PROJECT ABSTRACT

Local Train Timetable Android App

An offline local train timetable Android app with station search, next-train listings, halt-by-halt details and service advisories. Delivered built-to-order with full Kotlin source, the timetable dataset, report, PPT and viva kit.

01. Title

Local Train Timetable Android App — the suburban timetable in your pocket, fully offline: station-to-station search, upcoming trains with halt counts, and train detail views with timings and fares.

02. Introduction / Background

Suburban commuters plan their days around local trains, yet timetables live in cluttered PDFs or websites that need connectivity — exactly when a commuter in a basement station cannot load them. A missed fast train versus a slow one can cost 20 minutes. A dedicated app answers 'next trains from Dadar to Thane' in two taps, offline, with fast/slow indicators, halt counts and platform hints upfront.

03. Problem Statement

Timetable information is fragmented, connectivity-dependent and hard to query by station pair. This project builds a native Android app with a bundled offline timetable in a Room database, typeahead station search with swap, next-train listings computed from the device clock, halt-by-halt train details, line filters, service advisories and saved routes.

04. Objectives

- Bundle a structured stations–trains–halts timetable dataset in an offline Room database.
- Implement station-to-station search with typeahead, swap and line filtering.
- List upcoming trains with times, durations, halt counts and fast/slow tags.
- Show train detail: halt-by-halt timings, platform hints and fare table.
- Follow MVVM + Repository architecture with Material Design 3; document the live-feed integration point.

05. Proposed Methodology

The timetable dataset is compiled into a relational schema (stations, trains, halts, fares) and seeded into Room on first launch. The search screen queries trains serving both selected stations, ordered by next departure from the device clock; journey time and halt counts derive from the halt table. Detail screens list halts with scheduled times. The live-status screen is architected against a timetable-derived estimate layer with a defined Retrofit integration point for a future real-time feed.

06. System Architecture / Working

Bundled dataset → Room (stations, trains, halts, fares) → repository → ViewModels (search, detail, live) → Compose/View UI with Navigation component. Saved routes persist in preferences; background refresh is stubbed for the future feed.

07. Hardware / Software Requirements

- Android Studio with an Android 8.0+ device or emulator.
- Kotlin, Android Jetpack (MVVM, Room, Navigation), Coroutines/Flow, Material 3.
- No server needed; the dataset ships inside the APK.

08. Expected Outcomes

A working offline timetable app with search, listings, details and advisories — plus the dataset and seeding scripts, build instructions, a project report, PPT and viva Q&A.

09. Applications

- Daily suburban commuters checking next trains offline.
- New residents and visitors navigating fast vs slow services.
- A template for other cities' suburban timetables.

10. Future Scope

- Real-time running-status feed integration (point designed).
- Crowd-level estimates per train from user reports.
- Multi-language station names and voice search.