LockedMe.com Application

LockedMe.com
Prototype of the Application
Name : Shaik Akhila
GitHub : https://github.com/akhila31shaik/Project-Phase-1-.git

The prototype of the application is operated as a CLI (Command Line Program) without GUI. Its usage is to do file operations such as create new files along with content, delete a file or search a file from a specified directory and list them afterward in sorting order.

The implementation is done with the help of Java 8 and IDE IntelliJ.

Sprint Planning

The Implementation is done in two sprints which are mentioned below:

Sprint 1:

- Clarify the specification and requirements.
- Implement view content mechanism.
- Implement list of all files in sorted order.
- Implement functionality to close the program safely.

Sprint 2:

- Implement functionality to add create files along the content.
- Implement functionality to delete a file if it is present in that user specified directory.
- Implement functionality to search a file in the same directory.
- Documentation.

Source Code:
File directory

```java
package org.cisco.projectphase1.files;
import java.util.ArrayList;
import java.util.Collections;
import java.io.File;
import java.nio.file.FileSystems;
import java.nio.file.Path;
10 usages
public class Directory {
    4 usages
    public static final String name = "src/main/directory/";
    5 usages
    private ArrayList<File> files = new ArrayList<~>();
    1 usage
    Path path = FileSystems.getDefault().getPath(name).toAbsolutePath();
    1 usage
    File Dfiles = path.toFile();
    1 usage
    public String getName() { return name; }
    2 usages
    public ArrayList<File> addFiles() {

        File[] directoryFiles = Dfiles.listFiles();

        files.clear();
```

Service Component:

```java
package org.cisco.projectphase1.services;
import java.io.File;
import org.cisco.projectphase1.files.Directory;
4 usages
public class DirectoryService {
    3 usages
    private static Directory fileDirectory = new Directory();

    1 usage
    public static void PrintFiles() {

        fileDirectory.addFiles();

        for (File file : DirectoryService.getFileDirectory().getFiles())
        {
            System.out.println(file.getName());
        }
    }
    1 usage
    public static Directory getFileDirectory() { return fileDirectory; }

    public static void setFileDirectory(Directory fileDirectory) { DirectoryService.fileDirectory = fileDir

}
```

```java
import org.cisco.projectphase1.Welcome.FileOperations;
import org.cisco.projectphase1.Welcome.Screen;
import org.cisco.projectphase1.Welcome.WelcomePage;

10 usages
public class ScreenService {
    2 usages
    public static WelcomePage WelcomePage = new WelcomePage();
    1 usage
    public static FileOperations FileOperations = new FileOperations();



    2 usages
    public static Screen CurrentScreen = WelcomePage;


    4 usages
    public static Screen getCurrentScreen() {

        return CurrentScreen;
    }


    2 usages
```

Welcome Component:

```java
package org.cisco.projectphase1.Welcome;

import java.io.File;
import java.io.IOException;
import java.nio.file.FileSystems;
import java.nio.file.Path;
import java.util.ArrayList;
import java.util.InputMismatchException;
import java.util.Scanner;
import org.cisco.projectphase1.files.Directory;
import org.cisco.projectphase1.services.ScreenService;

3 usages
public class FileOperations implements Screen{
    4 usages
    private Directory dir = new Directory();

    5 usages
    private ArrayList<String> options = new ArrayList<>();

    1 usage
    public FileOperations() {

        options.add("1. Add a File to the directory");
        options.add("2. Delete A File from the directory");
```

```java
package org.cisco.projectphase1.Welcome;

public interface Screen {
    8 usages  2 implementations
    public void Show();

    2 usages  2 implementations
    public void NavigateOption(int option);

    3 usages  2 implementations
    public void GetUserChoice();
}
```

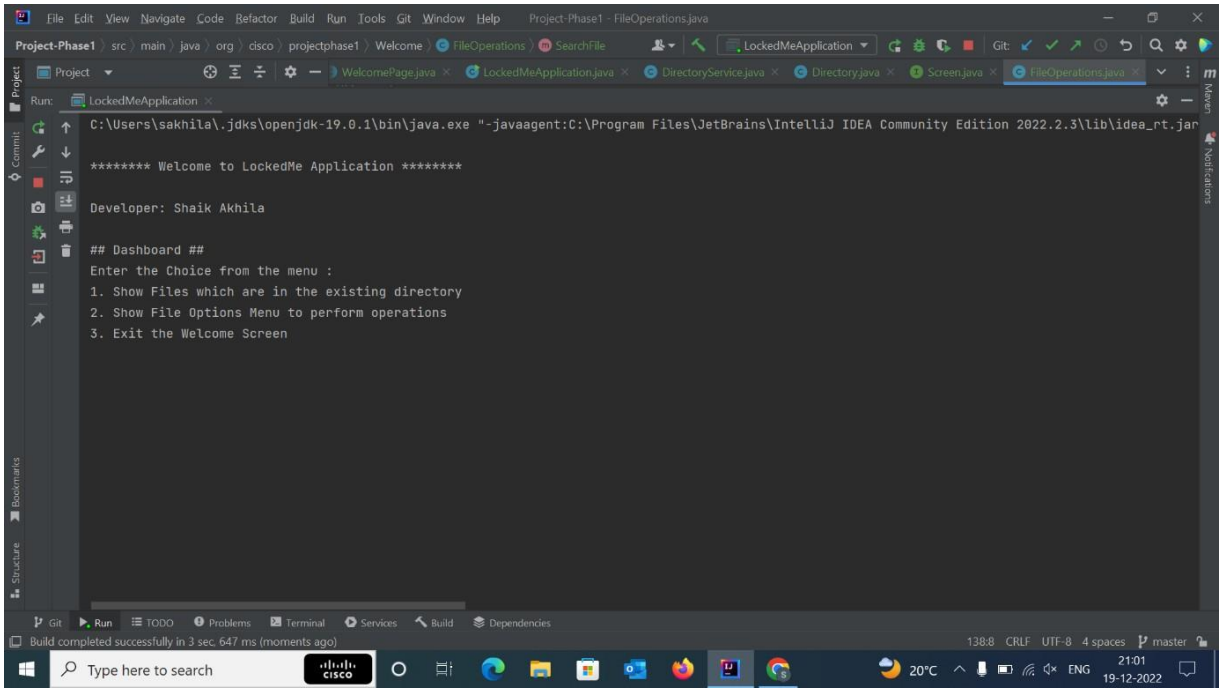
```java
import java.util.ArrayList;
import java.util.InputMismatchException;
import java.util.Scanner;

6 usages
public class WelcomePage implements Screen{
    1 usage
    private String Text = "\n******** Welcome to LockedMe Application ********\n";
    1 usage
    private String developerText = "Developer: Shaik Akhila";

    4 usages
    private ArrayList<String> options = new ArrayList<>();


    2 usages
    public WelcomePage() {
        options.add("1. Show Files which are in the existing directory");
        options.add("2. Show File Options Menu to perform operations");
        options.add("3. Exit the Welcome Screen");

    }

    1 usage
```

Main Class:

```java
package org.cisco.projectphase1;

import org.cisco.projectphase1.Welcome.WelcomePage;

public class LockedMeApplication {
    public static void main(String[] args) {
        WelcomePage wp = new WelcomePage();
        wp.userPage();
        wp.GetUserChoice();

    }
}
```

Screens:

This is the first screen that user will interact.



List all files:
This option will let user to see list of files in the specified directory in sorted order.

```
## Dashboard ##
Enter the Choice from the menu :
1. Show Files which are in the existing directory
2. Show File Options Menu to perform operations
3. Exit the Welcome Screen
1
List of Files in the Directory:
aa.txt
Aadhar.txt
Alexa.txt
```

File Operations:

This option will let user to provide several file operations with.

```
## Dashboard ##
Enter the Choice from the menu :
1. Show Files which are in the existing directory
2. Show File Options Menu to perform operations
3. Exit the Welcome Screen
2
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
```

Create a file:

This will allow user to create a file along with content inside it.

```
Please Enter the Filename:
Demo.txt
You are adding a file name to the directory: Demo.txt
File Added successfullu in the directory
File created: Demo.txt
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
4
## Dashboard ##
Enter the Choice from the menu :
1. Show Files which are in the existing directory
2. Show File Options Menu to perform operations
3. Exit the Welcome Screen
1
List of Files in the Directory:
aa.txt
Aadhar.txt
Alexa.txt
Demo.txt
Document.txt
## Dashboard ##
Run  TODO  Problems  Terminal  Services  Build  Dependencies
pleted successfully in 2 sec, 60 ms (2 minutes ago)
47:1  CRLF  UTF-8  4 spaces
```

Delete a file:

This will allow user to delete a file if it is present otherwise it will send a appropriate message.

```
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
2
Please Enter the Filename:
aa.txt
You are deleting a file named: aa.txtfrom the directory
File Deleted Successfully from the directory
Deleted File: aa.txt
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
```

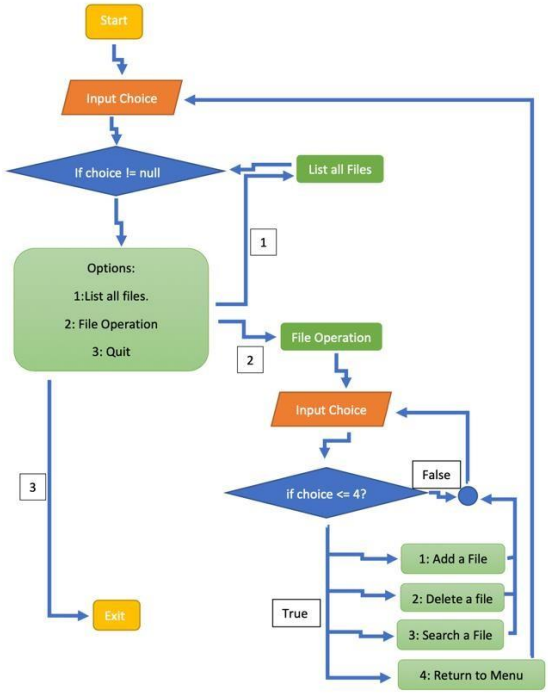
Search a file:
This will allow user to input a file name along with extension to begin the search procedure and give back the appropriate result.

```
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
3
Please Enter the Filename:
Alexa.txt
You are searching for a file named: Alexa.txt
Found Alexa.txt
The searched file was found in the directory
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
```

Quit
This will allow user to exit from the program safely.

```
File Options Menu
1. Add a File to the directory
2. Delete A File from the directory
3. Search A File from the directory
4. Return to Dashboard
4
## Dashboard ##
Enter the Choice from the menu :
1. Show Files which are in the existing directory
2. Show File Options Menu to perform operations
3. Exit the Welcome Screen
3


Process finished with exit code 0
```

Flow Diagram



- Core Concepts used in this project are mostly basic Java libraries such as Class & Objects, Packages, Interfaces, Collections, ArrayList, Access specifier, Try-catch block, File Handling Concepts, Error Exception handling, Inheritance, abstract, final, static methods.

Conclusion

1: The prototype is robust and platform independent.

2: User can easily use the prototype and safely exit out of it.

3: The prototype has a good interface with CLI (Command Line Interface).

4: As a developer, we can enhance it by introducing several new features such as appending in a file or overwriting a file and the file details for which user selected.

5: This prototype though is robust but user can only interact it with terminal or CLI so we can develop a good GUI interface for more better user-friendly.