Project 4 Task Writeup – Travel Destination Finder App (Jo's Travelog)

Name: Junxuan (Joshua) Liu
Andrew ID: junxuanl

Description

My application allows users to input an origin city and a budget, and fetches available travel destinations within the budget using the Amadeus Travel API. It also provides analytics on API usage and logs data for each request.

1. Implement a Native Android Application

a. Views in the Layout

The name of my Android application project is: TravelAssistantApp.

My application uses the following Views:

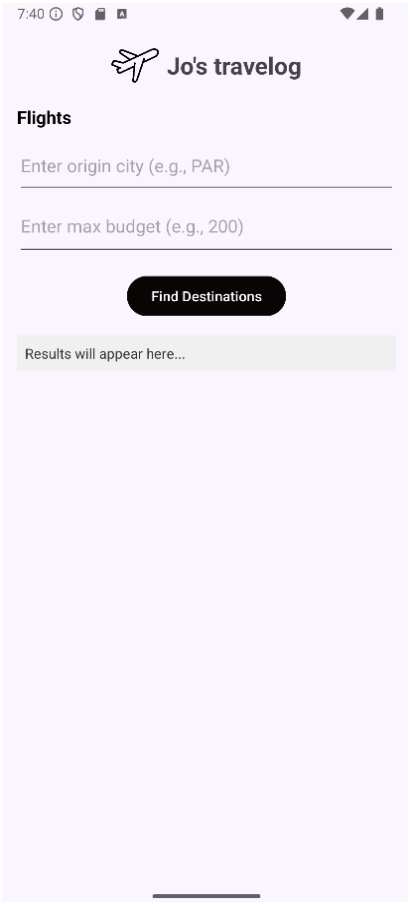
- TextView: Displays results and app instructions.

- EditText: Allows users to input the origin city and budget.

- Button: Triggers the data-fetching process.

- ImageView: Displays a travel-related icon for better user experience.

Screenshot before fetching data:



b. Requires User Input
The application prompts the user to enter the origin city and budget as input.

Example:
- Origin: "PAR" (Paris)
- Budget: "200"

c. Makes an HTTP Request
The Android app makes an HTTP GET request to the deployed web service using the following endpoint:

`https://laughing-space-meme-5gxg7r4v4959fp6vj-8080.app.github.dev/travel?origin=<origin>&maxPrice=<maxPrice>`

d. Receives and Parses JSON Reply
An example of the JSON reply:

```
Unset
{
   "data": [
      {
         "origin": "PAR",
         "destination": "CAS",
         "departureDate": "2022-09-06",
         "returnDate": "2022-09-11",
         "price": {
            "total": "161.90"
         }
      }
   ]
}
```

e. Displays New Information to the User
After receiving the data, the app updates the TextView to show the results.

Screenshot after data is displayed:

f. Repeatable Usage

The user can enter new input (origin city and budget) and fetch results repeatedly without restarting the app.

Example:

1. Enter Origin: PAR, Budget: 200, and fetch results.
2. Enter Origin: LON, Budget: 300, and fetch results.

2. Implement a Web Application Deployed on Codespaces

a. Using HttpServlet for API

The web application uses a Servlet named TravelServlet to handle requests.

- Endpoint: /travel
- Request Parameters: origin, maxPrice
- Response Format: JSON

b. Receives HTTP Requests
TravelServlet receives HTTP GET requests from the Android application with parameters for origin and budget.

c. Executes Business Logic
The servlet fetches travel data from the Amadeus API using an authenticated request, parses the response, and formats it to include only relevant information.

d. Replies with JSON
The servlet responds with a JSON object containing filtered travel data.

4. Log Useful Information
The web service logs the following information for each request:
- Timestamp of the request.
- Input parameters (origin and maxPrice).
- API response time in milliseconds.
- Response size (in bytes).
- API status (e.g., success or failure).
- Error messages (if any).

5. Store the Log Information in a Database
The logs are stored in a MongoDB Atlas database.

- Database Name: travelApp
- Collection Name: logs

Connection String:

mongodb+srv://junxuanl:Lljjxx0325@cluster0.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0

6. Display Analytics and Logs on a Web-Based Dashboard

Operations Analytics
The dashboard displays the following analytics:

1. **Top 5 Origin Cities:** Most frequently requested origin cities.
2. **Average API Response Time:** Average time taken for API responses.
3. **Most Frequent Budget Ranges:** Most common price ranges requested.

Screenshot of Analytics:

Travel Service Dashboard

Operations Analytics

Top 5 Origin Cities

- PAR: 3 requests
- MAD: 2 requests

Average API Response Time

Average API Response Time: 1693.6 ms

Most Frequent Max Price Requests

- Max Price: 200, 5 requests

Full Logs

All logs are displayed in a human-readable format on the dashboard using an HTML table.

Screenshot of Logs:

Travel Service Dashboard

Request Logs

Timestamp	Origin	Max Price	API Response Time (ms)	Status	Response Size (bytes)
2024-11-19 22:48:32	PAR	200	1307	success	7031
2024-11-19 22:53:09	MAD	200	2010	success	6965
2024-11-19 23:00:11	PAR	200	1243	success	7031
2024-11-19 23:10:06	PAR	200	1994	success	7031
2024-11-19 23:25:46	MAD	200	1914	success	6965

7. Deployment on Codespaces

- The web service is deployed on GitHub Codespaces.
- **Public URL:**

https://laughing-space-meme-5gxg7r4v4959fp6vj-8080.app.github.dev