Project Planning & Management

Project Overview

App Name: Routine - Habit Tracker App
Developed By: Jelly Bean Team

Technology Stack:

- **Frontend:** Kotlin with Jetpack Compose
- **Backend:** Firebase Authentication, Firestore
- **Architecture Pattern**: MVVM (Model-View-ViewModel)
- **API Handling:** Retrofit
- **Local Database:** Room Database
- **Live Data & State Management:** LiveData
- **UI Components:** RecyclerView, Custom UI elements (Jetpack Compose)

UI Design Screens: The following UI screens have been designed and implemented:

1. Welcome Screen: Introduction to the app.
2. Sign In Screen: User authentication via Firebase (Email & Social Login - Google, Apple, Facebook).
3. Register Screen: User account creation.
4. Reset Password Screen: Password recovery process.
5. Home Screen: Displays user habits, progress, and statistics.
6. Task Management Screen: Users can add, edit, and manage single or recurring tasks.
7. Timer Screen: Built-in focus timer for habit sessions.

8. Profile Screen: User settings, habit streaks, notifications, and customization options.

Screenshots & Design: The UI is designed in Figma and implemented in Jetpack Compose, following modern Material Design principles. The design is dark-themed with a minimalist and engaging user experience. (Refer to the UI screens provided above.)

Objectives:

- Provide users with an intuitive and efficient way to track habits and build routines.
- Improve habit consistency through reminders, streak tracking, and progress analytics.
- Enhance user engagement using gamification techniques.
- Offer AI-driven habit recommendations to personalize the experience.
- Ensure data security and seamless cloud synchronization using Firebase.

Project Scope:

- The application targets individuals looking to develop positive habits in areas like health, productivity, and personal development.
- Core features include habit tracking, reminders, a built-in timer, and analytics.
- The app will be available on Android, with potential expansion to iOS in the future.
- Users will have the ability to sync their data across multiple devices via Firebase.

Project Plan:

- Timeline: A Gantt chart illustrating the project schedule.
- Milestones: Key phases including planning, development, testing, and deployment.
- Deliverables: Functional application, documentation, and final report.
- Resource Allocation: Distribution of tasks and responsibilities across the team.

Task Assignment & Roles: Each team member has defined responsibilities, such as frontend development, backend integration, UI/UX design, and testing.

Risk Assessment & Mitigation Plan: Potential risks, such as feature delays, security vulnerabilities, or user engagement challenges, along with planned solutions.

KPIs (Key Performance Indicators): Metrics to measure project success, including response time, system uptime, and user adoption rate.

Literature Review

A comprehensive literature review was conducted to analyze existing habit-tracking applications, psychological studies on habit formation, and user engagement strategies. Key findings include:

- The importance of gamification in user motivation
- The effectiveness of reminders and streak tracking
- The role of AI-driven habit recommendations
- Challenges faced by users in maintaining consistency

These insights helped shape the design and functionality of Routine.

Feedback & Evaluation: Lecturer's assessment of the project based on implementation, usability, and design effectiveness.

Suggested Improvements: Areas where the project can be enhanced, such as additional features, performance optimization, or UI refinements.

Final Grading Criteria: Marks breakdown based on:

- Documentation
- Implementation
- Testing
- Presentation

Requirements Gathering

The requirements were gathered through surveys, interviews, and competitive analysis. Key functional and non-functional requirements identified include:

Stakeholders:

- **End-Users:** Want a user-friendly, customizable, and motivational tracking experience.
- **App Developers:** Focus on building a stable and high-quality app with positive feedback.
- **Marketing/Sales Team:** Aim to increase user base, revenue, and brand awareness.
- **Platform Providers:** Ensure compliance with their guidelines and generate revenue.
- **Health Professionals (Potential):** Value accurate tracking and data privacy.
- **Advertisers (Potential):** Seek effective user engagement and brand promotion.

Functional Requirements:

- User authentication and profile management
- Habit creation, modification, and tracking
- Customizable reminders and notifications
- Data synchronization with Firebase
- Offline mode support

Non-functional Requirements:

- High performance and smooth user experience
- Data security and privacy compliance
- Scalability for future enhancements

System Analysis & Design

Problem Statement & Objectives:

Problem Statement: People struggle to maintain consistency in their daily habits due to lack of motivation, poor habit-tracking mechanisms, and inefficient reminder systems. Existing habit tracker apps often lack personalization and engagement, leading users to abandon their habit-forming journey.

Objectives:

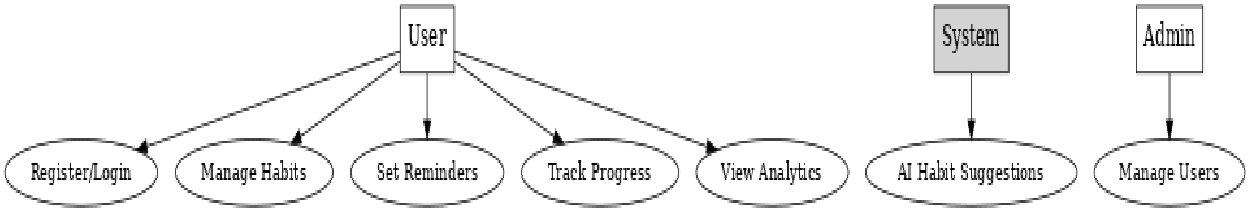
- Develop a user-friendly habit tracking application to help users build and maintain positive habits.
- Implement reminders and notifications to keep users on track.
- Provide streak tracking and gamification to enhance user engagement.
- Use AI-driven habit recommendations to personalize the experience.
- Ensure secure and seamless synchronization across devices using Firebase.

Use Case Diagram & Descriptions

Actors & Interactions:

- User: Can register, log in, create habits, edit habits, delete habits, set reminders, track progress, and view analytics.
- System: Handles authentication, stores and retrieves habit data, sends notifications, and provides insights.
- Admin: Manages system settings and monitors analytics.

Use Case Diagram:



Functional & Non-Functional Requirements

Functional Requirements:

- User Authentication: Register, login, and manage profiles via Firebase Authentication.
- Habit Management: Create, edit, delete, and track habits.
- Reminders & Notifications: Customizable habit reminders.
- Streak & Progress Tracking: Track user consistency and offer motivation.
- AI Habit Suggestions: Recommend habits based on user behavior.
- Data Synchronization: Cloud-based storage and retrieval via Firebase.
- Offline Mode: Basic functionality without internet.

Non-Functional Requirements:

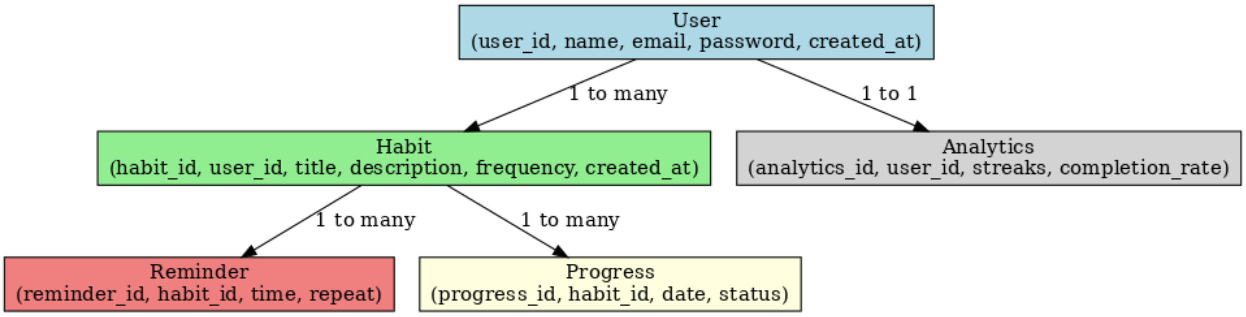
- Performance: Fast response time (<1s for most actions).
- Security: Secure user data using Firebase Authentication and Firestore rules.
- Scalability: Capable of handling thousands of users without performance loss.
- User Experience: Intuitive UI with a dark mode and smooth navigation.
- Reliability: 99.9% uptime with Firebase backend.
-

Software Architecture

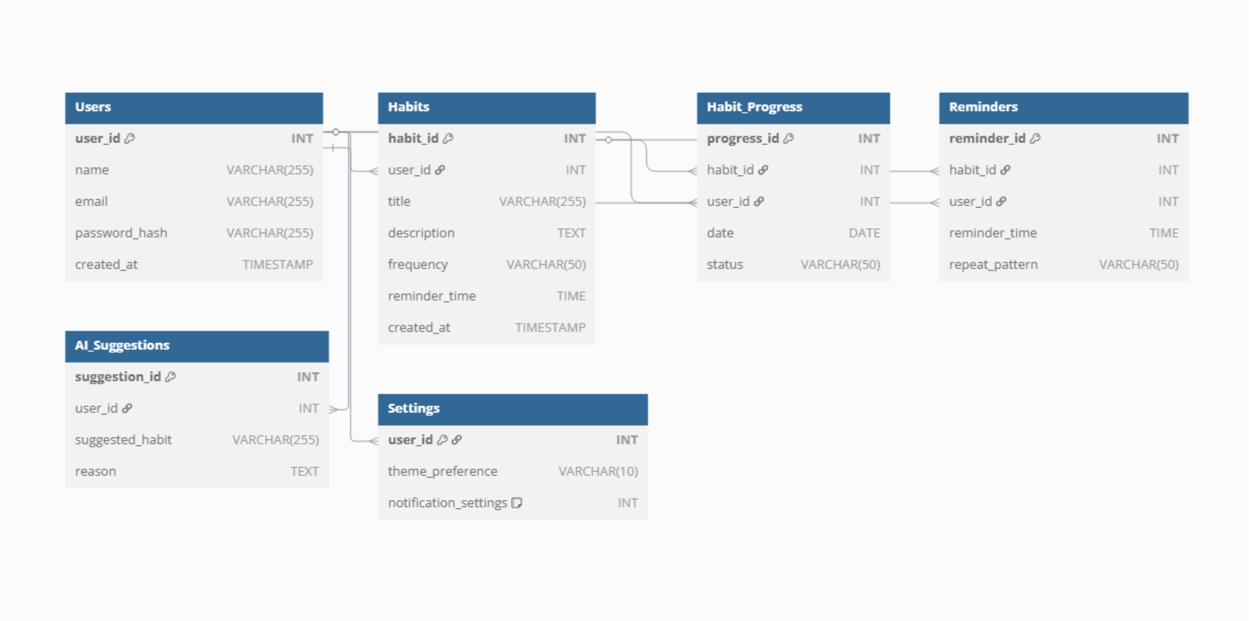
- Architecture Pattern: MVVM (Model-View-ViewModel) for clean UI separation.
- Frontend: Jetpack Compose (Kotlin).
- Backend: Firebase (Firestore for database, Authentication for login).
- API Handling: Retrofit for external API calls (if needed).
- Local Storage: Room Database for offline access.
- Live Data & State Management: LiveData for real-time updates.

Database Design & Data Modeling

ER Diagram (Entity-Relationship Diagram):


User
(user_id, name, email, password, created_at)
1 to many
1 to 1
Habit
(habit_id, user_id, title, description, frequency, created_at)
Analytics
(analytics_id, user_id, streaks, completion_rate)
1 to many
1 to many
Reminder
(reminder_id, habit_id, time, repeat)
Progress
(progress_id, habit_id, date, status)

Logical & Physical Schema:



Users	
user_id	INT
name	VARCHAR(255)
email	VARCHAR(255)
password_hash	VARCHAR(255)
created_at	TIMESTAMP

Habits	
habit_id	INT
user_id	INT
title	VARCHAR(255)
description	TEXT
frequency	VARCHAR(50)
reminder_time	TIME
created_at	TIMESTAMP

Habit_Progress	
progress_id	INT
habit_id	INT
user_id	INT
date	DATE
status	VARCHAR(50)

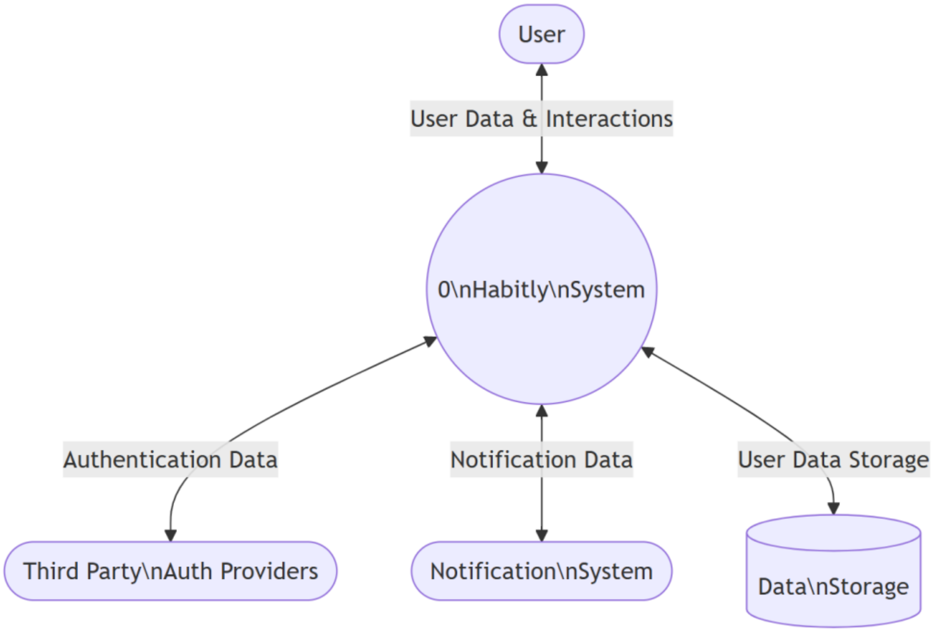
Reminders	
reminder_id	INT
habit_id	INT
user_id	INT
reminder_time	TIME
repeat_pattern	VARCHAR(50)

AI_Suggestions	
suggestion_id	INT
user_id	INT
suggested_habit	VARCHAR(255)
reason	TEXT

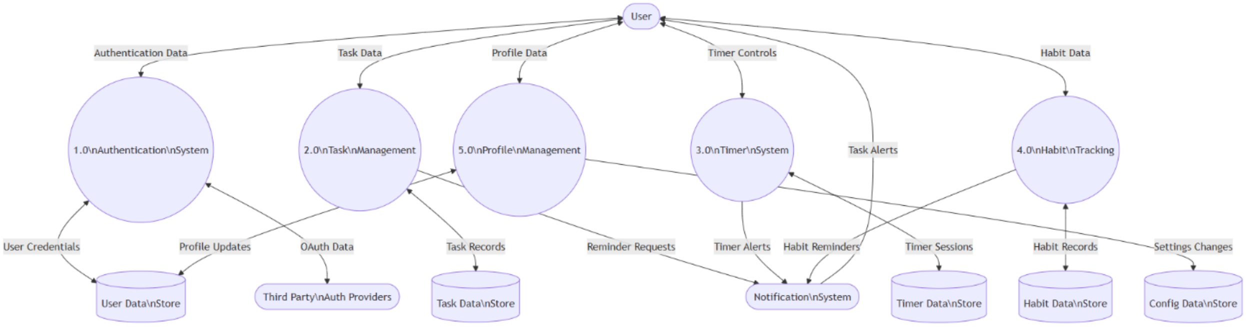
Settings	
user_id	INT
theme_preference	VARCHAR(10)
notification_settings	INT

Data Flow & System Behavior:

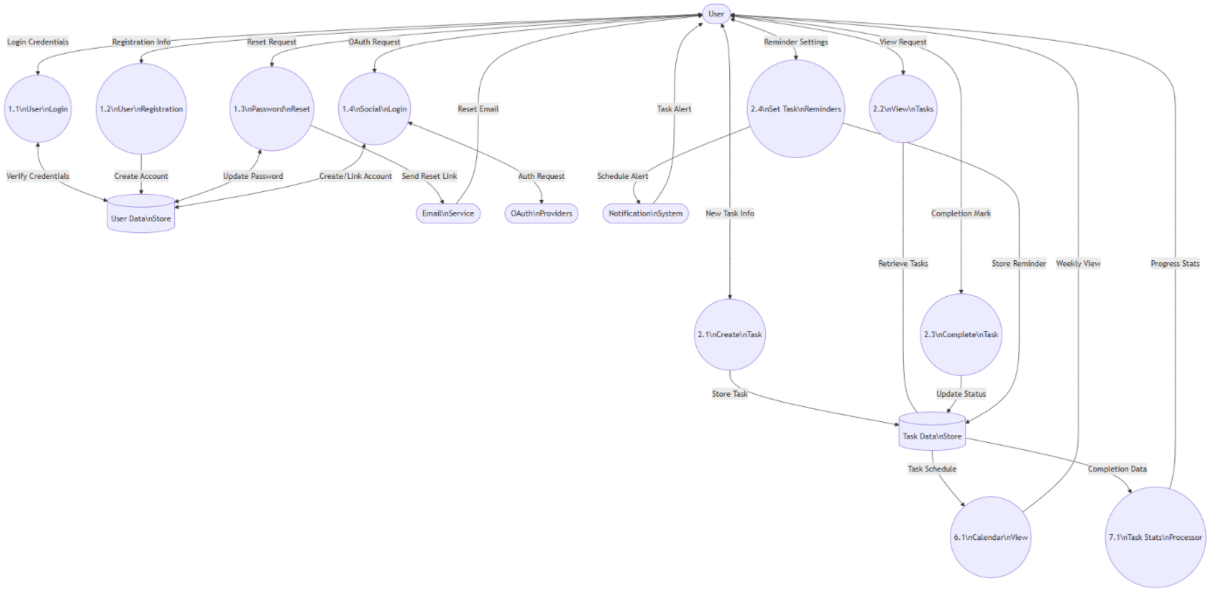
- DFD (Data Flow Diagram) Level 0:



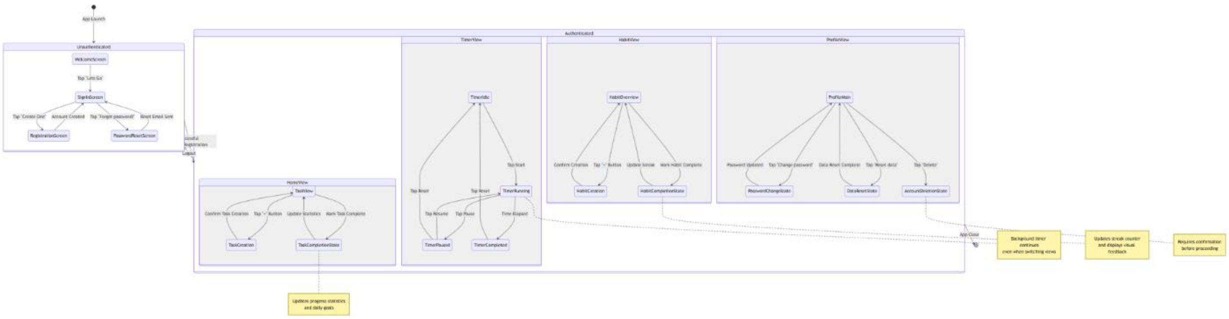
- DFD (Data Flow Diagram) Level 1:



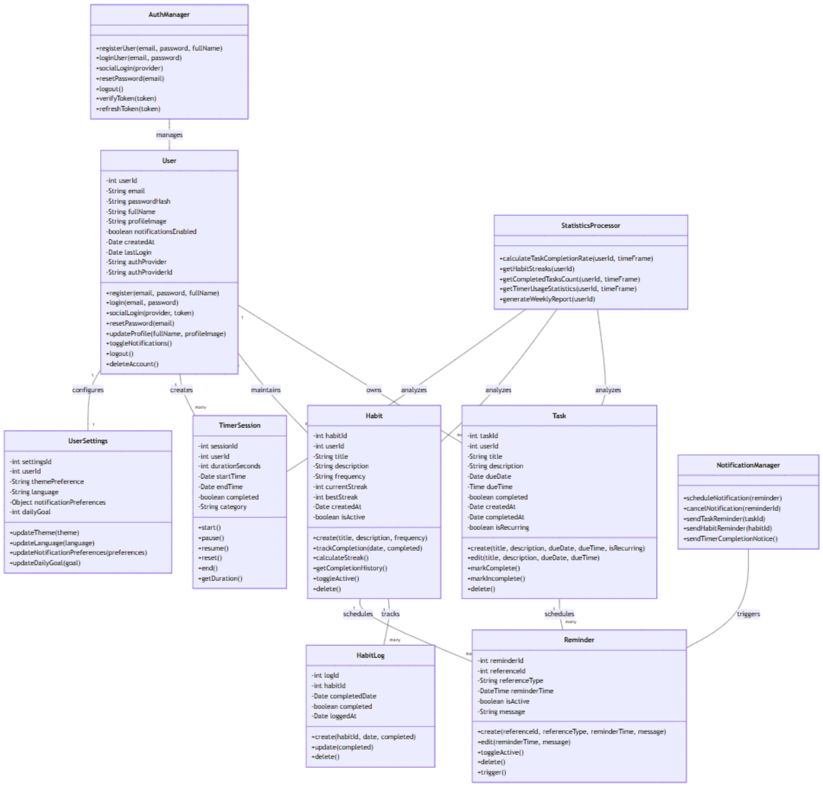
- DFD (Data Flow Diagram) Level 2: (Authentication & Task Management Focus)



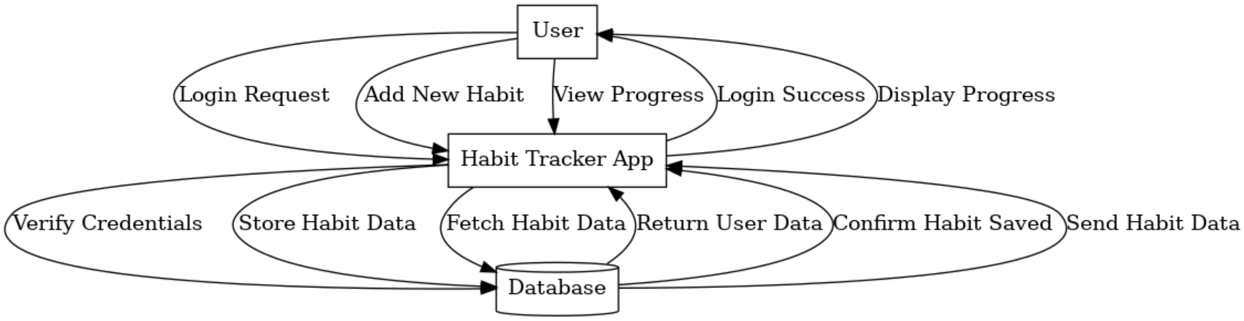
- State Diagram:



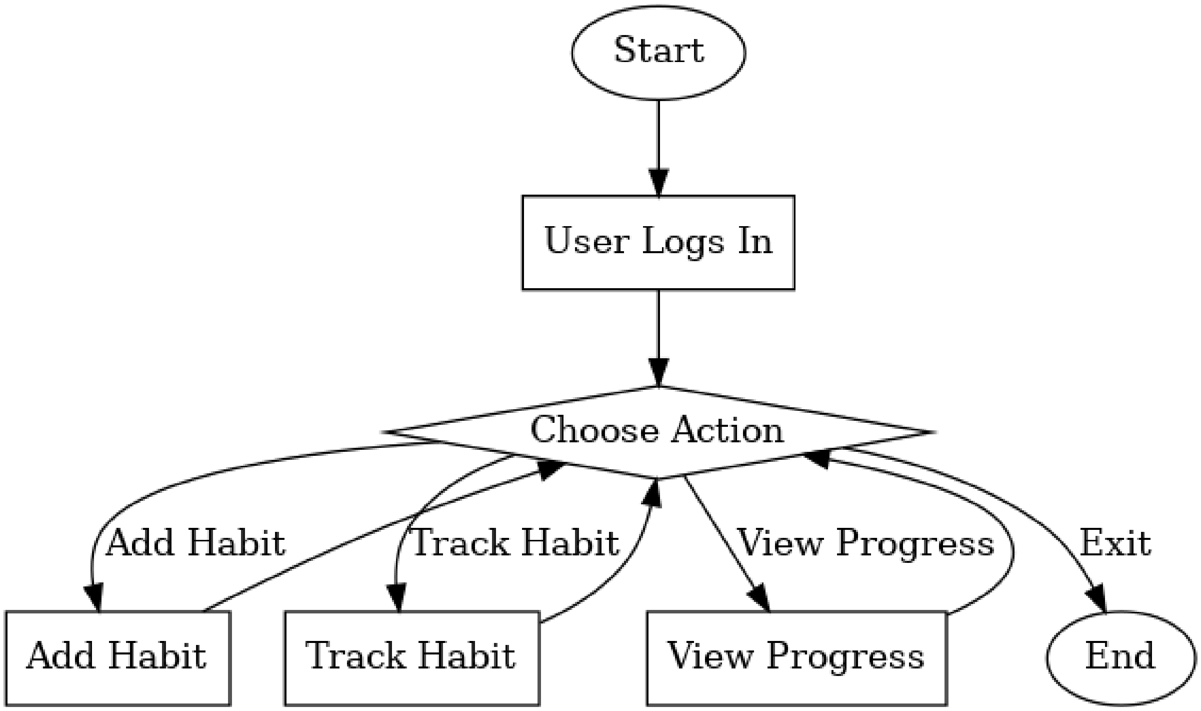
- Class Diagram:



- Sequence Diagrams:



- Activity Diagram:



UI/UX Design & Prototyping:

Wireframes & Mockups

Key Screens:

- Onboarding & Authentication (Sign-up, Login)
- Dashboard (Daily habits, progress overview)
- Habit Creation (Title, frequency, reminders)
- Habit Tracking (Calendar, streaks, completion status)
- Profile & Settings (Theme, notifications, account management)

UI/UX Guidelines

Design Principles:

Minimalist, intuitive, and user-friendly
Motivating color scheme (Blue for focus, Green for success)
High contrast & accessibility support

Typography & Colors:

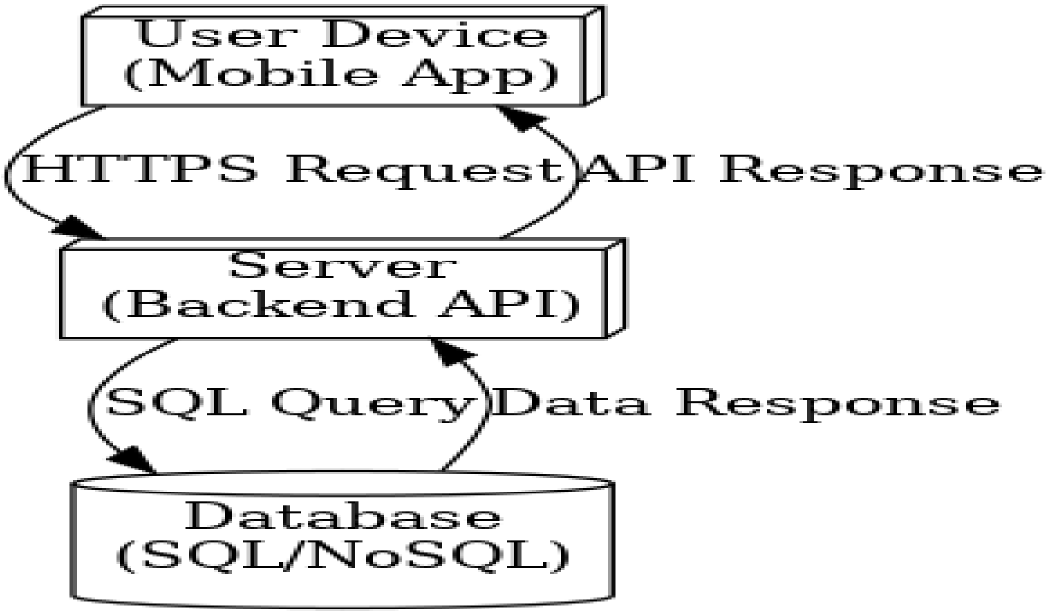
- Font: Roboto (clean & modern)
- Primary Colors:
  - Blue (#419CA7) – Focus & productivity
  - White (#FFFFFF) – Success & growth
  - Dark (#121212) – Clean background

System Deployment & Integration:

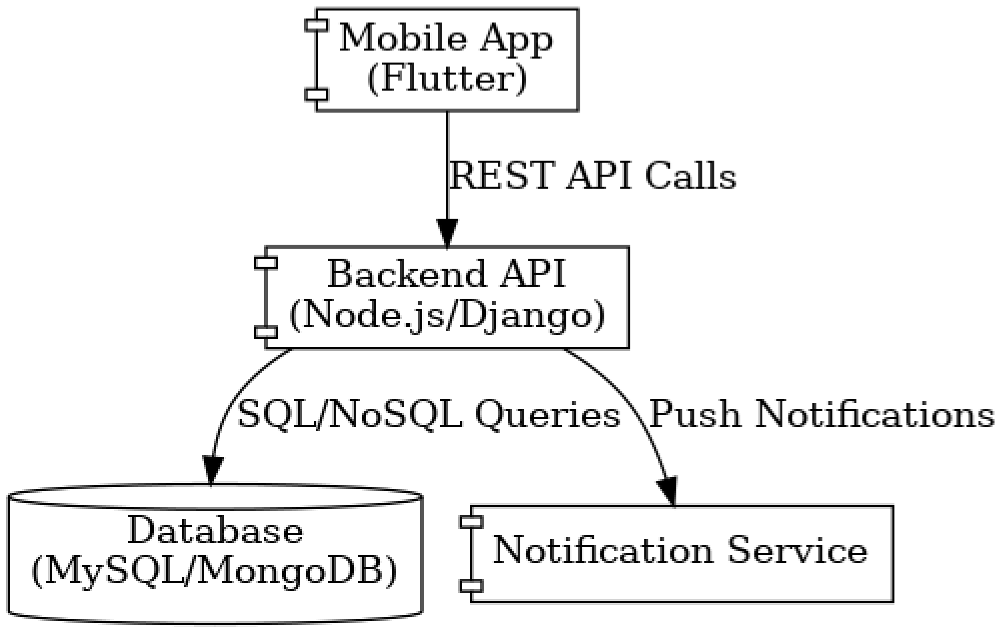
Technology Stack:

- Frontend: Jetpack Compose (Kotlin)
- Backend: Firebase Firestore & Authentication
- API Handling: Retrofit
- Local Database: Room Database

- Deployment Diagram:



- Component Diagram:



Additional Deliverables

API Documentation

Auth APIs

- POST /api/register – Register user
- POST /api/login – Authenticate user
- POST /api/logout – Logout

Habit APIs

- POST /api/habits – Create habit
- GET /api/habits – List user habits
- PUT /api/habits/{id} – Update habit
- DELETE /api/habits/{id} – Delete habit

Tracking APIs

- POST /api/habits/{id}/track – Mark habit as completed/missed

- GET /api/habits/progress – View progress

Reminder APIs

- POST /api/reminders – Create reminder
- GET /api/reminders – List reminders
- DELETE /api/reminders/{id} – Delete reminder

Settings APIs

- GET /api/settings – Get user settings
- PUT /api/settings – Update settings

Testing & Validation

- **Unit Tests** – Habit creation, tracking, and authentication.
- **Integration Tests** – API and database interactions.
- **UAT** – User experience testing and feedback.

Deployment Strategy

- **Hosting** – AWS / Firebase
- **CI/CD** – GitHub Actions / Fastlane
- **Scaling** – Caching, load balancing, auto-scaling