FANZONE

deti
universidade de aveiro
departamento de eletrónica, telecomunicações e informática

DECEMBER 16TH, 2025

FanZone

long live all the memories we made

AUTHORS

Carolina Reis, nº 131193
Carolina Silva, nº 113475

MOBILE COMPUTATION

Table of contents

THE PROBLEM

Concert Memories Fade, Connections Get Lost

Scattered Memories

- Fans take hundreds of photos/videos at concerts
- Media gets lost in camera rolls
- No dedicated space for concert memories

Missing Insights

- No way to track concert attendance
- Can't see stats about your concert journey
- No visualization of your music fan experience

Lost Connections

- Meet amazing people at concerts
- Exchange numbers but connections fade
- No platform to stay connected with fellow fans

Connectivity Issues

- Concert venues often have poor network coverage
- Can't rely on cloud-based apps during events
- Need offline-first solution

INTRODUCING FANZONE

A Mobile App for Music Fans to Capture, Share, and Relive Concert Experiences

CORE CONCEPT

FanZone is an offline-first Flutter mobile application that helps music fans:

- Organize concert memories in one place
- Capture photos, videos, and audio from events
- Connect with other attendees
- Track their concert journey with detailed stats
- Sync data seamlessly when online

VALUE PROPOSITION

"Your personal concert companion"

WHO IS FANZONE FOR?

Primary Users

01

Concert Enthusiasts

- Age: 18-35
- Attend 5+ concerts per year
- Active on social media
- Value experiences over possessions
- Want to preserve memories digitally

02

Social Music Fans

- Love meeting new people at events
- Seek community around shared music taste
- Want to stay connected post-concert
- Enjoy sharing experiences

03

Data-Driven Fans

- Track their concert attendance
- Interested in personal statistics
- Like visualizing their music journey
- Appreciate insights and trends

WHO IS FANZONE FOR?

Secondary Users

01

Event Organizers

- Create events on the platform
- Engage with attendees
- Share official media
- Build community

these and other concert organization companies...

CORE FEATURES

01 SECURE AUTHENTICATION

- Email/password signup and login
- Firebase Authentication integration
- Persistent sessions across app restarts
- Password reset functionality
- Profile management with display names and photos

02 EVENT MANAGEMENT

- Browse upcoming and past events
- Create new events with details (title, date, location, description)
- View event details and attendee lists
- Join events to track participation
- Filter events by date, location, or organizer

03 MEDIA CAPTURE & SHARING

- Take photos during events
- Record videos of performances
- Capture audio snippets
- Upload and share media with event community
- View media gallery from other attendees
- Media stored in Supabase Storage

04 OFFLINE-FIRST DESIGN

- Local SQLite database on device
- Works without internet connection
- Automatic sync when online
- Connectivity monitoring
- No data loss during network outages

CORE FEATURES

05 MOTION SENSORS INTEGRATION

- Accelerometer data capture
- Gyroscope tracking
- Magnetometer readings
- Real-time sensor data visualization
- Enhanced concert experience metrics

06 SOCIAL FEATURES

- Search for other users
- Follow/unfollow functionality
- View public profiles
- See follower/following lists
- Track mutual connections

07 PERSONAL STATISTICS

- Total events attended
- Events organized
- Media shared
- Followers/following count
- Visual charts and graphs (using fl_chart)
- Concert history timeline

08 LOCATION FEATURES

- Google Maps integration
- Event location mapping
- Find nearby events
- Location-based event discovery

SYSTEM ARCHITECTURE & TECHNOLOGIES

Key Architectural Decisions

Offline-First Approach

- Local SQLite as primary data source
- Backend as sync target
- Full control over data model

Firebase for Auth Only

- Handles user authentication
- Generates JWT tokens
- No Firestore (manual data control)

Custom Flask Backend

- Full control over business logic
- Token verification
- Custom data synchronization rules

Dual Database Strategy

- Local: SQLite (on-device)
- Remote: PostgreSQL (Supabase)
- Manual sync via REST API

SYSTEM ARCHITECTURE & TECHNOLOGIES

Technology Stack

01 Frontend (Mobile App)

Flutter (Dart)

- Cross-platform framework (iOS + Android)
- Single codebase
- Fast development
- Beautiful UI components
- SDK Version: ^3.9.2

02 Backend

Flask (Python)

- Lightweight web framework
- RESTful API design
- Easy to deploy
- Flexible and extensible

SYSTEM ARCHITECTURE & TECHNOLOGIES

Technology Stack

01 Frontend (Mobile App)

Key Flutter Packages:

- provider (State management)
- firebase_core & firebase_auth (Authentication)
- http (API calls)
- sqflite (Local database)
- supabase_flutter (Supabase client)
- sensors_plus (Motion sensors)
- fl_chart (Data visualization)
- image_picker, video_player, camera (Media)
- google_maps_flutter, geolocator (Maps & Location)
- connectivity_plus (Network monitoring)

02 Backend

Key Python Packages:

- Flask-SQLAlchemy (ORM)
- Flask-Cors (Cross-origin requests)
- psycopg2-binary (PostgreSQL driver)
- firebase-admin (Token verification)
- python-dotenv (Environment variables)

SYSTEM ARCHITECTURE & TECHNOLOGIES

Technology Stack

03 Database & Storage

PostgreSQL (Supabase):

- Relational database
- Hosted on Supabase
- Session Pooler for production
- Connection pooling

SQLite:

- Local on-device database
- Offline data storage
- Fast and lightweight

Supabase Storage:

- Media file storage
- Public URLs
- Scalable cloud storage

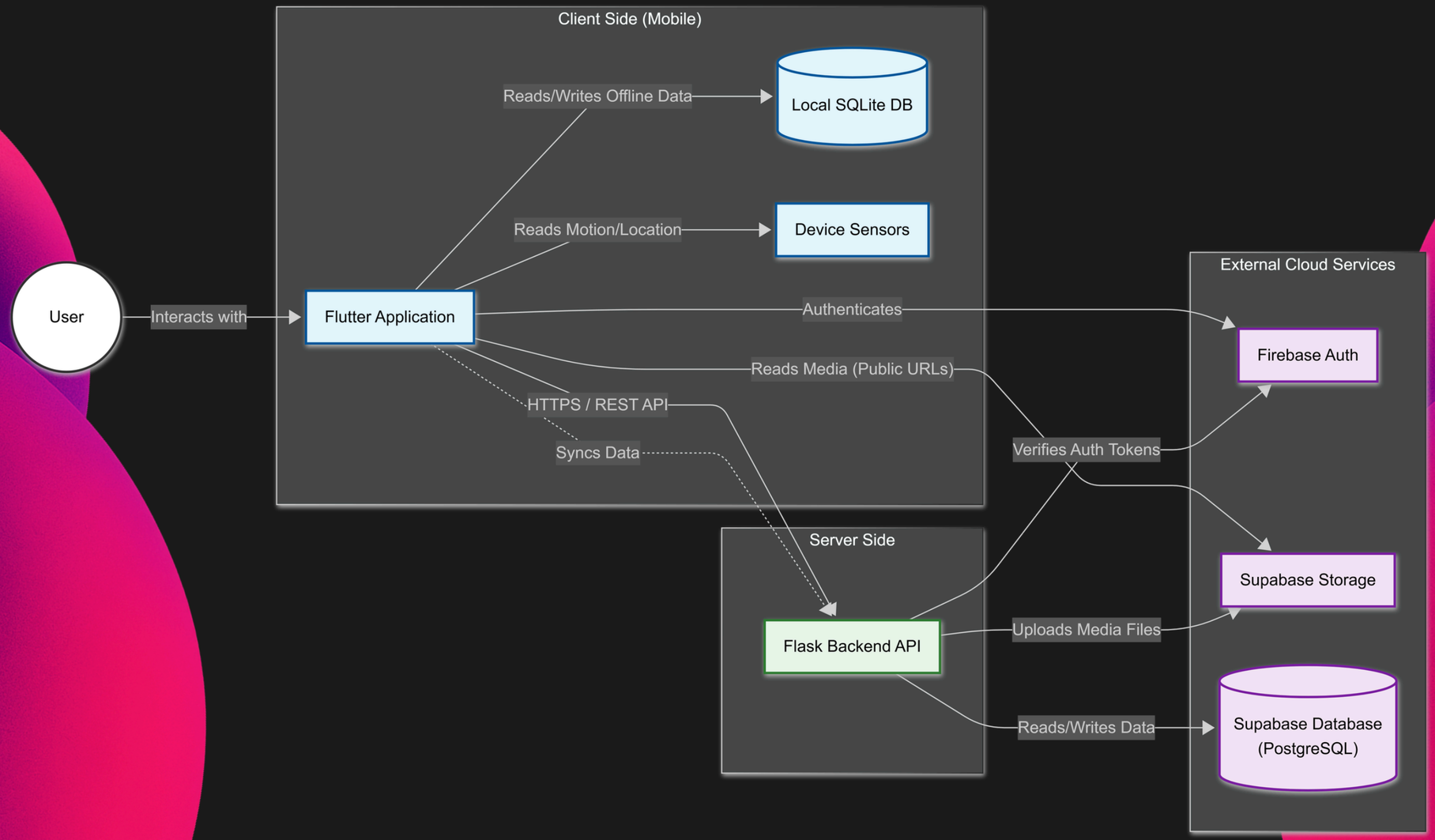
04 Authentication

Firebase Authentication:

- Email/password authentication
- JWT token generation
- Secure and scalable
- Password reset functionality

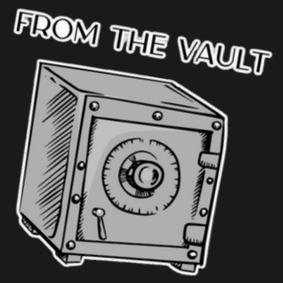
Firebase

SYSTEM ARCHITECTURE & TECHNOLOGIES



SECURITY & AUTH

Key Security Features



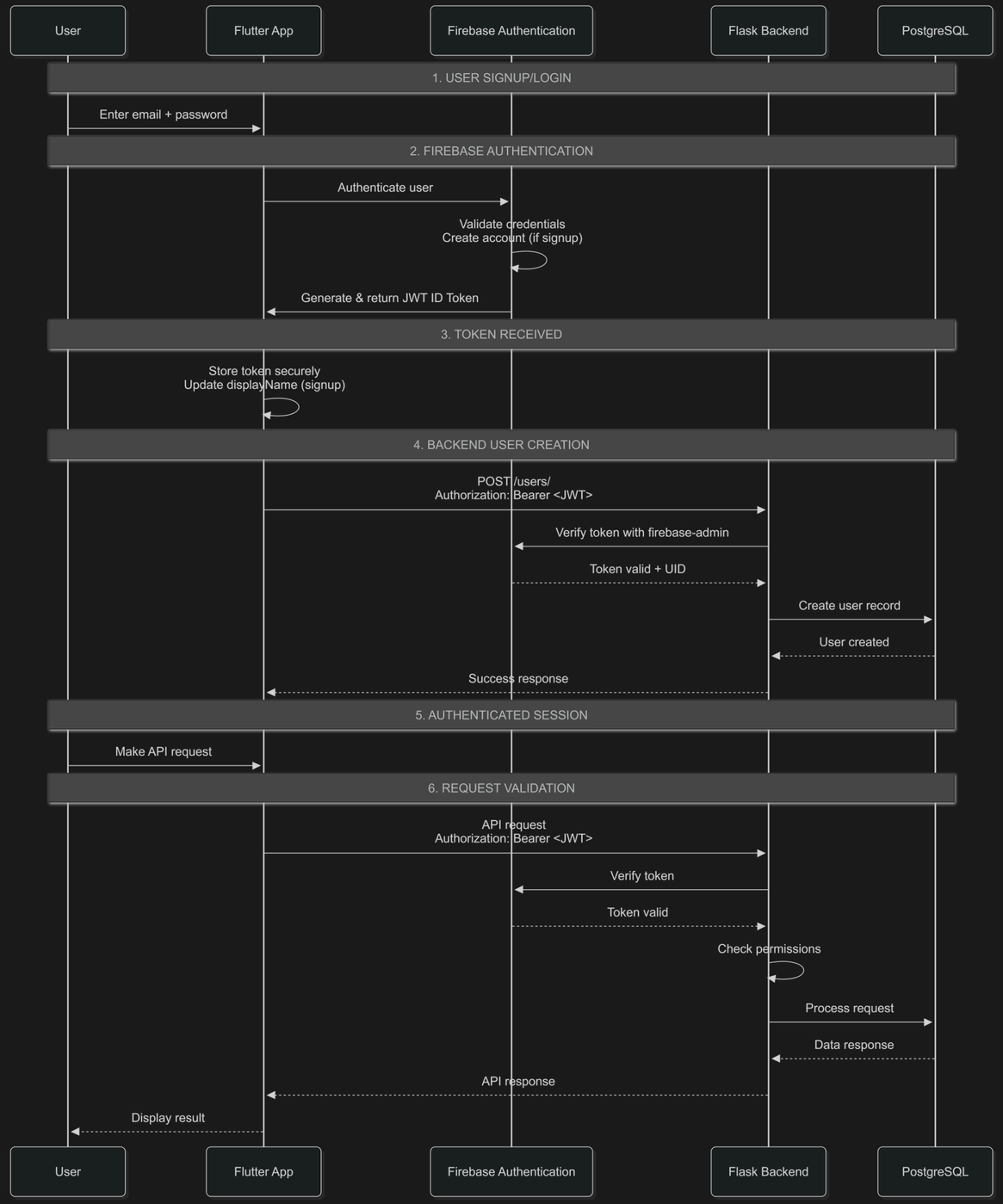
JWT Token-Based Auth:
- Stateless authentication
- Tokens expire automatically
- Secure transmission (HTTPS in production)

Error Handling:
- User-friendly error messages
- Firebase error code mapping
- Unauthorized access prevention

Token Verification:
- Every backend request verified
- Firebase Admin SDK validation
- User identity extracted from token (UID)

Dev Bypass Mode (Development Only)
- X-Bypass-Auth: true header
- Only enabled in local development
- Speeds up testing

SECURITY & AUTH



OFFLINE-FIRST DATA SYNC

Key Security Features

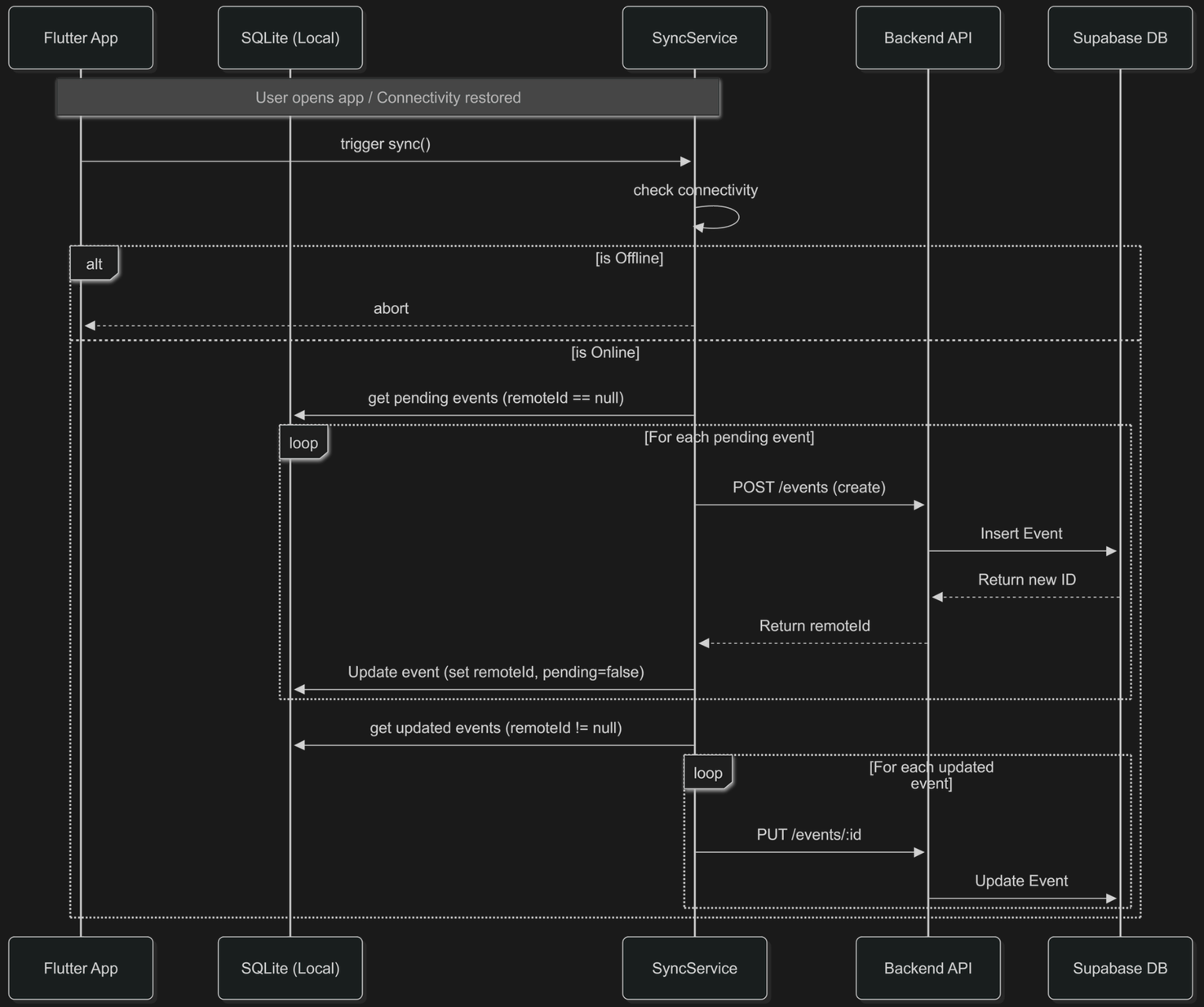
Concert Venue Reality:

- Poor or no network coverage
- Thousands of users on same network
- Data costs and roaming
- User expects app to work always

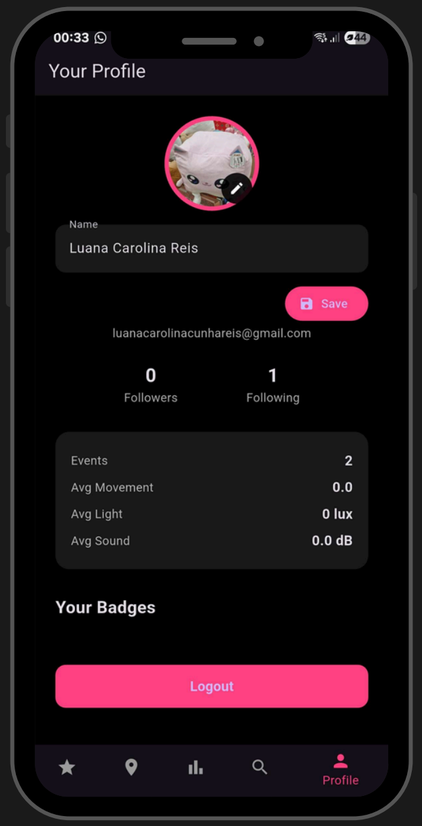
Our Solution:

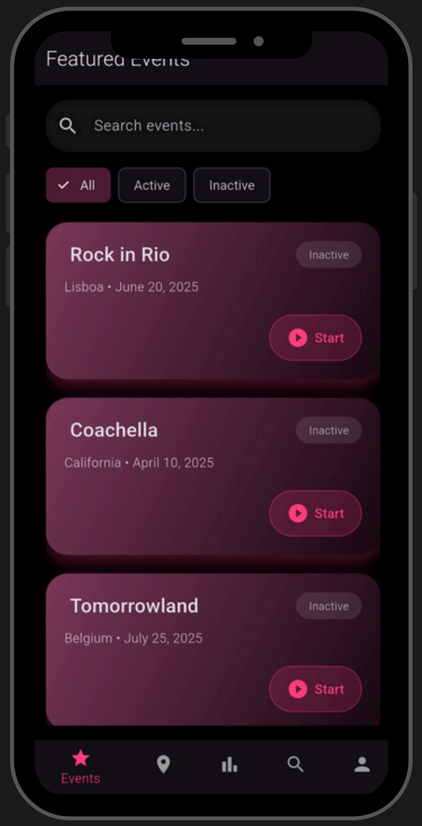
- Local SQLite database on device
- App works offline
- Automatic sync when online

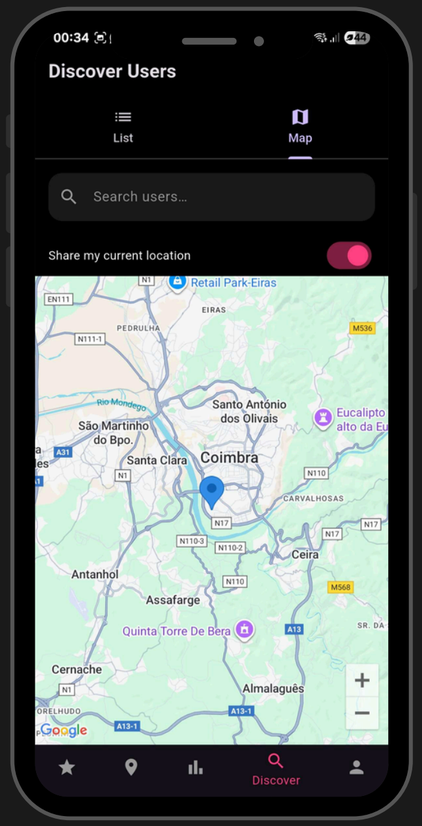
OFFLINE-FIRST DATA SYNC

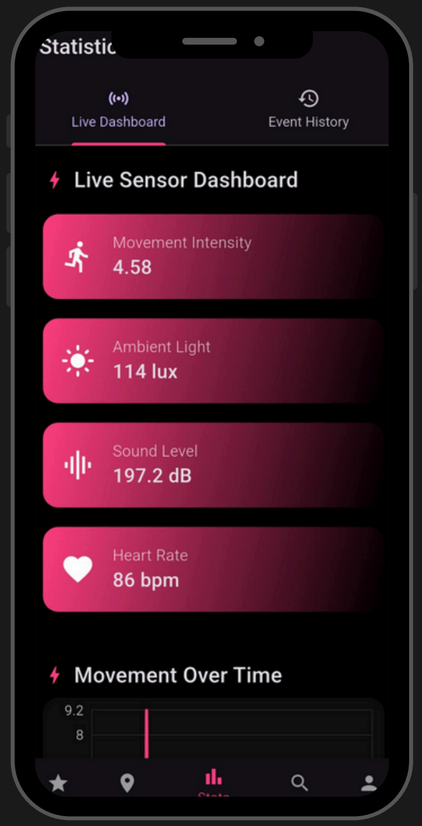


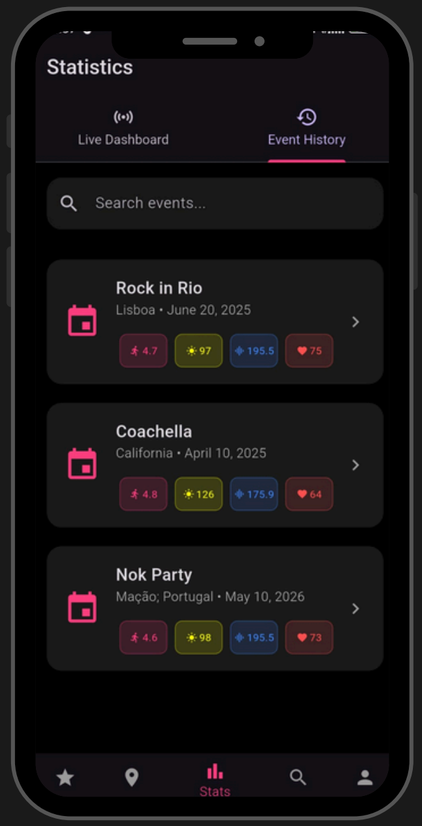
KEY SCREENS SHOWCASE



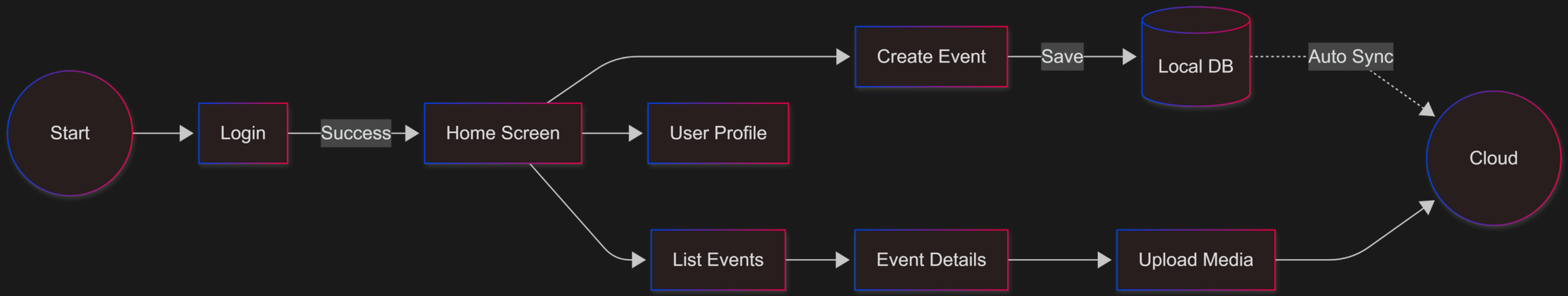








KEY SCREENS SHOWCASE



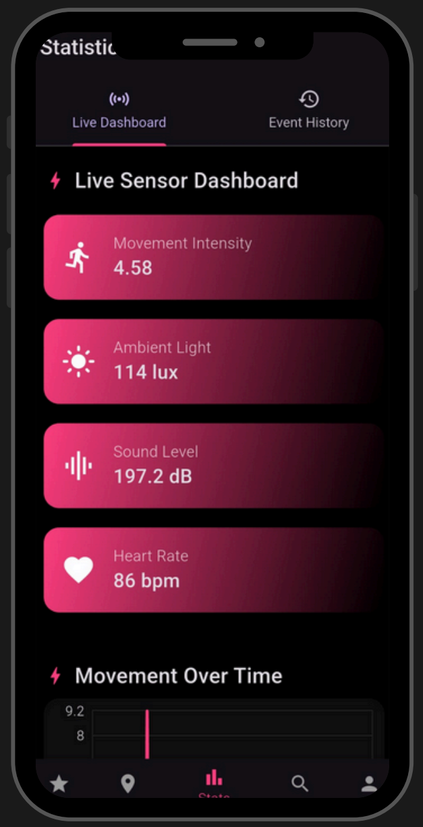
REAL-TIME CAPABILITIES

ACCELEROMETER (MOVEMENT)

- Measures device acceleration
- X, Y, Z axis data
- Captures movement intensity
- Use case: Detect dancing, jumping at concerts

AMBIENT LIGHT

- Estimates light intensity via camera
- Detects environment brightness changes
- Indicates stage and venue lighting



SOUND LEVEL

- Measures ambient sound via microphone
- Converts audio intensity to decibels (dB)
- Reflects crowd noise and energy

HEART RATE

- Receives BPM from Wear OS device
- Tracks user physical response
- Indicates engagement intensity

SERVER INFRASTRUCTURE

RESTful API Routes

Users Endpoints (/users/*)

- GET /users/ - Search users
- POST /users/ - Create user
- GET /users/<uid> - Get user profile
- PUT /users/<uid> - Update profile
- POST /users/<uid>/follow - Follow user
- DELETE /users/<uid>/unfollow - Unfollow user
- GET /users/<uid>/followers - Get followers
- GET /users/<uid>/following - Get following

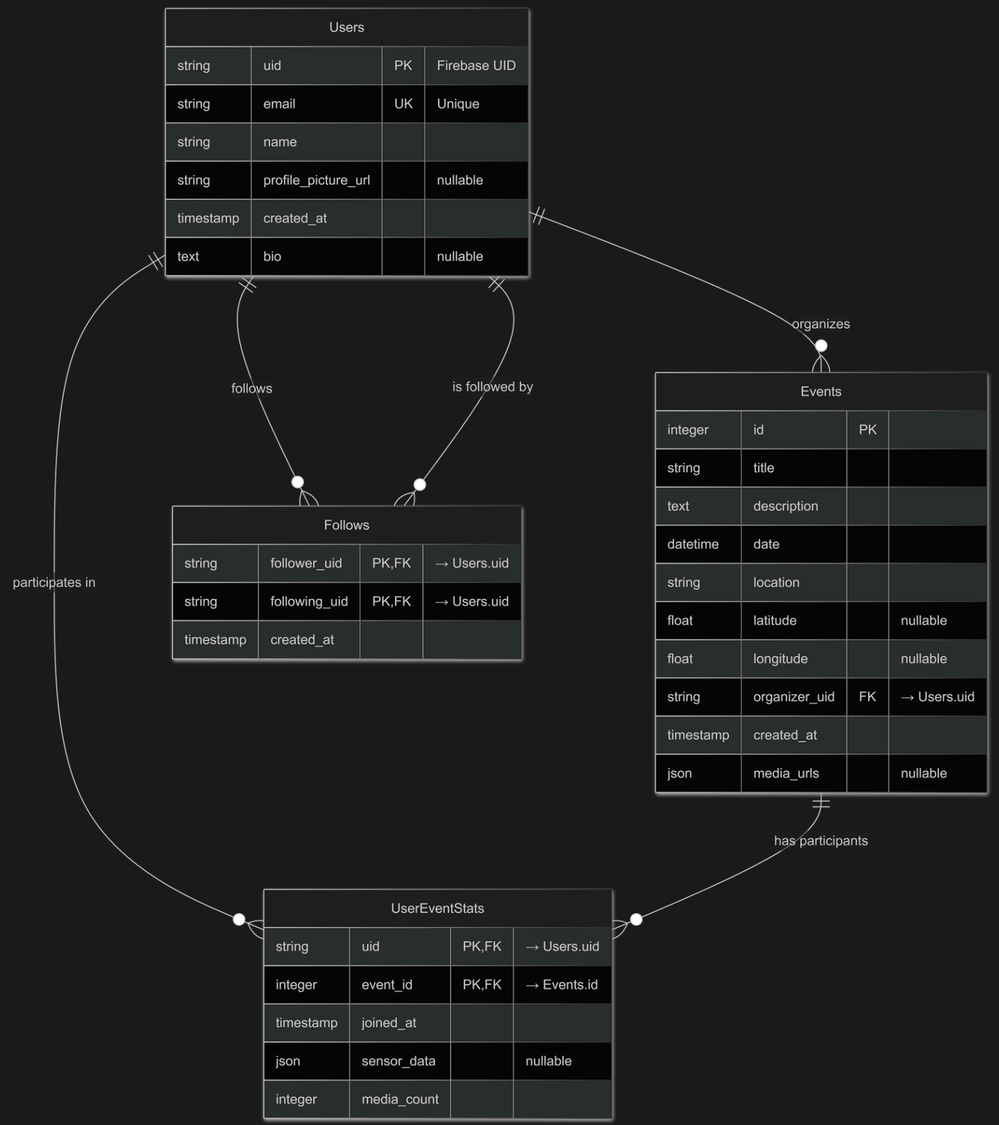
Events Endpoints (/events/*)

- GET /events/ - List events
- POST /events/ - Create event
- GET /events/<id> - Get event details
- PUT /events/<id> - Update event
- DELETE /events/<id> - Delete event
- POST /events/<id>/join - Join event
- DELETE /events/<id>/leave - Leave event
- GET /events/<id>/attendees - Get attendees

Storage Endpoints (/storage/*)

- POST /storage/upload - Upload media file

SERVER INFRASTRUCTURE



OVERCOMING DEVELOPMENT CHALLENGES

Challenge 1: Offline Data Synchronization

Problem: Poor venue connectivity & risk of data loss.

Solution: Offline-first architecture with local SQLite & background sync.

Challenge 2: Firebase Auth Integration with Custom Backend

Problem: Firebase Auth doesn't integrate directly with Flask.

Solution: JWT verification middleware using Firebase Admin SDK.

Challenge 3: Cross-Platform Testing

Problem: Different network/sensor behaviors on emulators vs devices

Solution: Centralized config.dart & environment-specific handling.

Challenge 4: Database Connection Pooling

Problem: Connection leaks & Supabase pooler limits

Solution: Environment-specific pooling (NullPool for Prod, small pool for Dev).

OVERCOMING DEVELOPMENT CHALLENGES

Challenge 5: Media Upload to Cloud Storage

Problem: Exposing Supabase keys on client-side.

Solution: Proxy uploads through backend using service role keys.

Challenge 6: Real-Time Sensor Data Handling

Problem: High-frequency data impacting UI performance & battery.

Solution: Throttled stream processing & aggregated storage.

Challenge 7: State Management Across App

Problem: Complex state sharing & prop drilling.

Solution: Provider pattern for efficient global state management.

LIVE DEMO

Add an Event

FUTURE ENHANCEMENTS & WRAP-UP

SCAN ME

https://fanzone-app.tech

FanZone

1. Music Integration:

- link events to artists/songs
- playlist generation from attended concerts

2. In-App Messaging:

- direct messaging between users
- event group chats
- push notifications

3. Gamification:

- badges for events milestones
- leaderboards for most active fans

4. Social Sharing & Community Features:

- Share to Instagram/TikTok/Twitter/...
- Fan groups for artists/genres

THANKS FOR YOUR ATTENTION!

AUTHORS

Carolina Reis, nº 131193
Carolina Silva, nº 113475

CURRICULAR UNIT

Mobile Computation

deti
universidade de aveiro
departamento de eletrónica, telecomunicações e informática