TaskFlow

REQUIREMENT SPECIFICATIONS
DOCUMENTATION

Pericles Lelos, Krish Mohan

Sam Fickle, Charles Timmons

Table of Contents

5.3 Detailed Use Cases……..

1) Introduction

1.1 Overview

This document lays out the requirements for the TaskFlow. TaskFlow is a web-based application capable of tracking various tasks and ranking them by priority. This program is intended to be simple to use with an intuitive interface.

1.2 Goals

TaskFlow Goals:

1. Provides easy and reliable authorization and authentication for users
2. Develops an interface that allows users to create new task cards and assign
3. Automatically assigns priority to each task based on factors like due date
4. Develops database to store user login info and user created tasks

1.3 Scope

- The TaskFlow web application will allow users to efficiently manage and prioritize tasks. The main features include user account creation, secure login/authentication, and task creation with automatic priority assignment based on due dates. Tasks can be edited, deleted, and viewed, with data securely stored in a Docker database.
- Out of Scope:
    1. Advanced task management features (e.g., project collaboration, subtasks)
    2. Third-party integrations (e.g., email notifications or external tools)
    3. Calendar page showing due dates

2) Design Constraints

2. 1 Product Environment

TaskFlow will be implemented in a REACT environment with data stored in an external database using Docker.

2.2 User Characteristics

Administrators/Developers - These users will have a base knowledge in navigating React development tools while also having access to the backend database. They will be responsible for maintaining and updating the software to ensure consistent quality within the web-app.

Primary User - These users will create accounts to track various tasks they need to get done.

3) Nonfunctional Requirements

3.1 Operational Requirements

1. The web application will be created using React and styled using Tailwind CSS
2. Data will be stored in a Docker database
3. The system should be available for users anytime
4. The system should be able to support over 10,000 users
5. The application's UI should be easy to grasp and simple to use

3.2 Performance Requirements

1. The program should authenticate users in under a minute
2. Upon logging in, user tasks should by displayed from the backend within 30 seconds
3. Creating a new task does not affect website performance

3.3 Security Requirements

1. TaskFlow Will require users to create an account to access the application
2. Users must be authenticated to be routed to the home page
3. Each user's tasks will not be accessible by any other user

4) Functional Requirements

1. Users will be able to sign up for an account
2. Users will be able to log into an account after signup

3. Users will be able to create tasks with a name, description, due date, etc
4. Users will be able to delete tasks
5. Users will be able to edit task details such as name and date
6. Administrators will be able to view, edit, create, and delete information stored in the database
7. Users will be able to sign out of their account

5) Use Cases

5.1 Actors

- Primary User – Creates an account, manages tasks.
- Administrator – Maintains and updates database content

5.2 Use Case List

1. Sign Up for an Account
2. Log In
3. Create a Task
4. Edit a Task
5. Delete a Task
6. View Tasks
7. Log Out
8. Administrator Database Management

5.3 Detailed Use Cases

Use Case 1: Sign Up for an Account

Actor: Primary User
Trigger: User wants to create an account to begin tracking tasks.
Preconditions: User is not logged in.
Main Flow:

1. User navigates to the sign-up page.

2. User enters required information (e.g., username, password).
   System validates input.
   System creates a new user account.
3. System logs the user in or redirects them to log in.
   Postconditions: A unique user account is stored in the database.
   Exceptions:

- Username already taken
- Invalid or incomplete input

Use Case 2: Log In

Actor: Primary User
Trigger: User attempts to access their task list.
Main Flow:

1. User enters login credentials.
2. System authenticates the user.
3. System routes user to their home page with tasks displayed.
   Exceptions:

- Incorrect password
- Account does not exist

Use Case 3: Create a Task

Actor: Primary User
Trigger: User chooses to add a new task.
Main Flow:

1. User selects “Create Task.”
2. User enters task details (name, description, due date, etc.).
3. System stores the new task in the database.
4. System calculates and assigns the task’s priority.
   Postconditions: Task is saved and appears in the user’s task list.
   Exceptions:

- Invalid data fields
- Database connection error

Use Case 4: Edit a Task

Actor: Primary User

Trigger: User chooses to modify an existing task.

Main Flow:

1. User selects a task to edit.
2. User updates relevant fields.
3. System saves the updated task details.

Postconditions: Task displays updated information.

Exceptions:

- Invalid or missing inputs

Task not found

Use Case 5: Delete a Task

Actor: Primary User

Trigger: User removes a task they no longer need.

Main Flow:

1. User selects “Delete Task.”
2. System requests confirmation.
3. System removes task from the database.

Postconditions: Task no longer appears in the task list.

Exceptions:

- Task not found
- Database error

Use Case 6: View Tasks

Actor: Primary User

Trigger: User navigates to their home/dashboard page.

Main Flow:

1. System retrieves all tasks associated with the user.

2. System displays tasks ordered by priority.
   Postconditions: User can see all tasks and their details.
   Exceptions:

- No tasks stored

- Database retrieval failure

Use Case 7: Log Out

Actor: Primary User
Trigger: User selects the log-out option.
Main Flow:

1. User clicks “Log Out.”
2. System ends the active session.
3. System returns user to login screen.
   Postconditions: User session is closed.

Use Case 8: Administrator Database Management

Actor: Administrator
Trigger: Admin performs system maintenance.
Main Flow:

1. Admin accesses backend tools or database interface.
2. Admin creates, edits, or deletes user/task records as needed.
   Postconditions: Database reflects updates made by admin.
   Exceptions:

- Unauthorized access attempt
- Invalid data modifications