CharlieChat

Group members: Jessica Huynh, Heather Lourie, Xinhao Wang, Thomas Zhang

Project goals

Create a web application that allows users to request MBTA information and give possible transportation options for the user, including times and prices. In addition to basic queries, the system will be able to determine information and direct the flow of the conversation from context and the conversation's history. For example, asking how to get to a certain station without specifying a starting location will assume the current location, or asking to get "there" via a cheaper route will find the cheapest route for the previously stated endpoint.

If there is time, we may also adapt the code we have into an AWS lambda function and create an Alexa skill.

What we expect to learn

We will learn to design a simple dialog- based discourse and how to incorporate speech recognition into a usable interface. We will also gain additional experience working in a group and responsibly and fairly dividing tasks.

On an individual level, those who don't have much experience with front-end web development will gain it, including how to call APIs, how to work with basic web technologies, and the client-server method of handling things.

Tools and data

The primary sources of data will be the MBTA's realtime API and the NextBus XML feed, which will provide locations about stops, upcoming buses and trains, and alerts. In addition, for things like the prices of particular modes of transportation not included in the data returned by the realtime API, we can manually keep track of that as fare changes are only done after long periods of public discussion.

Project structure

Front End: a front end framework that will use JavaScript+jQuery, HTML, and CSS; specific framework up in the air and we may not even use one and rely on just the basic web stack

Alexa skills kit: AWS lambda function that adapts the logic of the web app

API handling: to hide our MBTA API key, we can make POST and GET requests on the back end and on the front end call the relevant server methods by exposing some sort of RESTful API

Back End: most feasible options are Python with Django framework (for familiarity) and JavaScript (for ease in integrating with the front end)

Other utils: cloud platform, remote repository, continuous integration - likely a GitHub repo that we push to and deploy on Heroku with that GitHub repo for continuous integration

Task division and timeline

All group members will collaborate together during each step. Below are just primary focus, and may subject to change.

Jessica: Front End

Heather: API handling

Thomas: Alexa skills kit

Xinhao: Back End set up

Week1: initial web setup, get familiar with Django, MBTA API, Alexa skills kit, etc., deploy initial version

Week2: start working on separate functions

Week3: function integration

Week4: add additional features like context handling, etc., testing