Team Run Members:
Harut Kulakchian
Anthony Jimenez
Daniel Herrera
Osbaldo Bravo
Carlos Martine

Team Run Architecture Design

Mobile App

Stack:

Front End:

- React Native (javascript)
- React Navigation
- React Query
- React-native-maps

Back End:

- Java
- Springboot

Database:

- PostgreSQL

API's:

- Google Maps
- OpenAI

Package/Build Manager

- IOS - Xcode
- Android - Gradle
- Cross-platform (IOS & Android) - React Native

Task Assignments

Front End Development - Anthony Jimenez

Back End Development - Carlos Martinez

AI Implementation- Osbaldo Bravo

Database Schema and Queries- Harut Kulakchian

Maps and API Implementation - Daniel Herrera

URL

- https://lucid.app/lucidspark/f14f4b6e-8eea-46e5-8367-fbc9852026fe/edit?viewport_loc=889%2C-1424%2C10815%2C4755%2C0_0&invitationId=inv_0762e609-b596-4b69-b47d-e137bb406c89

For a Mobile App:

- **Database Tables (Models)**:

  User — account, streak, level

  Workout — body part, type, reps, weight, time

  WaterLog — amount ml, timestamp

Character — per body part stats & XP

Raid/Boss — required stats, rewards

Gear / UserGear — unlocks & equipped

ReminderSetting — water reminders

(Shown in the slides we provided)

- **Views**:

  Dashboard (“My Week”) — weekly workouts/water/boss fights.

  Log Workout — choose body part, exercise, reps/weight, plays animation.

  Water Tracker — quick amounts + progress bar + buffs.

  Warm-Up — preset or custom with timer/video + avatar.

  Raids/Boss — zone unlocks, challenge boss, rewards.

  Gear/Wardrobe — unlock/equip gear.

  Rest Mode — appears after 3 missed days, resume button.

  Settings — permissions, HR device link.

- **URLs**:

  /login, /log/workout, /log/water, /warmup, /week, /raids, /gear, /rest, /settings (API versions if we want: /api/workouts, /api/water, /api/summary/week, etc.)

Design Considerations

- **Decoupling:** controllers , services, repositories.
- **Anti-cheat:** prefer server computed stats: disable manual when auto logging active.
- **Streak/Rest:** rest after 3 missed days, resume button.
- **Accessibility/UX:** quick add, clear feedback, animations tied to actions.

For a Web App:

- **Deployment**:

For now we are still deciding on whether to develop specifically for IOS or Android or maybe even cross platform. We want this to be small scale so we're leaning towards only doing development for one of these systems.

In terms of deployment, we're going to go for a CI/CD pipeline so that there is a constant workflow going on from uploading to testing. This makes it so that as soon as changes are made they are tested and reviewed before being uploaded for users. This repeats until there isn't much to do but on the off chance that this does become big, we will continue this cycle.

For hosting providers, my team thought about using Railway since it would be perfect for an indie game. We don't anticipate a huge success so something small scale would suit this project better than something like Azure or Google Cloud.

Scripting is also something that we plan on using since this not only automates certain processes but it also takes away the human error element which is bound to happen if the processes are manually done.

- **Development/Deployment Environments**:

We will be using a VM for our development and deployment. This is because VMs offer a controlled environment where we can test various cases and situations that may occur in deployment. Also a container would perform best if we all have the same type of machine but since we use mac and windows it would be more beneficial to use a virtual machine as it will provide less complications when testing.

- **URLs**:
  - **Application Design**
  - **Project requirement**
  - **Wiki**

- **REST API**: Document if implementing, list methods and parameters, and describe in English.
  - POST /auth/register { username, email, password } Creates a new player account.
  - POST /auth/login { email, password } Authenticates a player
  - GET /player Parameter N/A Gets a player's full profile
  - POST /workout {activity, intensity(miles, weights,etc.), duration/sets} Submits player workout stats and calculates exp gain and rewards
  - GET /leaderboard Parameter N/A Gets top 50 or 100 players stats and puts them on display

- **Views**: (IWIP) = Image is a work in progress

  Dashboard ("My Week") (IWIP)— weekly workouts/water/boss fights.

  Log in screen — the first thing users will see when opening the application

  Log Workout — choose body part, exercise, reps/weight, plays animation.

  Water Tracker — quick amounts + progress bar + buffs.

  Warm-Up — preset or custom with timer/video + avatar.

  Raids/Boss (IWIP) — zone unlocks, challenge boss, rewards.

  Rewards — have your character grow stronger and unlock/equip gear.

  Rest Mode (IWIP)— appears after 3 missed days, resume button.

  Settings (IWIP)— permissions, HR device link

- **Database Schema**: List tables/documents with attributes and their types. Describe each table and attribute in English.

Users Example:

User_id	uuid	PK	User identifier
email	text	UNIQUE, NOT NULL	login email
password	text	NOT NULL	Password
created_at	Time stamp	DEFAULT(now)	Account creation

```
CREATE TABLE users (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  email TEXT UNIQUE NOT NULL,
  password_hash TEXT NOT NULL,
  created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);
```

Workout example:

user_id	uuid	FK, NOT NULL	User id
type	text	NOT NULL	Name of workout
reps	int	NOT NULL	# of reps
sets	int	NOT NULL	# of sets

```
CREATE TABLE workouts (
  id BIGSERIAL PRIMARY KEY,
  user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
  type TEXT NOT NULL,
  reps INT CHECK (reps IS NULL OR reps >= 0),
 sets INT CHECK (sets IS NULL OR reps >= 0),
  );
```