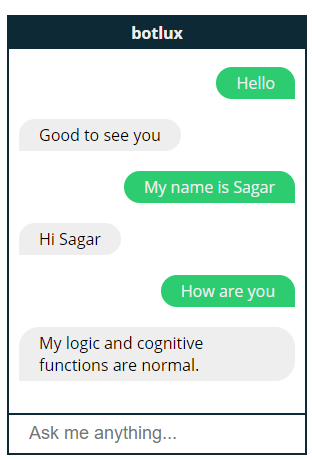
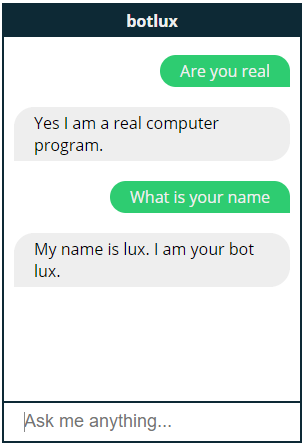
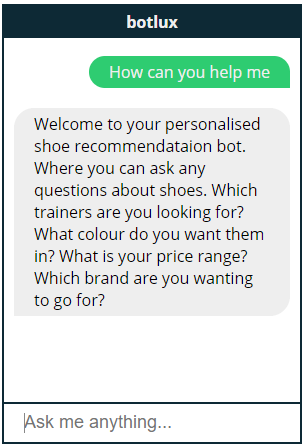
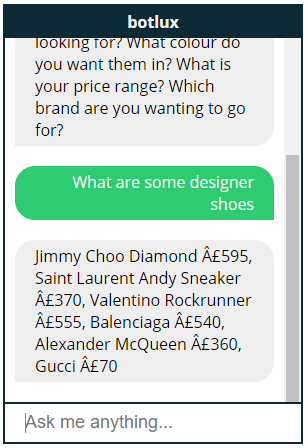
**Conversation Logs**

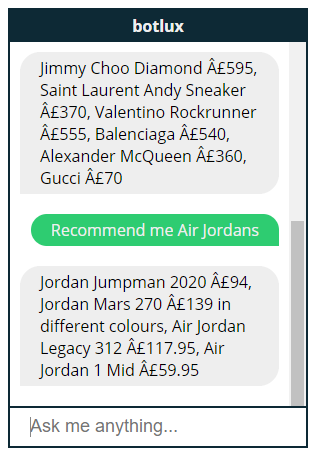
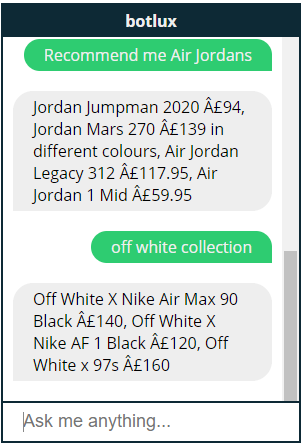
**Rule-based Component**

As you can see here, I am asking the bot questions from the AIML file and getting the correct responses.

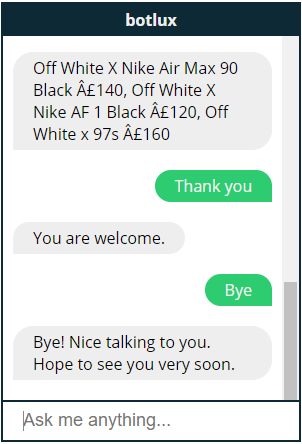


Here I have asked the user a question from the question-answer file and have gotten the correct answer. However, I have phrased the question exactly the same as said in text file so no similarity has been applied.



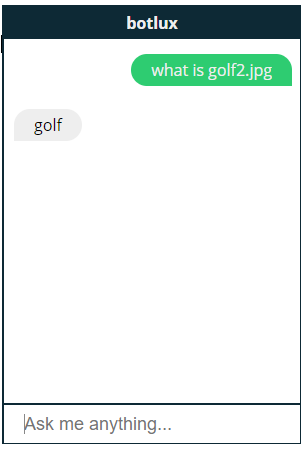
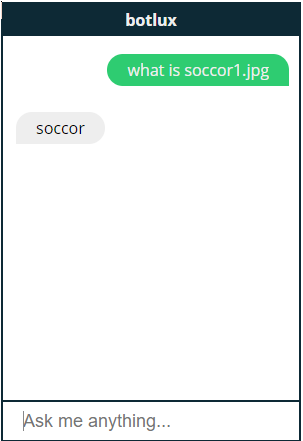
**Similarity-Based Component**

Here I have used just key words from the questions such as “Off white collection” and “recommend air jordans” and the bot has been able to use the similarity component to formulate an answer.



Here also the bot has correctly answered my questions and seems very humanlike.

**Image Classification Component**



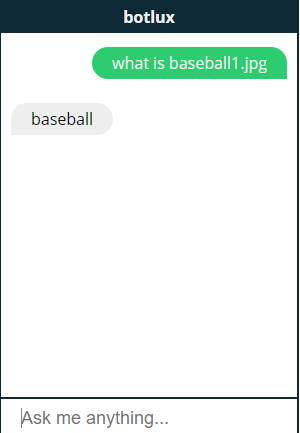
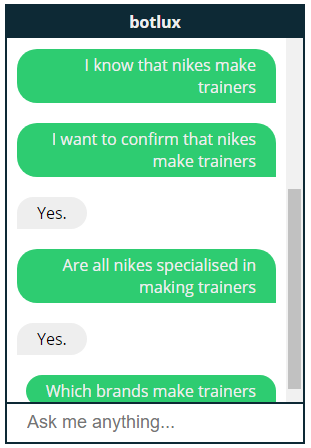
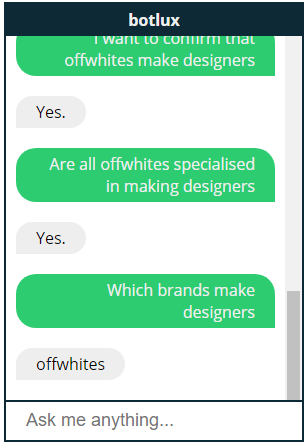
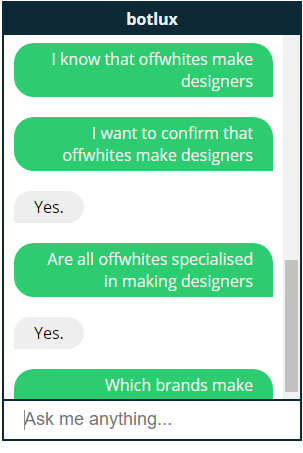
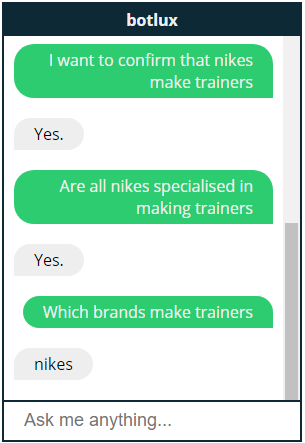
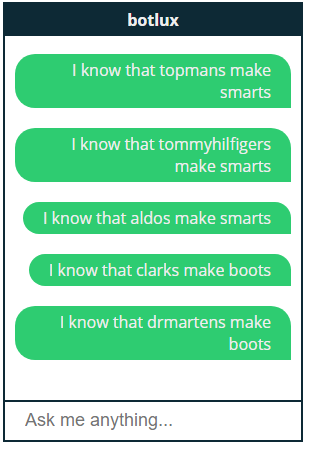
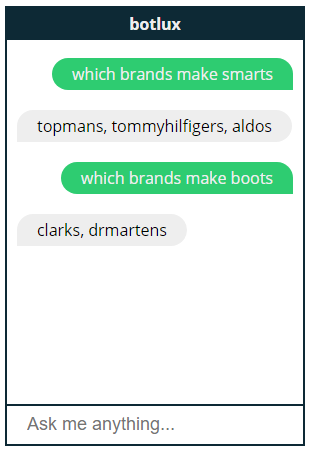


Image classification has now been implemented to the chatbot. This now uses images to detect what sport is being played in the image.

An image with name baseball1.jpg is stored in a folder. The user asks the chatbot what is baseball1.jpg.

The chatbot then uses the pre-trained CNN model built with Keras and Tensorflow. To analyse the image and give a response.

**Toy World Reasoning System**

Toy world reasoning has been implemented into the chatbot now. Using NLTK library and grammar file. Some questions that were in the text files have now been removed. As this new functionality acts like that but smartly.

Here, I have stored x in y and then displayed it with following answers. As shown above. For example, Nikes make trainers (x in y).

The confirming Nike makes trainers (are any x in y?)

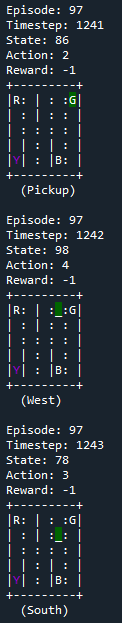
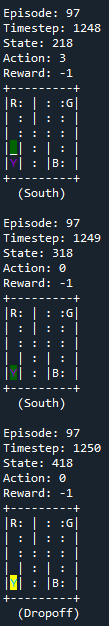
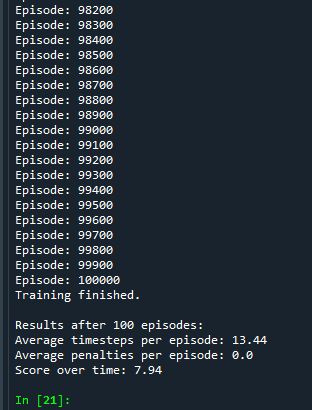
Then confirming Nike specialised in trainers (are all x in y?)

Lastly, it asks what brands make trainers (what is in y?)

Some complex examples can also be seen above on the left.

**Reinforcement Learning with OpenAI**

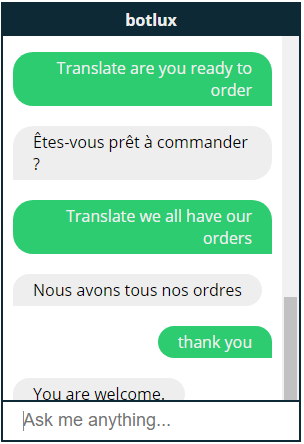
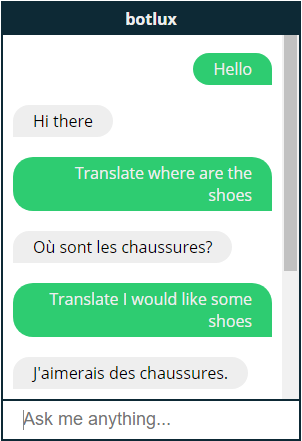
After implementing the feature in a chatbot, after the user enters, where is my delivery. It should show the user where the driver has picked the parcel up and when they are dropping it.

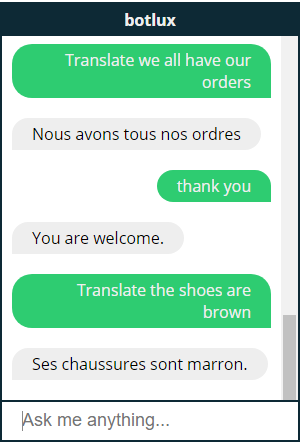




The Reinforcement Learning module has now been added to the chatbot. The deep q network creates a model to allow for each shoe deliver to the user. It helps the user see where their parcel is and when it will come to them.   
  
Above you can see that each delivery has several steps, which is the direction they move into until they reach the drop of destination. After dropping off a parcel they will either get reward or lose points from the reward. As you can see below after the last step of the 99th delivery the open AI module gained 20 points.

Apart from that it is visible that green is the position of the driver. The lines represent a wall so the driver must not pass through there. It needs to go in either direction to reach their destination. And yellow is when the parcel is dropped off.

**Sequence To Sequence Model**



This is built using a sequence to sequence network. Where a CNN model is created based on a dataset. This model allows the chatbot to accurately translate English into French.

There were only limited datasets available, otherwise if a dataset related to shoes was available it would have been better.