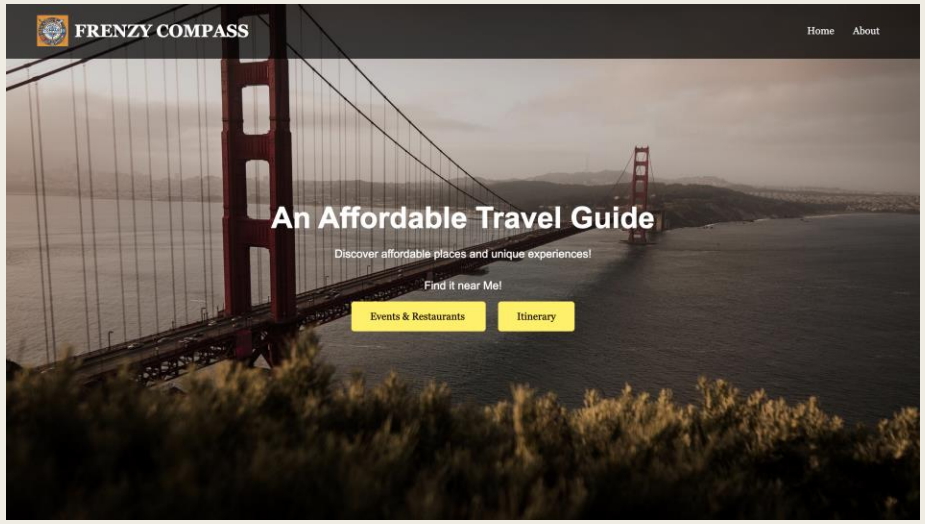
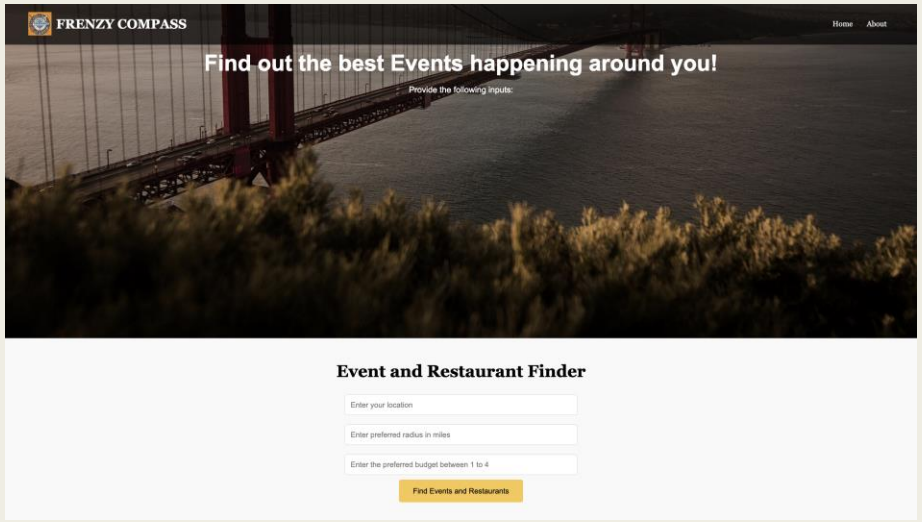
FRENZY COMPASS

An Affordable Travel Guide

Varun Rao, Garima Mathur, Amit Gangane

Introduction





The Frenzy Compass – An Affordable Travel Guide is a software that will help students efficiently plan their day out by suggesting them nearby activities, restaurants, and essential services based on their budget and locality when they are new to a city.

Location-Based Recommendations : Suggest nearby places (restaurants, attractions, events, grocery).

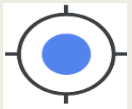
Budget & Preference Filters : Allow filtering based on budget and activity type.

Travel Time Optimization : Use Google Maps API to recommend the shortest travel routes.

APIs used:

Google Map APIs:

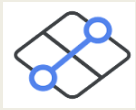
JavaScript Map API: Embeds dynamic maps on the web interface for user Interaction.



Geocoding API: Converts addresses into geographic coordinates for precise location search.

Places API: Finds nearby places like restaurants and attractions, filtered by user preferences.

Geolocation API: Detects the user's current location for personalized recommendations.



Distance Matirx API: Provides travel time and distance for multiple destinations..

Ticketmaster API:

Ticketmaster API: The Ticketmaster Discovery API allows you to search for events, attractions, or venues.

Business Problems and Solution

1. Finding Budget-Friendly Dining and Activities

- **Business Problem**:
International students like us who have just come to a new city often struggle to find affordable dining and to do activities in a new city.

- **Solution**:
The software solution will suggest nearby restaurants and activities based on the student's budget using Google Places API and Geolocation API, filtering options by price, rating etc.

2. Optimizing Travel Time

- **Business Problem**:
Students usually waste time traveling between distant attractions when they arrive in a new city, usually during the days when they have no classes.

- **Solution**:
Using our software which utilises Google Map API we will calculate the best suitable itinerary for an individual based on the selected activities, saving time by considering real-time data by giving nearest activity list.

3. Essential Services

- **Business Problem**:
Students usually need quick access to services like groceries etc to meet their daily needs but are unaware where they can find one especially in the initial days when they arrive in a new city.

- **Solution**:
The software uses Google Places API to find essential services nearby, ensuring easy access to essential services while traveling.

Design and Architecture

User Input
- Location
- Preferences (budget, activity type, distance)

```python
## this function will handle user input and output.
def main():
    print("Welcome to the Frenzy Compass - An Affordable Travel Guide!")

    location = input("Please enter a city name or ZIP code: ")

    while True:
        try:
            radius_miles = float(input("Please enter the search radius in miles: "))
            if radius_miles <= 0:
                raise ValueError("Distance must be positive.")
            break
        except ValueError as ve:
            print(f"Invalid input: {ve}. Please try again.")

    while True:
        try:
            budget = int(input("Please enter your budget level for restaurants (1=Inexpensive(under $10),
            if budget not in range(1, 5):
                raise ValueError("Budget level must be between 1 and 4.")
            break
        except ValueError as ve:
            print(f"Invalid input: {ve}. Please try again.")
```

```python
## This function will fetch the information of nearby restaurants based on user input using Google Maps API.
def find_nearby_restaurants(location, maxdistance_miles, budget):
    lat, lng = get_lat_long(location)
    if not lat or not lng:
        return "Invalid location. Please enter a valid city name or ZIP code."

    max_distance = miles_to_meters(maxdistance_miles)

    try:
        places_result = gmaps.places_nearby(location=(lat, lng), radius=max_distance, type='restaurant')
        if not places_result.get('results'):
            return f"No restaurants found near {location} within {maxdistance_miles} miles."

        restaurants = [
            {
                'name': place['name'],
                'address': place.get('vicinity', 'Address not available'),
                'price_level': place.get('price_level', 'Unknown'),
            }
            for place in places_result['results']
            if place.get('price_level') is None or place['price_level'] <= budget
        ]

        return restaurants if restaurants else f"No restaurants found within budget level {budget}."
    except Exception as e:
        return f"An error occurred while fetching restaurants: {e}"
```

Backend Processing (Python & API)
- Receive and validate input
- Query Google Maps APIs such as
  - **Places API** → Fetch nearby locations
  - **Distance Matrix API** → Calculate travel times
  - **Geocoding API** → Convert addresses
- Filter & optimize recommendations (distance, budget, activity)

Response Generation
- Sort and return optimized results

```python
restaurant_results = find_nearby_restaurants(location, radius_miles, budget)
if isinstance(restaurant_results, list):
    print(f"List of restaurants near {location} within {radius_miles:.2f} miles (Budget Level {budget}):\n")
    for i, restaurant in enumerate(restaurant_results, start=1):
        print(f"{i}. {restaurant['name']}\n   Address: {restaurant['address']}\n   Price Level: {restaurant['price_lev
else:
    print(restaurant_results)
```

Outputs:

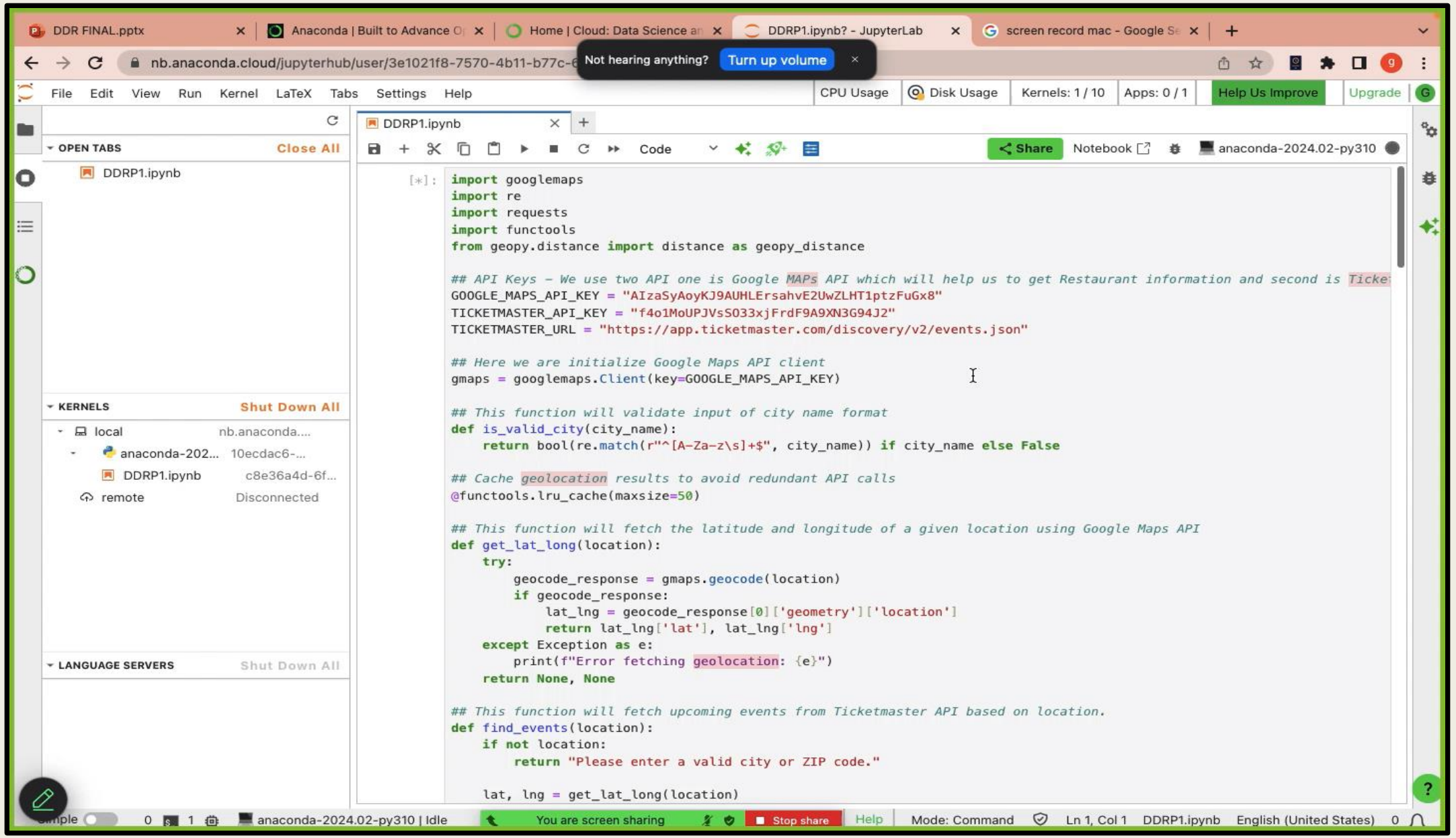
```
Welcome to the Frenzy Compass - An Affordable Travel Guide!

Please enter a city name or ZIP code:  94109
Please enter the search radius in miles:  4
Please enter your budget level for restaurants (1=Inexpensive(under $10),
2=Moderate($20-$40), 3=Expensive($50-$100), 4=Very Expensive(over $100)):  3
```

```
Restaurants near 94109 within 10.00 miles (Budget Level 3):

1. Fairmont San Francisco
   Address: 950 Mason Street, San Francisco
   (Unknown)
2. The Stinking Rose
   Address: 430 Columbus Avenue, San Francisco
   (Price Level: 3)
3. Hotel Shattuck Plaza
   Address: 2086 Allston Way, Berkeley
   (Unknown)
4. Perbacco
   Address: 230 California Street, San Francisco
   (Price Level: 3)
5. One Market Restaurant/Mark 'n Mike's NY Style Deli
   Address: 1 Market Street, San Francisco
   (Price Level: 3)
6. Zuni Café
   Address: 1658 Market Street, San Francisco
   (Price Level: 3)
7. Humphry Slocombe
   Address: One Ferry Building #8, San Francisco
   (Price Level: 2)
8. Yoshi's
   Address: 510 Embarcadero West, Oakland
   (Price Level: 2)
9. Marlowe
   Address: 500 Brannan Street, San Francisco
   (Price Level: 2)
10. Wayfare Tavern
   Address: 558 Sacramento Street, San Francisco
   (Price Level: 3)
```

```
Upcoming events near 94109:

1. Liquid Stranger w/ Ahee
   Date: 2025-02-07
   Venue: The Regency Ballroom
   Address: 1290 Sutter Street
   Tickets: https://www.ticketmaster.com/event/Z7r9jZ1A7oAaZ

2. Liquid Stranger - 18+
   Date: 2025-02-08
   Venue: The Regency Ballroom
   Address: 1290 Sutter Street
   Tickets: https://www.ticketmaster.com/event/Z7r9jZ1A7o7A8

3. Cordae
   Date: 2025-02-12
   Venue: The Regency Ballroom
   Address: 1290 Sutter Street
   Tickets: https://www.ticketmaster.com/event/Z7r9jZ1A7o0Fd

4. DHRUV
   Date: 2025-02-15
   Venue: Great American Music Hall
   Address: 850 O'Farrell St.
   Tickets: https://www.ticketmaster.com/event/Z7r9jZ1A78o-M

5. Malaa (18 & Over)
   Date: 2025-02-15
   Venue: The Regency Ballroom
   Address: 1290 Sutter Street
   Tickets: https://www.ticketmaster.com/event/Z7r9jZ1A7oE73
```

```
1. Trader Joe's
   Address: 555 9th Street, San Francisco
   Price Level: 2

2. Rainbow Grocery Cooperative
   Address: 1745 Folsom Street, San Francisco
   Price Level: 1

3. Trader Joe's
   Address: 3 Masonic Avenue, San Francisco
   Price Level: 2

4. Whole Foods Market
   Address: 690 Stanyan Street, San Francisco
   Price Level: 3

5. Whole Foods Market
   Address: 3950 24th Street, San Francisco
   Price Level: 3

6. 99 Ranch Market
   Address: 250 Skyline Plaza, Daly City

7. Trader Joe's
   Address: 401 Bay Street, San Francisco
   Price Level: 2

8. Bi-Rite Market
   Address: 3639 18th Street, San Francisco
   Price Level: 1

9. Trader Joe's
   Address: 265 Winston Drive, San Francisco
   Price Level: 2

10. Whole Foods Market
   Address: 399 4th Street, San Francisco
   Price Level: 3
```

Demo:


DDR FINAL.pptx
Anaconda | Built to Advance
Home | Cloud: Data Science
DDRP1.ipynb? - JupyterLab
screen record mac - Google
Not hearing anything?
Turn up volume
nb.anaconda.cloud/jupyterhub/user/3e1021f8-7570-4b11-b77c-6
File Edit View Run Kernel LaTeX Tabs Settings Help
CPU Usage
Disk Usage
Kernels: 1 / 10
Apps: 0 / 1
Help Us Improve
Upgrade
OPEN TABS
Close All
DDRP1.ipynb
KERNELS
Shut Down All
local nb.anaconda....
anaconda-202... 10ecdac6-...
DDRP1.ipynb c8e36a4d-6f...
remote Disconnected
LANGUAGE SERVERS
Shut Down All
DDRP1.ipynb
Code
Share
Notebook
anaconda-2024.02-py310
[*]:
import googlemaps
import re
import requests
import functools
from geopy.distance import distance as geopy_distance
API Keys - We use two API one is Google MAPs API which will help us to get Restaurant information and second is Ticke
GOOGLE_MAPS_API_KEY = "AIzaSyAoyKJ9AUHLErsahvE2UwZLHT1ptzFuGx8"
TICKETMASTER_API_KEY = "f4o1MoUPJVsSO33xjFrdF9A9XN3G94J2"
TICKETMASTER_URL = "https://app.ticketmaster.com/discovery/v2/events.json"
Here we are initialize Google Maps API client
gmaps = googlemaps.Client(key=GOOGLE_MAPS_API_KEY)
This function will validate input of city name format
def is_valid_city(city_name):
return bool(re.match(r"^[A-Za-z\s]+$", city_name)) if city_name else False
Cache geolocation results to avoid redundant API calls
@functools.lru_cache(maxsize=50)
This function will fetch the latitude and longitude of a given location using Google Maps API
def get_lat_long(location):
try:
geocode_response = gmaps.geocode(location)
if geocode_response:
lat_lng = geocode_response[0]['geometry']['location']
return lat_lng['lat'], lat_lng['lng']
except Exception as e:
print(f"Error fetching geolocation: {e}")
return None, None
This function will fetch upcoming events from Ticketmaster API based on location.
def find_events(location):
if not location:
return "Please enter a valid city or ZIP code."
lat, lng = get_lat_long(location)
anaconda-2024.02-py310 | Idle
You are screen sharing
Stop share
Help
Mode: Command
Ln 1, Col 1
DDRP1.ipynb
English (United States)

Limitation and Future Scope

As of now the limitation for the project is that we can depict only three variability, we aimed to make this software a one stop for all. Some more limitations that we address are:

- **Google Maps Dependency:** Limited coverage and potential high costs for API usage.
- **Data Accuracy:** Reliance on third-party data for places and reviews may lead to inaccuracies.
- **Dynamic Pricing:** Budget-based recommendations may be impacted by fluctuating prices.
- **Limited Filtering:** Basic filters may not cover all user preferences (e.g., dietary restrictions).

The future scope for this project are:

- **Live Updates:**
  Implementing live data for events, closures, and offering push notifications.
- **Expansion:**
  Extend service to more cities, add offline capabilities for limited internet access.
- **Payment & Booking:**
  Enable direct booking and payment for activities, restaurants, and events.


THANK YOU