Background of the problem statement

Specification Document:

Product Capabilities:

LockedMe.com is a command line application that allows users to manage their files in a user-friendly way. The application will provide the following capabilities:

1.Retrieve the file names in an ascending order

2.Add a user-specified file to the application

3.Delete a user-specified file from the application

4.Search for a user-specified file in the application

Product Appearance:

The application will be a command-line interface with a simple and user-friendly interface. The interface will provide the user with clear instructions and options to perform specific tasks.

User Interactions:

The user will interact with the application via the command line interface. The application will provide clear and concise instructions to the user on how to perform specific tasks. The user will be able to add, delete, search, and retrieve files with ease.

Sprints:

The development process will be broken down into three sprints, each lasting five working days. The sprints will be structured as follows:

Sprint 1:

- Set up the project and create the necessary files and directories
- Implement the functionality to retrieve the file names in an ascending order

Sprint 2:

- Implement the functionality to add a user-specified file to the application
- Implement the functionality to delete a user-specified file from the application

Sprint 3:

- Implement the functionality to search for a user-specified file in the application
- Add the navigation option to close the current execution context and return to the main context
- Add the option to close the application

Git and GitHub:

To store and track the enhancements of the prototype, we will use Git and GitHub. Each sprint will be developed on a separate branch, and once the sprint is complete, it will be merged with the main branch.

Java Concepts:

The application will be developed using Java programming language. The Java concepts that will be used in the project include:

- Object-oriented programming
- Exception handling
- File handling
- Collection framework
- Lambda expressions

Data Structures:

To ensure that the application is efficient and can handle large amounts of data, we will use data structures where sorting and searching techniques are used. The data structures that will be used in the project include:

- Arrays
- Lists
- Maps

Generic Features:

The application will include the following generic features:

- Retrieving the file names in an ascending order
- Navigation option to close the current execution context and return to the main context
- Option to close the application

Business-level Operations:

The application will include the following 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 for a user-specified file in the application
- Goal:

The goal of the company is to deliver a high-end quality product as early as possible. To achieve this, the development process will be agile and collaborative, with a focus on efficient and effective execution of tasks. Communication and coordination with all stakeholders, including project managers, designers, and QA engineers, will be essential to ensure the project is completed on time and to a high standard.

Project Details

Project Name: LockedMe

Developer Name: Karthikeyan P

Programming Language: Java

Sprints Planned and Achieved

Sprint 1:

- **Sprint Planning:**
  - Understanding the requirements and designing the user interface.
- **Tasks Achieved:**
  - Designed the user interface with three options to choose from: retrieve file names in ascending order, file operations, and exit.
- **Algorithm:**
  - Declare a constant DIRECTORY_PATH that holds the path to the directory where files will be stored.
  - Use a Scanner to read user input.
  - Display a welcome message to the user.
  - Display three options to choose from: retrieve file names, file operations, or exit.
  - Based on the user's input, call the corresponding method to retrieve file names, perform file operations, or exit the application.
  - In the retrieveFileNames () method:
    - Create a File object with the directory path.
    - Use the listFiles() method to get an array of files in the directory.
    - Check if the array is empty or not.
    - Sort the files in ascending order.
    - Display the names of the files.
  - In the fileOperations() method:
    - Use a do-while loop to keep displaying the menu until the user selects to go back to the main menu.
    - Display four options to choose from: add a file, delete a file, search for a file, or go back to the main menu.
    - Based on the user's input, call the corresponding method to add a file, delete a file, search for a file, or go back to the main menu.
  - In the addFile() method:
    - Use a Scanner to read the file name from the user.
    - Create a File object with the directory path and the file name.
    - Use the createNewFile() method to create the file.

        - Display a message to the user if the file was successfully created or if it already exists.
    - In the deleteFile() method:
        - Use a Scanner to read the file name from the user.
        - Create a File object with the directory path and the file name.
        - Use the delete() method to delete the file.
        - Display a message to the user if the file was successfully deleted or if it does not exist.
    - In the searchFile() method:
        - Use a Scanner to read the file name from the user.
        - Create a File object with the directory path and the file name.
        - Use the exists() method to check if the file exists.
        - Display a message to the user if the file exists or if it does not exist.
- **Core Concepts Used:**
    - File I/O
    - Control Flow (switch statement, do-while loop)

Sprint 2:

- **Sprint Planning**
    - Implementing the first option of retrieving file names in ascending order.
- **Tasks Achieved:**
    - Created a method to retrieve file names in ascending order from the specified directory.

- The retrieveFileNames() method was created to achieve this task. The method first creates a File object for the specified directory path and then calls the listFiles() method on this object to get an array of files in the directory. If the length of the array is 0, it prints a message saying the directory is empty. Otherwise, it sorts the array using the Arrays.sort() method and prints the names of the files in the directory in ascending order.
- The implementation of this feature is complete and has been tested successfully.

Sprint 3:

- **Sprint Planning**
    - Implementing the second option of file operations.
- **Tasks Achieved:**

1.Created a sub-menu with three options: add a file, delete a file, and search for a file.

2.Implemented a method for adding a file, which prompts the user to enter the file name and content and saves it in the specified directory.

3.Implemented a method for deleting a file, which prompts the user to enter the file name and deletes it from the specified directory.

4.Implemented a method for searching for a file, which prompts the user to enter the file name and searches for it in the specified directory. If found, it displays the file name and content. If not found, it displays a message that the file was not found.

- **Algorithm for adding a file:**

  1.Prompt the user to enter the file name and content.

  2.Verify that the file name is not empty.

  3.Create a File object with the specified file name and directory.

  4.If the file already exists, display a message that the file already exists and return.

  5.Otherwise, create a new file and write the content to it.

  6.Display a message that the file was added successfully.

- **Algorithm for deleting a file:**

  1.Prompt the user to enter the file name.

  2.Verify that the file name is not empty.

  3.Create a File object with the specified file name and directory.

  4.If the file does not exist, display a message that the file was not found and return.

  5.Otherwise, delete the file.

  6.Display a message that the file was deleted successfully.

- **Algorithm for searching for a file:**

  1.Prompt the user to enter the file name.

  2.Verify that the file name is not empty.

  3.Create a File object with the specified file name and directory.

  4.If the file does not exist, display a message that the file was not found and return.

  5.Otherwise, read the content of the file and display it.

6.Display a message that the file was found successfully.

- **Core concepts used in the project:**

  1.File handling: Reading and writing files using Java's File class.

  2.User input: Prompting the user for input and validating it to prevent errors.

Sprint 4:

- **Sprint Planning**
  - Testing and debugging the application.

Tasks Achieved: Tested the application and fixed any issues that arose.

- **Algorithms and Flowcharts of the Application:**

  1.Main Program Algorithm:

  1.1. Initialize the scanner object to read input from the user.

  1.2Print the welcome screen and display the option

available.

1.3Read the user input and perform the corresponding action until the user chooses to exit.

2.Retrieve File Names Algorithm:

2.1.Create a File object for the directory path.

2.2.Get a list of all files in the directory.

2.3.If the directory is empty, print a message indicating that.

2.4.If the directory has files, sort the files in ascending order.

2.5. Print the names of all files in the directory.

Sql code

Start -> Check if directory exists -> Get list of all files in directory -> Sort files in ascending order -> Display file names -> End

3.File Operations Algorithm:

3.1. Display the sub-menu options for adding, deleting, and searching for a file.

3.2.Read the user input and perform the corresponding action until the user chooses to go back to the main menu.

4.Add File Algorithm:

4.1.Read the name of the file to be added from the user.

4.2.Create a new File object with the specified name in the specified directory.

4.3.If the file already exists, print a message indicating that.

4.4.If the file is created successfully, print a message indicating that.

Sql code

Start -> Check if file exists -> Create file -> Display success message -> End

5.Delete File Algorithm:

5.1.Read the name of the file to be deleted from the user.

5.2.Create a new File object with the specified name in the specified directory.

5.3.If the file is deleted successfully, print a message indicating that.

5.4.If the file does not exist, print a message indicating that.

Sql code

Start -> Check if file exists -> Delete file -> Display success message -> End

6.Search File Algorithm:

6.1.Read the name of the file to be searched from the user.

6.2.Create a new File object with the specified name in the specified directory.

6.3.If the file is found, print a message indicating that.

6.4.If the file does not exist, print a message indicating that.

Sql code

Start -> Check if directory exists -> Get list of all files in directory -> Iterate over files -> Check if file name matches user input -> Display file path if match found -> Display error message if no match found -> End

Core Concepts Used in the Project:

- **Object-Oriented Programming:** The project may have used object-oriented programming concepts, such as creating and manipulating objects, encapsulation, and inheritance.
- **User Input:** The application reads input from the user using the Scanner class and validates the input to ensure it is of the correct type and within the specified range of valid values.
- **Output Formatting:** The application formats its output using string concatenation, printf statements, and other methods to make it more readable and user-friendly.
- **Sorting:** The application sorts the names of files in ascending order using the Collections.sort() method.
- **Documentation:** The project includes a README file that documents the purpose and functionality of the application and provides instructions for how to use it.
- **GitHub Repository Link:** https://github.com/karthikeyanvk18/LockedME

Conclusion and Unique Selling Points:

In conclusion, the LockedMe application provides users with a simple and user-friendly interface to perform file operations. The application allows users to retrieve file names in ascending order and perform file operations such as adding, deleting, and searching for files. The application does not close, exit, or throw an exception if the user specifies an invalid input.

Unique selling points of the LockedMe application include its simplicity and ease of use, as well as its ability to perform common file operations without the need for complex commands or knowledge of programming languages. Additionally, the application allows users to organize and manage their files in a convenient and efficient manner.