Test Plan for Google Maps (Geocoding) API

- **Objective**: To thoroughly validate the functionality, performance, and reliability of the Google Maps API endpoints i'll be using (geocode and nearbysearch).

- **Scope:** This testing will focus on the specified API endpoints:

  Geocoding:

    - Forward geocoding (address to coordinates):

      https://maps.googleapis.com/maps/api/geocode/json?address={address}&key={{API_KEY}}

    - Reverse geocoding (coordinates to address):

      https://maps.googleapis.com/maps/api/geocode/json?latlng={latitude},{longitude}&key={{API_KEY}}

  Nearby Search:

    - Finding nearby places by type and radius:

      https://maps.googleapis.com/maps/api/place/nearbysearch/json?location={latitude},{longitude}&radius={radius}&types={type}&key={{API_KEY}}

- **Schedule:** Testing will be conducted in an iterative manner, with each test case executed and results documented.

- **Resources:**

    - Google Maps API documentation (https://developers.google.com/maps/get-started)
    - Postman (https://www.postman.com/) for sending API requests and inspecting responses

Test Cases:

Test Case ID	**Description**	**Expected Result**	**Pass/Fail Criteria**
TC_01	Geocode API - Valid address	-Response status code: 200 OK -Response includes essential location data (latitude, longitude)	- Response contains a valid JSON object - Address components are mapped correctly

			(street, city, country, etc.) - Error message is absent
TC_02	Geocode API - Invalid address	-Response status code: 400 Bad Request	- Response contains an error message indicating the issue (e.g., "ZERO_RESULTS") - Error message is clear and informative
TC_03	Geocode API - Missing API key	-Response status code: 401 Unauthorized	- Response contains an error message indicating a missing or invalid API key - Error message guides the user to obtain a valid API key
TC_04	Nearby Search API - Valid location and radius	-Response status code: 200 OK Response includes an array of nearby restaurants (within the specified radius) - Each restaurant entry contains relevant details (name, rating, address)	- Response contains a valid JSON object
TC_05	Nearby Search API - Invalid location	-Response status code: 400 Bad Request	- Response contains an error message indicating the issue (e.g., "INVALID_REQUEST") - Error message is clear and informative
TC_06	Nearby Search API - Missing location parameter	-Response status code: 400 Bad Request	- Response contains an error message indicating a missing location parameter - Error message guides the user on how to provide a valid location

TC_07	Nearby Search API - Invalid radius value	-Response status code: 400 Bad Request	- Response contains an error message indicating an invalid radius value - Error message suggests a valid radius range (e.g., 1 to 50000 meters)
TC_08	Nearby Search API - Unsupported type (e.g., "hospital")	-Response status code: 400 Bad Request	- Response contains an error message indicating the type is not supported - Error message suggests valid types for nearby search (e.g., "restaurant", "cafe")

Testing Checklist:

General:

- **Headers:**
  - Verify that all required headers are included in the request (e.g., Content-Type, Authorization if using an API key).
  - Check for correct header formatting and values (e.g., Content-Type: application/json).
- **API Key:**
  - Ensure a valid Google Maps API key is included in the request (replace {{API_KEY}} with your actual key).
  - Consider testing with an invalid or missing API key to verify appropriate error handling.

Request Parameters:

- **Geocoding:**
  - Validate all required parameters for geocoding requests:
    - `address` (for forward geocoding)
    - `latlng` (for reverse geocoding)
  - Test with various address formats (full address, partial address, landmarks) and coordinate formats (latitude/longitude pairs).

        - Check for handling of special characters and edge cases in addresses.
- **Nearby Search:**
        - Verify all required parameters for nearby search requests:
            - `location` (latitude/longitude pair)
            - `radius` (search radius in meters)
            - `types` (comma-separated list of place types)
        - Test with different radius values within the allowed range (1 to 50000 meters).
        - Explore various place types supported by the API (e.g., "restaurant", "cafe", "atm").

Response:

- **Status Code:**
        - Verify that the response status code matches the expected value for each test case (e.g., 200 for success, 400 for errors).
- **Response Body:**
        - Ensure the response body is a valid JSON object.
        - Validate the structure and content of the JSON data according to the Google Maps API documentation for each endpoint.
        - Check for the presence of essential data elements (e.g., location data, address components, place information).

Error Handling:

- **Test with various error scenarios:**
        - Missing or invalid API key
        - Invalid request parameters (e.g., invalid address format, unsupported place type)
        - Non-existent locations
- **Verify that error responses are clear and informative:**
        - Include appropriate error codes and messages.
        - Provide guidance on how to resolve the error (if possible)

Tools and Services for Implementing the Test Plan:

I will use **Postman** for designing and executing API requests, and **Newman** for automating the execution of Postman collections. Postman's user-friendly interface will aid in creating requests and validating responses, while Newman will allow for automated testing and result analysis through command-line execution.

Three Example Tests Using Postman with Google Maps API

Test 1: Geocode API - Valid Address

1. **Create a GET request** in Postman.
2. **Set the base URL** to https://maps.googleapis.com/maps/api/geocode/json.
3. **Add the following parameters:**
    - address: Dhaka,Bangladesh
    - key: **AIzaSyAs1DEzpzs_mGsH46XMliN1CC9cg-x5BP0** {{API_KEY}}
4. **Send the request** and observe the response.

Expected Result:

- Status code: 200 OK
- Response body (JSON): Contains a valid JSON object with location data (latitude, longitude), address components, and no error message.

Test 2: Nearby Search API - Valid Location and Radius

1. **Create a GET request** in Postman.
2. **Set the base URL** to https://maps.googleapis.com/maps/api/place/nearbysearch/json.
3. **Add the following parameters:**
    - location: 23.780356,90.377641 (latitude,longitude)
    - radius: 500 (meters)
    - types: restaurant
    - key: **AIzaSyAs1DEzpzs_mGsH46XMliN1CC9cg-x5BP0** {{API_KEY}}
4. **Send the request** and observe the response.

Expected Result:

- Status code: 200 OK
- Response body (JSON): Contains a valid JSON object with an array of nearby restaurants within the specified radius. Each restaurant entry should include details like name, rating, and address.

Test 3: Nearby Search API - Missing Location Parameter

1. **Create a GET request** in Postman.
2. **Set the base URL** to https://maps.googleapis.com/maps/api/place/nearbysearch/json.
3. **Add the following parameters:**
    - radius: 500 (meters)
    - types: restaurant
    - key: **AIzaSyAs1DEzpzs_mGsH46XMliN1CC9cg-x5BP0** {{API_KEY}}
    - **Omit the** location **parameter** (intentionally missing)
4. **Send the request** and observe the response.

Expected Result:

- Status code: 400 Bad Request
- Response body (JSON): Contains an error message indicating that the location parameter is missing. The error message should be clear and guide the user on how to provide a valid location.