Project Name:

Project Management Tool

SRS Document

Table of Contents

1. Introduction

Purpose:

The purpose of this document is to define the requirements for developing a Project Management Tool that facilitates task creation, assignment, progress tracking, and collaboration. The intended audience includes project managers, team leads, and software developers.

Scope:

The Project Management Tool provides capabilities to:

- Create and store tasks with attributes (title, description, priority, status).
- Assign tasks dynamically to team members.
- Undo task actions for error recovery.
- Display pending tasks and assigned team tasks.
- Enable collaboration and task tracking.

The system will not include advanced analytics or machine learning features.

Definitions, Acronyms, and Abbreviations:

- **Task**: An individual unit of work with specific attributes.
- **Undo Stack**: A data structure to reverse task actions.
- **Task Queue**: A queue storing pending tasks for assignment or processing.

Overview:

This document outlines the functional and non-functional requirements for the Project Management Tool. It includes system features, interface requirements, and design constraints.

2. Overall Description

Product Perspective:

The tool is a standalone application but can be extended to integrate with external project management platforms or APIs.

- **Relationship to Other Systems**:
  The system can integrate with existing project tracking tools or notification systems for real-time updates.
- **System Diagrams**:
  - Task Queue: Handles task creation and prioritization.
  - Undo Stack: Manages undo actions for task updates.
  - Team Task List: Maintains dynamically assigned tasks.

Product Features:

1. Task creation and storage.
2. Task assignment to team members.
3. Undo functionality for error recovery.

4. Display of pending and assigned tasks.

User Characteristics:

- Target Audience: Team members and project managers.
- Technical Knowledge: Basic familiarity with terminal/console-based tools.

Constraints:

- The tool is CLI-based and lacks a graphical user interface.
- Limited to web-based or local execution only.

Assumptions and Dependencies:

- Assumes users have access to a C++ runtime environment.
- Assumes basic understanding of task attributes (priority, status).

3. System Features

Feature 1: Task Creation

- **Description**: Allows users to create tasks with attributes (title, description, priority, status).
- **Priority**: High
- **Stimulus/Response Sequences**:
    - Input: Task details (title, description, priority, status).
    - Processing: Adds task to the queue and undo stack.
    - Output: Displays task creation confirmation.
- **Functional Requirements**:
    - Generate unique task IDs.
    - Store task attributes in a queue.

Feature 2: Task Assignment

- **Description**: Assign tasks to team members dynamically.
- **Priority**: High
- **Stimulus/Response Sequences**:
    - Input: Username for task assignment.
    - Processing: Moves task from queue to team task list.
    - Output: Displays assigned task details.
- **Functional Requirements**:
    - Retrieve task from queue.
    - Dynamically add task to a user.

Feature 3: Undo Action

- **Description**: Undo the last task action (e.g., assignment or status update).
- **Priority**: Medium
- **Stimulus/Response Sequences**:

    - Input: Undo command.
    - Processing: Retrieves last task from stack and pushes it back into the queue.
    - Output: Displays undo confirmation.

- **Functional Requirements**:
    - Maintain an undo stack for all task actions.

4. External Interface Requirements

User Interfaces:

- Console-based interface with text-based menus for task creation, assignment, and tracking.

Hardware Interfaces:

- Standard input/output devices (keyboard, monitor).

Software Interfaces:

- Local execution in C++ runtime environment.

Communication Interfaces:

- None required; the tool operates locally.

5. Functional Requirements

1. Create tasks with attributes: title, description, priority, and status.
2. Assign tasks to specific users dynamically.
3. Display pending and assigned tasks.
4. Undo the last task action.

6. Non-Functional Requirements

Performance Requirements:

- The system should process task creation, assignment, and undo actions within 1 second.

Security Requirements:

- No external data storage or transmission; all data resides locally.

Usability Requirements:

- Simple menu-based navigation for all functions.

Reliability Requirements:

- Maintain task integrity during undo operations.

Maintainability and Supportability:

- Modular design allows for easy updates or addition of features.

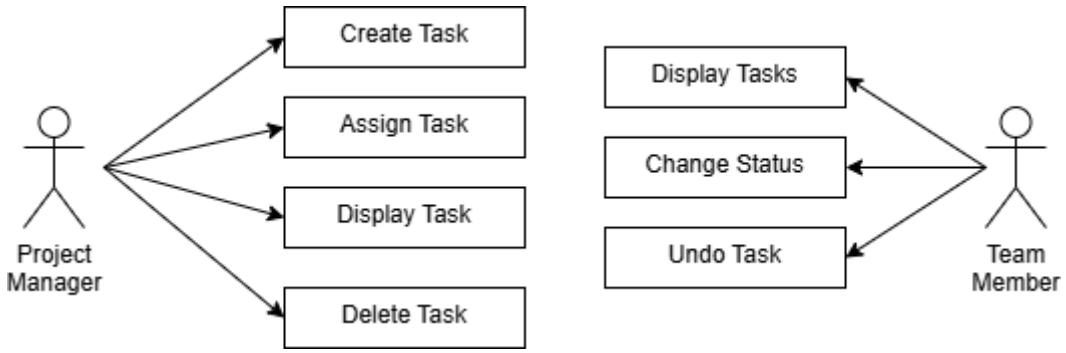
7. Design Constraints

- Must use standard C++ libraries (no external dependencies).
- Task storage limited to runtime memory (no database).

8. System Models

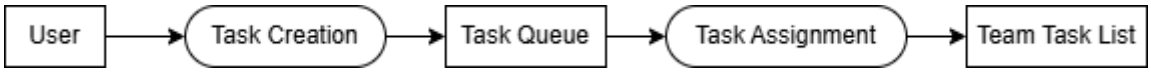
Use Case Diagram:

- Actors: User (Project Manager, Team Member).
- Actions: Create Task, Assign Task, Undo Action, Display Tasks.



Data Flow Diagram:

- Task creation → Task Queue → Task Assignment → Team Task List.



9. Other Requirements

Database Requirements:

- Not applicable (in-memory storage only).

Installation Requirements:

- C++ compiler and runtime environment required.

Legal or Regulatory Requirements:

- None.

10. Appendices

- Example Task:
  ID: 1, Title: "Design UI", Description: "Create wireframes for the landing page", Priority: High, Status: Pending.