Project Proposal: **Daily Outfit Recommendation App**

|  |  |
| --- | --- |
| **1.** | **Introduction** |
| **2.** | **Project Overview** |
| **3.** | **Problem Statement** |
| **4.** | **Objectives** |
| **5.** | **Domain Analysis** |
| **6.** | **Requirements** |
| **7.** | **Inception** |
| **8.** | **Use cases** |
| **9.** | **UML Diagram** |
| **10.** | **Tools & Technology** |
| **11.** | **Timeline** |
| **12.** | **Inspection Plan** |
| **13.** | **Prototype** |
| **14.** | **Ghantt Chart** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**1. Introduction**

For many people, selecting an outfit each day can be a time-consuming task, often leading to “decision fatigue” and frustration. With busy lifestyles and an increasing interest in personal style, there is a growing demand for tools that simplify daily fashion choices. This project proposes the development of a Daily Outfit Recommendation App to help users decide what to wear based on their wardrobe, weather conditions, calendar events, and personal style preferences.

The app will allow users to log their clothing items, generate daily outfit suggestions, and even receive advice based on fashion trends. By integrating weather and calendar APIs, the app will ensure that users are appropriately dressed for any occasion and weather condition. or many people, selecting an outfit each day can be a time-consuming task, often leading to “decision fatigue” and frustration. With busy lifestyles and an increasing interest in personal style, there is a growing demand for tools that simplify daily fashion choices. This project proposes the development of a Daily Outfit Recommendation App to help users decide what to wear based on their wardrobe, weather conditions, calendar events, and personal style preferences.

The app will allow users to log their clothing items, generate daily outfit suggestions, and even receive advice based on fashion trends. By integrating weather and calendar APIs, the app will ensure that users are appropriately dressed for any occasion and weather condition.

**2. Project Overview**The app allows users to:

* Log their wardrobe items.
* Get daily outfit suggestions based on weather and calendar events.
* Access fashion tips and trends for personalized style improvement.

By utilizing APIs for real-time weather and event planning, the app ensures users are always dressed appropriately for any occasion.

**3. Problem Statement**

People often struggle to decide on daily outfits, leading to wasted time and stress. Additionally, factors such as weather changes and specific events (work, casual outings, etc.) require consideration, which can further complicate the decision process. There is currently no comprehensive solution that offers personalized, practical outfit recommendations tailored to each user’s wardrobe and lifestyle.

**3. Objectives**

• **Develop a mobile app** that allows users to input their wardrobe items and style preferences.

• **Enable weather-based** outfit recommendations by integrating weather APIs.

• **Provide calendar-based** suggestions for special events or workdays.

• **Offer personalized styling tips** based on fashion trends and user data.

• Ensure **user data security** with encryption.

**4. Goals**

* **Reduce Decision Fatigue:** Simplify the process of outfit selection with automated recommendations.
* **Increase Confidence:** Enable users to feel prepared and stylish for events and daily routines.
* **Customization:** Personalize recommendations to fit users' individual styles, occasions, and weather conditions.
* **Seamless Integration:** Integrate external APIs (Weather and Calendar) into the app for real-time data use.
* **User Retention:** Build a user-friendly, visually appealing app to encourage daily usage.

**5. Domain Analysis**

1. **Customer**The app targets individuals who seek efficient and personalized outfit recommendations. Key characteristics include:
   * Users aged 18-45 who experience decision fatigue or lack inspiration in daily outfit selection.
   * Tech-savvy individuals who appreciate fashion and want to stay stylish while considering weather and event-specific needs.
   * Users in urban and suburban areas with access to smartphones and a varied wardrobe.
2. **Stakeholders**

**Primary Stakeholders:**

* + - **Users:** End-users of the app, including busy professionals and fashion enthusiasts.
    - **Development Team:** Designers, engineers, and QA testers responsible for creating and maintaining the app.

**Secondary Stakeholders:**

* + - **API Providers:** Weather and calendar service providers offering real-time data for outfit suggestions.
    - **Fashion Brands:** Potential partners for integrating fashion trends and personalized style recommendations.
    - **Sponsors or Investors:** Organizations or individuals funding the development and expansion of the app.

**6. Requirements**The requirements for the Daily Outfit Recommendation App are categorized as follows:

**Functional Requirements:**

* Ability for users to create an account and securely log in.
* Interface for users to input and organize their wardrobe items, including categories like tops, bottoms, footwear, and accessories.
* Integration with weather APIs to provide recommendations based on real-time weather conditions.
* Integration with calendar APIs to tailor outfits for specific events or days.
* Algorithm to suggest outfits based on user preferences, wardrobe inventory, weather, and event type.
* Notifications or reminders for users to check their daily outfit suggestions.

**Non-Functional Requirements:**

* The app should have a user-friendly and visually appealing interface.
* Ensure quick response times for data retrieval and outfit generation.
* Data encryption to secure user wardrobe and personal information.
* Compatibility with both iOS and Android platforms.
* Scalable architecture to handle a growing user base and data load.

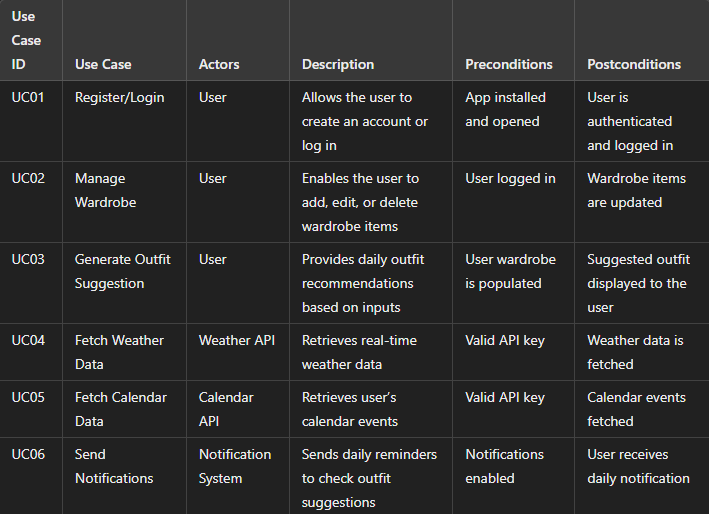
**7. Inception**

The idea stems from the growing need for personalized tools to simplify daily decisions, reduce stress, and improve individual style confidence. The project will begin with stakeholder interviews and surveys to finalize functional requirements and design concepts.

**8. Use Cases**

* **Register/Login:** Users can create an account to store their wardrobe securely.
* **Manage Wardrobe:** Add, organize, and delete clothing items.
* **Generate Outfit Suggestions:** Automated suggestions based on weather, events, and preferences.
* **Fetch Weather Data**: Integration with Weather API to recommend weather-appropriate clothing.
* **Fetch Calendar Data**: Integration with Calendar API to plan outfits for specific events.
* **Send Notifications:** Timely reminders for users to check daily outfit suggestions.
* **Provide Styling Tips:** Advanced suggestions incorporating fashion trends.

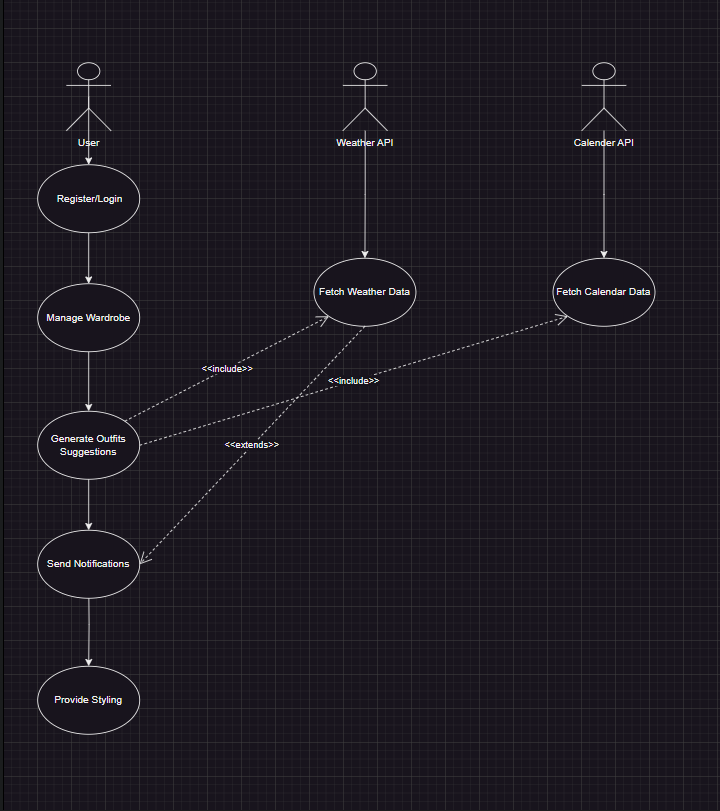
**9. Use Case Table**

****

**10. Test Case Table**

**10. UML Diagram**

The following UML Use Case Diagram represents the key components of the app:



**11. Requirements Analysis:**The requirements for the Daily Outfit Recommendation App are categorized as follows:

**Functional Requirements:**

* Users can create an account and securely log in.
* Input and organize wardrobe items, categorized as tops, bottoms, footwear, and accessories.
* Integration with weather APIs to provide real-time weather-based outfit recommendations.
* Integration with calendar APIs to suggest outfits for specific events or days.
* An algorithm that generates outfit suggestions based on user preferences, wardrobe inventory, weather, and event type.
* Notifications or reminders for daily outfit suggestions.

**Non-Functional Requirements:**

* User-friendly and visually appealing interface.
* Quick response times for data retrieval and outfit generation.
* Data encryption to secure user wardrobe and personal information.
* Compatibility with iOS and Android platforms.
* Scalable architecture to accommodate a growing user base and increasing data load.

**2. List of Actors**

* **Primary Actors:**
  + **Users:** Individuals who manage their wardrobe and receive outfit recommendations.
* **Secondary Actors:**
  + **Weather API:** Provides real-time weather data.
  + **Calendar API:** Supplies event-related data for outfit planning

**12. Extended Use Cases**

* **Use Case: Generate Outfit Suggestions**
  + **Actors:** User, Weather API, Calendar API
  + **Precondition:** User must be logged in and have wardrobe data uploaded.
  + **Basic Flow:**
    1. Retrieve real-time weather data.
    2. Fetch event details from the calendar.
    3. Generate personalized outfit suggestions using algorithms.
    4. Notify users with the daily suggestion.

**10. Tools & Technologies**

• **Frontend:** Flutter / React Native

• **Backend:** Firebase / Node.js

• **Database:** Firestore / MySQL

• **API Integrations:** Weather API (e.g., OpenWeatherMap) and Google Calendar API

• **Security:** AES Encryption

**11. Timeline**

|  |  |
| --- | --- |
| **Task** | **Duration** |
| Requirement Analysis | 2 weeks |
| Design and prototyping | 3 weeks |
| Backend and frontend Development | 7 weeks |
| Testing and Quality | 3 weeks |
| Final Deployment and User Training | 2 weeks |

**12.Inspection Plan**

To maintain quality throughout the development of the Daily Outfit Recommendation App, an inspection plan will be implemented at various stages. The plan includes the following inspections:

• **Requirement Analysis Inspection:** Verify that all functional and non-functional requirements align with project objectives. This will involve meetings with stakeholders and potential users to validate and finalize requirements.

• **Design Review:** Conduct thorough inspections of the UI/UX design to ensure the app is visually appealing, user-friendly, and aligns with modern design standards. Accessibility checks will be included to ensure usability for a wide audience.

• **Code Review:** Perform regular code inspections during development to maintain clean and efficient code. Peer reviews will be conducted to identify potential bugs early and enforce coding standards.

• **Unit Testing:** Conduct unit testing for individual features, such as outfit suggestion generation, calendar integration, and weather updates, to ensure each function operates as expected.

• **Integration Testing:** Test the interactions between different app modules (e.g., wardrobe database, weather API, and calendar API) to ensure seamless functionality and data flow.

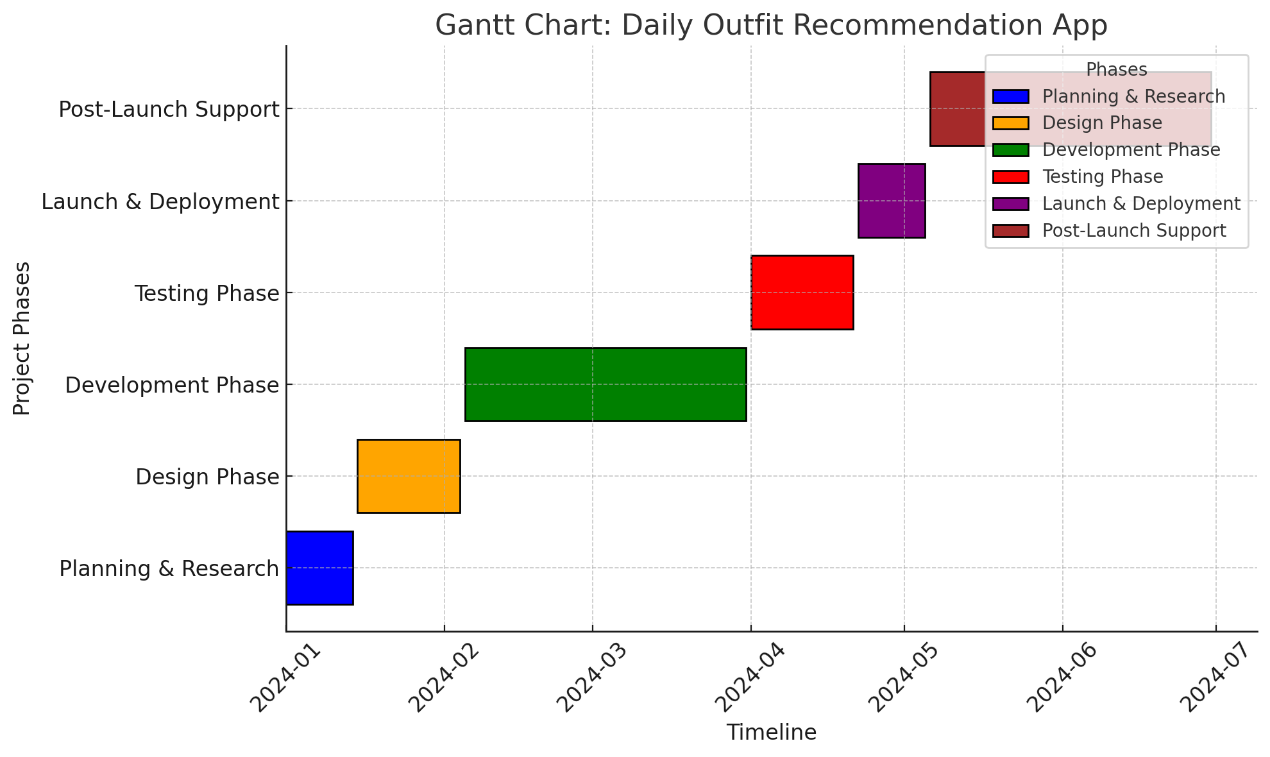
• **User Acceptance Testing (UAT):** Run a beta test with a select group of users to gather feedback on functionality, usability, and overall experience. Feedback will be used to make final adjustments before launch.

• **Final Inspection before Deployment:** Conduct a final inspection to confirm that all critical issues have been resolved. This includes checking app security, data handling, and overall performance to ensure a smooth and secure user experience.

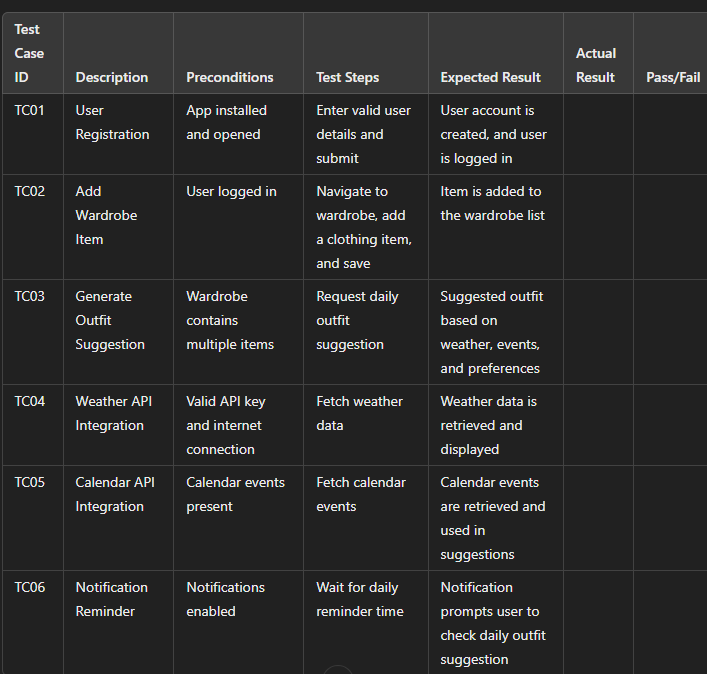
**13. Prototype**



**14. Ghantt Chart**



**15. Test Case Table**

****

**Expected Outcomes**

• Daily outfit suggestions tailored to user preferences, weather, and calendar events.

• Reduced time and stress associated with outfit selection.

• Enhanced user experience with seamless, secure data storage and retrieval.