Project Proposal

# Project Name

A Geographic Search Engine for travelling guidance.

# Project Team members

* Alexander Nestle ([nestle2@illinois.edu](mailto:nestle2@illinois.edu))
* Akhmad Rahadian Hutomo ([ahutomo2@illinois.edu](mailto:ahutomo2@illinois.edu))
* Yang Liu (yangl18@illinois.edu)

# Goal

The goal of this project is to create a search engine implementation that will allow users to query a database of businesses, places, and parks (which will be referred to as ‘places’ within this document) using natural language attributes such as ‘child friendly’, ‘Italian food’, or ‘beautiful building’ in order to quickly receive relevant places.

The Data-User-Service (DUS) Triangle for this project will be as follows:

# Summary

## View

The application will be a web application. The interface would only need to receive the user query and display the results. The places in the results will have a URL and/or Geo location that can be used to link the user to relevant pages such as Google Maps or business web page.

A web application would use a SPA framework such as React, Vue, or Angular. It will also use Bootstrap framework

As an option, a desktop application may be built using Python (one of the various desktop frameworks), or Java (Swing or FX) to ensure that it is cross platform.

## Backend

For a web application, simple API could be created for the front end to query. Python is the language best equipped for data interpretation, which would use Flask framework. However, I am most familiar with C# and ASP.NET Core.

Another option would be to create a python desktop application, which may be simpler and present less problems for us.

## Database and Queries

Currently, the place names and addresses were scraped from Facebook travel page recommendations, which were then used to query the Google Places API to receive place information and reviews. We may also collect more place/reviews from different sources (say, yelp, Kaggle) for any upcoming iterations of projects.

Topic modeling will be used to create the ‘queryable attributes’ of the places. This will be done by running the topic modeling algorithm (say LDA algorithm) against all the collected reviews. This will create the queryable topics (place attributes). During the topic modeling process, we will need to manually interpret the topics and name them appropriately.

Once the topics are created, we will run each review through our topic model. This will tell us the topics that each review included, and the most common topics will be used to describe each place, which is then queryable by the user.

Once the places are labeled with the topics, we will use TF-IDF and inverted index to enable fast user queries. This would be done by treating each list of place topics as a document.

The storage of the database was initially designed through a simple json file. But may subject to change to NoSQL database if the volume of data increases during project implementation and enhancement.

Example of initial JSON datasets:

{

"formatted\_address": "Brooklyn, NY 11201, USA",

"geometry": {

"location": {

"lat": 40.6896147,

"lng": -73.9858984

},

"viewport": {

"south": 40.6882965197085,

"west": -73.9872249802915,

"north": 40.6909944802915,

"east": -73.98452701970848

}

},

"name": "Brooklyn",

"place\_id": "ChIJq1RGWExawokRC5VK08ax-Ds",

"reviews": [

{

"author\_name": "hisham nofal",

"author\_url": "https://www.google.com/maps/contrib/114083506813599492082/reviews",

"language": "en",

"profile\_photo\_url": "https://lh3.ggpht.com/-2hGHNr4CZDQ/AAAAAAAAAAI/AAAAAAAAAAA/CN\_PCdB3E8E/s128-c0x00000000-cc-rp-mo-ba6/photo.jpg",

"rating": 4,

"relative\_time\_description": "11 months ago",

"text": "For bus is good",

"time": 1538667162

}

],

"types": [

"transit\_station",

"point\_of\_interest",

"establishment"

],

"url": "https://maps.google.com/?cid=4321399309968512267",

"html\_attributions": []

}

## Data Cleansing

A few data cleansing tasks has been done to the initial dataset to ensure there are no duplicates of places/ business entries, the place has a certain threshold of number of reviews and also count of the words inside the reviews

# Tasks

* Creating the dataset by scraping names from Facebook Travel Pages and querying Google Places API (completed for the first iteration)
* Implement data cleansing
* Developing topic modeling algorithm like LDA
* Manually label topics
* Function to compare each review for each place against the topic model and save the resulting topics to place
* Create inverted index for fast queries
* Evaluate and implement TF-IDF ranking function
* Develop Front end view
* Develop Backend implementation framework
* Create System Documentation

# Potential Problems

## Application Stack Restrictions

The assignment needs to be uploaded to a public repository for the TAs and professor to access. I am assuming they will both look at the code and run the project on their local system. We will need to account for this during develop and ensure that if we choose a web application that it is easily run locally.

## Data Cleansing

We may need to clean the data, such as removing review that are too small or contain erroneous text, to ensure that our topics are accurate.

## Database File Size

The JSON database file could become too large for memory. I do not forsee this happening as the test database I created contain around 2,000 places and 10,000 reviews and it was only 7.5 MB.