# Project 4: FindMyEvent - Ticketmaster Event Discovery System

## 1. Android Application

### User Interface Components

The Android application includes the following view:

- **TextView**: For displaying event names, dates, locations, and other text information
- **EditText**: For user input when searching for events
- **RadioButtons**: For selecting search type (Category, Country Code, All Events)
- **Button**: For initiating searches and viewing tickets
- **RecyclerView**: For displaying lists of events
- **CardView**: For styling event items in the list
- **ProgressBar**: For indicating loading status
- **ScrollView**: For the details screen to accommodate varying content length

### User Input

The app requires and processes user input in multiple ways:

- Search term input via EditText. You can search for Music, Dance, Sports, etc.
- You can search by Country Code: UK,US,AU,AS, etc.
- You can also search for all events.
- The search is capped at 10 results.

### HTTP Requests

The application makes HTTP requests to the web service using OkHttp:

- Requests are executed on a background thread using AsyncTask
- Different endpoints are called based on search type (category, country, all)
- Proper error handling is implemented for network failures

### Data Parsing

The app parses JSON responses from the web service:

- Uses Gson library to convert JSON to Java objects

- Maps the response to Event model objects
- Properly handles various data formats (dates, prices, locations)

## Information Display

Event information is clearly displayed to the user:

- List view shows basic event information (name, date, venue, location)
- Detail view shows comprehensive information including pricing and ticket links
- Empty state and error messages are shown when appropriate

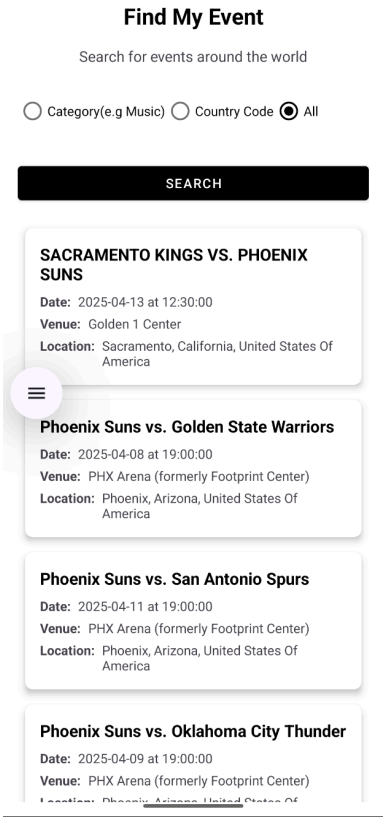
## Repeatable Functionality

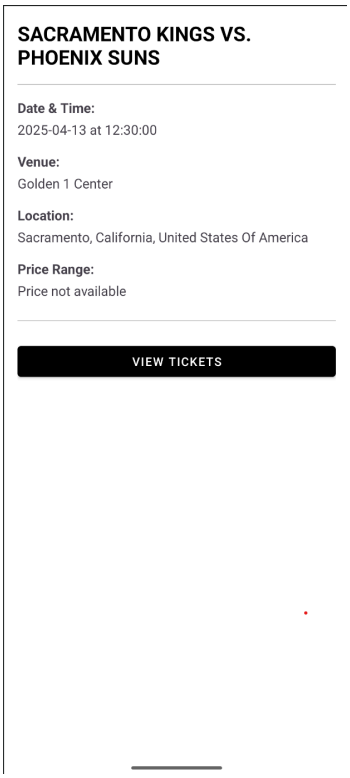
The application allows users to:

- Perform multiple searches without restarting
- Toggle between different search types
- View event details and return to the search results

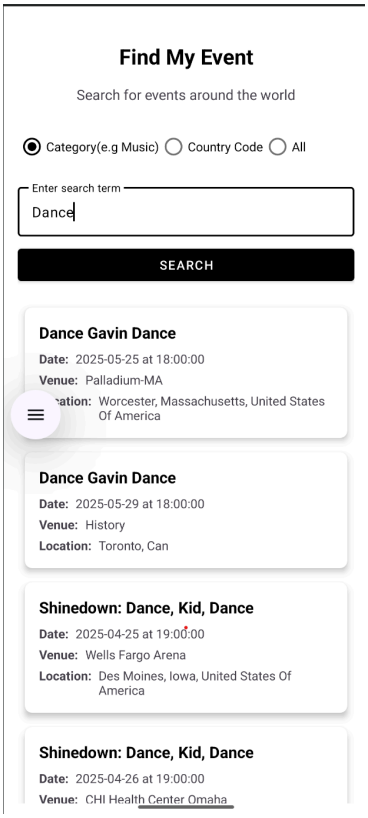
## Demo:

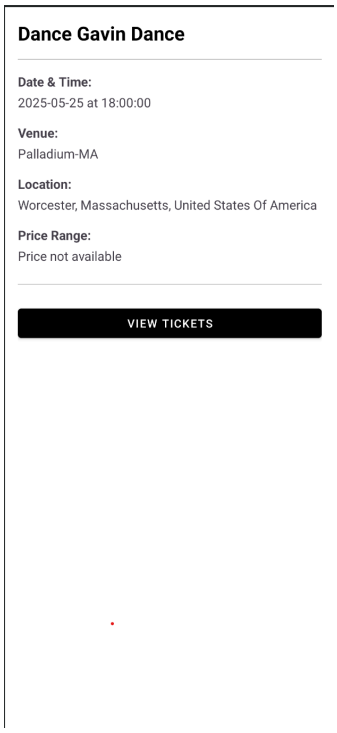
Search By All Events:



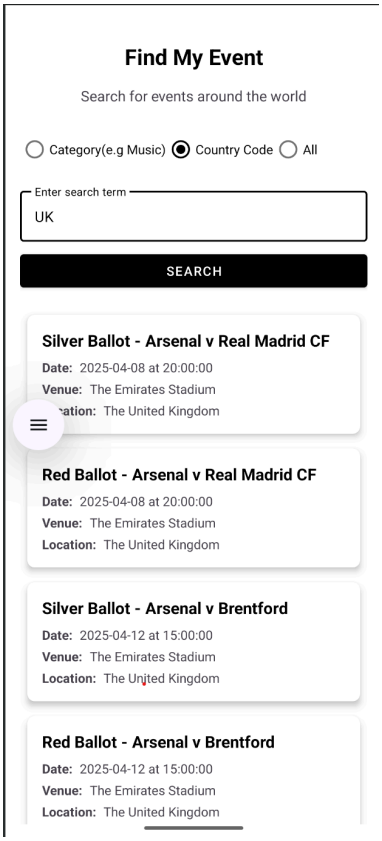


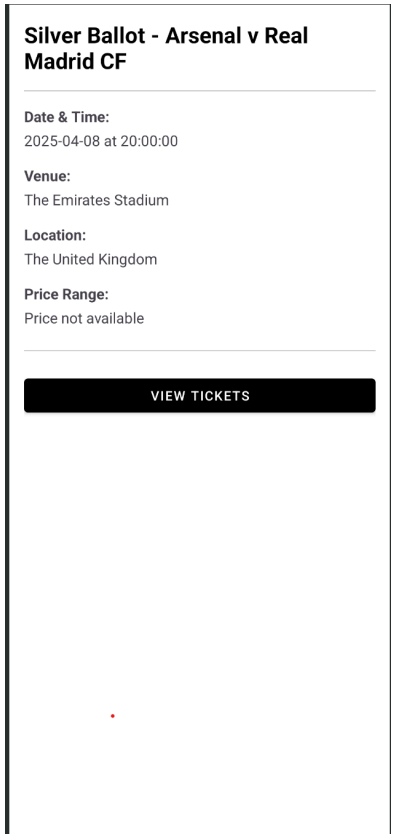

Search By Category:


Find My Event
Search for events around the world
Category(e.g Music)
Country Code
All
Enter search term
Dance
SEARCH
Dance Gavin Dance
Date: 2025-05-25 at 18:00:00
Venue: Palladium-MA
Location: Worcester, Massachusetts, United States Of America
Dance Gavin Dance
Date: 2025-05-29 at 18:00:00
Venue: History
Location: Toronto, Can
Shinedown: Dance, Kid, Dance
Date: 2025-04-25 at 19:00:00
Venue: Wells Fargo Arena
Location: Des Moines, Iowa, United States Of America
Shinedown: Dance, Kid, Dance
Date: 2025-04-26 at 19:00:00
Venue: CHI Health Center Omaha


Dance Gavin Dance
Date & Time:
2025-05-25 at 18:00:00
Venue:
Palladium-MA
Location:
Worcester, Massachusetts, United States Of America
Price Range:
Price not available
VIEW TICKETS

Search By Country Code:






# 2. Web Service

## API Implementation

The web service implements a RESTful API that:

- Receives HTTP requests from the Android application
- Processes parameters (searchType, query)
- Executes business logic to fetch data from Ticketmaster using Ticketmaster API
- Returns structured JSON responses

## Business Logic

The web service implements below business logic:

- Fetches event data from the Ticketmaster API
- Filters and processes the response

- Returns only the necessary information to the Android app
- Handles various search types (name/category, country, all events)

### Third-Party API Integration

The web service successfully integrates with the Ticketmaster API:

- Makes HTTP requests to Ticketmaster endpoints
- Passes appropriate parameters based on user search
- Processes XML/JSON responses from Ticketmaster
- Handles rate limiting and API errors appropriately

### Response Formatting

The web service formats responses appropriately:

- Returns JSON-formatted data to the Android app
- Includes only necessary information (no extraneous data)
- Properly structures the response for easy parsing
-

## 3. Logging and Dashboard

### Information Logging

The web service logs below pieces of information for each request/reply:

1. Client IP address
2. Search type
3. Search Query
4. Timestamp of request/response
5. Response status and time
6. Number of results returned

### Data Storage

All logs are stored persistently in MongoDB:

- Log entries include request and response details
- Data is properly structured for analytics
- Timestamps allow for chronological analysis

### Operations Analytics

The dashboard displays at least 3 interesting analytics:

1. Most popular search terms
2. Average response time for queries
3. Distribution of search types (category, country, all)

### Log Display

The dashboard provides a formatted display of logs:

- Shows detailed information for each request
- Formats data in a human-readable way (not as JSON or XML)
- Allows viewing historical activity

Webservice Landing page:

# Ticketmaster API Web Service

## API Endpoints

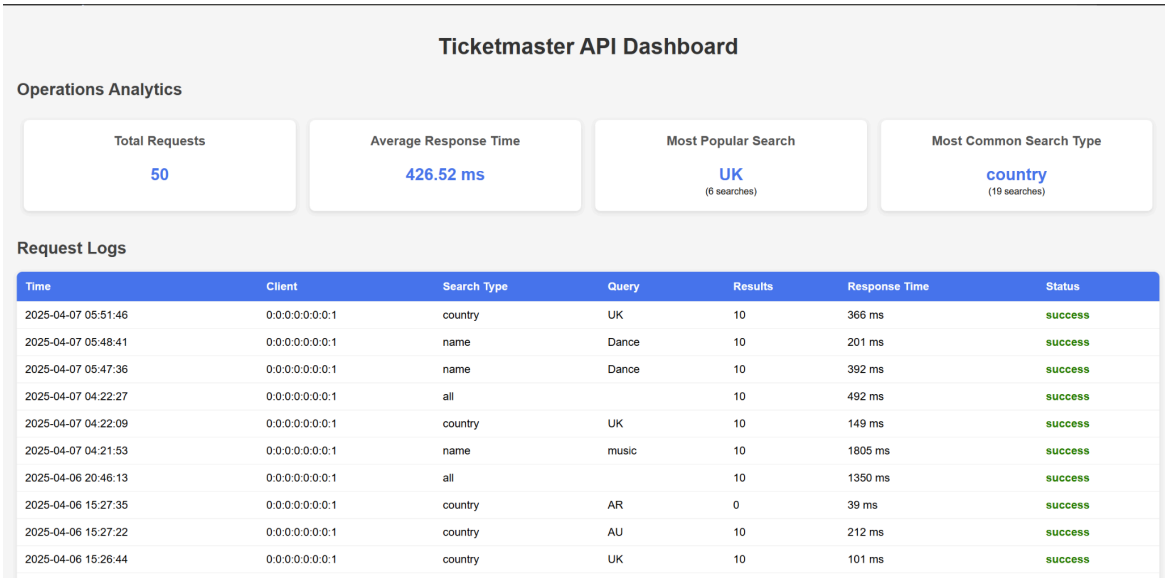
This service provides access to Ticketmaster event data.

- **GET /events** - Get all events
- **GET /events?searchType=name&query=YOUR_QUERY** - Search events by name
- **GET /events?searchType=country&query=COUNTRY_CODE** - Search events by country

## Dashboard

View analytics and logs for the API service:

Open Dashboard




Dashboard that maintain logs and other parameter.

# 4. Deployment

The web service is successfully deployed to GitHub Codespaces using Docker

To run the CodeSpace you will have to create a new codespace and update the base url in the APIService file in the Android app.

Remember to make the port public once the codespace is running.

```
// Replace with your deployed web service URL
1 usage
private static final String BASE_URL = "https://symmetrical-spork-x4rr7grwxrxh6995-8080.app.github.dev/";
```

# 5. Overall System Integration

The complete system demonstrates tight integration between:

- Android mobile app (client-side interface)
- Java web service (middleware and business logic)
- Ticketmaster API (third-party data source)
- MongoDB database (data persistence)
- Dashboard (operational monitoring)