LockedMe.com Project Documentation

Project and Developer Details

- **Project Name:** LockedMe.com
- **Developer:** Jaydeep Patil
- **Description:** A command-line prototype application for managing files in a specified directory. It includes features for displaying files, adding, deleting, and searching files with user-friendly menus and error handling.
- **Programming Language:** Java

Sprint Planning

Sprint Duration: 3 weeks (15 working days)

Sprint 1 (Week 1)

- **Objective:** Set up the project, GitHub repository, and implement basic file operations.
- Tasks:
    - Create the directory structure and initialize the project in an IDE.
    - Implement file display functionality (ascending order sorting).
    - Develop and test the ability to add files to the directory.
    - Push the initial version of the project to GitHub.
- **Outcome:** A functional program that can display and add files.

Sprint 2 (Week 2)

- **Objective:** Implement advanced business operations.
- Tasks:
    - Add functionality to delete and search for files.
    - Improve error handling for invalid inputs and exceptions.
    - Test all features for robustness.
    - Update GitHub with the latest changes.
- **Outcome:** Completed core business-level operations (CRUD) functionality.

Sprint 3 (Week 3)

- **Objective:** Finalize the project and prepare documentation.
- Tasks:
    - Add menus and ensure user-friendly navigation.
    - Document the algorithms, flowcharts, and core concepts.
    - Capture screenshots of the output and update the GitHub repository.
    - Prepare the final specification document.
- **Outcome:** A finalized, documented prototype for stakeholders.

Algorithms

Display Files Algorithm

1. Fetch all files from the specified directory.
2. Filter the files to include only regular files, excluding subdirectories.
3. Sort the list of files in ascending order.
4. Display the sorted list to the user.

Add File Algorithm

1. Accept the file name from the user.
2. Check if the file already exists in the directory.
3. If not, create the file and confirm success.
4. Handle exceptions for invalid paths or file creation errors.

Delete File Algorithm

1. Accept the file name from the user.
2. Check if the file exists in the directory.
3. If it exists, delete the file and confirm success.
4. If the file doesn’t exist, display an appropriate message.

Search File Algorithm

1. Accept the file name from the user.
2. Search the directory for a file with the specified name.
3. If the file is found, display its details.
4. If the file is not found, notify the user.

Flowchart

A simplified flowchart for the application is as follows:

```
Start
  |
Main Menu
  |-- Option 1: Display Files --> Fetch and Sort Files --> Display
  |-- Option 2: Business Operations
  |       |-- Add File --> Check and Create --> Confirm
  |       |-- Delete File --> Check and Delete --> Confirm
  |       |-- Search File --> Search and Display
  |-- Option 3: Exit Application --> End
  |
Invalid Input --> Display Error --> Return to Main Menu
```

Core Concepts Used

1. **Object-Oriented Programming (OOP):**
   - Encapsulation of file operations into specific methods.
2. **Collections Framework:**
   - Used ArrayList for sorting and managing file names.
3. **Exception Handling:**
   - Handled IOException and invalid user inputs gracefully.
4. **File Handling:**
   - Used java.io.File for creating, deleting, and searching files.
5. **Sorting Algorithms:**
   - Used Collections.sort() for sorting file names in ascending order.

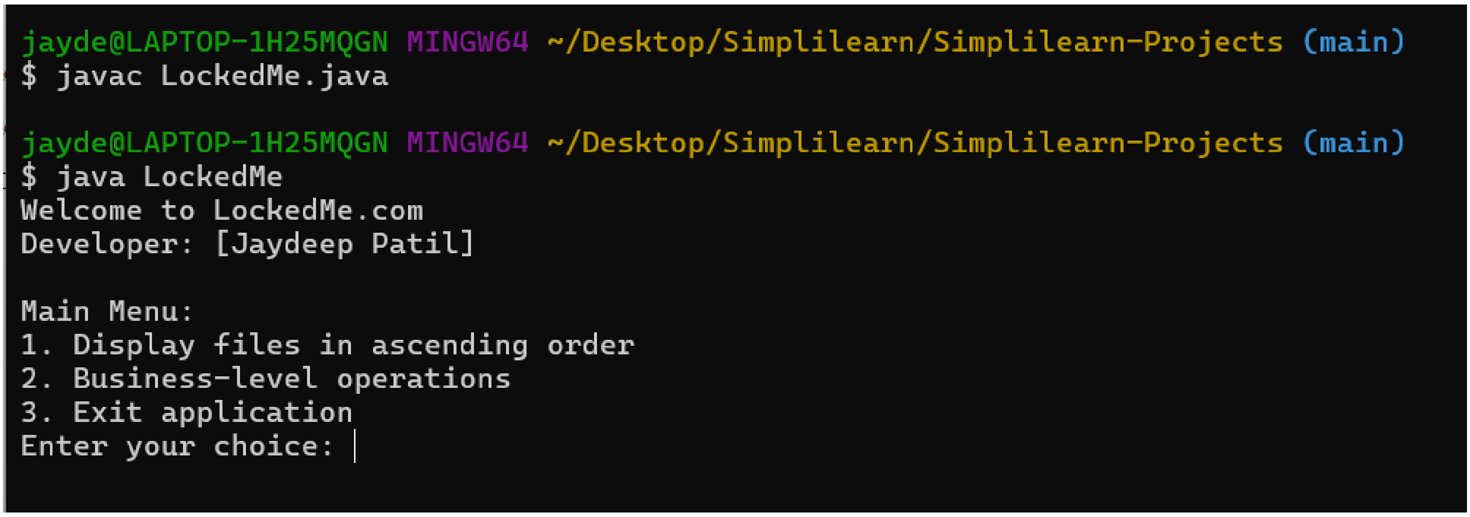
Conclusion and Enhancements

- **Unique Selling Points (USPs):**
  - Simple and user-friendly interface.
  - Robust error handling ensures the application doesn’t crash due to invalid inputs.
  - Lightweight and efficient operations with a modular design.
- **Potential Enhancements:**

1. Add support for managing subdirectories.
2. Enable batch operations for multiple files.
3. Develop a GUI version for better user experience.
4. Implement file encryption for enhanced security.
5. Add logging to track user actions for auditing purposes.

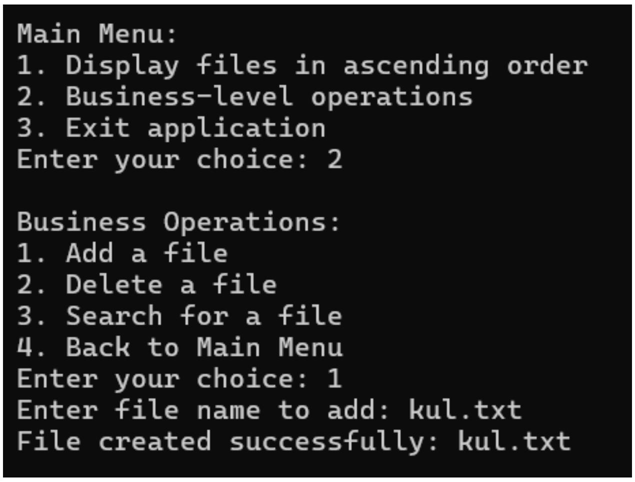
Screenshots

1. Run the Java Program



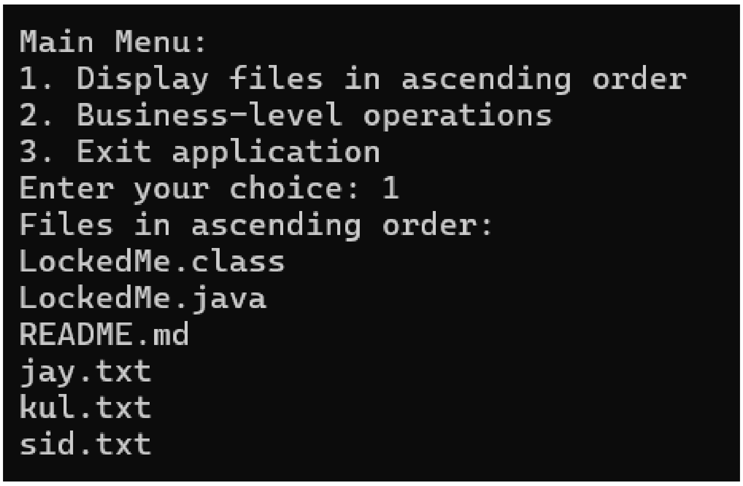
```
jayde@LAPTOP-1H25MQGN MINGW64 ~/Desktop/Simplilearn/Simplilearn-Projects (main)
$ javac LockedMe.java

jayde@LAPTOP-1H25MQGN MINGW64 ~/Desktop/Simplilearn/Simplilearn-Projects (main)
$ java LockedMe
Welcome to LockedMe.com
Developer: [Jaydeep Patil]

Main Menu:
1. Display files in ascending order
2. Business-level operations
3. Exit application
Enter your choice:
```

2. Main Menu



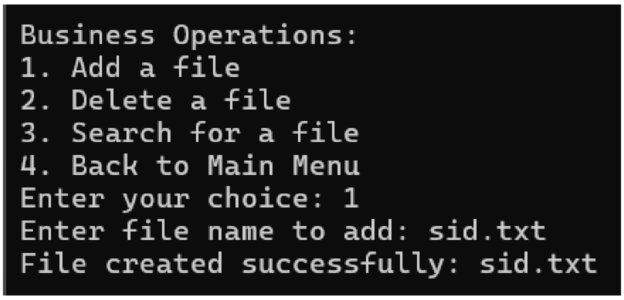
```
Main Menu:
1. Display files in ascending order
2. Business-level operations
3. Exit application
Enter your choice: 2

Business Operations:
1. Add a file
2. Delete a file
3. Search for a file
4. Back to Main Menu
Enter your choice: 1
Enter file name to add: kul.txt
File created successfully: kul.txt
```

3.Display Files in Order



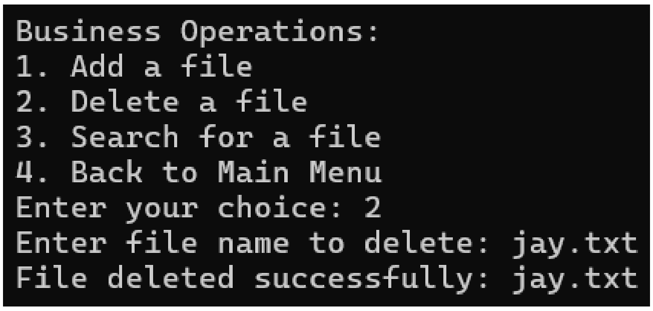
```
Main Menu:
1. Display files in ascending order
2. Business-level operations
3. Exit application
Enter your choice: 1
Files in ascending order:
LockedMe.class
LockedMe.java
README.md
jay.txt
kul.txt
sid.txt
```

4. Add a File



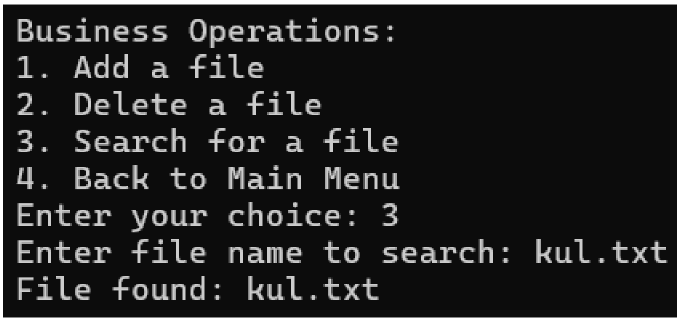
```
Business Operations:
1. Add a file
2. Delete a file
3. Search for a file
4. Back to Main Menu
Enter your choice: 1
Enter file name to add: sid.txt
File created successfully: sid.txt
```

5. Delete a File



```
Business Operations:
1. Add a file
2. Delete a file
3. Search for a file
4. Back to Main Menu
Enter your choice: 2
Enter file name to delete: jay.txt
File deleted successfully: jay.txt
```

6. Search a File



```
Business Operations:
1. Add a file
2. Delete a file
3. Search for a file
4. Back to Main Menu
Enter your choice: 3
Enter file name to search: kul.txt
File found: kul.txt
```