TripSage Phase 2
Software Code Documentation

Team Members

Mangalnathan Vijayagopal (mvijaya2)
Nischal Badarinath Kashyap (nkashya)
Amogh Rameshappa Devapura (arames22)
Niranjan Pandeshwar (nrpandes)
Sharath Bangalore Ramesh Kumar (sbangal2)

INTRODUCTION

Tripsage is an UI Based Application which assists users in devising their vacation plans by entering basic information. It also guides the users with the directions to be followed from Origin to Destination. Also included in the application is an important feature of helping the user to find tourist spots in their preferred destinations which are listed and asked to be selected by the user. These destinations are selected along the path devised by the google map which is retrieved from the google API.

The complete development was achieved using the following technologies

- Python3
- Django
- HTML
- CSS
- Google API’s

CODE FUNCTIONALITIES

This section covers major files used in Django and small descriptions of each file with their functionalities.

- Views.py - Backend logic is being written in this file in multiple functions
- urls.py - This is the start point of the project and contains the routes to which the web browser has to be routed once the user enters the URL
- models.py - we currently do not have any database usage in the project and this file contains the section which can be used for the database setup
- templates/ - This folder contains the html templates which have to be rendered with respect to each function's functionality.

In this software code documentation, we majorly concentrate on the view.py file as it contains the major part of the development work.
Now let us see each function listed in view.py along with the functionalities

- def home(request): This renders the homepage of the web url that is localhost:8000
- def getItemsForMapping(city,dict,subtype,locations) and
  def getMapString(city, dict, types, locations): Utility functions used for parsing the YAML and JSON data and also to build google maps URL from name of the place.
- def find_spots(request) : returns an html template with all the tourist spots which would interest the user. This would be decided with respect to the user preference.
- def myfunction(origin,destination): Fires an API query to the Google Cloud Server and receives the directions data in XML Format. This in return is converted into json format in order to have a better accessibility
- def directions(request):this is the major function which triggers all the other functionalities which receives the user inputs. Here we receive the origin destination and preference types. This will then format the directions by adding the Estimated Time of Arrivals at respective direction places. An

html template is rendered with the directions from origin to destination and a select option for the users to select the city they would like to review.

For getting the recommendation on various tourist places / restaurants / recreation places we have implemented an API places_recommendation.py. The API takes place as an input to recommend the places to visit.

- Places_recommendation :
    - Input: place name and type.
- Type:
    - adventures: tourist_attraction, stadium, zoo
    - kids: amusement_park, museum, food
    - relaxing: art_gallery, church, spa
    - other: hospital

FUNCTIONAL TESTING :

For functional testing of the code, a separate testing script is written to validate the output generated by TripSage.
The scripts are written in :
Phase2_TripSage/tripsage/planner/functionaltesing.py

The original code has been copied into the functional_testing.py.All the HTML and CSS dependencies have been removed in order to have an independent execution of the functional test. Initially there are two test cases that have been used for validation of the software.

Test Case 1: Here an origin and destination is parsed into the function and the google api is fired to receive the directions to be followed from origin to destination. This is further verified with our pre-configured values

Test Case 2: Here the preference types of users is initially given to the api which computes the tourist hot spots in the destination city which is by default considered to be the prefered viewpoint of the user. The received output is further verified with our pre-configured values.