**A REPORT**

**ON**

**TASKFLOW MANAGEMENT**

## **By**

-AP22110011253

-AP22110011255

-AP22110011275

-AP22110011294

-AP22110011307

***Under the Supervision of***

**Prof. Kavitha Rani Karnena  
Assistant Professor**

**Department of Computer Science and Engineering  
SRM University-AP**

**BACHELOR OF TECHNOLOGY  
IN  
COMPUTER SCIENCE AND ENGINEERING**

***Prepared in the partial fulfillment of the***

Project

|  |
| --- |
|  |

#### SRM UNIVERSITY, AP

**(April,2025)**

**Acknowledgement**

I would like to take this opportunity to express my heartfelt gratitude to all those who have supported me directly or indirectly throughout the development of this project. Their guidance, encouragement, and continuous support have been instrumental in the successful completion of the project.

I express my sincere thanks to **Dr. Kavitha Rani Karnena**, my project supervisor, for his invaluable guidance and consistent support throughout the course of this work. His deep knowledge, constructive feedback, and motivation have been a constant source of inspiration. Working under his supervision has greatly enriched my learning experience and contributed to my academic growth.

I would also like to thank the **faculty members and peers at SRM University - AP** for their insightful suggestions, technical discussions, and support, which have helped shape this project. Their contribution has been vital in navigating the challenges and refining the features of the app.

I am also thankful to my **family and friends** for their unwavering encouragement, patience, and motivation throughout this journey. Their belief in my abilities has provided the emotional support necessary to stay focused and dedicated.

This project, titled **"Task Flow"**, is an Android application developed in Java to manage and track tasks with features like user authentication, offline data storage via Room Database, task filtering, and automatic status updates. Every element of the app has been developed with the end-user in mind, and I remain open to any suggestions or feedback for further enhancement.

Lastly, I extend my sincere appreciation to everyone who has contributed in any capacity to the successful realization of this project. Your support has been truly invaluable.

**ABSTRACT**

Task Flow is an Android-based task management application developed in Java that empowers users to efficiently organize, track, and manage their daily activities. Designed with user productivity and simplicity in mind, the application offers a seamless experience for creating, viewing, and filtering tasks, complete with titles, descriptions, deadlines, and status indicators.

A key feature of Task Flow is its ability to automatically update the status of tasks to Completed once their deadline has passed, reducing manual oversight and enhancing time management. The application stores all task-related data locally using the Room Database, ensuring full offline functionality and persistent data access regardless of internet connectivity.

To safeguard user data and personalize the task management experience, secure authentication is implemented, allowing users to sign up and log in to their accounts. The intuitive and responsive user interface is designed using Android best practices to ensure smooth navigation and accessibility across a range of devices.

Additional features include task filtering by status (Pending or Completed), enabling users to easily monitor progress and focus on outstanding tasks. The app is developed using Android Studio, with compatibility for Java 8 or higher and the Android SDK.

Through its simple yet powerful feature set and offline capabilities, Task Flow offers a reliable personal productivity tool tailored for modern mobile users.

**Table of Contents Page No**

**Acknowledgement--------------------------------------------- I**

**Abstract--------------------------------------------------------- II**

**Chapter 1. Introduction**

**Chapter 2. Overall Description**

**Chapter 3. Specific Requirements**

**Chapter 4. Non Functional Requirements**

**Chapter 5. Testing**

**Conclution------------------------------------------------------ III**

**Introduction**

**1. Introduction**

**1.1 Purpose:**

The primary objective of the **Task Flow** project is to develop a mobile application that provides a simple yet effective platform for users to manage and track their daily tasks. Unlike overly complex productivity tools, Task Flow is designed with a strong emphasis on **ease of use**, **accessibility**, and **offline support**, catering to users of all age groups, including those with minimal technical experience.

This application enables users to create tasks with titles, descriptions, deadlines, and statuses. A standout feature of the app is its ability to **automatically update a task's status to “Completed”** once the deadline has passed, helping users stay on track effortlessly. In addition, all task data is stored locally using **Room Database**, allowing users to manage their tasks without needing an active internet connection.

The goal is to provide a **responsive, secure, and offline-capable** task management solution that streamlines productivity and caters to real-world usage scenarios.

**1.2 Scope:**

The **Task Flow** Android application is focused on improving productivity through simple task creation and management features. The app is ideal for personal use, enabling users to stay organized and informed about their to-dos in real-time. The following features are covered under the scope:

**Task Management:**

* **Create Task:** Users can add new tasks with detailed input fields such as title, description, deadline, and initial status.
* **View Tasks:** Users can view all created tasks in an organized list format.

**Task Status Automation:**

* The system automatically updates the task’s status to “Completed” once its deadline has passed, helping users track overdue items without manual updates.

**Task Filtering:**

* Tasks can be filtered based on their current status: **Pending** or **Completed**, allowing users to focus on relevant tasks.

**Offline Functionality:**

* Using **Room Database**, task data is stored locally, ensuring full functionality even without internet access.

**Authentication:**

* Basic **user authentication** is provided, allowing users to securely sign up and log in to their personalized task workspace.

**User Interface:**

* The application features a **clean, intuitive, and responsive UI**, making it accessible to both tech-savvy users and beginners.

**1.3 Definitions, Acronyms, and Abbreviations:**

**Definitions:**

* **Task Flow:** An Android application for managing and tracking daily tasks.
* **Task:** A unit of work that includes a title, description, deadline, and status (Pending or Completed).
* **Room Database:** A persistence library in Android that provides an abstraction layer over SQLite to allow fluent database access.
* **Deadline:** The time by which a task is expected to be completed.
* **Status:** The current state of a task (Pending or Completed).

**Acronyms and Abbreviations:**

* **UI:** User Interface
* **SDK:** Software Development Kit
* **DB:** Database
* **API:** Application Programming Interface

**1.4 References:**

1. Android Developers Documentation – <https://developer.android.com>
2. Room Persistence Library – <https://developer.android.com/jetpack/androidx/releases/room>

**1.5 Overview:**

As outlined in the purpose, this app is aimed at **individual users** seeking a straightforward and effective task tracking tool. The app is designed to work well for:

* **Busy Professionals and Students:** Who require task reminders and deadline tracking.
* **Older Adults and Non-Tech-Savvy Users:** Thanks to the simple and clean interface, the app is easy to navigate.
* **Offline Users:** Who want full functionality without depending on internet availability.

**2. Overall Description**

The **Task Flow** app aims to deliver a minimal, user-friendly, and offline-capable task management solution for Android users. Designed with simplicity and practicality in mind, the app helps users organize their daily tasks while ensuring accessibility even without internet connectivity. The application follows a modular and scalable architecture, allowing for future feature additions and maintenance.

It prioritizes clean UI design, data persistence through **Room Database**, and automatic task status tracking to provide an efficient and seamless user experience for all age groups—including non-technical and older users.

**2.1 Product Perspective**

Task Flow is a **standalone mobile application** built for Android platforms using Java. It focuses on simplifying task tracking and management by storing all data locally, enabling **offline-first functionality**. By integrating Room Database, the app avoids dependence on external servers, making it lightweight and robust.

**Functional Requirements:**

* Add, edit, view, and delete tasks.
* Auto-update task status to *Completed* after the deadline passes.
* Filter tasks based on status (Pending/Completed).
* Secure login and registration system.

**User Roles and Permissions:**

* **General Users**: Can create and manage their own tasks after logging in.
* **Authentication**: Ensures only authorized users can access their task data.

**Security and Privacy:**

* Uses secure local storage through Room.
* Authentication ensures tasks are tied to user identity.
* Sensitive data such as credentials is handled securely (e.g., hashed passwords).

**Scalability and Performance:**

* Optimized for handling hundreds of local tasks efficiently.
* Architecture supports future integration with cloud backup or sync features.

**Integration with External Systems:**

* Currently standalone and offline; can later be integrated with cloud storage (e.g., Firebase) for backup and sync.

**Usability and Accessibility:**

* Simple, responsive, and intuitive UI supports users of all skill levels.
* Designed for smooth navigation and minimum user input.

**Compliance with Industry Standards:**

* Adheres to Android development best practices, including secure data handling and efficient UI/UX design patterns.

**Flexibility and Customization:**

* Supports enhancements like tagging, notifications, or calendar view in future updates.

**Maintenance and Support:**

* Easily maintainable codebase using MVVM architecture.
* Minimal dependencies and clean modular structure.

**Reporting and Analytics:**

* Optional future upgrade to provide basic productivity stats (e.g., tasks completed per week).

**2.2 Product Functions**

* **User Registration/Login:** Allows users to securely register and log in to manage their personal task list.
* **Create Task:** Users can add new tasks with title, description, deadline, and initial status.
* **View Tasks:** Displays all tasks in a scrollable, filtered list.
* **Edit/Delete Task:** Modify or remove existing tasks.
* **Auto Status Update:** System sets tasks as *Completed* when the deadline passes.
* **Filter Tasks:** Filter tasks by status (Pending/Completed).
* **Offline Data Storage:** Task data is saved locally using Room, ensuring data availability without internet.

**2.3 User Characteristics**

**User Module:**

* **User Login/Registration** – Secure access to personal task data.
* **Create Tasks** – Add tasks with titles, descriptions, deadlines.
* **View Task List** – Display a list of tasks with sorting/filtering.
* **Auto Update Status** – Background service updates task status after deadline.
* **Edit/Delete Task** – Modify or remove tasks.
* **Filter Tasks** – Toggle between Pending and Completed views.
* **Offline Access** – Full functionality without internet.

**2.4 Constraints**

* **Local Storage Only** – No cloud sync or backup in current version.
* **Single Device Access** – Data is not shareable across devices unless cloud integration is implemented.
* **Authentication Simplicity** – Basic auth may not use advanced encryption or MFA.
* **Android-Specific** – Not available on iOS or web.
* **Limited Scalability** – Meant for personal task management, not enterprise-scale.
* **Battery Usage** – Auto status update logic must be optimized to avoid battery drain.

**2.5 Assumptions and Dependencies**

**Assumptions:**

* Users have Android devices running version 5.0 (Lollipop) or later.
* Users will interact with the app daily or weekly to manage tasks.
* Room Database is sufficient for storing local task data.
* Basic email/password authentication is sufficient for user identification.

**Dependencies:**

* **Android SDK** – Required for building and running the application.
* **Room Database** – For storing and querying task data.
* **Java 8 or higher** – For writing app logic and UI components.
* **Android Jetpack Libraries** – For architecture (ViewModel, LiveData, etc.).
* **UI Libraries** – Optional third-party libraries for enhanced UI/UX.

**3. Specific Requirements**

**3.1 External Interfaces**

**User Interface (UI)**

* The application will feature a modern, intuitive Android UI built with native **Java XML layouts**.
* Follows Material Design guidelines for consistency and accessibility.
* Responsive layout for multiple screen sizes (phones and tablets).
* Light and dark theme support for enhanced usability.

**Hardware Interfaces**

* Runs on Android smartphones or tablets (minimum Android 5.0 – Lollipop).
* No special hardware dependencies—standard device sensors and storage are sufficient.
* Supports both touch-based navigation and keyboard input.

**Software Interfaces**

* **Room Database (Jetpack)** is used for local data persistence.
* Follows **MVVM architecture** using ViewModel and LiveData components.
* Utilizes **Java 8**, Android SDK, and **Jetpack Libraries** for core functionality.
* **AlarmManager/WorkManager** may be used for background updates or status changes.

**Communication Interfaces**

* All data is stored and processed locally—no network required.
* Future versions may optionally integrate with cloud APIs (e.g., Firebase).
* Secure storage is handled via Android’s internal storage and Room’s encryption capabilities (if extended).

**3.2 Functions**

**Account Management**

* Users can register and log in to access their personal task database.
* Authentication system includes:
  + Username/email and password validation.
  + Secure password storage (hashed).

**Task Management**

* Users can:
  + Create, view, update, and delete tasks.
  + Set a deadline for each task.
  + Filter tasks by *Pending* or *Completed* status.

**Auto Status Update**

* The system checks the current time and updates the task status to *Completed* if the deadline has passed.

**Data Storage & Sync**

* Tasks are stored in a local **Room Database**.
* Optional future feature: Sync tasks across devices using Firebase.

**Security & Authentication**

* Local-only authentication with encrypted Room storage.
* Credentials stored using best practices for Android (e.g., SharedPreferences + encryption or EncryptedSharedPreferences).

**Notifications**

* Optional feature to notify users about upcoming or overdue tasks.

**3.3 Performance Requirements**

* **App Launch Time**: Under 2 seconds on supported Android devices.
* **Task Query Time**: List of up to 500 tasks should load in under 1 second.
* **Background Status Update**: Auto status update should execute within 3 seconds after checking deadlines.

**3.4 Logical Database Requirements**

**Database Type**

* **Room (SQLite)** is used for persistent local storage.
* Entities include User, Task, and possibly Category or Tag for future upgrades.

**Data Integrity**

* Ensures that task deadlines and user relationships are consistent.
* All CRUD operations are handled via DAOs with LiveData observers.

**Backup & Recovery**

* Local backup via Android's auto-backup service.
* Optional export/import as .json or .csv for manual backups.

**3.5 Design Constraints**

**Compliance**

* Designed for personal use with local storage; however:
  + Follows Android secure coding practices.
  + Ensures sensitive information is stored securely.

**Scalability**

* Modular design allows for:
  + Cloud integration (Firebase) in future.
  + Adding new modules (e.g., calendar sync, notifications).

**Cross-Platform Compatibility**

* Runs only on Android OS (version 5.0+).
* Not currently compatible with iOS or desktop.

**3.6 Software System Quality Attributes**

**Security**

* User credentials are securely stored.
* Future-ready for biometric login or two-factor authentication.

**Usability**

* Clean and simple UI using Material Design principles.
* Designed for all user types, including non-technical users.

**Reliability**

* Room ensures ACID properties for local transactions.
* Minimal crash risk due to simple, local architecture.

**Maintainability**

* MVVM architecture ensures clean separation of concerns.
* Easy to extend and debug due to modular structure.

**Portability**

* Compatible with all Android devices meeting minimum OS requirements.
* APK file can be easily distributed via Google Play or sideloaded.

**Non-Functional Requirements**

**4.1 Performance Requirements**

* **API Responsiveness**: Ensure that all backend APIs respond within 3 seconds under normal load conditions.
* **Concurrent Users**: Support at least 1,000 concurrent users without significant performance degradation.
* **Report Generation**: Generate financial reports and account statements for up to a year within 5 seconds.
* **Transaction Throughput**: Handle a minimum of 500 transactions per minute during peak usage hours.​

**4.2 Security Requirements**

* **Authentication**: Implement JWT-based authentication with optional OAuth2 login for enhanced security.
* **Role-Based Access Control (RBAC)**: Utilize Spring Security to enforce RBAC across Admin, Customer, and Employee roles.
* **Data Encryption**: Ensure all data exchanges occur over HTTPS with SSL/TLS encryption.
* **Password Management**: Hash all sensitive data, such as passwords, using bcrypt.
* **Compliance**: Adhere to GDPR, PCI DSS, and RBI regulations.
* **Monitoring**: Integrate real-time monitoring and alerts for security breaches or anomalies using Spring Boot Actuator.​

**4.3 Usability Requirements**

* **Responsive Design**: Develop a frontend using React.js that provides a seamless experience across desktops, tablets, and smartphones.
* **Accessibility Standards**: Ensure the UI complies with WCAG 2.1 accessibility standards.
* **User Assistance**: Offer user-friendly onboarding screens and a comprehensive help section to assist new users.​

**4.4 Reliability Requirements**

* **System Uptime**: Guarantee 99.99% uptime with automated failover mechanisms in case of server failures.
* **Data Backup**: Implement automated daily backups and real-time replication to prevent data loss.
* **Disaster Recovery**: Establish a disaster recovery plan with a Recovery Time Objective (RTO) of less than 30 minutes and a Recovery Point Objective (RPO) of less than 15 minutes.​

**4.5 Maintainability Requirements**

* **Architectural Design**: Adopt a modular and layered architecture in both frontend (React.js) and backend (Spring Boot) to enhance maintainability.
* **Code Simplification**: Use Lombok to reduce boilerplate code in Java development.
* **Logging and Exception Handling**: Implement structured logging and proper exception handling mechanisms to simplify debugging and monitoring.​

**4.6 Portability Requirements**

* **Deployment Flexibility**: Ensure the application is easily deployable across cloud platforms such as Railway.app, Render, and AWS.
* **Cross-Platform Compatibility**: Guarantee seamless operation across Windows, macOS, Linux, Android, and iOS devices.
* **Browser Support**: Support all major modern browsers, including Chrome, Firefox, Safari, and Edge.​

**4.7 Scalability Requirements**

* **System Architecture**: Design the system architecture to support horizontal scaling through load balancers and distributed cloud deployments.
* **Database Configuration**: Configure MongoDB to allow sharding and replication, efficiently managing increasing data loads.​

**4.8 Audit and Compliance Requirements**

* **Audit Logging**: Maintain detailed audit logs for critical operations such as fund transfers, loan approvals, and user logins.
* **Data Retention**: Retain transaction and user data according to legal and compliance requirements, for a minimum duration of 7 years.

**Testing Description**

The *Task Flow* Android app was thoroughly tested to ensure it met all functional and performance requirements. Task creation, editing, and deletion features were tested manually and validated against expected outcomes. The app was verified to correctly auto-update task statuses to “Completed” after the specified deadline passed.

Authentication workflows, including user login and registration, were tested using both valid and invalid credentials to ensure secure access and proper error handling. The task filtering functionality for viewing pending and completed tasks was verified through multiple test runs with varied task data.

Responsiveness and layout behavior were evaluated across various Android devices and screen sizes, ensuring consistent UI performance. The app was tested in both online and offline modes, and task data remained accessible without an internet connection.

Performance under load was examined by adding a large number of tasks, and the app maintained smooth scrolling and responsiveness. Additionally, compatibility tests were run on multiple Android versions (API 26 to 33) to confirm consistent functionality and UI rendering.

All core features performed as expected without crashes, lags, or data inconsistencies.

**Conclusion**

The **Task Flow** Android app successfully provides an intuitive and efficient solution for task management, enabling users to easily create, track, and manage their tasks. Developed using **Android Studio**, **Java**, and **Room Database** for local storage, the app ensures a seamless user experience with offline capabilities and automatic task status updates based on deadlines.

With features such as **user authentication**, **task management**, **automatic status updates**, and **offline support**, Task Flow offers a robust platform for individuals to stay organized and on top of their tasks, regardless of internet connectivity. The **task filtering** option enhances usability by allowing users to quickly view tasks based on their current status, while the **responsive UI** ensures smooth navigation across different devices.

By utilizing **Room Database** for local storage, Task Flow guarantees data persistence even without an internet connection, offering users a reliable solution for task tracking on the go. The app is fully functional with **Android SDK** support and is built with **Java 8**, ensuring compatibility with modern Android devices.

In conclusion, Task Flow is a comprehensive, user-friendly, and offline-capable task management app that meets the needs of individuals seeking to improve their productivity and task organization. Its features and functionality make it a valuable tool for managing tasks efficiently, both online and offline.