figure 6.3.

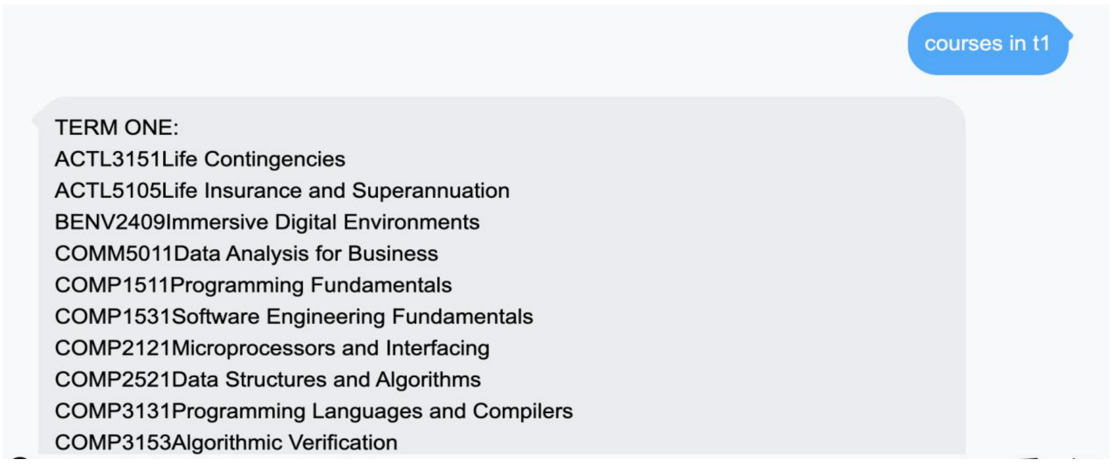

tell me more about comp9900

comp9900 :
Title: Information Technology Project
course_uoc: 6 UOC
course_url: https://www.handbook.unsw.edu.au/Postgraduate/courses/2020/COMP9900/?browseByInterest=cd74ce13db96df002e4c126b3a96194f&
school: School of Computer Science and Engineering
offering_terms: TERM ONE,TERM TWO,TERM THREE

Figure 6.3.

(3)Which course we can choose in each term

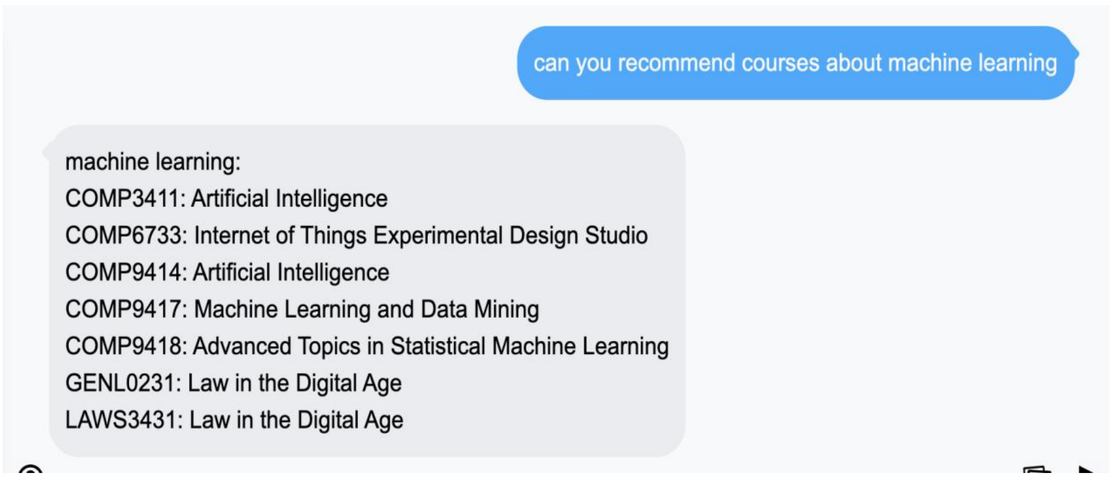
Our chatbot can provide course lists offered in each semester for users,see the Figure 6.4.


courses in t1

TERM ONE:
ACTL3151Life Contingencies
ACTL5105Life Insurance and Superannuation
BENV2409Immersive Digital Environments
COMM5011Data Analysis for Business
COMP1511Programming Fundamentals
COMP1531Software Engineering Fundamentals
COMP2121Microprocessors and Interfacing
COMP2521Data Structures and Algorithms
COMP3131Programming Languages and Compilers
COMP3153Algorithmic Verification

Figure 6.3.

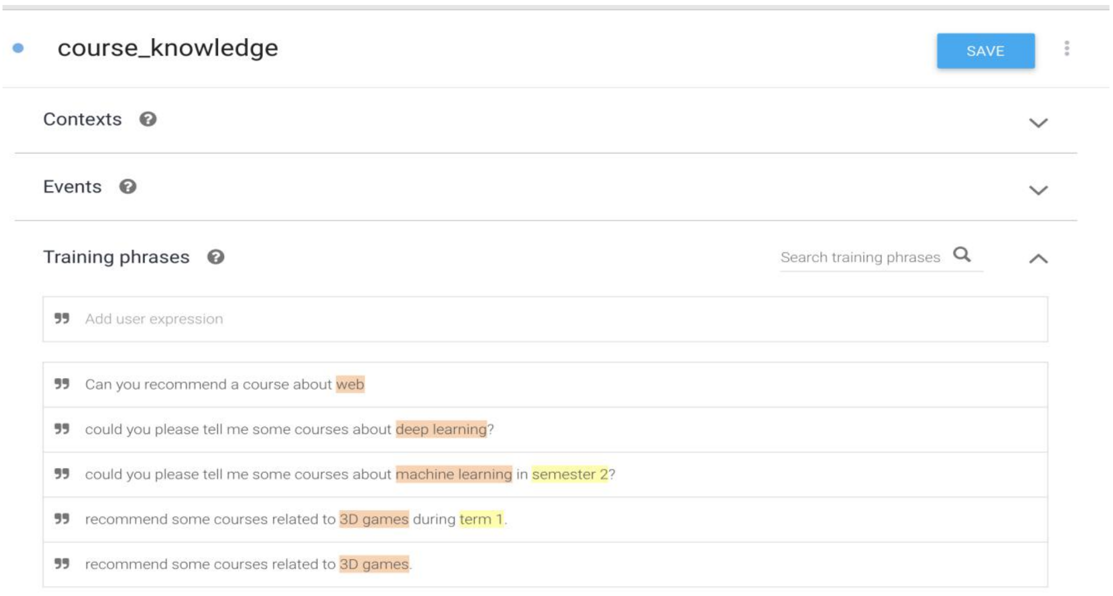
(4)Course recommendation
Our chatbot can recommend courses for users according to their insterststs.
It is not simply matching the key words on course title, our bot will check each description of courses and select courses including knowledge point you asked,see the Figure 6.4.


can you recommend courses about machine learning

machine learning:
COMP3411: Artificial Intelligence
COMP6733: Internet of Things Experimental Design Studio
COMP9414: Artificial Intelligence
COMP9417: Machine Learning and Data Mining
COMP9418: Advanced Topics in Statistical Machine Learning
GENL0231: Law in the Digital Age
LAWS3431: Law in the Digital Age

Figure 6.4.

For course part, mainly combined with dialogflow and database, using dialogflow as a platform linking fronted and backend part extracting useful information for our backend and we can process user query.
For example, course recommendation part, in dialogflow, we input training query and set key words,see the Figure 6.5.And our backend can search in database based on key values,shown on Figure 6.6.


course_knowledge

SAVE

Contexts

Events

Training phrases

Search training phrases

Add user expression

Can you recommend a course about web

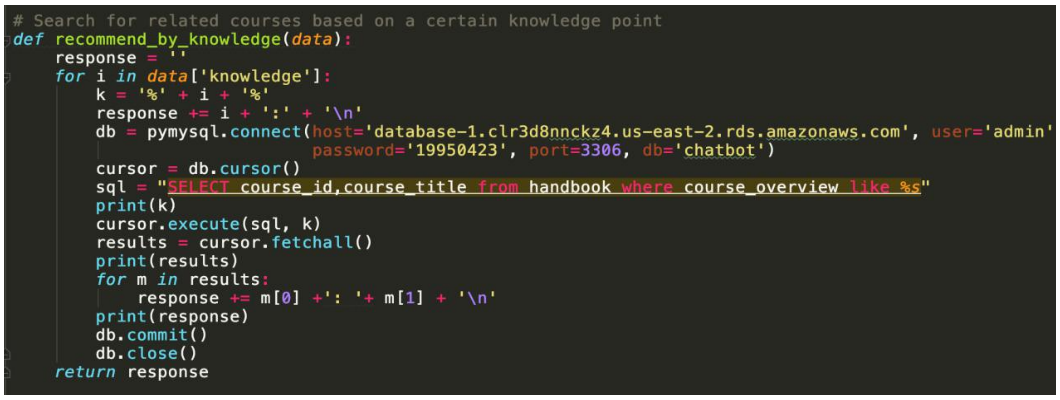
could you please tell me some courses about deep learning?

could you please tell me some courses about machine learning in semester 2?

recommend some courses related to 3D games during term 1.

recommend some courses related to 3D games.

Figure 6.5.

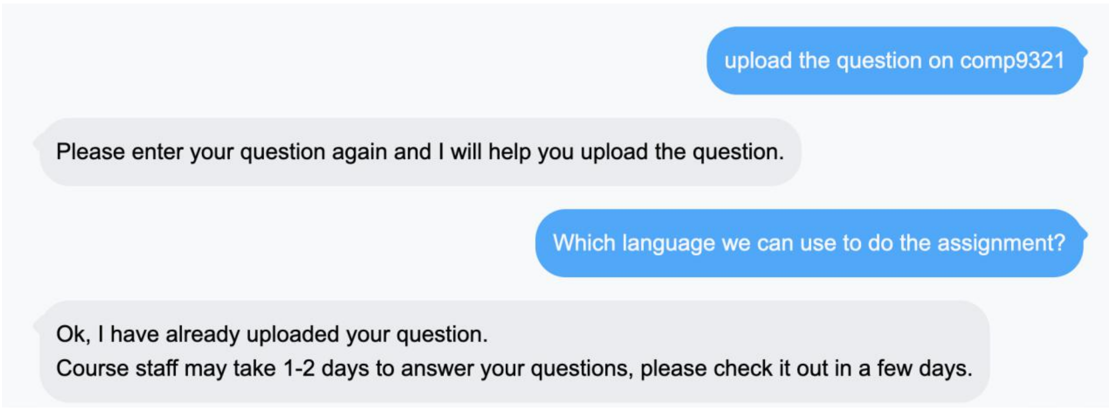


```python
# Search for related courses based on a certain knowledge point
def recommend_by_knowledge(data):
    response = ''
    for i in data['knowledge']:
        k = '%' + i + '%'
        response += i + ':' + '\n'
        db = pymysql.connect(host='database-1.clr3d8nnckz4.us-east-2.rds.amazonaws.com', user='admin'
                             password='19950423', port=3306, db='chatbot')
        cursor = db.cursor()
        sql = "SELECT course_id,course_title from handbook where course_overview like %s"
        print(k)
        cursor.execute(sql, k)
        results = cursor.fetchall()
        print(results)
        for m in results:
            response += m[0] +': '+ m[1] + '\n'
        print(response)
        db.commit()
        db.close()
    return response
```

Figure 6.6.

(3)Q&A part

For this part, students can ask questions for chatbot, and chatbot can match the most similar questions and give solutions. Students need to tell bot which course the question belongs to, cause if no matched question, it will be uploaded to lecturer or tutor.
Firstly, we need to input what courses we enrolled, and we collected question list of these three courses on forum. And we provide question upload feature for students to upload questions to staff,see the Figure 7.1.



upload the question on comp9321

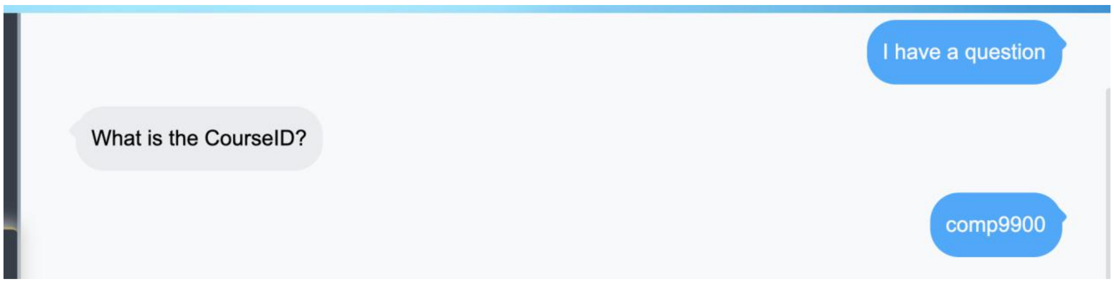
Please enter your question again and I will help you upload the question.

Which language we can use to do the assignment?

Ok, I have already uploaded your question.
Course staff may take 1-2 days to answer your questions, please check it out in a few days.

Figure 7.1

Then users can ask questions to bot by providing question and which course the question belongs to,see the Figure7.2 and 7.3.

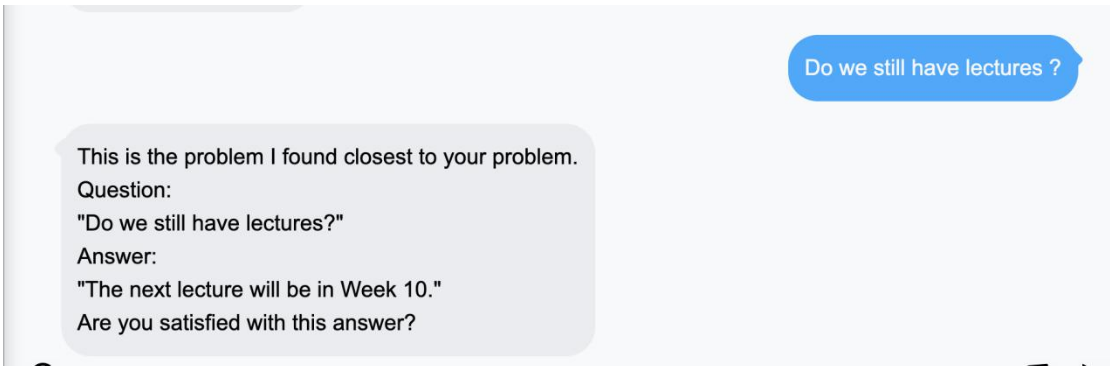


I have a question

What is the CourseID?

comp9900

Figure 7.2

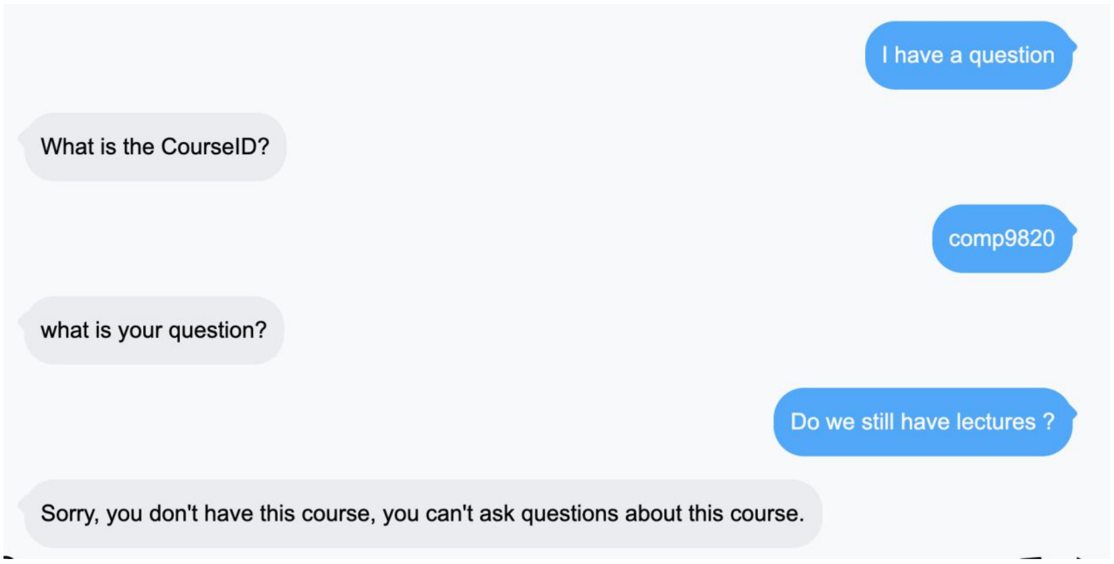


Do we still have lectures ?

This is the problem I found closest to your problem.
Question:
"Do we still have lectures?"
Answer:
"The next lecture will be in Week 10."
Are you satisfied with this answer?

Figure 7.3

If users asked questions on courses they did not enrolled, it will tell users they have no authorization of that course,see the Figure 7.4.



I have a question

What is the CourseID?

comp9820

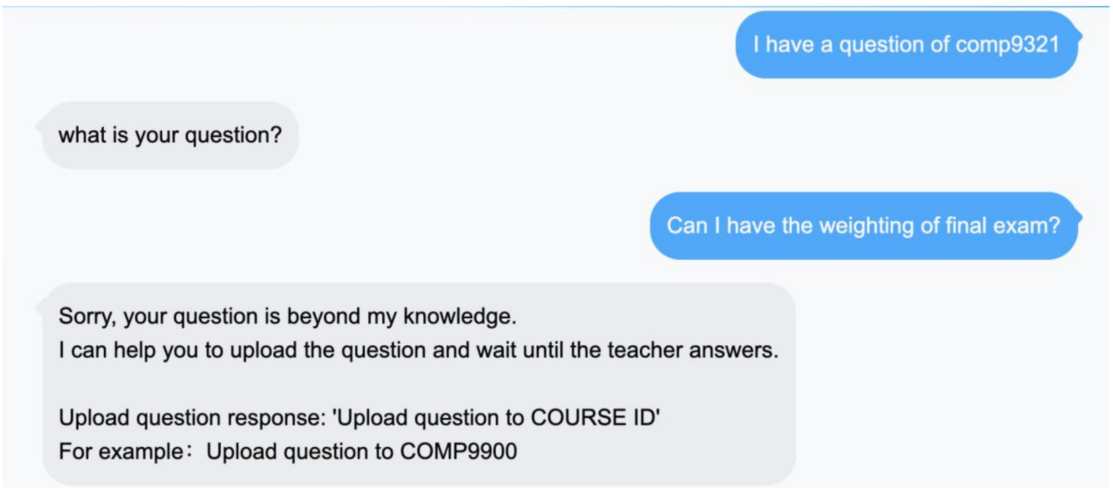
what is your question?

Do we still have lectures ?

Sorry, you don't have this course, you can't ask questions about this course.

Figure 7.4

And if our bot cannot find similar question like you asked, it will use upload function we built for students to upload questions for staff to resolve it,see Figure 7.5.



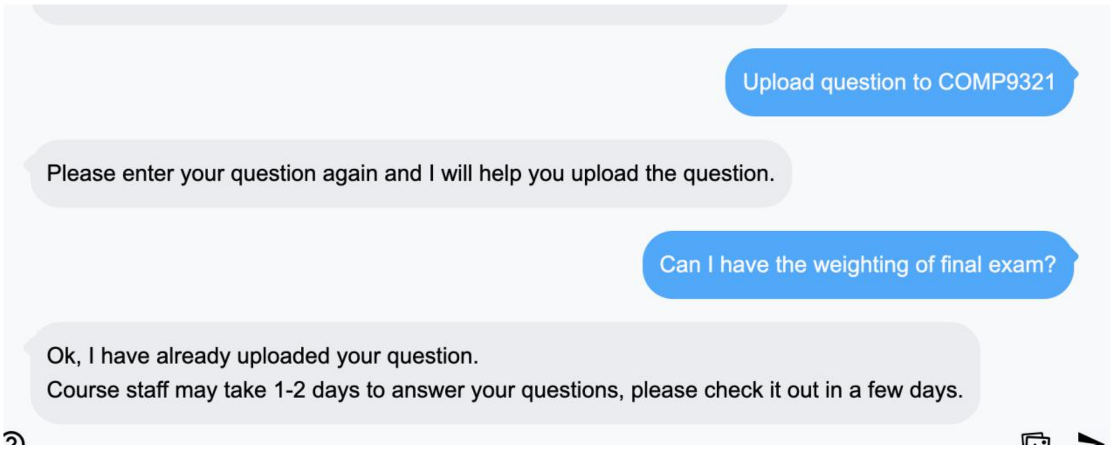
I have a question of comp9321

what is your question?

Can I have the weighting of final exam?

Sorry, your question is beyond my knowledge.
I can help you to upload the question and wait until the teacher answers.

Upload question response: 'Upload question to COURSE ID'
For example： Upload question to COMP9900


Upload question to COMP9321

Please enter your question again and I will help you upload the question.

Can I have the weighting of final exam?

Ok, I have already uploaded your question.
Course staff may take 1-2 days to answer your questions, please check it out in a few days.

Figure 7.5

At similarity question matching part, we match questions through comparing similarity of vector of words to resolve similar questions. We build a frequent term list through 5000 questions to create high usage and low valuable words, and delete them in questions in order to make the match more accurate. We set if there are many similar questions, it will return top 3 most similar questions, if there are less than three questions,then the system will return all of them with answers.

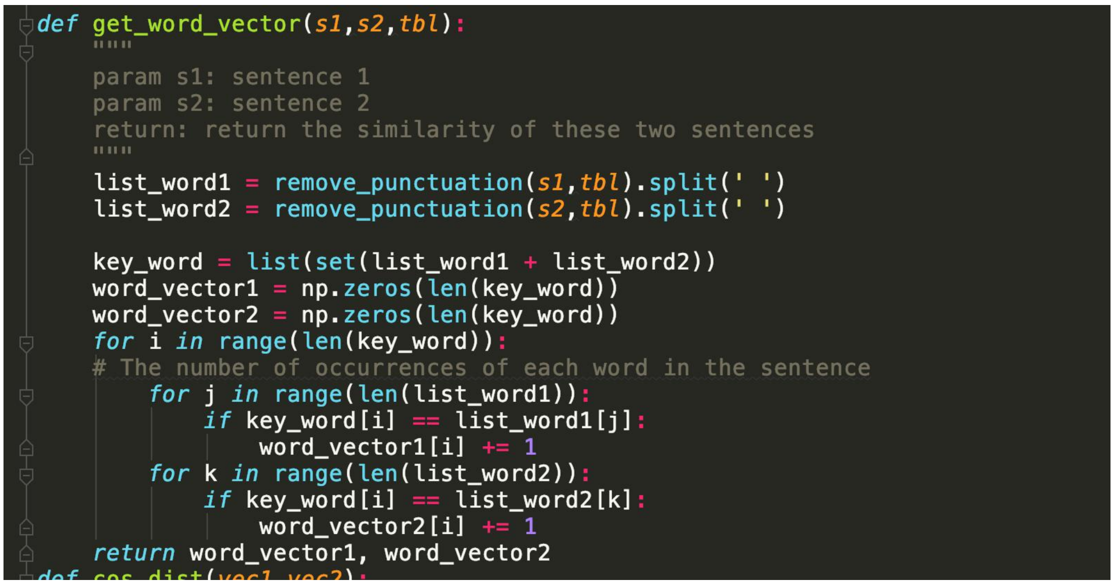

```
def get_word_vector(s1,s2,tbl):
    """
    param s1: sentence 1
    param s2: sentence 2
    return: return the similarity of these two sentences
    """
    list_word1 = remove_punctuation(s1,tbl).split(' ')
    list_word2 = remove_punctuation(s2,tbl).split(' ')

    key_word = list(set(list_word1 + list_word2))
    word_vector1 = np.zeros(len(key_word))
    word_vector2 = np.zeros(len(key_word))
    for i in range(len(key_word)):
    # The number of occurrences of each word in the sentence
        for j in range(len(list_word1)):
            if key_word[i] == list_word1[j]:
                word_vector1[i] += 1
        for k in range(len(list_word2)):
            if key_word[i] == list_word2[k]:
                word_vector2[i] += 1
    return word_vector1, word_vector2
```

Figure 7.6 get vecoter of words

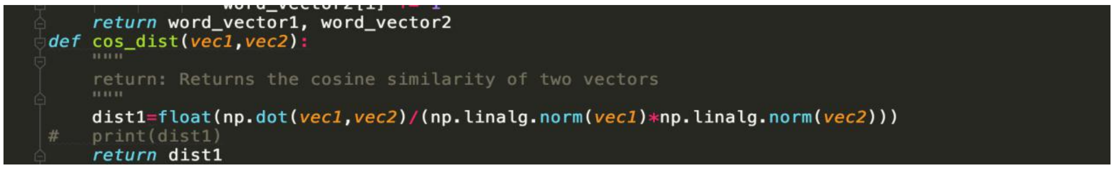
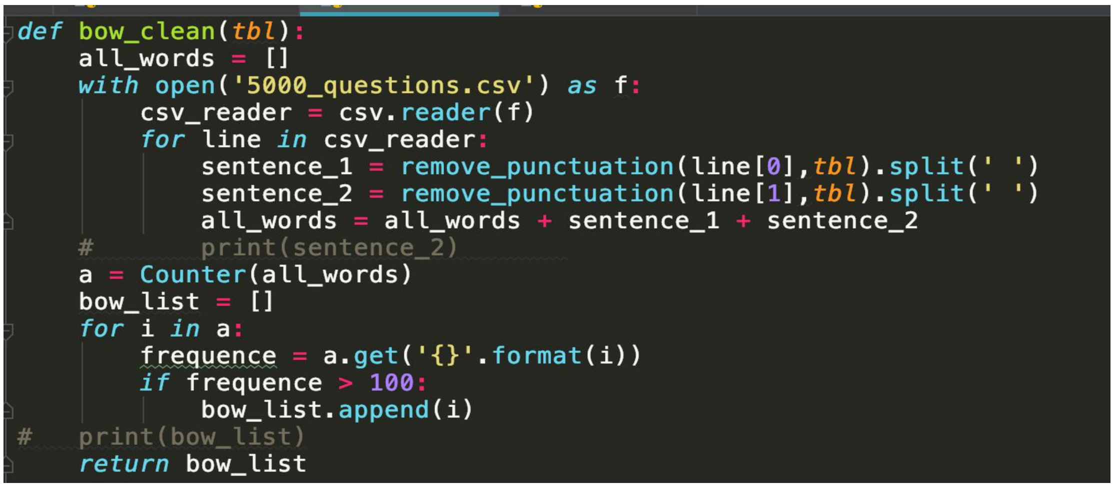

```
    return word_vector1, word_vector2
def cos_dist(vec1,vec2):
    """
    return: Returns the cosine similarity of two vectors
    """
    dist1=float(np.dot(vec1,vec2)/(np.linalg.norm(vec1)*np.linalg.norm(vec2)))
#   print(dist1)
    return dist1
```

Figure 7.7 get cosine distance of vectors

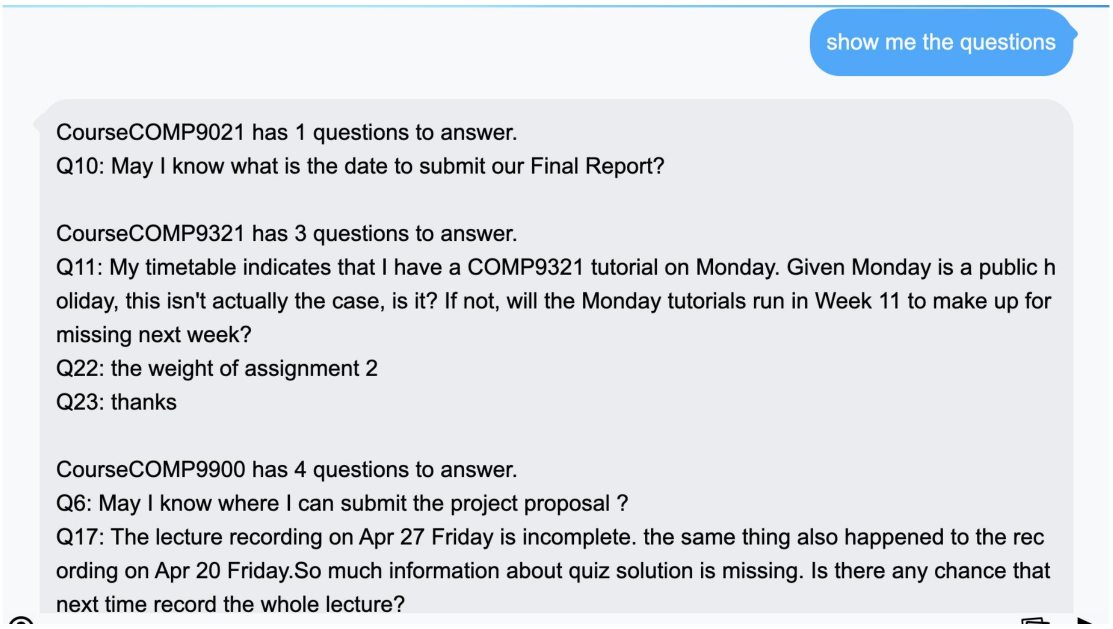


```
def bow_clean(tbl):
    all_words = []
    with open('5000_questions.csv') as f:
        csv_reader = csv.reader(f)
        for line in csv_reader:
            sentence_1 = remove_punctuation(line[0],tbl).split(' ')
            sentence_2 = remove_punctuation(line[1],tbl).split(' ')
            all_words = all_words + sentence_1 + sentence_2
    #           print(sentence_2)
    a = Counter(all_words)
    bow_list = []
    for i in a:
        frequence = a.get('{}'.format(i))
        if frequence > 100:
            bow_list.append(i)
#   print(bow_list)
    return bow_list
```

Figure 7.8 generate bow list

3.3.2 Function for staff

At staff part, it is mainly to help staff to see questions asked by students,and solve them for students. For example, staff typed: show me the questions, chatbot will show the questions unresolved. On the interface, staff can recognize how many questions have resolved and how many not resolved. All questions will be divided by different courses. And staff can see the detail of unresolved questions by typing such like: show me the questions,see the Figure 8.1.



show me the questions

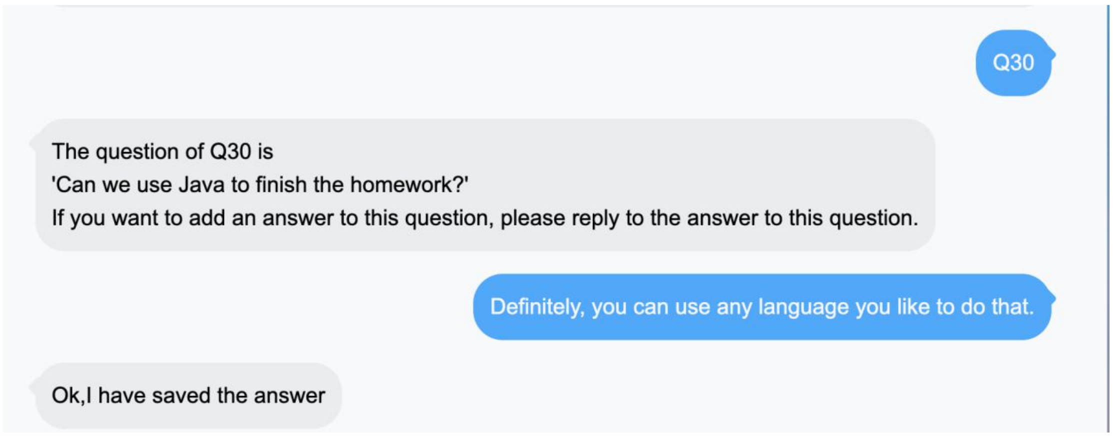
CourseCOMP9021 has 1 questions to answer.
Q10: May I know what is the date to submit our Final Report?

CourseCOMP9321 has 3 questions to answer.
Q11: My timetable indicates that I have a COMP9321 tutorial on Monday. Given Monday is a public h oliday, this isn't actually the case, is it? If not, will the Monday tutorials run in Week 11 to make up for missing next week?
Q22: the weight of assignment 2
Q23: thanks

CourseCOMP9900 has 4 questions to answer.
Q6: May I know where I can submit the project proposal ?
Q17: The lecture recording on Apr 27 Friday is incomplete. the same thing also happened to the rec ording on Apr 20 Friday.So much information about quiz solution is missing. Is there any chance that next time record the whole lecture?

Figure 8.1

If lecturer or tutor wants to answer question just type the question id like below and give the answer,see the Figure 8.2. After answering the question, it will be saved to database and show to students if they ask similar question,see thr Figure 8.3. We achieve this part similar to course information part.



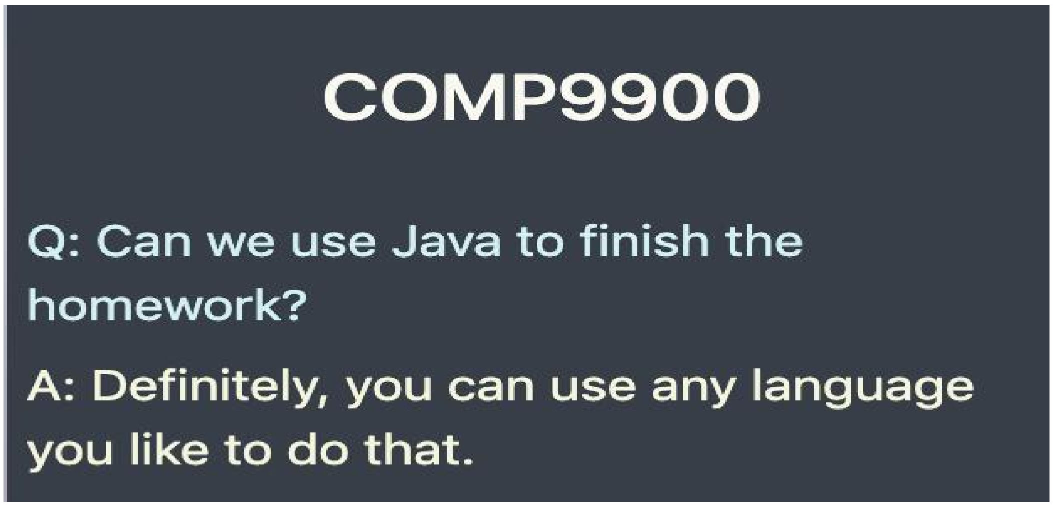
Q30

The question of Q30 is
'Can we use Java to finish the homework?'
If you want to add an answer to this question, please reply to the answer to this question.

Definitely, you can use any language you like to do that.

Ok,I have saved the answer

Figure 8.2



COMP9900

Q: Can we use Java to finish the homework?

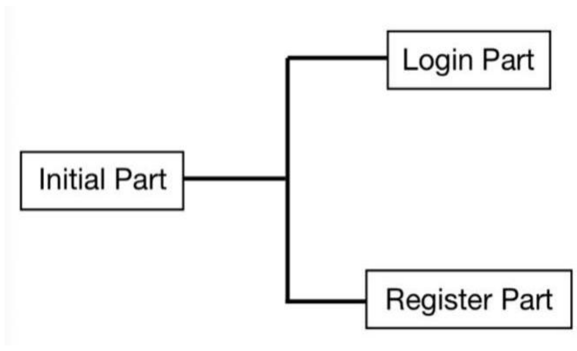
A: Definitely, you can use any language you like to do that.

Figure 8.3

3.4 User Interfaces

Our goal was to develop a mature and complete web application, which will implement the functionality that most software on the market can achieve. Also, it can effectively achieve user interaction with Chatbot. The user interface is implemented by HTML, CSS, Flask, Jinja and JavaScript (jQuery, Ajax).

The interface can be divided into three parts: Login/Register part, Function part and Chatbot part, each part all have independent and complete functions. Users can sign up an account and then they can log in securely. Besides, they can use function part to do a great number of things, such as change their name, email or password. The most important part is chat part, users can ask questions by text or pictures. As for different login identities: Student and Staff, student can select the courses they are studying and upload questions about their courses, student can select the courses they are teaching and answer questions about their courses. Please refer to the following figure 9.1 for the approximate function, which are listed the detailed functions below.



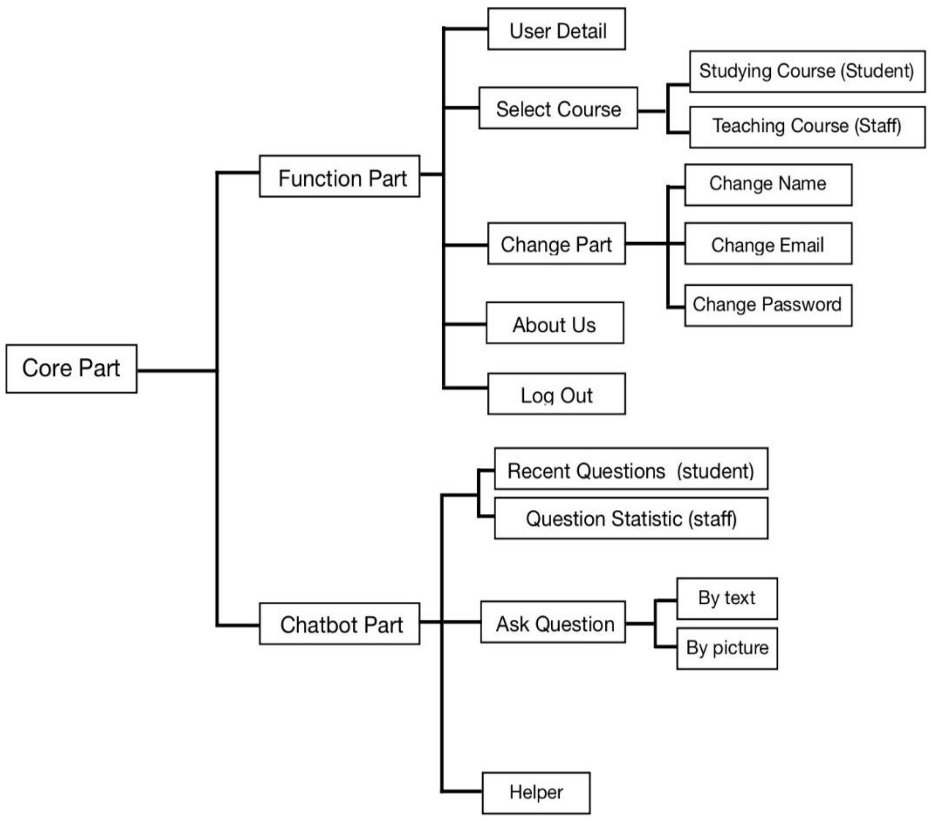


Figure 9.1

3.4.1 Initial Part

Initial part is the most basic part of our Chatbot web application. It can be roughly divided into three parts, the picture switching function, login part and register part. Please seen the figure 9.2 & 9.3 below.

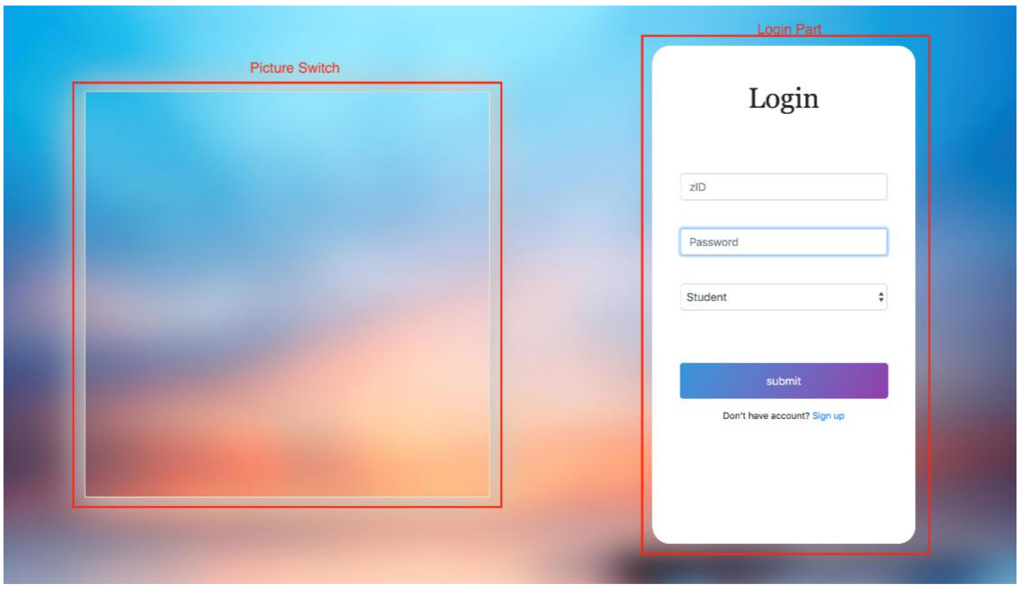


Figure 9.2 Picture Switch and Login Part

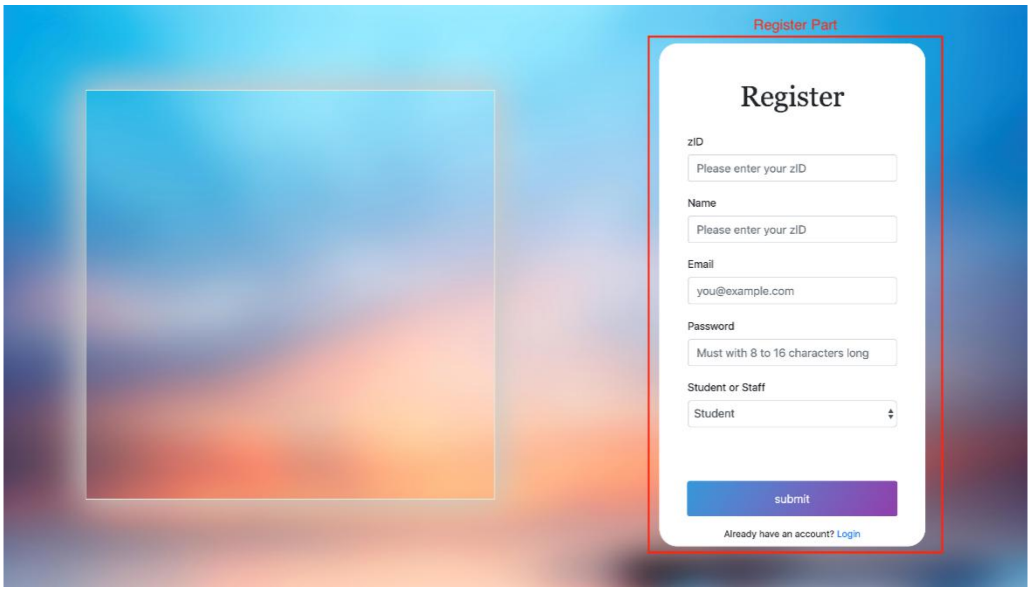


Figure 9.3 Register Part

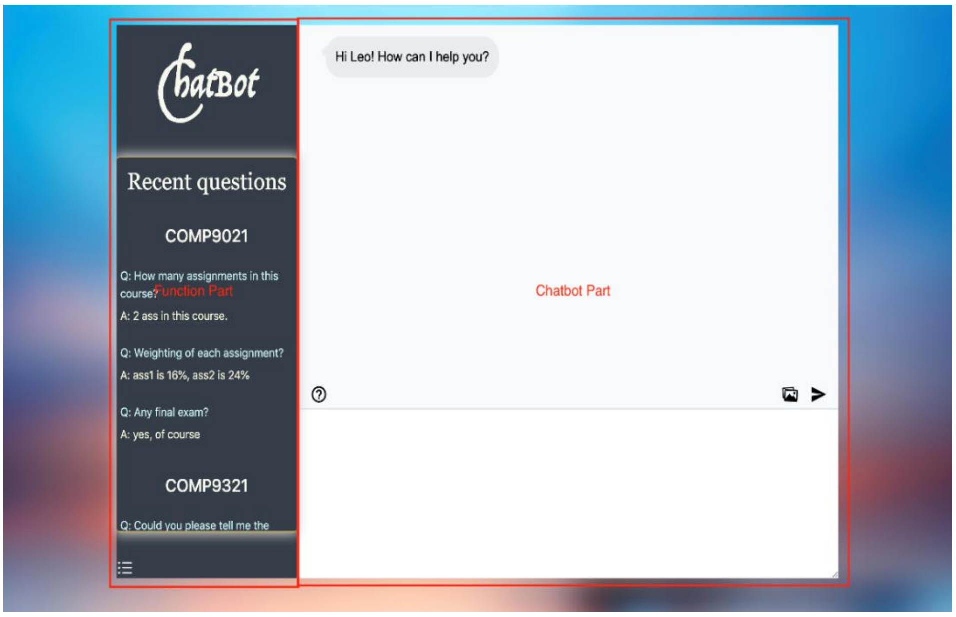
In picture switching function, the interval of picture switching is three seconds, which is include 5 pictures. it is used to make a preliminary demonstration of our Chatbot web application and making log in/register interface not too monotonous. We mainly use the JavaScript to achieve this function.
The Log in/Register page is implemented by Flask and Jinja. we use the Flask to create a database form (Please refer to the database section for details), and using Jinja to show the form to interface easily. When the user registers, he must input valid information. Data will be

sent to back-end, if the information of the form is valid, then registration will be successful and the information of user will be saved. It is same for the log in interface. When user log in success, back-end will return a key to the front-end as the user's unique identity.

3.4.2 Core Part

As we can see from the figure below, the core part is divided into two parts: Function part and Chatbot part, which have the different functions. The interface of core part is one-piece design. In other words, it is completed by only one html page file, the switching of the content is implemented by the display and hiding of the each div part. And most of these switching animations are used CSS and animation.css to achieve.

Figure 9.4



Recent questions

COMP9021

Q: How many assignments in this course?

A: 2 ass in this course.

Q: Weighting of each assignment?

A: ass1 is 16%, ass2 is 24%

Q: Any final exam?

A: yes, of course

COMP9321

Q: Could you please tell me the

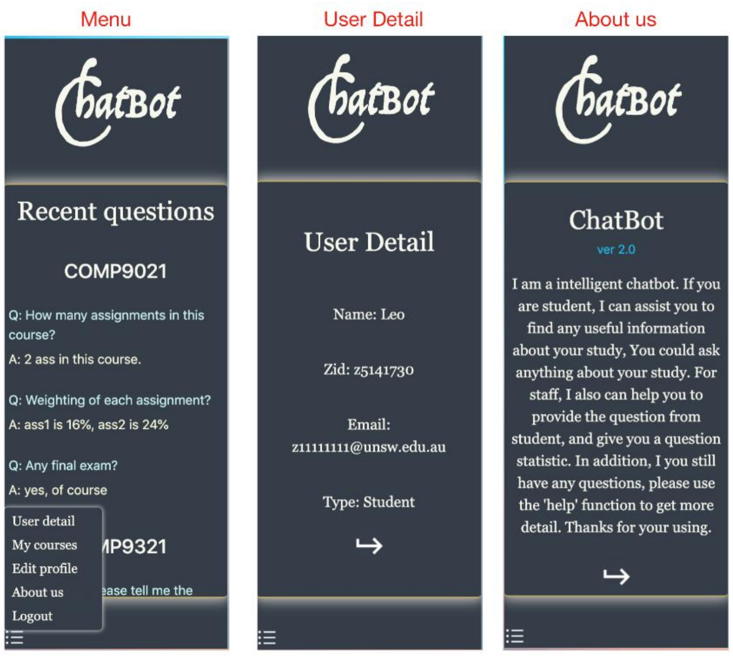
Hi Leo! How can I help you?

Function Part

Function part is composed of five parts, which is user detail, select courses, change part, about us and logout. They switch through a menu button at the bottom. Because most of these parts need to interact with the backend to get information, considering the delay waiting time, so we use a gradient display to smooth this process.

User detail is the first function of this part, which can show the detail of user' s information, such as name, ZID, Email and their type (student or staff). The implementation of this feature is through Ajax to post the unique identity key to backend, backend verifies this and retrieves the user's information from the database then send to frontend.

About us is a simple feature that is used to introduce our web app. The reason why we added this feature it is because we wanted to bring our web app closer to a product which on the market.


Menu

ChatBot

Recent questions

COMP9021

Q: How many assignments in this course?

A: 2 ass in this course.

Q: Weighting of each assignment?

A: ass1 is 16%, ass2 is 24%

Q: Any final exam?

A: yes, of course

User Detail

ChatBot

User Detail

Name: Leo

Zid: z5141730

Email: z11111111@unsw.edu.au

Type: Student

About us

ChatBot

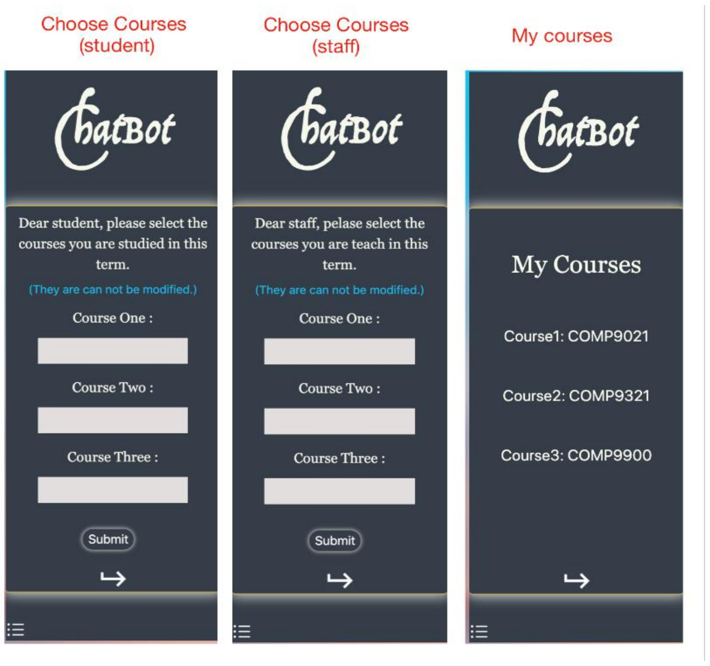
ChatBot

ver 2.0

I am a intelligent chatbot. If you are student, I can assist you to find any useful information about your study, You could ask anything about your study. For staff, I also can help you to provide the question from student, and give you a question statistic. In addition, I you still have any questions, please use the 'help' function to get more detail. Thanks for your using.

Figure 9.5 Function Part

Select courses is an important part, which has a relationship with many of the functions behind. In order to simulate a real course processing, this feature is different for students and staff. They both need to use this function and choose three courses when they login at the first time, for student, they should choose three courses which they are studying, and teacher should choose the course they are teaching. When they are selected and submitted, the data will be sent to the backend and recorded in database. It is worth mentioning that this data cannot be modified. When they submit successfully, this page will change to a showing courses page.


Choose Courses (student)

ChatBot

Dear student, please select the courses you are studied in this term.

(They are can not be modified.)

Course One :

Course Two :

Course Three :

Choose Courses (staff)

ChatBot

Dear staff, pelase select the courses you are teach in this term.

(They are can not be modified.)

Course One :

Course Two :

Course Three :

My courses

ChatBot

My Courses

Course1: COMP9021

Course2: COMP9321

Course3: COMP9900

Figure 9.6 Function Part

Change part mainly used to implement the modification of user data. In our web application, user can change their name, email and password (ZID is primary key so is cannot be modified). This feature is a very common but hard to achieve, this is because the dynamic modification of the user information will bring many unexpected bugs. The implementation of this feature is also the use of Ajax, frontend send the new information to backend, if it is valid then the database of backend will be modified.

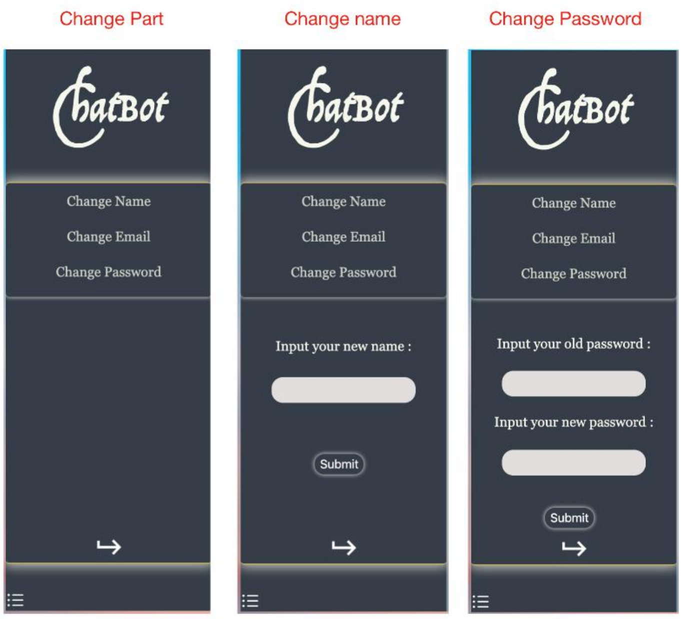
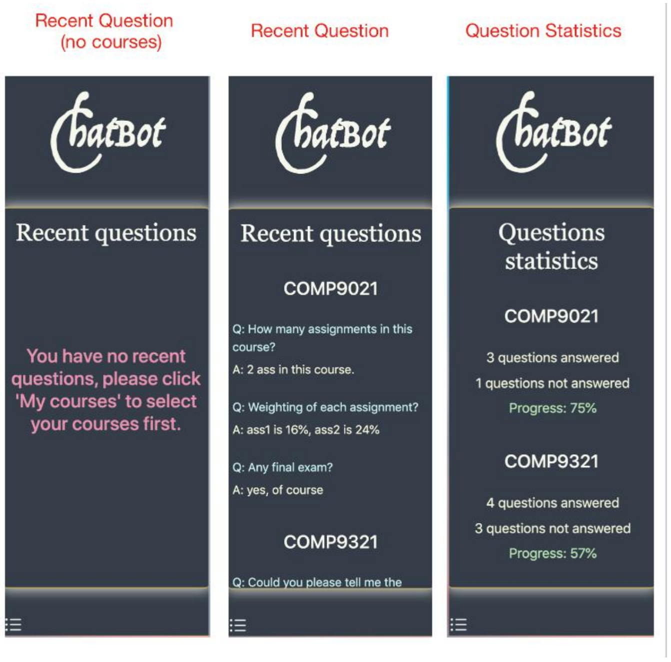


Figure 9.7 Function Part

Chatbot Part

Chatbot part is consisted of three parts. Among, recent questions and questions statistics has a close relationship with the previous function part. Please seen the figure below.



You have no recent questions, please click 'My courses' to select your courses first.

COMP9021

Q: How many assignments in this course?

A: 2 ass in this course.

Q: Weighting of each assignment?

A: ass1 is 16%, ass2 is 24%

Q: Any final exam?

A: yes, of course

COMP9321

Q: Could you please tell me the

COMP9021

3 questions answered

1 questions not answered

Progress: 75%

COMP9321

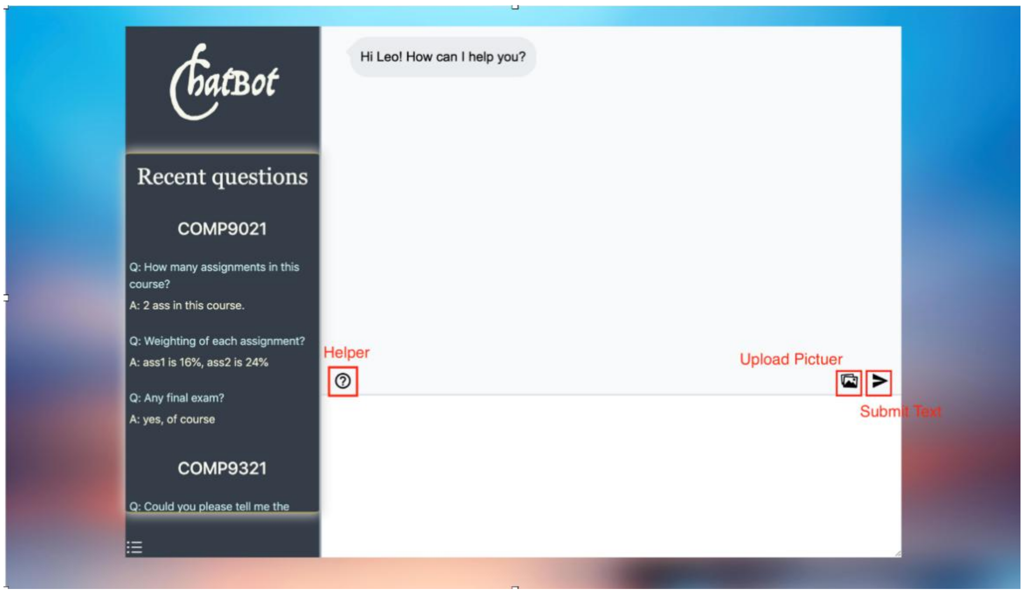
4 questions answered

3 questions not answered

Progress: 57%

Figure 9.8 Chatbot Part

For this part, the functions that students and staffs can used are different. Firstly, when they are not choosing their courses, this interface will not show anything just like the first part of figure. When they choose their courses, students will display the recent question function. It can display the latest questions of the course selected by the student in flashback form, and these questions are answered by the staff. This function is implemented by sending the course code of the student to the backend, and backend filters the latest questions answered by the course from the database. We think it is an interesting and useful feature. As for staff, it will show the question statistics rather than posting recent questions, including answered and not answered questions, which will help staffs get understand their courses deeply.
Next, we will introduce the helper and ask question part.



Hi Leo! How can I help you?

Recent questions

COMP9021

Q: How many assignments in this course?
A: 2 ass in this course.

Q: Weighting of each assignment?
A: ass1 is 16%, ass2 is 24%

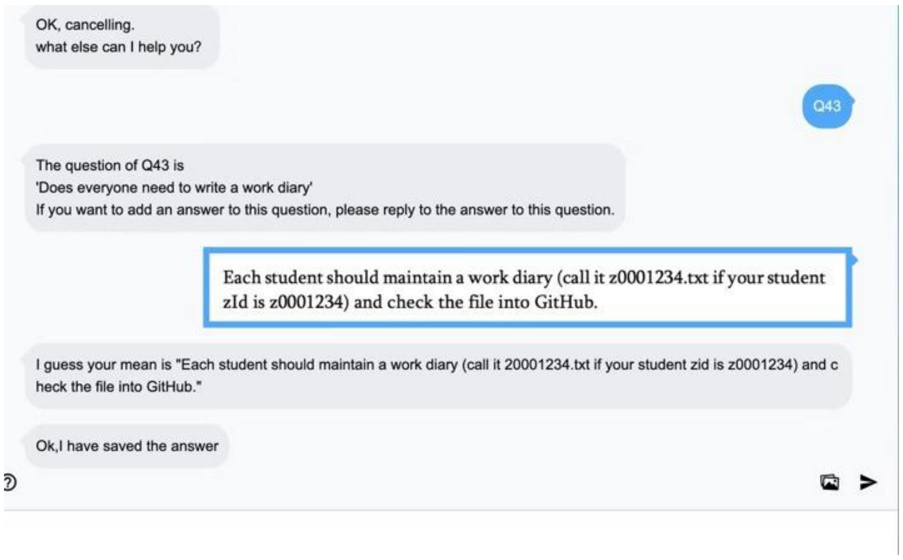
Q: Any final exam?
A: yes, of course

COMP9321

Q: Could you please tell me the

Figure 9.9 Chatbot Part
Ask question is the most important part. In other words, it is the core of the whole web application. For the implementation of this part, we refer to styles of Messenger and WeChat. In our thought, a tidy and simple chat interface will be more liked by users, and the functions are very practical but not need too many. In this part, we consume a lot of time in the CSS, including page layout and colour. Besides, we added animation to each button, making it less stiff when clicked.
The chat interface is divided into two parts, the lower part is the input and function button box, and the upper part is to display the information under the chat. As for the lower part, we use the textarea type rather than input type in HTML, it is because textarea is more adaptable to input content, including automatic line breaks and warping. And we use JavaScript to add a more special design that user can use ENTER button to submit their information, also they can be used ENTER + Ctrl to wrap. In addition, we added a hidden word limit, which will automatically extract the first part when the number of words exceeds 255.
As for chat part, which is implement by dynamically adding div messages. We divide the content of information into human and robotic, represented by different regions and colours. When user input a message and submit it, which will extract and processed by JavaScript, then it will send to backend, through a series of treatments the message will return to frontend. Besides, we add to waiting message for robotic to make it

more humanize. And using regular expressions to identify URL and convert to link form.


OK, cancelling.
what else can I help you?

Q43

The question of Q43 is
'Does everyone need to write a work diary'
If you want to add an answer to this question, please reply to the answer to this question.

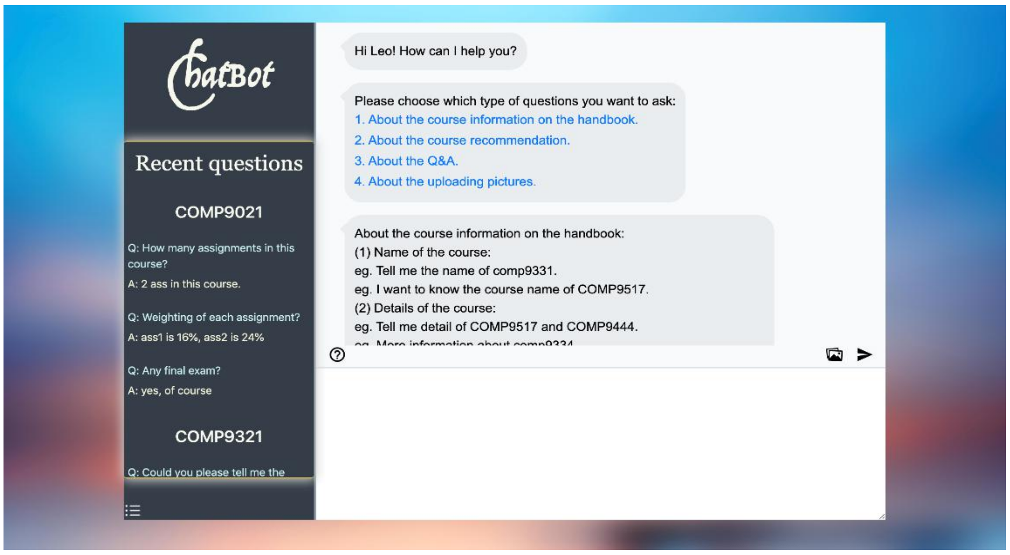
Each student should maintain a work diary (call it z0001234.txt if your student zId is z0001234) and check the file into GitHub.

I guess your mean is "Each student should maintain a work diary (call it 20001234.txt if your student zid is z0001234) and c
heck the file into GitHub."

Ok,I have saved the answer

Figure 9.10 Chatbot Part

In addition, image recognition is also an interesting part. As shown in the figure above, it allows the user to express the problem in the form of a picture. Then web application will extract the text in the image. Although it is not always effective, it will be some convenience and fun for the user. When user uploading the picture, frontend will use the FormData type to save the picture and send to backend through Ajax. Backend will get the picture and save it to cache and to recognize this picture.



Recent questions

COMP9021

Q: How many assignments in this course?
A: 2 ass in this course.

Q: Weighting of each assignment?
A: ass1 is 16%, ass2 is 24%

Q: Any final exam?
A: yes, of course

COMP9321

Q: Could you please tell me the the

Hi Leo! How can I help you?

Please choose which type of questions you want to ask:
1. About the course information on the handbook.
2. About the course recommendation.
3. About the Q&A.
4. About the uploading pictures.

About the course information on the handbook:
(1) Name of the course:
eg. Tell me the name of comp9331.
eg. I want to know the course name of COMP9517.
(2) Details of the course:
eg. Tell me detail of COMP9517 and COMP9444.

Figure 9.11 Chatbot Part

Helper is a very useful function. It will provide users with useful tutorials in the form of conversations to help them better use this web app. Users clicks the ‘?’ button to get help information, and according to the tutorial to get the help they want, at the same time, the tutorial also provides a great number examples, which can help users acquire this web application more effectively.

4. Project management tools and system

Since the team size is small and the team members can't invest all of them in this project in a limited time, in other words, each member has multiple projects at the same time, and the team members are good at different parts. So we choose to divide the work for project development and keep a certain frequency of code sharing and discussion to unify the progress.

4.1 Query project information and select project title

Before the start of the project, team members first conducted an understanding of all the project topics and necessary information, and we measured different topics according to the technology that our team members are familiar with it. The final team members are very interested in the chatbot. As students, we have a better understanding of the course system, so after a week of inquiries and two team discussions, we identified our topic.

4.2 Develop project plans and division of labor

Among the members of our team, each person is good at different technologies: good at front-end, good at database and back-end construction, familiar with dialogflow, and good at web crawlers and data processing. So we handed the front end to a member, and a back-end developer and front-end developer to hand over, two members to assist in the background development, and mainly responsible for the approximate matching and data processing part.

In addition to the clear division of labor, the team developed a monitoring plan at the beginning of the project, including code integration twice a week, content discussion and progress. Each group meeting is a mission's deadline, so that our weekly tasks will not be postponed and the final time will not be able to complete the functions we designed.

4.3 preparation stage

In the preparation stage

(1) We learned how to combine dialogflow with flask. Read the documentation for dialogflow

(2) Go to the school's course website to find the right data.

(3) Design the functions of our robots, conceive the inconveniences encountered in students' lives, put these time-consuming operations into our functions, and analyze the practicality and feasibility of each function.

4.4 Implementation phase

During the project's ongoing phase:

(1) We set the deadline for each feature to complete.

(2) In the first meeting of the week, we only do the scheduling and code integration.

(3) On the second meeting of the week, we will set up the next week's mission on our project management software Trello. And mark the task on the member responsible for this task.

(4). Each member will follow up on the tasks on Trello so that each team member can understand the overall amount of completion. Avoid project delays.

(5). Task sequence:

(1) First choose to build a database on the Amazon cloud, establish a database, and ensure that the work can be taken at will.

(2) Complete the backend of the log in and registration interface

(3) Build a front-end framework and make an initial dialogue interface to ensure that the back-end can be tested.

(4) Complete the construction of the dialogflow function

(5) dialogflow connection flask back-end

(6) Adding an approximate matching algorithm

(7) Overall optimization, adding a quick browsing system.

4.5 Project optimization and documentation

After all the functions are basically completed, we have a long-term meeting, continuous testing, and also give our products to more people to test, find problems and solve them to make our project products reach team capacity. Optimal within the range.

In the report part, we first write their own parts according to their respective roles in the project, and the other parts are written in meeting. And overall optimization before submission.

5. Challenges and future improvements

5.1 Challenge

Front-end:

Display problems with different front-end window sizes.

approximate matching mode:

On the approximate matching model of the problem, there is a certain limit on the length of the sentence. If the length of the problem in the database is long, the accuracy of the approximate match may be reduced. Therefore, when the problem is derived from the database for matching, the length of the sentence is truncated, which results in the accuracy of the matching is not very high.

Back-end:

The contextual application of dialogflow is difficult to fully meet expectations. Many times we want to get the student's question or the teacher's answer. We need all the course ID and input text, but if the text contains our other intone parameters, it will match For other problems, we spent a long time designing the entity and intent of dialogflow to make it meet our expectations.For example, when a student asks a question with a course ID in the forum, sometimes they misidentify the problem and make chatbot think that the user is asking for information about the course. And return the wrong answer

5.2 future improvement

1. If there is more time and opportunity, a function should be added in the future: the student interface can add different colors, and the questions that I ask questions can be displayed in special colors for the students to display.
2. We think there is still a function that may improve the performance of the teacher user. Maybe sorting the questions the teacher sees is a good direction.
3. 3. If there is an opportunity to optimize in the future, our chatbot will add an email alert function.

6. Team effort

➢ Scrum Master

Student Name: Dong Zhu Student Number: z5165957

The scrum master is responsible for the overall work schedule and progress monitoring of the team, writing the project proposal and most of the framework of the final report. Organizing team members to participate in group face to face discussions, assisting the team members in solving problems and test weekly tasks. In addition, the person in charge is responsible for all the data preparation and data cleaning, data processing, etc. of the entire project, and cooperates with the team members to import the data into the database. Also responsible for the approximate matching model of the data, including writing code, training, testing and the application of the model. Participated in functional testing during the final front-end back-end docking and testing.

➢ Back-end & Database Developer

Student Name: Yaqi Yang Student Number: z5143675

Designed and built the database, and built the database on the Amazon cloud RDS, connecting the database to the back end.
Write the backend code for login registration and the corresponding HTML file.
The design implements some of the functions of dialogflow and connects the dialogflow to the backend. Achieve most of the basic problems of chatbot.
In addition, in the two back-end developers, it is responsible for docking with the front-end, providing feedback on the back-end according to the front-end requirements, and providing corresponding supplementary code.

➢ Back-end Developer

Student Name: Jinpeng Jiao Student Number: z5143932

Back-end developer is responsible for the tech-support and process data to front-end and service for users.
Processing the course information extraction from dialogflow, writing part of course related questions, mainly writing question and answer part. Combining the matching model into back-end.Processing question dataset in order to use in back-end.
In addition, in the two back-end developers, it is responsible for processing code for the question matching, participating in model training and apply the model, connecting question matching and back-end.

➢ Front-end Developer

Student Name: Bohan Zhao Student Number: z5141730

Front-end developer is responsible for the all interface design and a part of connection with the back-end. Firstly, designing the interface style and construct a rough page layout. In additional, front-end developer should typesetting each area of the page, adding function and css for them, and adding some animations to them. Finally, adjusting the layout of the page again and add some detail to the page. Besides, front-end developer should complete some interaction with the backend, such as extract information or put back information.

7. Conclusion

With the rapid development of natural language processing and intelligent chat assistants, this project of chatbots has greatly enhanced our project development capabilities and teamwork capabilities. In the realization of chat robot, this project mainly uses Google Dialogflow to realize the natural language processing of input sentences, and realizes the answer extraction process based on the information in keywords and database in the intelligent question answering system. The chat robot mainly implements user management, problem matching and answering, uploading course questions, course information query and post question’ answers, etc. Moreover, the robot can perfect the contents of its database from the dialogue.
Throughout the project, the team members not only completed the tasks they were responsible for on time, but also performed well in solving the team problems. At the same time, based on the tutor recommendation on lab, we have improved many details, such as classifying the problem according to the course ID, modifying the front-end UI interface, and changing the default statement returned by the robot.

8. User documentation/manual

Environment: python3.6+
IDE: pyCharm(https://www.jetbrains.com/pycharm/download/)

Required packages:

```
Bootstrap_Flask==1.1.0
pytesseract==0.3.0
Flask_SQLAlchemy==2.4.1
Flask==0.12.4
Werkzeug==0.14.1
requests==2.18.4
PyMySQL==0.9.3
Flask_WTF==0.14.2
numpy==1.16.0
WTForms==2.2.1
```

```
dialogflow==0.6.0
Flask_Migrate==2.5.2
Flask_Uploads==0.2.1
flask_bootstrap==3.3.7.1
Pillow==6.2.1
```

Steps:

1.Clone and navigate to chatbot directory

2.Install the required packages

```
pip install -r requirements.txt
```

3.Run the python server

```
python run.py
```

4. Open “http://127.0.0.1:5000” in your browser.

Reference

[1] Nye, B. D., Graesser, A. C., & Hu, X. (2014). AutoTutor and Family: A Review of 17 Years of Natural Language Tutoring. International Journal of Artificial Intelligence in Education, 24(4), 427—469.

[2] Architecture for building Conversational Agents that support Collaborative Learning http://www.cse.unsw.edu.au/~cs9900/19T3/For-ConvrdstionalAgents/KumarAndRoseInPress.pdf

[3] Kaggle dataset website. https://www.kaggle.com/quora/question-pairs-dataset