eGain SWE Demo

Chatbot to resolve lost package

Design Choices

A chatbot used to track a package should ideally be as simple and straight-forward of a process to provide the best end-user experience. The ideal scenario goes; a user opens the chat, inputs their tracking information, and then is presented with an estimated delivery date and a current location. To flesh out the chatbot a bit, features such as changing delivery address or instructions, canceling order if possible, and reordering if a package is declared lost after a certain amount of time allow for the program to complete its purpose thoroughly and likely resolve any issue that may arise in the package tracking process without involving a representative.

Technical Implementation

Seeing as how the project is simply a barebones demo, it is entirely constructed in python and expected to run in the terminal, but in a production environment, a lot of this data would be retrieved via SQL queries to the database as well as API calls for information such as address validation (google maps API) and other logic that is sudo-coded in via printouts or seed data. Even with the fully functional logic present, integrating it into a website would not be challenging as the foundation as been established and good design/compartmentalization was used to allow for scalable growth of menu options, responses, and features present in this chatbot’s toolbox.

Challenges Faced

The greatest challenge faced when approaching this project was properly planning the scope of the user’s expectations in terms of features present and how each menu should be structured. The point was to make it easy for a user to complete a likely objective put in place when accessing the chatbot, and as such any solution can be achieved with minimal user input as the structure in place presents only necessary and likely options relevant to a given package’s status. It was a bit difficult to decide how to design logic given that no database exists or APIs are present to be called, but a bit of imagination was put into play as print statements output what would likely be expected had said information been available to the backend logic.

Future Improvements

Given more time to fully develop the chatbot, the first changes would be to design a schema befitting the requirements or implement an existing one, adding a user interface and designing a simple API to allow for communication between the two, and lastly adding QOL features and possibly offloading some of the functionality of the chatbot onto the website hosting it, such as notifications on the site for bad weather impacting delivery windows, border closing, etc. Once the logical side had been developed more, a feature that would greatly improve ease-of-use would be transitioning from user input to buttons to eliminate the biggest source of user errors through hard-coded inputs.