

Project 4 Task 2 – MuscleWiki 2.0

Lekhya Reddy Rachapalli (lrachapa)

Overview

My distributed system provides a workout finder that allows users to select a muscle group and optional equipment and then fetches exercise ideas from a third-party API ([ExerciseDB](#)). It reshapes the data, logs usage analytics. There is a dashboard that provides logs and operational analytics via a JSP interface backed by MongoDB Atlas.

1. Native Android Application

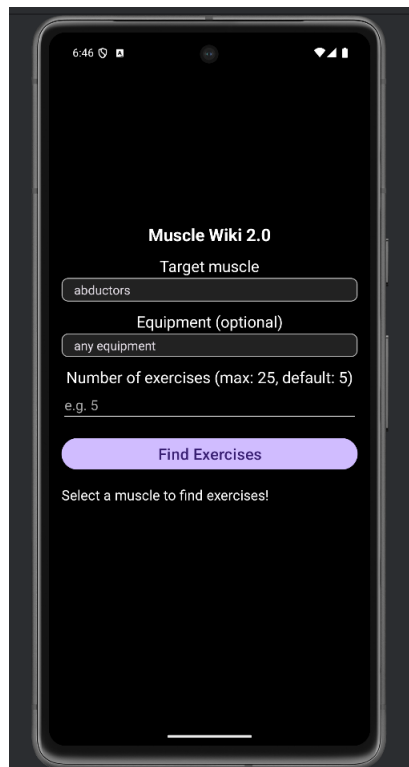
Android Studio project: MyExerciseApplication

a) Uses at least three different kinds of views:

My app uses:

- Spinner (for muscle + equipment selection)
- EditText (for exercise limit)
- Button (Search)
- TextView (status updates)
- ImageView (GIF display in dialog)
- ListView (results list)

File: activity_main.xml

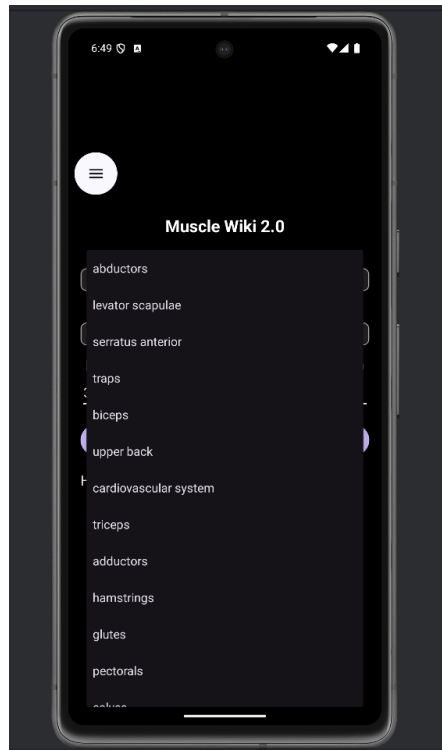


Before search

b) Requires input from the user:

Users select:

- Target muscle
- Optional equipment
- Number of exercises (1–25; default is 5)



Selecting muscle

c) Makes an HTTP GET request to web service

The mobile app issues this GET request:

<https://fuzzy-tribble-r4654xj577gicp9w4-8080.app.github.dev/exerciseapp/api/exercises?muscle=biceps&equipment=dumbbell&limit=3>

when the muscle chosen is biceps, equipment is dumbbell and limit is 3

This is implemented in: MainActivity.java → FetchExercisesTask.doInBackground()

d) Receives and parses a JSON reply

Example of the JSON returned is:

```

{
  "exercises": [
    {
      "gifUrl": "https://static.exercisedb.dev/media/QQwxa8g.gif",
      "equipments": [
        "dumbbell"
      ],
      "instructions": [
        "Step:1 Sit on an exercise ball with your feet flat on the ground and your back straight.",
        "Step:2 Hold a dumbbell in one hand with an underhand grip, resting your elbow on the exercise ball.",
        "Step:3 (intermediate variation) Keeping your upper arm stationary, exhale and curl the dumbbell up towards your shoulder.",
        "Step:4 Pause for a moment at the top, then inhale and slowly lower the dumbbell back to the starting position.",
        "Step:5 Repeat for the desired number of repetitions, then switch arms."
      ],
      "targetMuscles": [
        "biceps"
      ],
      "name": "intermediate style dumbbell preacher curl over exercise ball"
    }
  ],
  "totalExercises": 85,
  "query": {
    "muscle": "biceps",
    "limit": 1,
    "equipment": "dumbbell"
  },
  "returned": 1,
  "status": "ok"
}

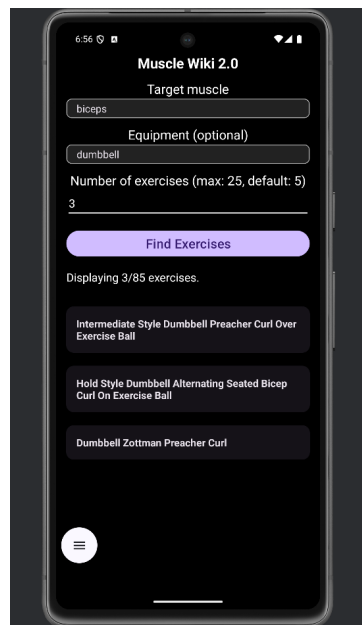
```

Parsing occurs in: MainActivity.java → parseAndDisplay()

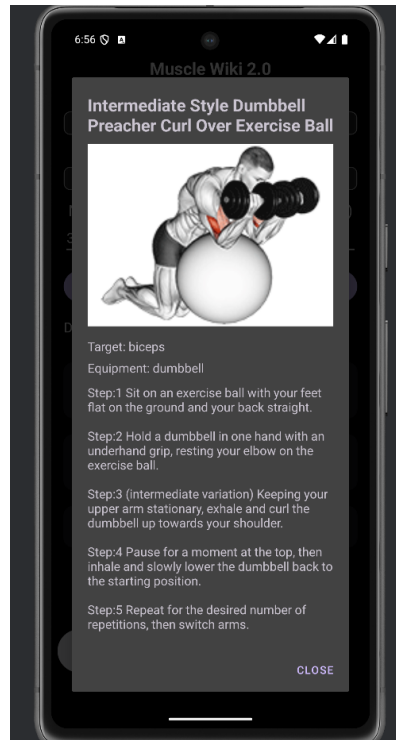
e) Displays new information

The app displays:

- Exercise cards in a ListView
- Exercise names
- GIFs loaded with Glide
- Detailed popup dialog with full instructions



Results

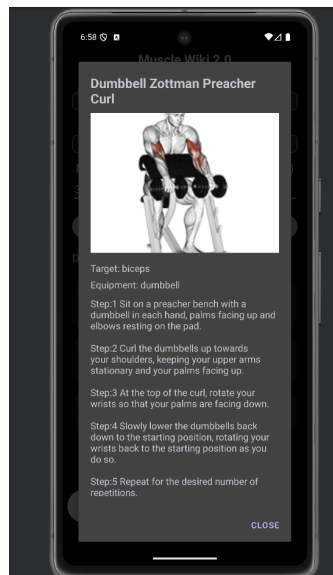


After clicking on an exercise

f) Repeatable

User can repeatedly:

- Modify inputs
- Fetch results again
- Open different exercise dialogs



Choosing a different exercise

2. Web Application

MyExerciseApplicationWebService is deployed at:

<https://fuzzy-tribble-r4654xj577gjcp9w4-8080.app.github.dev/exerciseapp/>

a) Uses an HttpServlet to implement API

Main servlet: MyExerciseServlet.java

Path: /api/exercises

This servlet:

- Validates inputs
- Calls external ExerciseDB API
- Reshapes JSON
- Logs analytics to MongoDB
- Replies with cleaned JSON

b) Receives HTTP requests from Android

Example request:

GET /exerciseapp/api/exercises?muscle=biceps&equipment=dumbbell&limit=3

Handled in:

MyExerciseServlet.doGet()

c) Executes business logic

- Input validation
- Constructing ExerciseDB URL
- Fetching + parsing third-party results
- Extracting:
 - name
 - gifUrl
 - targetMuscles
 - equipment
 - instructions
 - Cleaning JSON for mobile

d) Replies using JSON

Reply structure is custom-designed: status, query, totalExercises, returned, exercises[]

4) Log useful information

timestamp: chronological ordering

clientId: monitoring usage

userAgent: device analytics

muscle: most requested muscle group

equipment: most popular equipment

limit: usage trend

servletStatus: success or not
errorMessage: for debugging
thirdPartyURL: traceability
thirdPartyStatus: health
latencyMs: performance monitoring
returnedToClient: result count

ORGANIZATION: Lekhya Reddy's Org -> PROJECT: Project 0 -> CLUSTER: ExerciseApp

Back to Clusters

Data

ExerciseApp

DATABASES: 3 COLLECTIONS: 8

+ Create Database

Search Namespaces

myexerciseapplication_db

myexerciseapplication_logs

project4_db

sample_mflix

myexerciseapplication_db.myexerciseapplication_logs

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 12.87KB TOTAL DOCUMENTS: 27 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

Filter Type a query: { field: 'value' }

Reset Apply Options

INSERT DOCUMENT

1-20 of many results

System Status: All Good

©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

MongoDB

5) Store logs in MongoDB

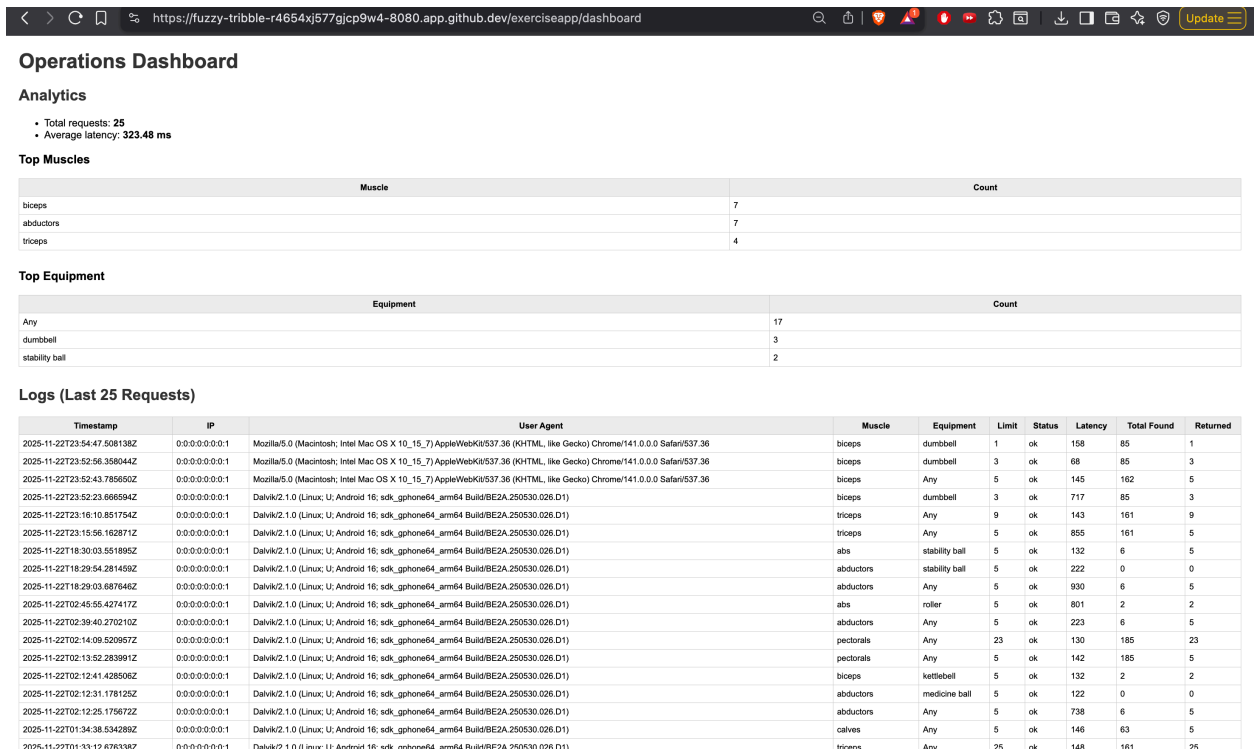
Database: myexerciseapplication_db

Data persists across restarts

6) Display operations analytics + complete logs

Implemented in MyDashboardServlet.java and dashboard.jsp. It shows:

- Total number of requests
- Average API latency
- Top 3 muscle groups
- Top 3 equipment types
- Logs



Dashboard

7) Works in Codespaces

The servlet is deployed and publicly accessible via:

<https://fuzzy-tribble-r4654xj577gjcp9w4-8080.app.github.dev/exerciseapp/api/exercises>

Android app works when:

- The port (8080) is public
- WAR is copied/deployed
- Tomcat is running manually

The commands to be run are:

cd tomcat

bin/startup.sh

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1
@lekhy2617 → /workspaces/Exercise-Application-Dashboard (main) $ cd tomcat
@lekhy2617 → /workspaces/Exercise-Application-Dashboard/tomcat (main) $ bin/startup.sh
Using CATALINA_BASE: /workspaces/Exercise-Application-Dashboard/tomcat
Using CATALINA_HOME: /workspaces/Exercise-Application-Dashboard/tomcat
Using CATALINA_TMPDIR: /workspaces/Exercise-Application-Dashboard/tomcat/temp
Using JRE_HOME: /usr/local/sdkman/candidates/java/current
Using CLASSPATH: /workspaces/Exercise-Application-Dashboard/tomcat/bin/bootstrap.jar:/workspaces/Exercise-Application-Dashboard/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
@lekhy2617 → /workspaces/Exercise-Application-Dashboard/tomcat (main) $
```

Commands to run

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1

| Port | Forwarded Address | Running Process | Visibility | Origin |
|----------|------------------------------|---|------------|----------------|
| 8080 | https://fuzzy-tribble-r46... | /usr/local/sdkman/candidates/java/current/bin/java -Dj... | Public | Auto Forwarded |
| Add Port | | | | |

Make the port public