Phase -1 Assessment

Technical Specification Document

By

Kaviyapavai Natarajan

Created On 21-May-2021

Last updated on 07-June-2021

# Introduction:

## Problem Description:

Company Lockers Pvt. Ltd. hired you as a Full Stack Developer. They aim to digitize their products and chose LockedMe.com as their first project to start with. You are asked to develop a prototype of the application. The prototype of the application will be then presented to the relevant stakeholders for the budget approval. Your manager has set up a meeting where you are asked to present the following in the next 15 working days (3 weeks):

* Specification document - Product’s capabilities, appearance, and user interactions
* Number and duration of sprints required
* Setting up Git and GitHub account to store and track your enhancements of the prototype
* Java concepts being used in the project
* Data Structures where sorting and searching techniques are used.
* Generic features and three operations:
  + Retrieving the file names in an ascending order
  + Business-level operations:
    - Option to add a user specified file to the application
    - Option to delete a user specified file from the application
    - Option to search a user specified file from the application
    - Navigation option to close the current execution context and return to the main context
  + Option to close the application

The goal of the company is to deliver a high-end quality product as early as possible.

# Requirements:

* Plan more than two sprints to complete the application.
* Document the flow of the application and prepare a flow chart.
* List the core concepts and algorithms being used in the application.
* Code to display the welcome screen with application name and developer details.
* Code to display the user interface such as options displaying the user interaction information.
* Features to accept the user input to select one of the options listed.
* Options should return the current files in ascending order, and able to perform add, delete, search, exit to main menu the files.
* Finally, option to close the application.

# Sprint Planning:

The following sprint planning has been done.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task Name** | **Priority** | **Owner** | **Start Date** | **End Date** | **Status** |
|  |
| **Sprint 1 - Requirement Mapping** | | | | | |  |
| Identify the customer requirement | High | Kaviya | 20-May-21 | 23-May-21 | Done |  |
| Map out the features required | Medium | 23-May-21 | 24-May-21 | Done |  |
| **Sprint 2 - Feature Designing** | | | | | |  |
| Design the features | High | Kaviya | 24-May-21 | 27-May-21 | Done |  |
| Design user flow | Low | 27-May-21 | 29-May-21 | Done |  |
| **Sprint 3 -Code Writing** |  | | | | |  |
| Write the code for separate functions | High | Kaviya | 30-May-21 | 03-Jun-21 | Done |  |
| Write the code for driver class | High | 03-Jun-21 | 05-Jun-21 | Done |  |
| Testing the code | High | 05-Jun-21 | 07-Jun-21 | Done |  |

# Flow chart:

# Solutions:

## Technical Design Documentation:

This application allows user to display the sorted file list. It also allows user to add files, search files, delete files and exit to main menu. It provides an option to exit out of the application.

The approach towards the solution is prepared as a flow chart and hereby attached along with the documentation.

Driver class starts by displaying the application name and developer details and version of the application. Followed by the options to choose from main menu and main exit for user interface. Declaring scanner object to read from the keyboard and a variable to store the user’s input temporarily. Declare a variable to store the path name entered by the user.

A while loop is used for the program to continue giving options to choose from until main exit is chosen. An else if statement is used to display the course of action to be executed. If condition checks whether the option chosen is main menu. Else if checks if the option chosen is main exit. Else block of code executes if the user input is invalid.

When the main menu is selected by the user, switch case statements are used to choose the following options to sort file, add file, search file, delete file and exit to main menu. User is prompted to select any option to perform their task.

The business logic is achieved by creating separate classes for each action/task like sorting files, adding files, searching files and deleting files.

Sorting files class makes use of the import java.io. file. It creates a file object to refer to the directory of files. Reading files in the directory into an array and using the java inbuilt method to get the list of files in an array and iterating over the files to get the name of the list of files. Arrays. Sort() method is used to sort the files and for each is used to get the names of files.

Adding files class also imports java.io.file. It declares the private variable to hold the pathname. Constructors with and without parameter is declared. A method to add file by the user is declared and it gets the path name from the user in which they want to add their files. It creates a file object to refer to the path name and by using inbuilt file method the user specified file is created and added to the specified folder. This throws a IO Exception, so the code is surrounded in try catch block. It also lists the files in the folder after adding the user specified file.

Searching file class also imports java.io. file. It declares the variable as private, and constructors are declared. It has a method to search the user specified file using exists () method which checks whether the file denoted by the abstract pathname exists, if it did it also compares the name using equals () method in an if condition. Only when the comparison turns out true, the method will return the file found. This is enclosed in a try catch block to catch any IOException being thrown.

Deleting files class imports java.nio.files. It declares a private variable and constructors. The method implemented in this class uses the path object to get the path name and uses the files inbuilt method. It deletes the user specified file if it exists and if not, it returns a file not found message. This is also enclosed by try catch block to catch any IO Exception.

Testing the code yielded the desired output. The output file/screenshots are attached for reference.

## Core Concepts Used:

* By creating objects of these classes in the user interface main driver class, we achieve encapsulation.
* Used Java file and I/O’s inbuilt methods for adding, deleting and searching user specified files.
* sorting is done on the files by reading files into an array.
* Exception handling is used to catch any exceptions from the code.
* GitHub and GitBash is used to store and update the project progression.
* Thus, by using all the above concepts source code optimization and increased performance is achieved.