Location Finder

Problem statement:

This website shows you the details of the locations mentioned in the drop down list.It takes Initial location as RCOEM and finds nearby Locations as per the categories mentioned

Explanation of project:

This Python project is a location finder that uses geographical data to identify and display nearby locations on a map based on a given category. Here's a breakdown of its functionality:

Imports: The project imports necessary libraries like pandas for data manipulation, geopy for distance calculation, ipyleaflet for interactive maps, ipywidgets for interactive widgets in Jupyter Notebook, and IPython.display for HTML display.

Custom CSS Styling: Defines custom CSS styles for the widgets used in the project.

Functions:

load_locations_dataset(): Loads a dataset containing location information.
filter_locations_by_category(df, category): Filters locations by a given category from the dataset.
get_nearby_locations(user_location, locations_df, radius=5.0): Finds nearby locations within a specified radius from a given user location using latitude and longitude coordinates.
display_locations_on_map(user_location, locations): Displays locations on an interactive map using ipyleaflet's Map and Marker components.
Displaying Custom Styles and Information: Displays custom HTML styling and information about the project, default location, and its functionality.

Loading Location Data:

Loads the location dataset (location_dataset.csv) using the load_locations_dataset() function.
Creates a dropdown widget (Dropdown) for selecting different categories of locations like hospitals, petrol pumps, food places, and bank/ATMs.
Event Handling:

Defines a function on_button_clicked(b) which is triggered when the "Show Map" button is clicked.
Retrieves the selected category from the dropdown.
Filters locations based on the selected category using filter_locations_by_category().
Calculates nearby locations using get_nearby_locations() based on default user location (set to RCOEM) and the filtered locations.
Displays the nearby locations on the map using display_locations_on_map().
Display Widgets:

Displays the category dropdown (category) and the "Show Map" button (button) created using ipywidgets.
Sets up the button click event to trigger the processing and display the map.
Overall, this Python project combines pandas for data manipulation, geopy for geospatial calculations, ipyleaflet for interactive mapping, and ipywidgets for creating an interactive UI within Jupyter Notebook to find and display nearby locations based on categories.

Library and Dataset used:

Here's a list of the libraries used in the code:

Pandas (import pandas as pd):
Pandas is a powerful data manipulation and analysis library in Python.

Geopy (from geopy.distance import geodesic):
Geopy is a library used for geocoding (converting addresses into geographic coordinates) and calculating distances between coordinates.

ipyleaflet (from ipyleaflet import Map, Marker):

ipyleaflet is an interactive maps library for Jupyter Notebooks.

ipywidgets (import ipywidgets as widgets):
ipywidgets provides interactive HTML widgets for Jupyter Notebooks. It's used in this code to create interactive dropdown menus (widgets.Dropdown) and buttons (widgets.Button) for user interaction.

IPython.display (from IPython.display import display, HTML):

IPython.display provides a way to create and display HTML content within IPython environments, useful for customizing the display of content in Jupyter Notebooks.

Dataset:

category	name	latitude	longitude
hospital	Alexis Multispeciality Hospital	21.1858	79.0795
hospital	VIMS Hospital	21.15762	79.08241
hospital	Central Govt Health Scheme	21.170278	79.066586
hospital	Kunal Hospital	21.18361	79.07919
petrol_pump	Reliance Petrol Pump	21.173607	79.061308
petrol_pump	Bharat Petroleum	21.174902	79.056391
petrol_pump	Indian Oil	21.178329	79.069714
petrol_pump	HP Petrol Pump	21.174661	79.055336
food_place	Haldirams	21.1636	79.0801
food_place	Pizza Hut	21.171469	79.079356
food_place	Dominos	21.181126	79.046067
food_place	Zero Degrees	21.164254	79.079039
food_place	Hi Pepper Pizza	21.164946	79.079803
Bank_Atm	Induslnd	21.170937	79.067344
Bank_Atm	lakshmi vilas	21.173666	79.070696
Bank_Atm	Axis bank	21.178647	79.06263
Bank_Atm	Uco bank	21.176699	79.061268
Bank_Atm	Bank of barodra	21.180901	79.059481

Code:

```
import pandas as pd
```

```
from geopy.distance import geodesic
from ipyleaflet import Map, Marker
import ipywidgets as widgets
from IPython.display import display, HTML

# Custom CSS for styling
custom_css = """
    .widget-label {
        color: #336699;
        font-weight: bold;
    }
    .widget-text {
        width: 200px;
        padding: 5px;
        margin-bottom: 10px;
    }
    .widget-dropdown {
        width: 200px;
        padding: 5px;
        margin-bottom: 10px;
        background-color: #f2f2f2;
    }
    .widget-button {
        padding: 10px 20px;
        background-color: #336699;
        color: white;
        border: none;
        cursor: pointer;
    }
    .widget-button:hover {
        background-color: #005f8b;
    }
"""

def load_locations_dataset():
    dataset_path = "location_dataset.csv"  # Replace this with your
dataset path
    return pd.read_csv(dataset_path)

def filter_locations_by_category(df, category):
    return df[df['category'] == category]

def get_nearby_locations(user_location, locations_df, radius=5.0):
```

```
    nearby_locations = []
    for index, row in locations_df.iterrows():
        location_coords = (row['latitude'], row['longitude'])
        distance = geodesic(user_location,
location_coords).kilometers
        if distance <= radius:
            nearby_locations.append({
                'name': row['name'],
                'category': row['category'],
                'latitude': row['latitude'],
                'longitude': row['longitude'],
                'distance_km': distance
            })
    return nearby_locations

def display_locations_on_map(user_location, locations):
    map_center = user_location
    m = Map(center=map_center, zoom=12)

    for loc in locations:
        marker = Marker(location=(loc['latitude'], loc['longitude']),
title=loc['name'])
        m.add_layer(marker)

    display(m)

# Applying custom styles to widgets
style = HTML('<style>' + custom_css + '</style>')
display(style)

# Default latitude and longitude
default_latitude = 21.1766
default_longitude = 79.0616

# Displaying heading and default location
display(HTML('<h1>Location Finder</h1>'))
display(HTML('<h2>Initial locationis set to RCOEM</h2>'))
display(HTML('<h3>This website shows you the details of the
locations mentioned in the drop down list.It takes Initial location as
RCOEM and finds nearby Locations as per the categories
mentioned </h3>'))
display(HTML(f'<p>Default location: Latitude {default_latitude},
Longitude {default_longitude}</p>'))
```

```
# Load the locations dataset
locations_df = load_locations_dataset()

# Example categories
categories = ['hospital', 'petrol_pump', 'food_place', 'Bank_Atm']
category = widgets.Dropdown(description='Category:',
options=categories, style={'description_width': 'initial'})

# Displaying the category dropdown
display(category)

def on_button_clicked(b):
    user_location = (default_latitude, default_longitude)
    cat = category.value

    # Filter locations by category
    filtered_locations = filter_locations_by_category(locations_df,
cat)

    # Get nearby locations
    nearby_locations = get_nearby_locations(user_location,
filtered_locations)

    # Display nearby locations on map
    display_locations_on_map(user_location, nearby_locations)

# Button for triggering the processing
button = widgets.Button(description="Show Map",
style={'button_color': '#336699'})
button.on_click(on_button_clicked)
display(button)
```

Output:

Location Finder

Initial locationis set to RCOEM

This website shows you the details of the locations mentioned in the drop down list.It takes Initial location as RCOEM and finds nearby Locations as per the categories mentioned

Default location: Latitude 21.1766, Longitude 79.0616

Category: food_place

Show Map