COMPANY LOCKERS PVT. LTD.

LockedMe.com

GitHub Repository Link: https://github.com/pratinavhingonia22/LockedMe

Developed by: Pratinav Hingonia

Project Overview:

The project entails creating the prototype application LockedMe.com for Company Lockers Pvt. Ltd. The goal is to digitally transform their goods and offer a prototype for budget approval. Java will be used to create the application's command-line interface. It will include capabilities like file adding, deletion, and search. The project will be carried out in sprints, and the code will be kept on GitHub. Delivering high-quality products piecemeal while offering a user-friendly experience are the objectives.

Sprints:

Sprint 1: Project Setup and User Interface

Tasks:

- Set up the development environment (Eclipse/IntelliJ).
- Create a new Java project for LockedMe.com.
- Initialize Git and create a new repository on GitHub.
- Push the initial project code to the GitHub repository.
- Design the welcome screen with application name and developer details.
- Implement the user interface with options for user interaction.
- Add functionality to accept user input and execute corresponding actions.

Sprint 2: File Management Operations

Tasks:

- Implement the first option to retrieve file names in ascending order.
- Read the files from the root directory.
- Sort the file names in ascending order.
- Display the sorted file names.
- Implement the second option with file management operations.

  A) Adding Files

  - Add a file to the existing directory list:
  - Accept user input for the file name.
  - Add the filename to the collection of file names.

  B) *Delete Files*

  - Delete user specified file from the existing directory list:
  - Accept user input for the file name.
  - Remove the file name from the collection of file names.
  - Handle case sensitivity to ensure the correct file is deleted.
  - Display appropriate messages for successful or unsuccessful operation.

  C) Searching Files

- Search a user specified file from the main directory:
- Accept user input for the file name.
- Search for the file name in the collection of file names.
- Handle case sensitivity to retrieve the correct file.
- Display the result upon successful or unsuccessful search.

D) *Implement navigation option to return to the main context*

- Implement the third option to close the application.

Sprint 3: Source Code Optimization and Refinement

Tasks:

- Refactor the code for improved readability and maintainability.
- Optimize the source code for performance.
- Review and update exception handling to handle invalid inputs.
- Add necessary comments and documentation to the code.
- Perform thorough testing and debugging to ensure correctness.
- Create the final specification document.
- Push the updated source code and specification document to the GitHub repository.

Data Structures Used:

1) *Array*: Implicit usage of arrays may be seen in the form of Java arrays, which store the file names after they have been fetched from the directory. Before being arranged in alphabetical order and shown to the user, the file names are first saved in a structure like an array.

2) *ArrayList:* Before placing the names of the files in ascending order, they are stored in ArrayList for easy access as it enables dynamic scaling and the simple addition of components, it is appropriate for storing and manipulating the list of file names.

Within the programme, these data structures are used for the purposes of managing and manipulating the file names. While the array stores the names of the files that were retrieved, the ArrayList makes it possible to modify the list and arrange it in an effective manner.

Algorithms and Core Concepts Used:

Algorithms:

1) ***Sorting:***
   The programme sorts the file names in descending order by employing the Collections.sort() function that is available within the Java Collections framework. The temporal complexity of this technique is O(n log n), where n is the total number of files in the directory.

Core Concepts Used:

1) ***File Handling:***
   In order to interface with the file system, the programme makes use of the file handling capabilities that are provided by Java. It creates files, deletes files, and searches for files in the directory that you provide. Other activities it conducts include looking for files.

2) ***User Input:***
   The programme makes use of the Scanner class in order to receive input from the user via the command line interface. It presents the user with a menu of options and requests the names of files before doing a variety of tasks.

3) ***Conditional Statements:***
   The flow of the programme may be controlled by the user's decisions through the use of conditional statements such as if-else and switch-case. They choose the appropriate activities to carry out based on the menu item that has been picked.

4) ***Exception Handling:***
   The program code has try-catch blocks in order to manage any potential exceptions that may arise during the processing of file operations such as the creation and deletion of files. It makes certain that the programme continues to function without any hiccups and offers error messages when they are required.

5) ***Collections:***
   Before putting the file names in alphabetical order, the programme stores them in its memory using an ArrayList collection. The ArrayList component facilitates dynamic scaling, in addition to facilitating the simple insertion of file names to the list.

6) ***Looping:***
The code uses loops, such as for-each and while, to traverse over files, show menu options, and perform repetitive chores until the user decides to quit the programme. This continues until the user chooses to exit the application.

7) ***OOP:***
The programme utilizes an object-oriented approach by separating its code into classes and methods. It encourages reusability of code, modularity, and maintainability of the system.

8) ***Modularity:***
The code is modular, with distinct methods for different activities such as presenting menus, receiving user input, conducting file actions, and showing results. Additionally, the code displays results in a modular fashion. Readability and manageability of the code are both improved as a result.

Flow of the Application:

The following is an outline of the process that the LockedMe.com application goes through:

1) The programme begins by presenting the "welcome" screen, on which is shown the name of the application as well as information about the creator.

2) The user is presented with the primary menu, from which they may select several available alternatives. You have the choice of examining all of the files, carrying out activities on the files, or quitting the application.

3) If the user chooses the option to see all files, the programme will fetch the file names from the directory that the user requested and show them in ascending order if that option is selected.

4) If the user selects the option for file operations, a sub-menu will be presented. On this sub-menu, the user will have the choice to either add a file, remove a file, search for a file, or return to the main menu.

5) If the user chooses the option to add a file, they will be requested to input the name of the file once they have made that selection. After that, the programme will make an effort to generate a new file in the directory using the given name. If the operation was successful, a message congratulating you on your accomplishment is displayed; otherwise, the appropriate error message is presented.

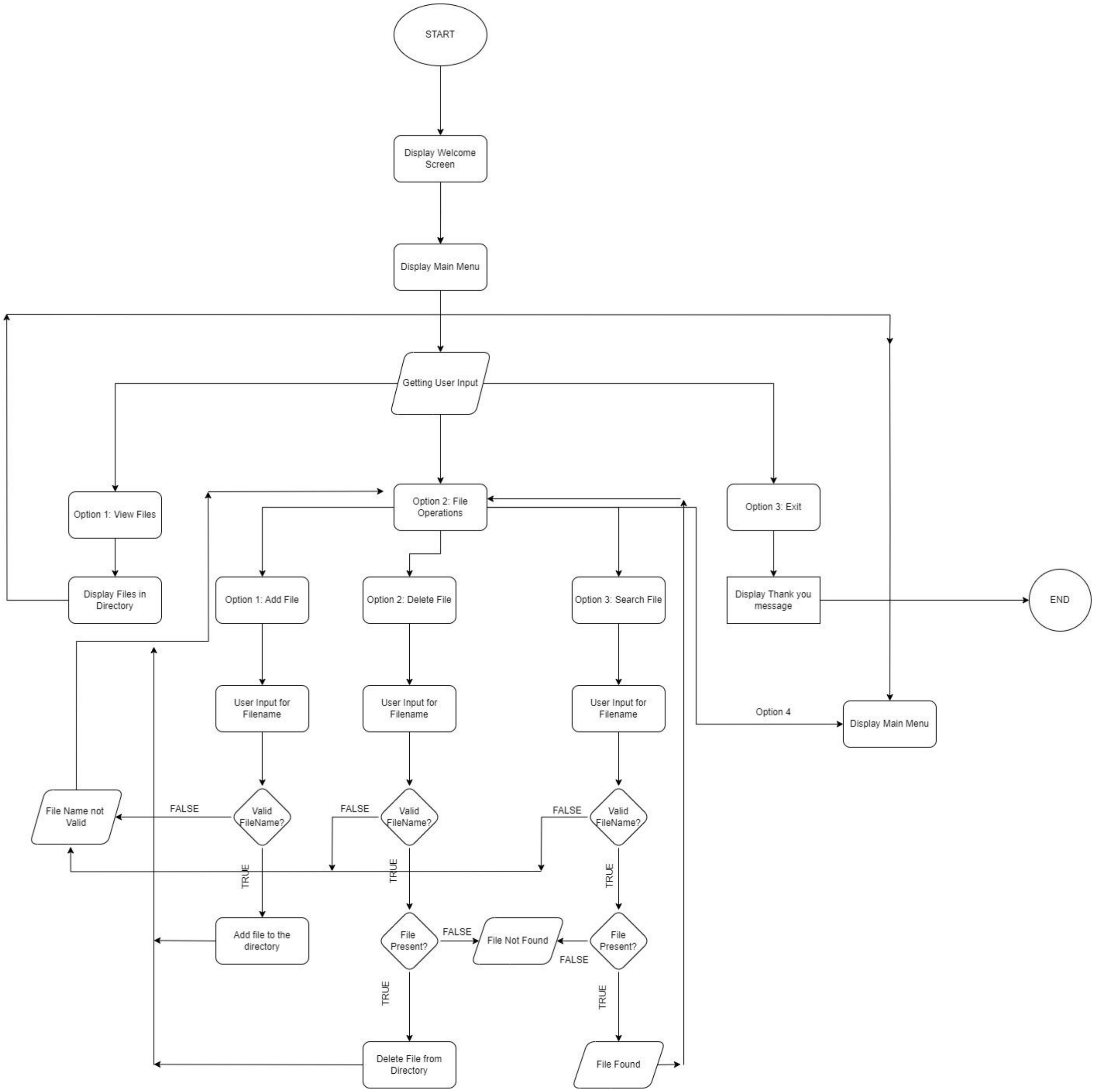
6) The user is invited to input the name of the file that is to be removed after selecting the option to delete a file from the drop-down menu. The application verifies if the file is present, and if it is, it removes it from the system. If the file was successfully removed, it will display a message of success; otherwise, it will display an error message if the file could not be located.

7) When the user selects the option to look for a file, they will be requested to type in the name of the file they are looking for. The application searches for the file in the directory that was supplied and then shows a message that indicates whether or not the file was found.

8) Following completion of any file operation, the user is presented with a menu of available file operations from which they can select a different action or return to the main menu.

9) If the user chooses to depart the programme by selecting the option to do so from the primary menu, a farewell message will be displayed, and the application will then close.

Flow Chart:


START
Display Welcome Screen
Display Main Menu
Getting User Input
Option 1: View Files
Option 2: File Operations
Option 3: Exit
Display Files in Directory
Option 1: Add File
Option 2: Delete File
Option 3: Search File
Display Thank you message
END
User Input for Filename
User Input for Filename
User Input for Filename
Option 4
Display Main Menu
File Name not Valid
FALSE
Valid FileName?
FALSE
Valid FileName?
FALSE
Valid FileName?
TRUE
TRUE
TRUE
Add file to the directory
File Present?
FALSE
File Not Found
FALSE
File Present?
TRUE
TRUE
Delete File from Directory
File Found

Conclusion on Enhancement of Application:

For Company Lockers Pvt. Ltd., the LockedMe.com application is acting as a prototype for the company's digitalization of file management. When seen from the perspective of a Full Stack Developer, the programme may be upgraded and improved in a number of different ways.

1) ***User Interface:*** At the moment, the programme only supports input via the command line. The user experience may be considerably improved by building a graphical user interface (GUI) for the user interface. This can also make the programme more user-friendly and aesthetically appealing.

2) ***Error Handling:*** The application's functionality can be improved by putting in place dependable systems for handling errors. This involves giving clear error messages, verifying user inputs, and gently managing any exceptions that may arise. The programme will become more robust and user-friendly after improved error handling has been implemented.

3) ***File Operations:*** The programme is presently capable of supporting fundamental file operations such as the addition of new files, the deletion of existing files, and the search for files. Enhancing the application's functionality and user-friendliness may be accomplished by integrating supplementary features such as file editing, choices for managing file information, and file sorting capabilities, for example.

4) ***Security:*** The use of security methods such as user authentication, authorisation, and encryption may ensure that the files maintained by the programme are kept private and intact if the appropriate safeguards are in place. This is of the utmost importance, particularly in the case when the application is utilized in a production setting.

Unique Selling Points of The Application:

1) ***Simplified File Management:*** LockedMe.com offers a method that is both straightforward and effective for the management of files. Users are able to quickly see, add to, remove from, and search for files in the designated directory, which simplifies the responsibilities associated with file administration.

2) ***Personalization and Adaptability:*** The programme may be altered to function with any directory, enabling users to organize their file collections in accordance with the unique requirements of their situations. The capability of the programme to execute actions on files contained inside a certain directory as well as the option to select the directory in which those files are located contributes to the application's versatility.

3) ***Scalability*** is a feature of this programme, which enables it to manage enormous file repositories and directories. Even when dealing with a large number of files, efficiency may be optimized by the application of algorithms and data structures that are very effective.

4) ***Open Source and Expandable:*** LockedMe.com has the potential to be built as an open-source project, which will enable programmers to contribute to the application's expansion, improvement, and further personalization. Its extensibility and adaptability to particular requirements are enabled by its modular design as well as its strict adherence to coding best practices.