Term Project: *Task Manager Application*

Design

Table of Contents

1 Introduction

Introduce this document and the project it describes. You should also summarize the remaining content of the document here.

This document intends to outline the description of the Task Manager application, a Java program I shall create as a course requirement. The general goal of this application is to help unique users keep track of personal life tasks, through a simple and straight forward user interface.

1.1 Purpose and Scope

You should describe this document by giving its purpose, scope.

This design document attempts to serve as a comprehensive outline of the features to be included in the finished application, and is important to maximize productive of time spent implementing. This document shall consider the overall design and process of implementation, first in a wide scope, and then in details of components. After outlining important design considerations, this document will explore a complete system overview, to create a high level understanding of the project. Following this, a system architecture analysis will outline the systems which shall compose the application. Details of this architecture will follow is section 5.

1.2 Target Audience

Describe the target audience for this document.

The potential target audience of this application is very broad. It could include any person or professional who is interested in software assistance in keeping track of task progress. This audience may prefer a more simple software solution than the complicated ones available on the market.

1.3 Terms and Definitions

Define any terms or acronyms you will be using in the remainder of this document.

The Task Manager Application (TMA) is the title of the project.
A major part of this project is designing the graphical user interface (GUI).
This project will be built with the help of an IDE (Integrated Development Environment) as well as a JDK (java development kit).

2 Design Considerations

Describe the purpose of this section and provide an outline of its subsections. Only a few sentences are expected here.

Design considerations are the real and tangible constraints, or development obstacles which I will need to account for. No project exists without certain fundamental requirements and considering here the constraints, dependencies and methodology will make implementation more efficient.

2.1 Constraints and Dependencies

What constrains, either functional or non-functional were you required to adhere to in designing and implementing your system?

As in most software projects, time is the largest constraint and most important to account for. I balance progress of the project between a job and other coursework so it is important not to make implementation goals too ambitious in order to adhere to deadlines. I am required to implement the application in Java, with a Graphical user interface. I must allow for multiple users on a single system.

2.2 Methodology

Describe the software engineering methodology or methodologies you will use, justify your choice(s), and briefly discuss how you implemented them.

The software engineering methodology I am using for the Task Manager Application is best described as incremental. I accordance with the flow of class progression and time constraints, incremental fits best as courses tend to progress linearly. In general I will focus my attention on one part of the software life-cycle at a time from requirements, design, analysis, implementation and testing. There will be some overlap of attention between adjacent stages.

3 System Overview

A complete, high level system description goes here. This section should provide a high level abstraction of the entire project, and prepare the reader for the system architecture discussion. You should provide supporting diagrams and tables where appropriate.

Though I would not consider this system very complicated by relative standards, a complete overview of the entire application shall be a major step in driving the projects focus.

The task manager application as a functional system, will work from two major modules. The first module will be user account management and authentication. Like other applications, users will need to register with a unique name and password, per the system. There are considerations in this module such as simple password encryption, and saving and loading user data. Upon successful login, a user will be transferred to the second module, which shall be the management application.

The management module is the primary goal of the project. It will mainly be divided into 3 aspects, being “to-do”, “in progress” and “completed”. Users will be able to add new tasks as strings, to the “to-do” window. Tasks can be transferred linearly between the 3 stages, and removed once they are “complete” The images captures the intended look and feel of the finished modules

4 System Architecture

A short summary of the system-level architecture should be here, and an introduction to the following subsystems. Only a few sentences are expected here.

Both modules, the user authentication as well as the management module will require specific implementations of unique Java architecture. They will require certain libraries and dependencies to be implemented efficiently, as well as powerfully. JavaFX is a Java framework which intents to replace Swing as a cross platform, GUI component. As such, I will utilize it heavily for all of my projects architecture.

4.1 The Task Managing Module

Provide a detailed, high level description of the first subsystem, component or object of your system. Your choice of terms should reflect the development methodology you chose in the design considerations section.

You should provide a detailed description of the interface. A high level description of the internal architecture is also appropriate here. You may want to outline any important algorithms as well.

Be sure to include supporting diagrams, such as UML diagrams, interaction diagrams and data flow diagrams, wherever appropriate.

The task managing module will exist as an embedded system of frames. The top level container of this system will be a “Stage” in JavaFX. This is essentially the window view. Though not very significant on its own, this stage will house some very important components. Through use of the “stackpane” layout, the stage will contain the primary scene, which is the component to house all other fundamental aspects of this module. The stage will also contain some basic window controllers such as close, maximize and minimize which are implement by default in JavaFX.

4.1.1 The Primary Scene and Layout

Some of the larger subsystems, components or objects may need to be broken down further to complete the high-level picture. The same guidelines for content apply in this section as in its parent section.

4.1.2 The Three Task Status Containers

Don’t forget to remove the surrounding < and > symbols.

4.2 The Account Management Module

Don’t forget to remove the surrounding < and > symbols.

4.2.1 User registration

Don’t forget to remove the surrounding < and > symbols.

4.2.2 User Login/Authentication

Don’t forget to remove the surrounding < and > symbols.

5 Detailed System Design

5.1 <Subsystem, Component or Object 1>

Here, you will provide a highly detailed description of your subsystem, component or object. You will describe your implementation in a fair amount of detail. It is quite common to see pseudo code or even source listed in this section for portions of the object. List the data structures you will use, helper functions that you will need, and any other facet of the implementation that helps the reader understand your implementation. It is unlikely that you will need to introduce any new diagrams in this section.

5.1.1 <Sub-subsystem, sub-component or sub-object 1-1>

Don't forget to remove the surrounding < and > symbols.

5.1.2 <Sub-subsystem, sub-component or sub-object 1-1>

Don't forget to remove the surrounding < and > symbols.

5.2 <Subsystem, Component or Object 2>

Don't forget to remove the surrounding < and > symbols.

5.3 <Subsystem, Component or Object 3>

Don't forget to remove the surrounding < and > symbols.