UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

NAAC GRADE A+
Accredited University

MOBILE APPLICATION DEVELOPMENT

UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

PROJECT REPORT ON
FITNESS TRACKING APPLICATION

SUBMITTED BY:

RITINDER KAUR

UID:24MCI10092

SECTION:24MAM1A

SUBMITTED TO:

MS. GURPREET KAUR

E14273

BRANCH:MCA[AI&ML]

Student Name :Ritinder Kaur
UID: 24MCI10092
Branch : MCA(AIML)
Section/Group :24MAM-1A
Semester : 2nd
Date of submission:16/04/25
Subject Name : Mobile App Development Lab
Subject Code : 24CAH-653

1. Abstract

The Fitness Tracker App is a mobile application developed for Android that allows users to record and monitor key fitness metrics including workouts, water intake, and calorie consumption. It provides a secure login and registration system so each user's data remains private and personalized. The application features a simple interface for logging exercises (type and duration), tracking daily water consumption, and recording meals along with calories. All data is stored locally on the device using an SQLite database, enabling offline access and persistence of records. This report outlines the objectives, design, implementation, results, testing, and security considerations of the Fitness Tracker App, demonstrating how it fulfills its goal of helping users track their fitness activities on a daily basis.

2. Introduction

In today's health-conscious world, many individuals track their fitness activities to maintain a healthy lifestyle. Mobile applications serve as convenient tools for logging workouts, diet, and hydration habits. The **Fitness Tracker App** was conceived to provide an all-in-one solution for personal fitness tracking. This Android application enables users to **log their daily workouts**, **track water intake**, and **monitor calorie consumption** through meal logging. By consolidating these features into a single app, users can gain insights into their daily health-related activities and progress over time.

The Fitness Tracker App is designed with ease-of-use in mind, featuring a clean interface and straightforward workflow. Users can create a personal account, log in securely, and then navigate to different sections for logging fitness data. Each log entry is time-stamped and stored locally so that users can review their history

at any time without needing an internet connection. The app leverages a local SQLite database for data persistence, ensuring that information remains available even if the app is closed or the device is offline. The following sections of this report detail the project's objectives, the technologies and design approach used, implementation specifics, the results obtained, testing and validation procedures, security considerations, and conclusions drawn from the development of the Fitness Tracker App.

3. Objectives

The primary objectives of the Fitness Tracker App project are as follows:

- **Secure User Authentication:** Implement a registration and login system to allow users to create personal accounts and ensure that each user's data is isolated and accessible only after a successful login.
- **Workout Logging:** Provide a feature for users to record their workout sessions by specifying the workout type (e.g., running, cycling) and duration (in minutes), and maintain a chronological log of all workouts.
- **Water Intake Tracking:** Enable users to track their daily water consumption by logging the amount of water (in milliliters) they drink, helping users monitor their hydration habits over time.
- **Calorie Tracking:** Allow users to log their meals with a description and the number of calories consumed, creating a daily food diary that helps monitor calorie intake.
- **Data Persistence:** Store all user information and fitness logs in a persistent local database so that data is saved between sessions and can be reviewed later, even without network connectivity.
- **User-Friendly Interface:** Design a simple and intuitive user interface for all features, making it easy to navigate between logging activities and to view past records without confusion or clutter.

By achieving these objectives, the project seeks to create a functional app that assists users in maintaining their fitness routines and encourages consistent tracking of health metrics.

4. Technology Stack

The Fitness Tracker App is built using the following technologies and tools:

- **Android Studio & Android SDK:** The development environment and platform used for building the Android application. The app targets the Android OS and was developed and tested using Android Studio.
- **Kotlin Programming Language:** All application code is written in Kotlin, a modern, statically-typed language officially supported for Android development. Kotlin provides concise syntax and null-safety features which helped reduce errors.
- **XML Layouts (Android UI):** User interfaces for each screen (activities) are designed using XML layout files. Standard Android UI components such as EditText (text input fields), Button, TextView, and ListView are used to build the interactive forms and display lists of logs.
- **SQLite Database:** The app uses a built-in SQLite database for local data storage. Android's SQLiteOpenHelper class (implemented via a custom DatabaseHelper class) is utilized to create and manage the database and tables that store users and fitness logs.
- **AndroidX Libraries:** The application uses AndroidX support libraries (e.g., AppCompatActivity) for backward compatibility and modern Android components. Toast messages are used from the Android framework to provide feedback to users on actions (e.g., error messages, success notifications).

This technology stack was chosen to ensure the app is lightweight, runs entirely on-device, and leverages the robust capabilities of the Android platform for data storage and UI.

5. System Design

The system is designed with a modular structure, separating concerns between user interface components (activities for different features) and the data storage layer (SQLite database). The high-level architecture consists of multiple activities orchestrating user interactions, and a single database that persists all relevant data. The major components of the system design are:

- **User Authentication Module:** This includes the **Login** and **Registration** activities. New users can register an account, and existing users can log in. Authentication ensures that each user's data is kept separate. Only after

logging in can a user access the tracking features, which protects personal fitness data from unauthorized access.

- **Main Dashboard:** After a successful login, the user is presented with a main menu (Main Activity) that serves as a dashboard. It displays a welcome message with the user's name and provides navigation buttons to each tracking section: Workouts, Water, Calories. It also includes a Logout option. The design here focuses on simplicity, allowing users to clearly choose which activity they want to log or review.
- **Workout Logging Module:** The Workout Activity allows users to input details of workouts. Its design includes input fields for workout type and duration and a list view to display all logged workouts. This module is responsible for capturing exercise data and showing the user's workout history.
- **Water Intake Module:** The Water Activity is designed for logging water consumption. It consists of an input for the amount of water consumed and a list view showing all recorded water intake entries. It helps users ensure they meet daily hydration goals by maintaining a log of water intake.
- **Calorie Tracking Module:** The Calorie (Meal Logging) Activity allows users to record meals and calories. It includes input fields for meal description and calorie amount, along with a list of past meal entries. This module provides an overview of daily calorie intake and can help users manage their diet.
- **Local Database (SQLite):** A single SQLite database (FitnessTracker.db) underpins the entire application, storing all persistent data. A custom DatabaseHelper class defines the schema and provides methods to interact with the data. The database contains multiple tables to support the app's features:
  - *users* – Stores user account information with fields: **user_id** (primary key), **username** (unique), and **password**. This table is used for login authentication and ensuring usernames are unique.
  - *workouts* – Stores workout log entries with fields: **workout_id** (primary key), **user_id** (foreign key referencing a user), **type** (text describing the workout type), **duration** (integer minutes), and **date** (text timestamp). Each entry represents a workout a user has logged.
  - *water_intake* – Stores water consumption logs with fields: **water_id** (primary key), **user_id** (foreign key), **amount** (integer volume of water in milliliters),

and **date** (text timestamp). Each record corresponds to a logged water intake event.

- *meals* – Stores meal (calorie) logs with fields: **meal_id** (primary key), **user_id** (foreign key), **meal** (text description of the food or meal), **calories** (integer number of calories), and **date** (text timestamp). Each entry is a meal the user has recorded for calorie tracking.

6. Implementation

The Fitness Tracker App was implemented in Kotlin following the design outlined above. Each major feature corresponds to an Activity and associated database functions. The key implementation details for each feature are described below:

- **User Registration:** Implemented in RegisterActivity, this feature allows new users to create an account. The registration screen provides input fields for a username, password, and password confirmation. When the user taps the Register button, the app validates the input – ensuring none of the fields are empty and that the password and confirmation match. It then interacts with the database via the DatabaseHelper. A check is performed using isUsernameTaken to ensure the chosen username isn't already in use. If the username is unique, a new user record is inserted into the **users** table by calling addUser with the provided credentials. The user receives feedback through Toast messages: for example, an error message if the username is already taken or the passwords don't match, and a success message if registration is completed. On successful registration, the RegisterActivity calls finish(), which returns the user to the login screen so they can proceed to log in with the new credentials.
- **User Login:** The login functionality is handled in LoginActivity. The login screen contains fields for username and password and buttons for "Login" and "Register". When the user attempts to log in, input validation checks that the username and password fields are not empty (showing a Toast error if they are). The app then calls DatabaseHelper.getUserId(username, password) to verify the credentials against the **users** table. If a matching user is found (i.e., the credentials are correct), the login is considered successful: the app retrieves the user's unique ID and proceeds to launch the Main Dashboard (MainActivity). An Intent is used to start MainActivity, and the user's ID and username are passed along to the next screen. The LoginActivity is

finished/closed upon success so that the user cannot navigate back to it without explicitly logging out, thereby maintaining a basic session. If the credentials are invalid, a Toast message ("Invalid username or password") is displayed to inform the user that login failed. The login screen also has a Register button which simply opens the RegisterActivity, allowing new users to sign up.

- **Water Intake Logging:** Water tracking is implemented in WaterActivity in a manner similar to workouts but simpler, as it involves only one numeric input. The UI consists of an EditText for the amount of water (in milliliters) and an "Add" button, along with a ListView showing past water intake logs. Upon opening this screen, the app calls DatabaseHelper.getWaterLogs(userId) to retrieve all water intake records for the user from the **water_intake** table. Each record is formatted (for example, as "2025-04-16 10:00 - 250 ml") and displayed in the ListView via an ArrayAdapter. When the user enters an amount and presses the Add button, the input is validated to ensure it is not empty and represents a positive integer (amount of water in ml). If the field is empty or the value is zero/non-numeric, the app shows a Toast error (e.g., "Please enter a valid water amount") and does not proceed
- **Calorie Tracking (Meal Logging):** The calorie tracking feature is handled by CalorieActivity. Its user interface includes an EditText for the meal or food item description, another EditText for the number of calories, and an "Add Meal" button, along with a ListView of past meal entries. On activity start, DatabaseHelper.getMealLogs(userId) is executed to fetch all existing meal records for the user from the **meals** table. These are returned as formatted strings (e.g., "2025-04-16 12:30 - Breakfast: 300 cal") combining the timestamp, meal name, and calories, and displayed in the ListView using an ArrayAdapter. To add a new entry, the user inputs the meal name and calorie count and taps the Add button. The implementation validates that neither field is empty; it also checks that the calorie input is a positive integer. If validation fails (e.g., missing meal name or non-numeric calories), a Toast message ("Please enter meal and calories") is shown to prompt the user.

7. Result

The completed Fitness Tracker App effectively meets its goals, offering a smooth, interactive experience for users to track workouts, water intake, and calorie consumption. Users can register, log in, and seamlessly log activities, with entries

instantly displayed on-screen and saved using SQLite, ensuring data persists across sessions. The user interface is clean and intuitive, with clearly labeled input fields and easy navigation. Logs include timestamps for context, and the app runs efficiently offline, making it ideal for use in gyms or during outdoor activities. Overall, the app is a simple yet powerful tool for daily fitness tracking, demonstrating strong Android development and UI design skills.

8. Testing and Validation

Thorough testing was conducted to validate that each component of the Fitness Tracker App works correctly and reliably. The testing covered functional verification of each feature, input validation checks, and overall integration to ensure the user flow is smooth. Below are the key areas tested and their outcomes:

- **Login Authentication:** The login feature was validated by testing both failure and success paths. Entering an incorrect username or wrong password for an existing user resulted in an "invalid credentials" Toast and remained on the login screen, as it should. When the correct username and password were entered, the app successfully transitioned to the Main Dashboard, carrying over the user's information. It was also verified that upon successful login, using the back button would not return to the login screen (because the LoginActivity finishes on success), thereby simulating a basic session management. This ensures that users cannot accidentally go back to a stale login screen or bypass the login.
- **Workout Logging Functionality:** The workout log screen was tested by inputting various workout entries and checking their appearance in the list. For instance, a test entry like *Type: "Running", Duration: 30* was added – after tapping "Add Workout", the new entry appeared at the top of the list with the current date/time and text "Running: 30 min", confirming the data was saved and retrieved correctly. Input validation was also tested: leaving the type or duration blank, or entering a non-numeric or zero duration, correctly triggered an error Toast and no entry was added to the list (ensuring bad data is not stored). Multiple workouts were added in succession to verify that the list view updates continuously and maintains all entries (each new entry pushing the previous ones down). The ordering of the log (newest first) was confirmed by checking timestamps. These tests demonstrated that workout logging is reliable, and data integrity (only valid workouts are logged) is maintained.

- **Water Intake Logging:** Similar tests were performed on the water tracking screen. A valid water intake entry (e.g., *250 ml*) was added and immediately showed up in the list with a timestamp (e.g., "2025-04-16 18:30 - 250 ml"). Attempts to add invalid data – such as an empty input or "0" ml – were correctly rejected with an error message and no list update, indicating the validation logic works. Adding multiple water entries (for example, logging water multiple times a day) resulted in a list of entries, all correctly formatted and present. The persistence of these entries was also checked by navigating away and back (or restarting the app) and seeing that the list still contained the previously added values, confirming they were stored in the database.
- **Calorie Tracking (Meal Logging):** The meal logging feature was tested by adding sample meal entries. For example, adding *"Breakfast – 400 cal"* and *"Lunch – 600 cal"* in sequence. After each addition, the entry appeared in the calories list view with the timestamp and details ("Breakfast: 400 cal", etc.). The app was tested for preventing invalid inputs: trying to add a meal without a name or without calories (or with a non-numeric calorie value) prompted an error and did not add anything to the list, as expected. It was also observed that the calorie logs list retains all entries and shows the latest meal first. By adding several entries, it was confirmed that all are retained and displayed properly, verifying that there is no unintended limit or overwrite happening in the database.

9. Security Considerations

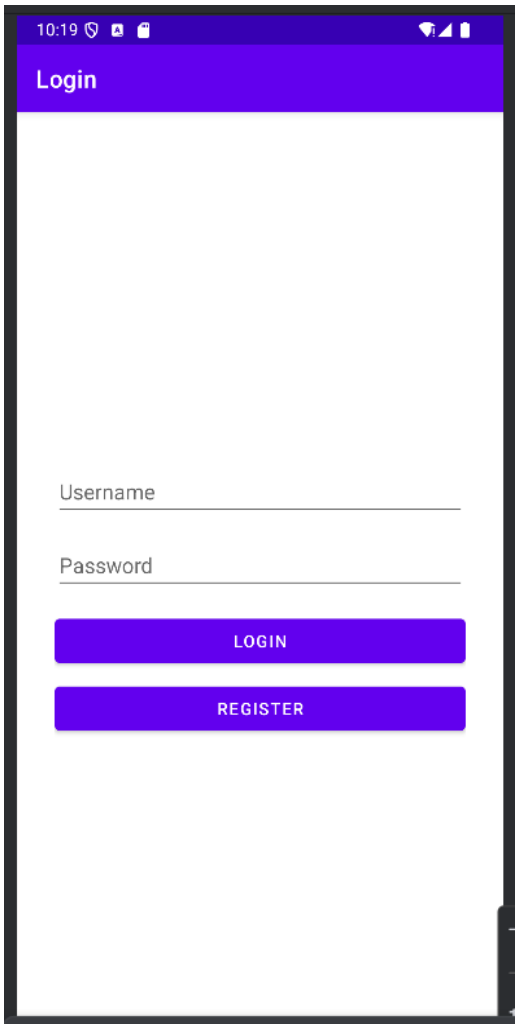
While the Fitness Tracker App is a client-side application with data stored locally, several security and privacy considerations were acknowledged during development:

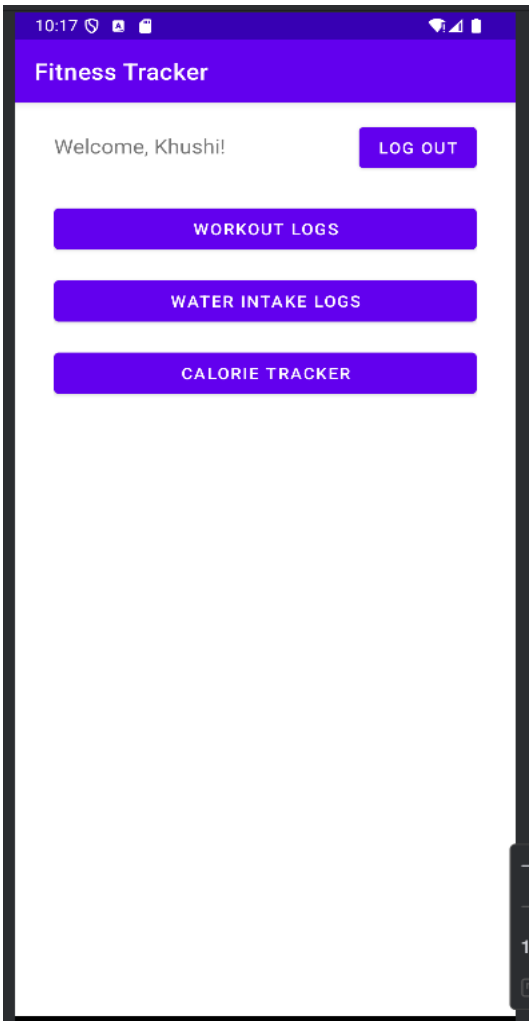
- **Account Security:** The app requires users to authenticate (log in) before accessing any fitness data. This protects the user's personal logs from casual access by others who might use the same device. Each user's data is keyed by a user ID, and the application only queries records for the logged-in user, ensuring data isolation between accounts. This design means that even if multiple people use the app on one device, they cannot see or modify each other's fitness logs without knowing the correct credentials.
- **Password Storage:** User passwords are stored in the SQLite database in plain text form, which is a **security weakness** in the current implementation. This was acceptable for a small-scale college project, but in a real-world scenario, storing plaintext passwords is risky. Anyone with access to the device's raw database file could potentially read user credentials. A more secure approach

would be to store hashed and salted password representations so that the actual passwords are not retrievable even if the database is compromised. Implementing password hashing was identified as a potential improvement to enhance security.

- **Data Privacy on Device:** All fitness data (workouts, water intake, meals) resides in a local database on the user's device. Android sandboxes app data by default, which means no other app can directly access the Fitness Tracker App's database. However, if the device is rooted or someone gains physical access to the internal storage, the data could be extracted. No encryption is applied to the stored data in this implementation. For better privacy, especially for sensitive personal health data, one could consider encrypting the database or at least certain fields (though this adds complexity). On the upside, since data never leaves the device, there is no exposure over a network or cloud – eliminating risks of data interception during transmission.

10.Output:





11. Conclusion

The development of the Fitness Tracker App resulted in a successful implementation of a multi-feature Android application for personal health tracking. The project achieved all its stated objectives: users can securely register and log in, and once authenticated, they have access to an intuitive interface for logging workouts, water intake, and calorie consumption. The use of a local SQLite database fulfills the requirement of data persistence, allowing users to accumulate and review their fitness data over time. Each component of the app – from the user authentication to the logging of different activities – works in concert to provide a seamless user experience.

This project not only provides a practical tool for end users but also demonstrates solid understanding of Android development principles. Key takeaways include the effective use of Activities for separate concerns, managing state and navigation between screens, implementing robust input validation, and handling data storage and retrieval using SQLite. The simplicity of the app's design proved to be a strength, as it made the application easy to test and ensured reliability. The end result is a lightweight and responsive app that can run entirely offline, which can be particularly useful for users who prefer not to depend on internet connectivity for tracking their daily health metrics.

While the current version of the Fitness Tracker App is fully functional, there are opportunities for future enhancements. For instance, implementing password hashing would greatly improve security for user credentials. Additional features like graphical summaries of data (charts of weekly workouts or daily water intake), reminders to log activities, or syncing capabilities for backup could further increase the app's utility and appeal. Nonetheless, as it stands, the project successfully delivers a focused solution for fitness tracking. It provides users with greater awareness of their daily exercise, hydration, and nutrition, thereby supporting healthier habits. In conclusion, the Fitness Tracker App project is a meaningful step toward leveraging mobile technology to promote personal fitness, and it serves as a strong foundation for any further development or improvements in this domain.