# ISYS30221 Artificial Intelligence 2021-22

## Coursework Documentation Template

## 1- About this submission

|  |  |
| --- | --- |
| Student Name | Alarna Oyaide |
| Student ID | N0931110 |
| Chatbot Topic | Anime |
| Tasks implemented in this submission (a,b,c,or d) | A,B,C |
| Files inventory (excluding this file) | Animekb.csv, generalanime.csv, mybot-basic.py, mybot-basic.xml, CNNmodel.py, saved\_model.pb, keras\_metadata.pb |
| Demo video URL | [Recording-20230323\_112518.webm](https://myntuac-my.sharepoint.com/:v:/g/personal/n0931110_my_ntu_ac_uk/EbXTFqTG9xxJs8UskMII3ssBXWPQ4AuaiGwiT-afM41UrQ?e=0PwA28) |
| Checklist | I will submit this file separately (without compression) into DropBox  All other files are zipped and will be submitted into DropBox  The demo video is recorded as instructed, and the sharing link is inserted above  I have made sure that the demo video is shared according to the instructions, so that I allowed everybody in the university to view it.  All the sections below are populated accordingly. |

## 2- Design notes (shrink/grow as needed, add images where applicable)

|  |  |
| --- | --- |
| General explanations of the system and its goals | The purpose of my system is a functional chatbot which specialises in info about anime. The code is a basic design of a chatbot that uses different agents to generate responses to user inputs. The chatbot uses a Q&A CSV file to provide answers to common questions, and an AIML agent to provide more complex responses. The chatbot also integrates with other APIs to provide additional functionality such as weather reports and Wikipedia summaries. The chatbot can also learn new information from the user and update its knowledge base. The main loop of the chatbot reads user inputs, selects an appropriate response agent, processes the input, and generates a response. The chatbot can also handle special commands such as exiting the program or checking the validity of a statement. |
| The system requirements, i.e., the list of what the system should do/have from a user’s perspective | * Python 3.x interpreter should be installed on the system where the code will run. * Required Python libraries such as wikipedia, nltk, csv, aiml, pandas, json, and requests should be installed. * A valid API key for OpenWeatherMap should be available for using the weather functionality. * A CSV file containing question-answer pairs should be available in the specified format. * A properly formatted AIML file (mybot-basic.xml) should be available for chatbot response generation. * Pandas Library: The code uses the Pandas library to read the CSV file containing the question and answer pairs. * NLTK Library: The code uses the NLTK library to parse logical expressions and perform resolution proofs. You will need to install this library before running the code. |
| The employed AI techniques, and the explanation of program codes and the supplied files. | The code imports several libraries, including wikipedia, nltk, csv, aiml, pandas, json, and requests. These libraries provide various functions, such as reading and writing data to files, processing natural language, and accessing external APIs.    The code initializes a Wikipedia agent, which allows the chatbot to search for and retrieve information from Wikipedia.    The aiml library is used to create an AIML (Artificial Intelligence Markup Language) agent, which can understand and respond to natural language queries.    The code reads in a knowledge base (KB) from a CSV file and stores it in a list called kb. The KB is represented using first-order logic expressions. The code checks the consistency of the KB using a resolution theorem prover from the nltk library.    The code then enters a main loop, where it prompts the user for input, processes the input, and generates a response. The code first checks if the user input matches a question-answer (Q&A) pair in a CSV file. If there is a match, the code returns the corresponding answer. Otherwise, the code uses the AIML agent to generate a response based on the user input.    The code then post-processes the response for commands. If the response begins with a # character, the code interprets it as a command. The code currently supports three commands:     * Exit the chatbot (#0)      * Search Wikipedia for a given query (#1)      * Get the weather information for a given location (#2)     If the response matches a pattern for a command related to updating the KB or checking its consistency, the code updates the KB and checks its consistency using the resolution theorem prover. If the KB is consistent, the code prints a confirmation message. Otherwise, the code prints an error message and retracts the last KB update.    Finally, if the response does not match any command, the code prints the response to the user. |

## 3- Conversation log (insert text, screenshots and/or images as required)

|  |
| --- |
| Task A:      Task B:    Task C:    The CNN model was trained for 10 epochs on the training dataset, with a batch size of 10. During training, the model achieved an accuracy of 35.19% in the first epoch, and the accuracy increased gradually until the 9th epoch, where it reached 50%. However, in the final epoch, the accuracy dropped again to 35.19%. During evaluation, the model achieved an accuracy of 50% and a loss of 0.6947. These results suggest that the model is not performing very well on the given dataset, and may need to be improved or modified to achieve better accuracy. |

(no word count is necessary)