Project 4 Task 2 - BikeLaneApp

Name: Jan Park / AndrewID: jaeunp

Description:

The application allows users to search for bike lanes in Pittsburgh based on street name or notes, optionally filtering by year added. It displays results in a ListView and provides a Mapbox-powered map for the area. The system also logs each mobile interaction and stores logs in a MongoDB Atlas collection, which are used to power a web-based analytics dashboard.

Requirements:

1. Implement a native Android application

Project name: BikeLaneMobile

a. Has at least three different kinds of Views in your Layout (TextView, EditText, ImageView, or anything that extends android.view.View). In order to figure out if something is a View, find its API. If it extends android.view.View then it is a View.

Views used:
The layout contains 1) EditText (for user input of street keyword),

2) Spinner (dropdown for selecting year)

3) ListView (to show bike lane results)

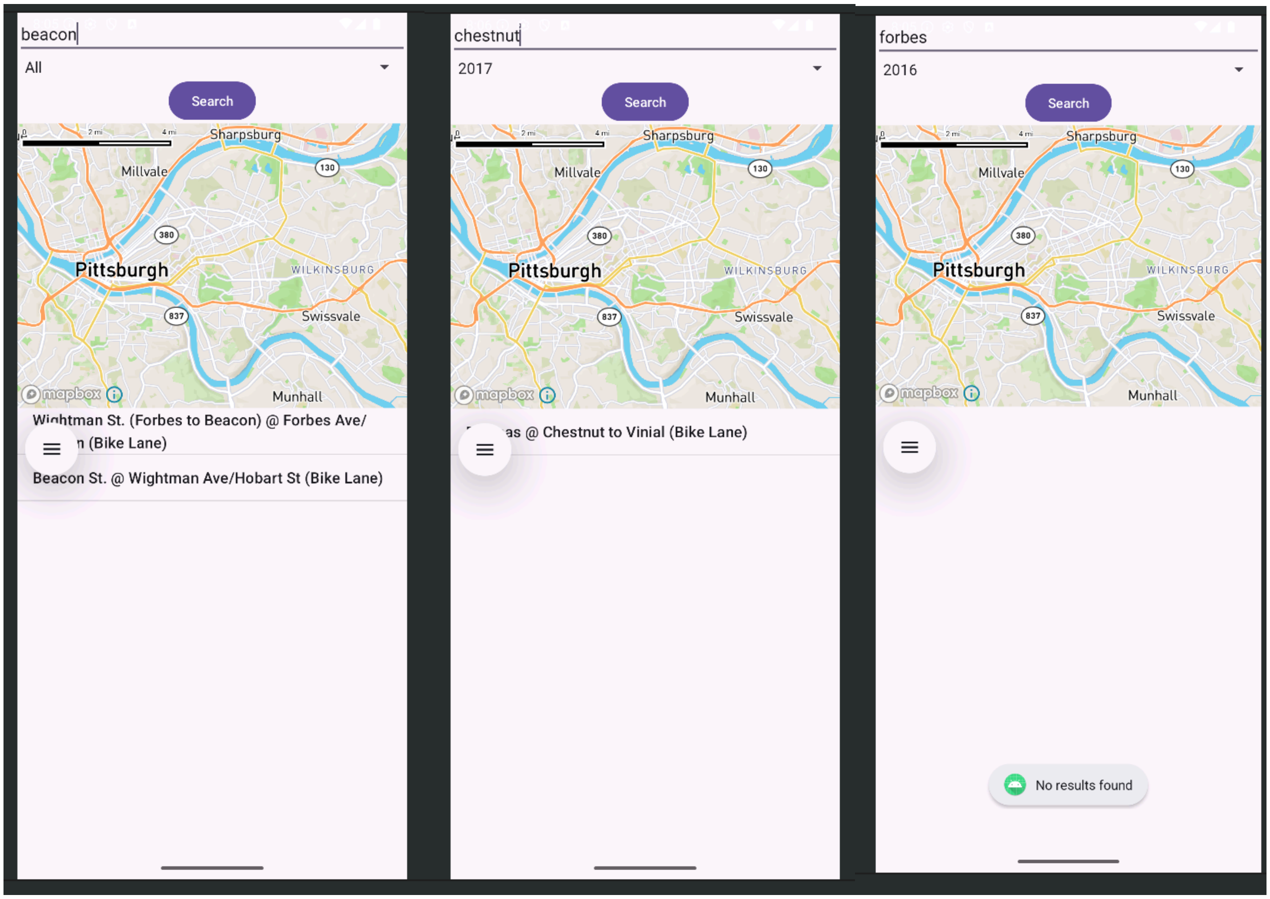
4) MapView (Mapbox map to show city area)

5) Button (to trigger search)

These all extend android.view.View.

b. Requires input from the user

Users enter a search keyword and optionally choose a year.



Wightman St. (Forbes to Beacon) @ Forbes Ave/ …n (Bike Lane)

Beacon St. @ Wightman Ave/Hobart St (Bike Lane)

…as @ Chestnut to Vinial (Bike Lane)

No results found

c. Makes an HTTP request (using an appropriate HTTP method) to your web service. Note that this request must be done using a background thread (see Lab 8's use of BackgroundTask)

The HTTP GET request is performed in the fetchBikeLanes() method using HttpURLConnection in a background thread:

HttpURLConnection conn = (HttpURLConnection) new URL(apiUrl.toString()).openConnection();

conn.setRequestMethod("GET");

Example request:

https://literate-xylophone-7vv4rgvg65rqfwp7r-8080.app.github.dev/api/bikelane?q=Forbes&year=2019

d. Receives and parses an XML or JSON formatted reply from your web service

The mobile app parses a JSON array containing information such as BLName, CrossSt, and Note.

e. Displays new information to the user

Each matched bike lane is formatted as [Name] @ [CrossSt] (Note) and displayed in the ListView.

f. Is repeatable (I.e. the user can repeatedly reuse the application without restarting it.)

Users can enter new inputs and rerun the query. No need to restart the app.

2. Implement a web service

Project name: BikeLaneApp (Java Servlet based)

a. Implement a simple (can be a single path) API.

Implemented using BikeLaneServlet.java, which responds to /api/bikelane.

b. Receives an HTTP request from the native Android application

The servlet extracts q (keyword) and year from the request parameters.

c. Executes business logic appropriate to your application. This includes fetching XML or JSON information from some 3rd party API and processing the response.

Continuing from the web site for task 1 (https://data.wprdc.org/dataset/88550525-db51-4fa2-9418-b23c1329d6e6/resource/841de570-9de1-4568-87a1-f52dfb1b7622)

WPRDCClient.searchBikeLanesRaw() fetches data from

https://data.wprdc.org/dataset/88550525-db51-4fa2-9418-b23c1329d6e6/resource/841de570-9de1-4568-87a1-f52dfb1b7622/download/2019-bike-lanes.geojson

and filters data by keyword and year, then returns matched entries.

d. Replies to the Android application with an XML or JSON formatted response. The schema of the response can be of your own design.

The servlet sends a JSON array with matched results back to the Android client.

4. Log useful information

At least 6 pieces of information is logged for each request/reply with the mobile phone. It should include information about the request from the mobile phone, information about the request and reply to the 3rd party API, and information about the reply to the mobile phone. (You should NOT log data from interactions from the operations dashboard.)

Logged information for each user request:

1) keyword: what the user searched for
2) year: year filter (if any)
3) timestamp: time when the request started
4) resultCount: how many results were found
5) durationMs: time taken to complete the request
6) endpoint: which endpoint was used (/api/bikelane)

This allows analytics such as identifying popular search terms, performance bottlenecks, and uncommon queries.

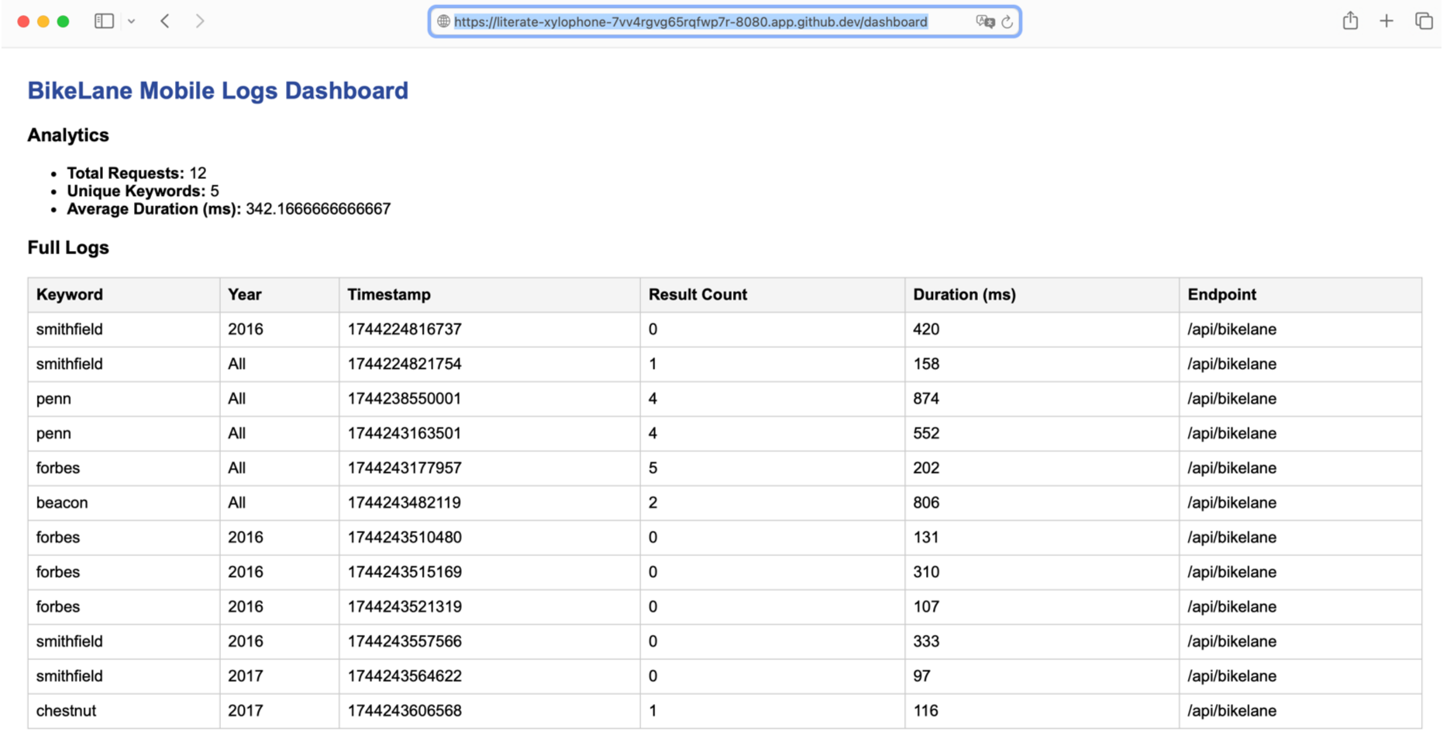
5. Store the log information in a database

Connection string: mongodb+srv://parkjaeunjan:distributedsystems891215@cluster0.xvy1z.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0

Database: bikelane_logs
Collection: mobile_logs

Logs are inserted via LogServlet.java when the Android app calls /api/log.

6. Display operations analytics and full logs on a web-based dashboard



https://literate-xylophone-7vv4rgvg65rqfwp7r-8080.app.github.dev/dashboard

BikeLane Mobile Logs Dashboard

Analytics

- **Total Requests:** 12
- **Unique Keywords:** 5
- **Average Duration (ms):** 342.1666666666667

Full Logs

Keyword	Year	Timestamp	Result Count	Duration (ms)	Endpoint
smithfield	2016	1744224816737	0	420	/api/bikelane
smithfield	All	1744224821754	1	158	/api/bikelane
penn	All	1744238550001	4	874	/api/bikelane
penn	All	1744243163501	4	552	/api/bikelane
forbes	All	1744243177957	5	202	/api/bikelane
beacon	All	1744243482119	2	806	/api/bikelane
forbes	2016	1744243510480	0	131	/api/bikelane
forbes	2016	1744243515169	0	310	/api/bikelane
forbes	2016	1744243521319	0	107	/api/bikelane
smithfield	2016	1744243557566	0	333	/api/bikelane
smithfield	2017	1744243564622	0	97	/api/bikelane
chestnut	2017	1744243606568	1	116	/api/bikelane