

Simulation and Modelling
(UCS751)
Project Report
KnowBot

Submitted by:

(102183010) Eknoor Kaur Walia

(102183013) Gurpreet Kaur

(102183016) Ishita

BE Fourth Year- COE



Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala

NOV, 2023

Problem Description:

In the current educational landscape, students often face challenges in accessing immediate help for their academic queries. Traditional methods of education are not always able to cater to the individual learning pace and style of each student. Moreover, the availability of teachers or tutors is limited to specific hours, which may not align with the students' study schedules and also all the study resources are scattered which makes it difficult to cope with.

The objective is to design an educational chatbot that can provide 24/7 assistance to students, answering their queries and helping them understand and learn concepts effectively. The chatbot should be capable of providing personalized learning experiences, adapting to each student's pace and style of learning. It should also be able to conduct quizzes for revision, provide formative feedback, and solve general doubts.

The chatbot should be user-friendly, engaging, and efficient in delivering accurate information. It should be designed to handle multiple queries at a time and provide immediate responses. The chatbot should also have multilingual capabilities to cater to students from different linguistic backgrounds.

The ultimate goal of this project is to enhance the learning experience of students, making education more accessible, personalized, and efficient.

Need:

- **Student Onboarding and Orientation:** For new students, navigating the college/university/school environment can be challenging. Chatbots can help simplify the starting process by providing virtual tours of the online platform for i.e LMS, webkiosk and other helpful sites and its features, offering step-by-step guidance on course registration and enrolment, and answering frequently asked questions about college/university/school policies and procedures
- **Personalized Learning and Academic Support:** These bots can offer personalized learning experiences for students, adapting to their individual needs, and offering support in various subject areas.
- **Homework Assistance:** Education bots can help students understand complex topics, provide study material, and offer ideas for completing assignments.
- **Exam Preparation and Study Guides:** They can help students prepare for exams by providing study guides, previous year question papers practice questions and instant feedback etc.
- **Immediate Feedback:** Chatbots provide instant feedback to students on their queries, and related to assignments.

Our work:

Our project KnowBot aims to centralise the information available on various official TIET websites and other education resources to help students access the right information within no time.

This project implements a simple conversational AI system capable of recognizing user intents and generating appropriate responses. The system is designed to understand natural language input, classify it into predefined intents, and provide relevant information or responses based on the recognized intent.

Key Components:

Intent Definition:

- Intents are defined in a JSON file ([intents.json](#)), containing patterns associated with different user intentions.

Natural Language Processing (NLP):

- The system uses Natural Language Processing techniques, including tokenization, lemmatization, and removal of stopwords and punctuation, to process both the predefined intent patterns and user input.

Intent Classification:

- The project employs a simple word-matching approach to classify user input into predefined intents. The system counts occurrences of words associated with each intent to determine the most likely intent.

Response Generation:

- Responses for each intent are defined in the JSON file. The system randomly selects a response from the list associated with the recognized intent.

Text-to-Speech Conversion:

- The generated response is converted to speech using the Google Text-to-Speech (gTTS) library. The resulting audio is saved as a WAV file and played back to the user.

User Interaction:

- The system interacts with the user by printing a welcome message and prompting for input. The conversation can continue through multiple interactions.

Algorithmic description:

Import Required Libraries:

- Import the necessary libraries like gTTS for text-to-speech, translate for translation, nltk for natural language processing, numpy for numerical operations, etc.

Load Intents from JSON File:

- Load the intents from a JSON file (intents.json) using the json library.

Tokenization and Preprocessing:

- Tokenize the patterns in each intent using the nltk.word_tokenize method.
- Lemmatize the words, convert to lowercase, and remove stopwords and punctuation.

Build Vocabulary and Classes:

- Create lists (words and classes) to store unique words and classes from the intents.
- Create a list (documents) of tokenized words along with their corresponding intent tags.

User Interaction:

- Print a welcome message and get user input as a chat message.

Processing User Input:

- Tokenize and preprocess the user's input similar to the intents.

Tagging and Classification:

- Initialize a dictionary (final_tag) to store the count of each intent tag.
- For each word in the user's input, check if it matches any word in the documents for each intent.
- Update the count for the corresponding intent tag in the final_tag dictionary.

Selecting the Intent Tag:

- Identify the intent tag with the maximum count as the selected tag.
- If the count for the selected tag is None or 0, set the selected tag as 'default'.

Generating Response:

- Retrieve responses associated with the selected intent tag.
- Choose a random response from the list of responses.

Text-to-Speech Conversion:

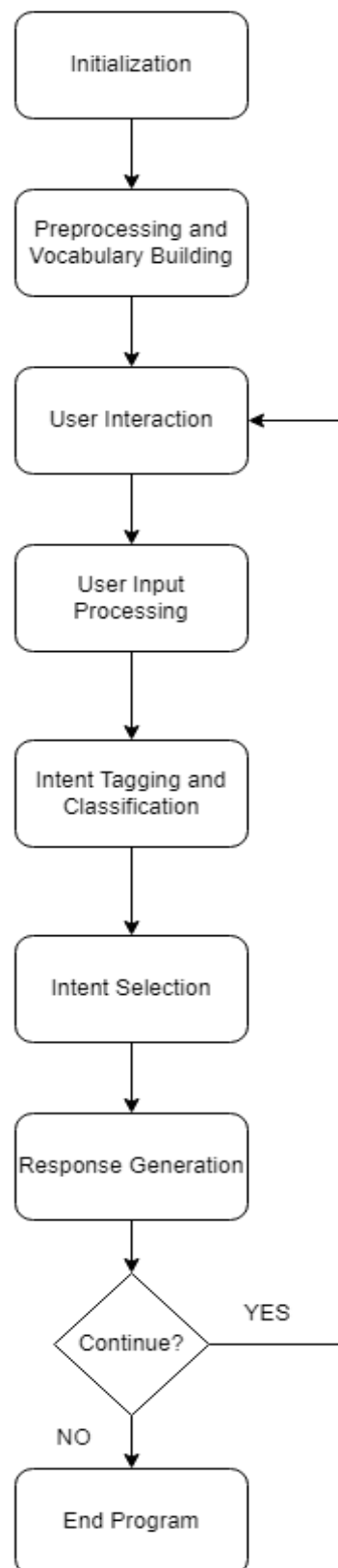
- Convert the response to speech using gTTS.
- Save the speech as a WAV file (1.wav).
- Play the audio using IPython.display.Audio.

Explanation:

- The code starts by importing necessary libraries and loading intents from a JSON file.
- It then preprocesses the intents' patterns and builds a vocabulary and class list.
- The user is prompted for input, and the input is processed similarly to the intents.
- The code tags the user's input with intents by counting occurrences of words associated with each intent.
- The intent with the maximum count is selected, and a random response from that intent is chosen.
- The response is converted to speech, saved as a WAV file, and played.

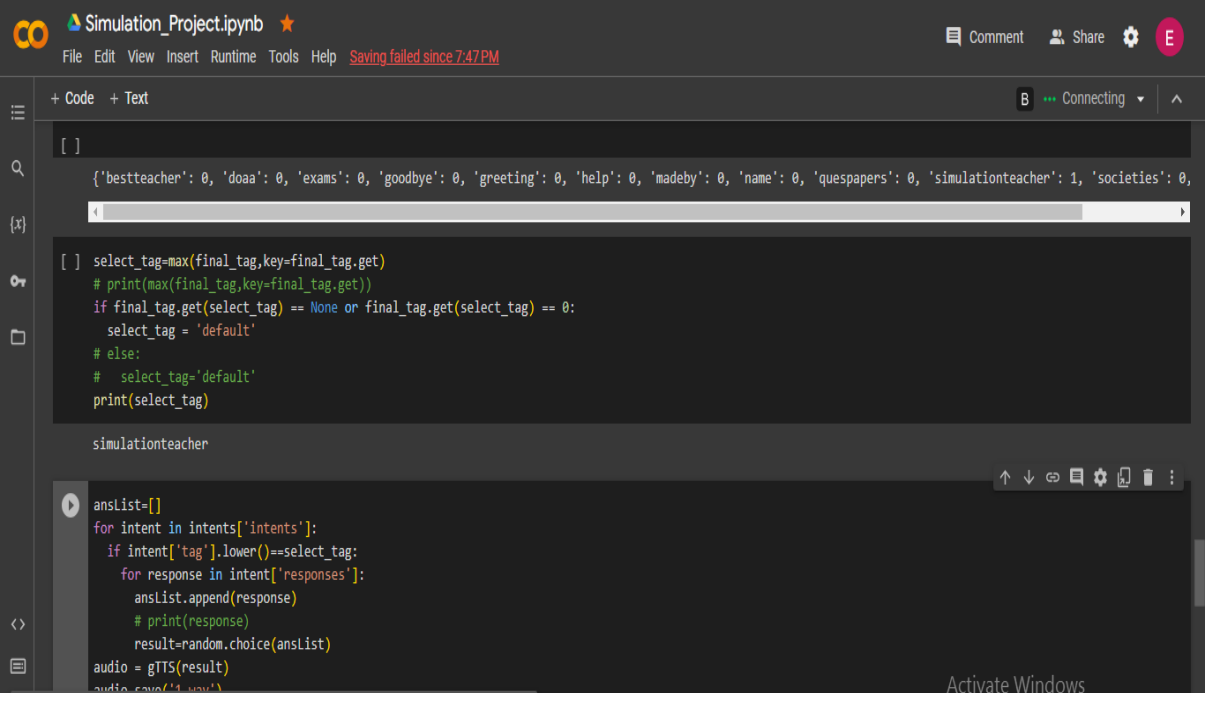
KnowBot uses intent matching based on word occurrences, and it selects the intent with the highest count.

Project workflow:



Results:

user input:



The screenshot shows a Jupyter Notebook titled "Simulation_Project.ipynb". The code defines a dictionary of intents and a loop to process user input. The dictionary contains keys like 'bestteacher', 'doaa', 'exams', 'goodbye', 'greeting', 'help', 'madeby', 'name', 'quespapers', 'simulationteacher', and 'societies'. The loop iterates over the intents, selecting a tag based on the user input and then printing the selected tag.

```
[ ]
```

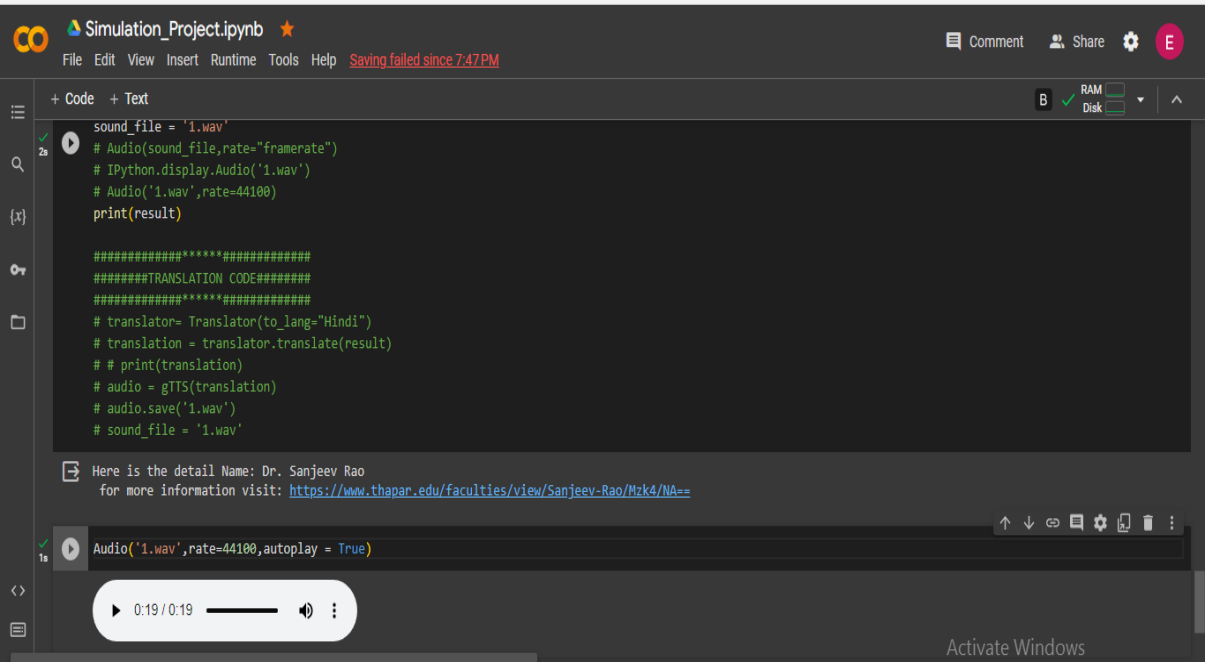
```
{'bestteacher': 0, 'doaa': 0, 'exams': 0, 'goodbye': 0, 'greeting': 0, 'help': 0, 'madeby': 0, 'name': 0, 'quespapers': 0, 'simulationteacher': 1, 'societies': 0,
```

```
[ ] select_tag=max(final_tag,key=final_tag.get)
# print(max(final_tag,key=final_tag.get))
if final_tag.get(select_tag) == None or final_tag.get(select_tag) == 0:
    select_tag = 'default'
# else:
#     select_tag='default'
print(select_tag)
```

```
simulationteacher
```

```
ansList=[]
for intent in intents['intents']:
    if intent['tag'].lower()==select_tag:
        for response in intent['responses']:
            ansList.append(response)
            # print(response)
            result=random.choice(ansList)
audio = gTTS(result)
audio.save('1.wav')
```

KnowBot response:



The screenshot shows the same Jupyter Notebook interface. The code defines a function to play audio and a translation code snippet. The function uses the 'gTTS' library to generate audio from text and save it to a file. The translation code snippet uses the 'GoogleTranslator' library to translate text from English to Hindi.

```
sound_file = '1.wav'
# Audio(sound_file,rate="framerate")
# IPython.display.Audio('1.wav')
# Audio('1.wav',rate=44100)
print(result)
```

```
#####TRANSLATION CODE#####
#####TRANSLATION CODE#####
# translator= Translator(to_lang="Hindi")
# translation = translator.translate(result)
# # print(translation)
# audio = gTTS(translation)
# audio.save('1.wav')
# sound_file = '1.wav'
```

```
Here is the detail Name: Dr. Sanjeev Rao
for more information visit: https://www.thapar.edu/faculties/view/Sanjeev-Rao/Mzk4/NA==
```

```
Audio('1.wav',rate=44100,autoplay = True)
```

Novelty and comparison:

The proposed educational chatbot introduces several innovative features that set it apart from existing solutions:

- **Campus-Specific Information:** The chatbot is tailored specifically for our campus, making it unique. It is equipped with detailed knowledge about our specific academic programs, faculty, events, and resources. This allows it to provide accurate and relevant responses to queries related to our campus.
- **Audio Capability:** Unlike traditional text-based chatbots, our educational chatbot includes an audio feature. This allows students to interact with the chatbot using voice commands, making it more accessible and user-friendly. The audio feature also enables the chatbot to provide auditory learning materials, catering to auditory learners and enhancing the overall learning experience.
- **Multimodal Learning:** By incorporating both text and audio, the chatbot supports multimodal learning. This caters to different learning styles and preferences, thereby improving the effectiveness of learning.

These novel features make our educational chatbot a powerful tool in enhancing the learning experience on our campus, providing personalized, accessible, and efficient academic support to all students.

Future directions:

- **Academic Support and Guidance:**

Personalized Learning Paths: Knowbot will provide personalized study plans based on individual learning styles, past performance, and upcoming coursework, optimizing academic success.

Subject-Specific Assistance: Offering tailored support in various subjects, from explanations of complex concepts to providing additional resources like videos, articles, or practice problems.

- **Administrative Assistance:**

Course Registration and Scheduling: Assisting students in selecting courses, creating schedules, and managing conflicts, considering preferences and prerequisites.

Deadline Management: Sending reminders for assignment due dates, exam schedules, and project milestones to help students stay organized and focused.

- **Collaboration and Networking:**

Study Groups Facilitation: Connecting students with similar academic interests or courses to form study groups, fostering collaboration and peer learning.

Networking Opportunities: Providing information about seminars, workshops, and networking events related to academic disciplines or career paths.

- **Career Development:**

Resume Building and Career Advice: Assisting in crafting resumes, cover letters, and offering guidance on internships, job applications, and career paths.

Interview Preparation: Providing resources and mock interview sessions to help students prepare for job interviews and internship opportunities.

- **Campus Information and Services:**

Campus Navigation: Guiding students around campus, highlighting important locations such as libraries, dining halls, and academic buildings.

Service Information: Offering details on campus services like counseling, health centers, and academic resource centers.

- **Integration and Adaptability:**

Integration with Learning Management Systems: Seamlessly integrating with college platforms to access course materials, grades, and academic updates.

Adaptive Learning: Continuously learning from interactions to personalize recommendations and responses for individual students.