Software Requirements Specification
for
Task Tracker
(.NET CORE)

Prepared by

PIXYBYTE

TPBMZY#1 Task Tracker

June 8th

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to specify the requirements for the development of a Task Tracker application. The application aims to help individuals manage their tasks and projects effectively.

1.2 Scope

The Task Tracker application will provide features to create, update, and track tasks. It will allow users to set deadlines, assign priorities, and monitor the progress of tasks. The application will be developed for desktop platforms.

1.3 Definitions, Acronyms, and Abbreviations

- SRS: Software Requirements Specification
- UI: User Interface
- CLI: Command-Line Interface

1.4 References

Any references related to the project can be listed here.

2 Overall Description

2.1 Product Perspective

The Task Tracker application will be a standalone software that operates on desktop platforms through a CLI. It will have a user-friendly UI using CLI and provide essential task management features.

2.2 Operating Environment

The Task Tracker application will be developed to run on Windows, macOS, and Linux operating systems. It will require a compatible desktop environment.

2.3 Design and Implementation Constraints

The application will be developed using C# programming language and the .NET Core Framework.

3 Functional Requirements

3.1 Menus

- Initially, the user should be met with a UI screen displaying the list of tasks by default.
- There should be options to 'Add Task', 'Edit Task', or 'Exit'.
- When adding a task, a new UI menu should appear with the required fields or inputs for a new task, indicating what is optional and what is not.
- When editing a task, a new UI menu should appear with all relevant information of the current task with inputs for which information to edit.
- OPTIONAL: For adding or editing, there should be a 'Cancel' option along with the 'Confirm' option.

3.2 Task Creation

- The user should be able to create a new task by providing a mandatory title.
- The user should be able to create a new task by additionally and optionally providing the description, duration, deadline, priority, and tags of the task.
- The system should generate a unique identifier for each task.
- Each newly created task should be open by default.
- OPTIONAL: The user should be able to duplicate an already existing task.

3.3 Task Update

- The user should be able to update the details of a task, including its title, description, duration, deadline, priority, and tags. This included updating both completed and open tasks.
- The user should have the option to mark a task as complete or reopen a completed task.
- The user should have the option to delete a task.
- OPTIONAL: The user should have the option to designate a task as 'Not Pursuing', indicating that it is not complete, but not should not remain open.

3.4 Task Tracking

- The user should be able to view a list of all tasks.
- OPTIONAL: The user should be able to filter tasks based on deadline or priority and sort tasks based on criteria such as tags.

3.5 OPTIONAL: Advanced Features

3.5.1 Notifications

- The system should provide notifications or reminders for approaching task deadlines. For a CLI application, the application can just execute a 'BEEP' sound when a task is due.

3.5.2 Linked Duplicates

- When duplicating a task, the user should have the option to link this duplicate task to the original task. This will mean that any edits made to the original task will reflect on all linked duplicates and vice versa.

3.5.3 Data

- Store user data locally.

4 Non-Functional Requirements

4.1 Usability

- The application should have an intuitive and user-friendly UI, allowing users to navigate and interact with tasks effortlessly.
- The UI should provide clear indications of task status, priority, and progress, optionally, making use of colors.

4.2 Performance

- The application should have fast response times when loading tasks and updating information
- The system should be able to handle a large number of tasks without significant performance degradation.

5 Appendices

5.1 Appendix A