**Trip Scheduler**

**Introduction**

If I were going on a trip, as a first step, I would try to gather a list of tourist attractions in that city. Then I would choose the ones that I was interested in. For example, if I was interested in culture then I would plan on visiting Museums. On the other hand, if I was interested in Art I would visit more Art Galleries in that city. The most attractive places visit in that city, would vary from person to person.

Once the places of interest were identified, I would sort the list then I would plan on the details like which places can be visited on which day etc. It would make sense to group places that are located near each other and visit them together in the same day. Geographic location of the places being visited would be used to make this possible.

The Trip Scheduler would help the tourist develop this kind of a plan. The Tourist would enter the address of the city and the number of days he/she would be staying in the city. Trip Scheduler would analyze this input and produces the following results:

1. List of Tourist attractions that can be visited on each day of their stay.
2. City Map with the locations of the tourist attractions marked. The places would be marked with colors and each color representing the day on which the places are visited.

## Data

GEOCODERS data is used to identify the latitude and longitude of the City.

The following FOURSQUARES data are needed by the tourist scheduler tool:

1. List of venues in a specified a geographic location.
2. Categories of each venue. This is required since the tool is going to produce the result based on the user choice of categories.
3. Location data (latitude and longitude), and rating information for each venue. This is required to pick the top N places to visit and to categorize them based on the proximity to each other.

Foursquare data is the main dataset being used to get the above data.

1. Venues/Categories API: This API provides the list of Venue Categories supported in the dataset. The tourism related Categories are selected from the received response and displayed to the user. The user can then select their interested categories from this list. For example, assume the response from the dataset contains the following categories: Restaurants, Shopping Malls, Museums, Sports Activities, Art Galleries, Parks and Palaces. The tourist could choose, Museum, Art Galleries and Parks as interested categories. The tool would select places only from these categories.
2. Venues/Explore API: This API provides the list of venues that matches the selected categories in the specific location. The fields of interest are Venue name, Venue ID, Venue Category, Venue Latitude and Venue Longitude.
3. Venues/Venue ID API: This API is used to get the details of each venue. The fields of interest are ‘Rating’ and the ‘Likes’. The Venues are then sorted based on this ‘Rating’ and ‘Likes’ values. From this sorted list, the top N venues are picked up.

The top N venues would be segmented using the K-Means clustering algorithm. The attributes used for clustering would be the latitude and longitude of the venue in order to get the tourist places segmented based on the geographical location. Each segment represents the day on which the places are visited.

## Methodology

### Step 1: Import needed libraries, turn off warnings, load foursquare credentials

### Step 2: Input the location to visit (city), number of days, and categories

### Step 3: Get the latitude and longitude of the location (Geocoders)

### Step 4: Get the list of supported venue categories (Foursquare)

### Step 5: Get applicable tourism categories

### Step 6: Get the list of Venues matching the tourism categories

### Step 7: Convert to Dataframe with columns, Venue Name, ID, Category, Latitude and Longitude.

### Step 8: Get the details of each travel venue

#### Not implemented because it requires multiple calls to the venue/venueID api and venue/venueID calls are in Sandbox account.

### Step 9: Add the columns 'Rating' and 'Likes' to the dataframe based on the values received from the Venue details

#### Not implemented because it requires multiple calls to the venue/venueID api and venue/venueID calls are in Sandbox account.

### Step 10: Sort the dataframe based on the Rating and Likes. Pick up top N travel venues.[¶](https://render.githubusercontent.com/view/ipynb?commit=b95803db6086ebec862d859824f2e9372673133a&enc_url=68747470733a2f2f7261772e67697468756275736572636f6e74656e742e636f6d2f6472626167756961722f636f75736572612d63617073746f6e652d6e65772f623935383033646236303836656265633836326438353938323466326539333732363733313333612f43617073746f6e65322e6970796e62&nwo=drbaguiar%2Fcousera-capstone-new&path=Capstone2.ipynb&repository_id=147009055&repository_type=Repository#Step-10:-Sort-the-dataframe-based-on-the-Rating-and-Likes.--Pick-up-top-N-travel-venues.)

### Step 11: Form a new dataframe by dropping everything except Lat and Long

### Step 12: Use K-Means clustering to segment venues based on the Latitude and Longitude values

### Step 13: Display the list of venues (for each day for 3 days)

### Step 14: Use Folium to visualize the travel venues on the City map.¶

## Results

#### Day 1

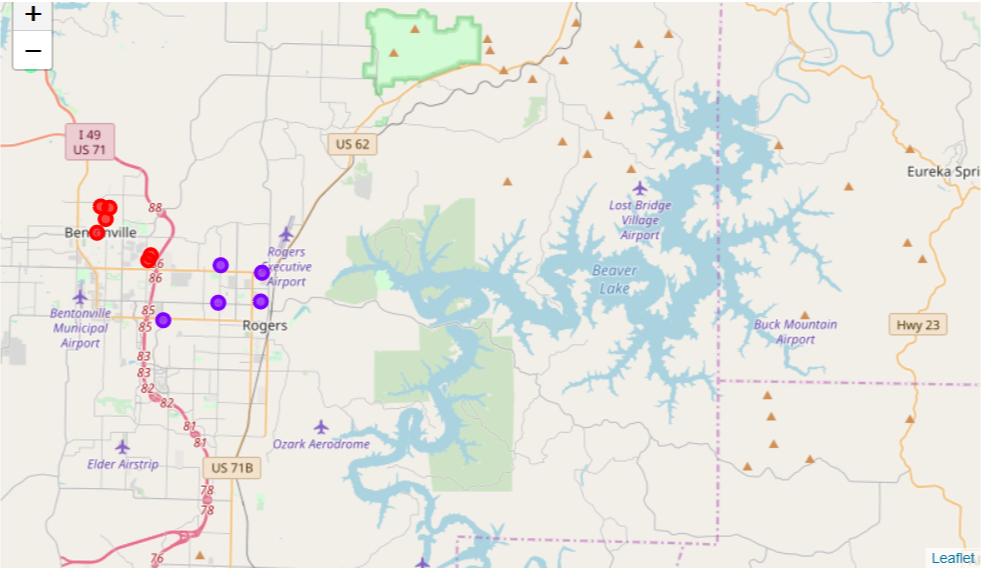
|  | **Name** | **Category** |
| --- | --- | --- |
| **12** | Buckminster Fuller's Fly's Eye Dome | Museum |
| **18** | The Land of Happy Monkeys | Zoo |
| **86** | Park Street Baptist Church | Church |
| **103** | S.O.D POD | Shrine |
| **27** | Crystal Bridges Trail | Trail |
| **71** | First United Methodist Church Bentonville | Church |

#### Day 2

|  | **Name** | **Category** |
| --- | --- | --- |
| **77** | The Church of Jesus Christ of Latter-day Saints | Church |
| **15** | Daisy Airgun Museum | Museum |
| **41** | Foerster Park | Park |
| **89** | oakley chapel | Church |
| **104** | Church | Church |

#### Day 3

|  | **Name** | **Category** |
| --- | --- | --- |
| **57** | Bentonville Bike Trail | Park |



## DISCUSSION

The number of places to be covered on any day can be longer than the number we would like to visit because only latitude and longitude were used in the K-Means algorithm to do the clustering.

Future improvements could add options such as hours to visit at a venue, number of hours users spend on average at each venue, etc, then the algorithm could do further splits on this list.

## CONCLUSION

When someone is planning a trip to an unknown city, the planning becomes difficult due to the lack of knowledge about the city. This tool can help. The tool uses preference of venue categories, number of days stay, ratings of venues, and venue likes. This helps the traveler by making it easy for the user to make a schedule without going to multiple web pages.