David Westgate

The Task Management Assistant

Requirements

Table of Contents

1 Introduction....3
1.1 Purpose and Scope....3
1.2 Target Audience....3
1.3 Terms and Definitions....3
2 Product Overview....4
2.1 Users and Stakeholders....4
2.1.1 Productive individuals in the general population....4
2.1.2 David Westgate....4
2.2 Use cases....4
2.2.1 Final Test....4
2.2.2 Assessment....4
2.2.3 Future use by myself....4
3 Functional Requirements....5
3.1 Three column-like containers, to contain task items. *5*
3.1.1 Aspect 1: Add and removing tasks. 5
3.1.2 Aspect 2: Task transitions between columns. 5
3.2 Registering users....*5*
3.3 Saving and Loading....*5*
4 Nonfunctional Requirements....6
4.1 Adjusting text and display attributes....*6*
4.1.1 Aspect 1: Adjusting font size....6
4.1.2 Aspect 2: Adjusting typeface....6
4.2 Having a appreciable color scheme....*6*

4.3 Using Icons..*6*

5 Milestones and Deliverables..7

5.1 Creating the multi-user system...*7*

5.1.1 Stage 1: Create a system of registration..7

5.1.2 Stage 2: Create a system of authentication..7

5.2 Loading Saved user data...*7*

1 Introduction

This Document is the first stage in my creating of the Task Management Program. This document identifies crucial preliminary requirements of the project at hand and other background data which will help keep the development of the project focused for its duration.

1.1 Purpose and Scope

The task management program, is intended to help the user manage the status of his or her desired tasks. Every tasks exists in a different state of completion, and this software will help track the status of each task for future reference.

1.2 Target Audience

This program could be useful to any achieving person on a modern system. More specifically, this program would be useful to productive adults or students, or other professional. Any user would have a computer capable of running the modern Java Runtime Environment, such as Windows , OS X, or Linux.

1.3 Terms and Definitions

JRE: Java Run time Environment
JDK: Java Development Kit
JVM: Java Virtual Machine
IDE: Integrated Development Environment
TMA: Task Management Application

2 Product Overview

The Task Management Application aims to bring simplicity, and ease of use back to the concept of software assisted task managing. By graphically displaying three columned lists of “To Do”, “In Progress”, and “Done”, the application will very simply allow the user to create and remove task items, and transition them between the states. The product will allow for multiple users via registering a username and password. A possible limitation, or “feature” by industry standards is that a users data will only be saved on the local machine, but could be transferred by file sharing.

2.1 Users and Stakeholders

Here the intended users are explored and their interests. The existence of users drives the entire project giving it purpose. These users include general productive individuals, and also myself.

2.1.1 Productive individuals in the general population

This program is meant to be utilized by any productive person, who is inclined to already utilize a computer. This could be a professional managing their tasks at work. A homeowner organizing their house duties and projects, or a student managing homework and reading. Specifically, it is known that this program will be utilized by at least one member of the general population, our courses TA.

2.1.2 David Westgate

David Westgate, this documents author, is the sole executor of this project. David will manage the entirety of this projects requirements, design, testing, analysis, implementation and possible maintenance.

2.2 Use cases

The use cases are the scenarios under which the program is expected to be used. They are carefully considered during development, as they help set implementation goals. The use cases include the Final Test, Assessment, and future use by myself.

2.2.1 Final Test

In the final test by the developer, myself, I will run the application. My goal in this test will be to uncover any remaining bugs by attempt to use each feature of the program in a unique manner. Knowing which bugs remain, I will then settle with the fact, as the final test will likely precede the due date of the project by just a few minutes.

2.2.2 Assessment

During the assessment use case, the quality of implementation will be judged by my courses Teaching Assistant or Instructor. For personal reasons, this is the most important use case.

2.2.3 Future use by myself

Concluding the formal graded lifecycle of this project, this program seems very much like something I would use in the future. It would be simplistic, and allow me to avoid using a cloud based service to help manage my tasks. Over a period of time, I could discover bugs from long term use that were never apparent during lifecycle testing and development.

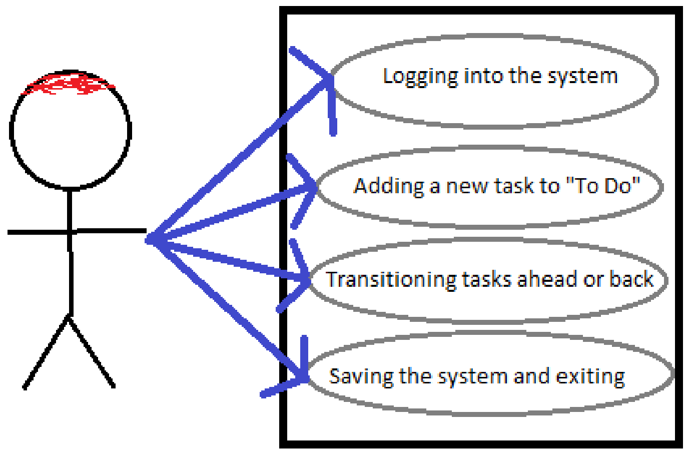
Logging into the system
Adding a new task to "To Do"
Transitioning tasks ahead or back
Saving the system and exiting

Illustration 1: David, using this Application in the future. Circa 2017

3 Functional Requirements

The functional requirements here, are the target implementation goals required to meet the basic needs of the project description. They are the primary focus of development and should all be achieved my submission. For this project, having three column containers to contain tasks, registering users, and saving/loading are the focused functional requirements

3.1 Three column-like containers, to contain task items.

Having three graphical column contains, such as rectangles which are able to contain task items, which are basically other rectangles which themselves contain a short sentence or phrase, is the fundamental basic of this project. The columns themselves will be respectively titled "To Do", "In Progress" and "Done".

3.1.1 Aspect 1: Add and removing tasks.

Introducing task items into the system, as well as removing them from the system is vital. There will be an ability to add task items into the "To Do" column, which will prompt for the users text description of the task. Tasks will need to be deleted, and vanish forever, as the user pleases from any column. Tasks may also be edited within any progress column.

3.1.2 Aspect 2: Task transitions between columns.

The ability to transition tasks, from "To Do" to "In Progress" and from "In Progress" to "Done" is also vital to the purpose of this project. This transition is a very large part of the management portion of the tasks themselves. These two different transitions will need to be bi-directional. Transitioning in reserve may be necessary in all sorts of real life scenarios when setbacks happen.

3.2 Registering users

To allow for multiple accounts on the same system, users shall need to register. Basically, a user will need to provide his or her desired username, and password. If a username is taken, the user shall be prompt to pick a different one. The user must then always remember these credentials to stay on top of their task management.

3.3 Saving and Loading.

When a user logs in each time with their credentials, it is expected they will have the same exact graphical display of their task management system, as the last time they logged off. This implies a well functioning saving and loading implementation. Saving must work well in two situations. When the user hits the “Save” button during normal use, and when the user logs out, they will be prompted to save. Loading will happen automatically when the user logs in.

4 Nonfunctional Requirements

Nonfunctional requirements are important, but are not vital to the general success of the project. It is best to have completed all nonfunctional requirements by project but they are not necessary for a working and testable prototype. This projects nonfunctional requirements are using graphical icons whenever possible, having an appreciable color scheme, and the ability to adjust text attributes.

4.1 Adjusting text and display attributes

Adjusting text attributes is important as not all users read and work best at default text settings. A user will likely prefer a customized font size and typeface to suit his or her needs. The attributes should save and persist in a configuration file, along with the core task information.

4.1.1 Aspect 1: Adjusting font size

This program will provide a way for the user to change the font size, either by increasing or decreasing it. A user might prefer an increased size for easier readability, or smaller size for denser information delivery.

4.1.2 Aspect 2: Adjusting typeface

Having a few different options for a typeface makes a subtle difference with usability. A user may be more pleased with the application in general if his or her desired typeface persists between sessions, and may even be more likely to come back and use the program more.

4.2 Having a appreciable color scheme

Though not important to core functionality, having a good color scheme between borders, backgrounds, state columns, and task boxes makes the program more inviting. This will assist in the application appearing more user-friendly, user attentiveness and less eye strain.

4.3 Using icons

Though the context of this assignment is in an English language classroom and I am expecting English language users, implementing icons is good practice. In my finished application were to ever take on an extended audience, icons make a much better, universal language. Icons could serve for the save, exit, and transition buttons. The use of icons also saves valuable screen space from otherwise lengthy text.

5 Milestones and Deliverables

Milestones in this project are the important points by which certain concrete aspects of the project will be accomplished. Aside from the basics of setting up the project environment and workspace in an IDE, as well as setting up a working repository, the first major milestone will be creating the multi-user support, as all following work would create content that is unique among users. This first major milestone can be broken down into creating a system to allow unique users to register, and also creating a system to authenticate a user that is tied to a saved data file. The second milestone shall be to load the saved user data into working data sets or structures.

5.1 Creating the multi-user system.

Creating the multi-users system primarily involves the use of the text based fields of username and password, and managing them into data files. I will generally need to work with external data files, implement a simple encryption for passwords, to not be in plain text, and consider a good formatting practicing for storing this data. By the end of this stage I will be ready to work on implementing the task management portion of this program which can then be unique to each user.

5.1.1 Stage 1: Create a system of registration

Creating a system to allow for registration comes first. Basically I will encode a simple box, that when clicked, created a popup window. This window will prompt for a username, and twice for the same password. It will reject usernames that are already taken, and reject passwords which do not match, and shall except anything else. It will save all excepted combinations out to file. This will involve a mild about of work in Java.

5.1.2 Stage 2: Create a system of authentication

The second stage is to permit a user to log in, so that all further work is tied to ones own account. By default, when the software is to run, the user will be prompted for a username and password in separate fields. Implementing this functionality will first require comparing the username to the data file for a match, posting a message if not found. If found, then a password match will be checked for, with a unique error if there is a password mismatch. Finally, if the user is authenticated the remainder of their work is tied to their username.

5.2 Loading Saved user data

If a user could not load his or her own data, this application would become very useless upon the first time the user exits. When a user is successfully authenticated the application will look for a data file tied to the current username. The application must then be able to parse this data, and create a data structure for task management each stage. Each data entry item in the file will represent a single task and all of its relevant data and metadata which will be placed into one of the three data structures. Loading may also be required to do nothing, like on the users first time logging in.