**LockedMe-Virtual Key for Your Repositories**

This document contains sections for:

* [Sprint planning and Task completion](#Sprint_plan) of the project.
* [Core concepts used in project](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Core_concepts)
* [Flowchart of the Application](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Flow).
* [Demonstrating the product capabilities, appearance, and user interactions.](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Product_capability)
* [Conclusions](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Conclusions)

The code for this project is stored at

Github URL:

<https://github.com/manijoy/lockedme.com>

The project is developed by **Tangudu Manikanta**

## Sprints planning and Task completion

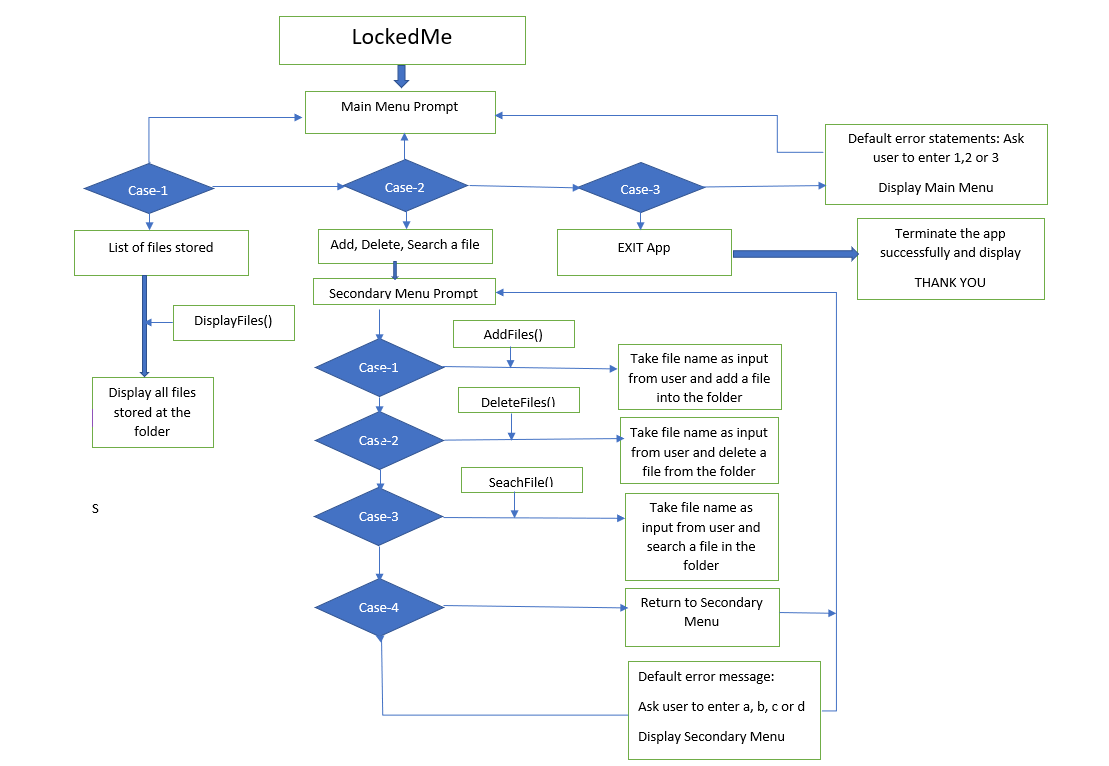
We are creating application for files management. The application is planned to be completed in 3 sprints of time for one developer and one tester, and each sprint is of 2 weeks. Tasks assumed to be completed in the sprints are:

* Creating a flowchart for the application
* Initializing git repository to track changes as development progresses.
* Writing the Java program to fulfill the requirements of the project.
* Testing the Java program with different kinds of User input
* Pushing code to GitHub.
* Creating this specification document highlighting application capabilities, appearance, and user interactions.

## Core concepts used in project

Collections framework, File Handling, Sorting, Flow Control, Recursion, Exception Handling, Streams API

## Flowchart of the Application



## Demonstrating the product capabilities, appearance, and user interactions

To demonstrate the product capabilities, below are the sub-sections configured to highlight appearance and user interactions for the project:

1. [Creating a new project in Eclipse](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_1)
2. [Writing a program in Java for the entry point of the application (**LockedMe.java**)](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_2)
3. [Writing a program in Java to display Menu options available for the user (**MENU OPTIONS.java**)](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_3)
4. [Writing a program in Java to handle Menu options selected by user (**HandleOptions.java**)](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_4)
5. [Writing a program in Java to perform the File operations as specified by user (**FileOperations.java**)](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_5)
6. [Pushing the code to GitHub repository](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_6)

## **Step 1:** Creating a new project in Eclipse

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next.
* Create a Project named as “Lockedme”.
* Select your project and go to File -> New -> Package.
* Create a Package name “com.lockedme”,and click on finish.
* Right click Package-> New -> Class.
* Create a Class named as “Lockedme.java”, and click on finish.
* Check the checkbox “public static void main(String[] args)”, and click on “Finish.”

## **Step 2:** Writing a program in Java for the entry point of the application (**LockedMe.java**)

package LockedMe

import java.io.File;

import java.io.IOException;

import java.util.Arrays;

import java.util.Scanner;

public class LockedMe {

File folder\_name;

static String FOLDER\_DIR;

public LockedMe() {

FOLDER\_DIR = System.getProperty("user.dir");

folder\_name = new File(FOLDER\_DIR+"/files");

if (!folder\_name.exists())

folder\_name.mkdirs();

System.out.println("Folder : "+ folder\_name.getAbsolutePath());

}

private static final String WELCOME\_SCREEN =

"\*\*\*\*\*\* LockedMe.com \*\*\*\*\*\*\*\*"+

"\*\*\*\*\*\* This application is developed by \*\*\*\*\*\*\*\*"+

"\*\*\*\*\*\* MANIKANTA TANGUDU \*\*\*\*\*\*\*\*";{

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/n”) }

## **Step 3:** Writing a program in Java to display Menu options available for the user (**MAIN\_ MENU\_SCREEN**)

* 1. [Displaying Welcome Screen](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_3_1)
  2. [Displaying Initial Menu](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_3_2)
  3. [Displaying Secondary Menu for File Operations available](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_3_3)

**Step 3.1:** Writing method to display Welcome Screen

package LockedMe;

public LockedMe() {

FOLDER\_DIR = System.getProperty("user.dir");

folder\_name = new File(FOLDER\_DIR+"/files");

if (!folder\_name.exists())

folder\_name.mkdirs();

System.out.println("Folder : "+ folder\_name.getAbsolutePath());

}

private static final String WELCOME\_SCREEN =

"\*\*\*\*\*\* LockedMe.com \*\*\*\*\*\*\*\*"+

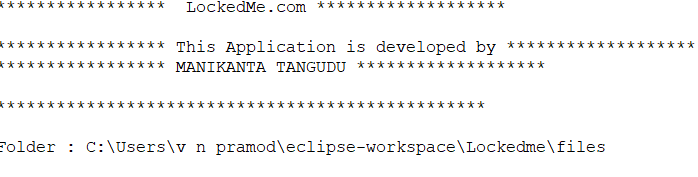
"This application is developed by"+

"\*\*\*\*\*\*MANIKANTA TANGUDU \*\*\*\*\*\*\*\*";{

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

**Output:**



**Step 3.2:** Writing method to display Initial Menu

**private static final String MAIN\_MENU\_SCREEN =**

**"\n |\*\*\* MAIN MENU \*\*\*| \n"+**

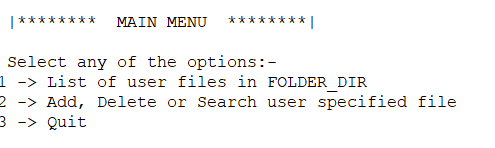
**"\n Select any of the options:- \n"+**

**"1 -> List of user files in FOLDER\_DIR\n"+**

**"2 -> Add, Delete or Search user specified file\n"+**

**"3 -> Quit ";**

**Output:**

****

**Step 3.3:** Writing method to display Secondary Menu for File Operations

private static final String SECONDARY\_MENU\_SCREEN =

"\n |\*\*\* SECONDARY MENU \*\*\*| \n"+

" \nSelect any of the options: \n"+

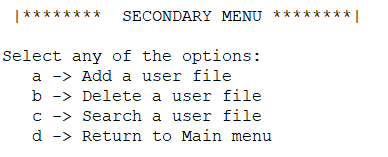
" a -> Add a user file\n"+

" b -> Delete a user file\n"+

" c -> Search a user file\n"+

" d -> Return to Main menu";

**Output:**

****

## **Step 4:** Writing a program in Java to handle Menu options selected by user (**HandleOptions.java**)

* 1. [Handling input selected by user in initial Menu](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_4_1)
  2. [Handling input selected by user in secondary Menu for File Operations](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_4_2)

**Step 4.1:** Writing method to handle user input in initial Menu

void showPrimaryMenu() {

System.out.println(MAIN\_MENU\_SCREEN);

try{

Scanner scanner = new Scanner(System.in);

int option = scanner.nextInt();

switch (option){

case 1 : {

showFiles();

showPrimaryMenu();

}

case 2 : {

showSecondaryMenu();

}

case 3 : {

System.out.println("\*\*\*Thank You\*\*\*\*");

System.exit(0);

}

default:

System.out.println("Please Select options 1, 2 or 3");

showPrimaryMenu();

}

}

catch (Exception e){

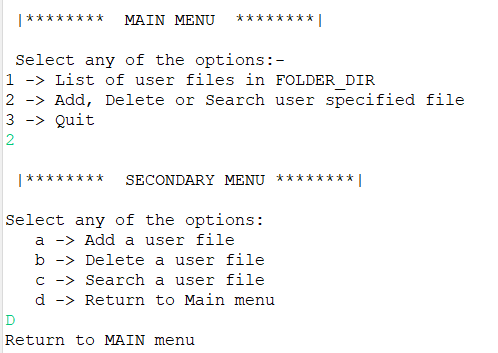
System.out.println("Please Select options 1, 2 or 3");

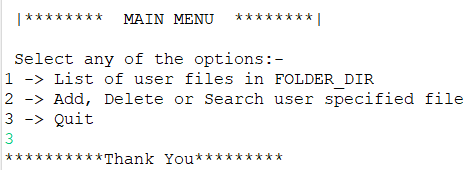
showPrimaryMenu();

}

}

**Output:**

****

****

**Step 4.2:** Writing method to handle user input in Secondary Menu for File Operations

void addFile(String filename) throws IOException {

File filepath = new File(folder\_name +"/"+filename);

String[] list = folder\_name.list();

for (String file: list) {

if (filename.equalsIgnoreCase(file)) {

System.out.println("File " + filename + " already exists at " + folder\_name);

System.out.println("Try with another filename");

return;

}

}

filepath.createNewFile();

System.out.println("File "+filename+" successfully added to "+ folder\_name);

}

void deleteFile(String filename) {

File filepath = new File(folder\_name +"/"+filename);

String[] list = folder\_name.list();

for (String file: list) {

if (filename.equals(file) && filepath.delete()) {

System.out.println("File " + filename + " successfully deleted from " + folder\_name);

return;

}

}

System.out.println("Error occurred while Deleting File..");

System.out.println("Please enter existing file name to delete..");

}

void searchFile(String filename) {

String[] list = folder\_name.list();

for (String file: list) {

if (filename.equals(file)) {

System.out.println("FILE FOUND : File " + filename + " exists at " + folder\_name);

return;

}

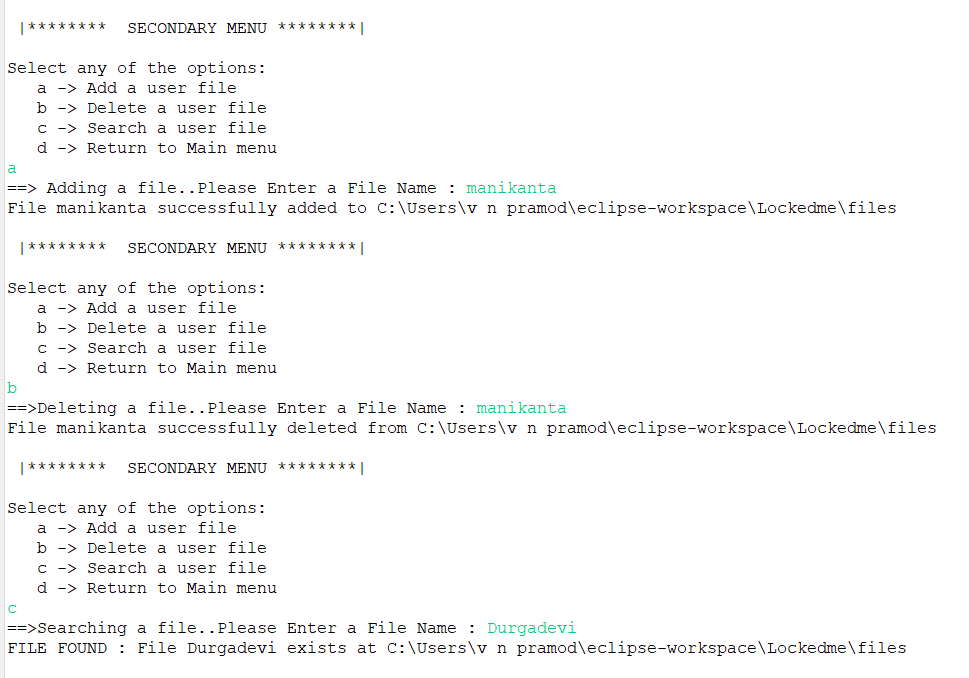
}

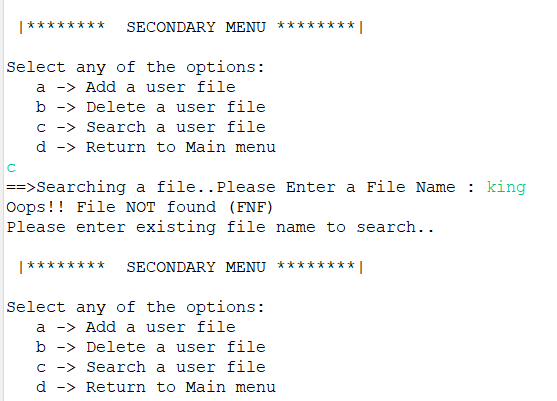
System.out.println("Oops!! File NOT found (FNF)");

System.out.println("Please enter existing file name to search..");

}

**Output:**

****

****

## **Step 5:** Writing a program in Java to perform the File operations as specified by user

* **FileOperations** consists methods for -:
  1. [Creating “main” folder in project if it’s not already present](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_5_1)
  2. [Displaying all files in “main” folder in ascending order and also with directory structure.](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_5_2)
  3. [Creating a file/folder as specified by user input.](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_5_3)
  4. [Search files as specified by user input in “main” folder and it’s subfolders.](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_5_4)
  5. [Deleting a file/folder from “main” folder](file:///C:\Users\jjj\Downloads\LockedMe%20-%20Virtual%20Key%20for%20Repositories.docx#Step_5_5)

**Step 5.1:** Writing method to create “main” folder in project if it’s not present

package LockedMe;

import java.io.File;

import java.io.IOException;

import java.util.Arrays;

import java.util.Scanner;

public class LockedMe {

File folder\_name;

static String FOLDER\_DIR;

public LockedMe() {

FOLDER\_DIR = System.getProperty("user.dir");

folder\_name = new File(FOLDER\_DIR+"/files");

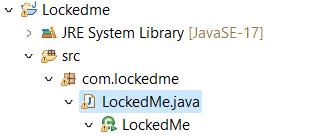
if (!folder\_name.exists())

folder\_name.mkdirs();

System.out.println("Folder : "+ folder\_name.getAbsolutePath());

}

**Output:**

**\**

**Step 5.2:** Writing method to display all files in “main” folder

void showFiles() {

if (folder\_name.list().length==0)

System.out.println("empty folder");

else {

String[] list = folder\_name.list();

System.out.println("The files in "+ folder\_name +" are :");

Arrays.sort(list);

for (String str:list) {

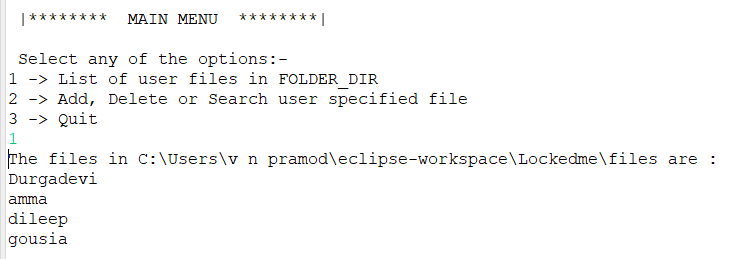
System.out.println(str);

}

}

}

**Output:**



**Step 5.3:** Writing method to create a file/folder as specified by user input.

**void addFile(String filename) throws IOException {**

**File filepath = new File(folder\_name +"/"+filename);**

**String[] list = folder\_name.list();**

**for (String file: list) {**

**if (filename.equalsIgnoreCase(file)) {**

**System.out.println("File " + filename + " already exists at " + folder\_name);**

**System.out.println("Try with another filename");**

**return;**

**}**

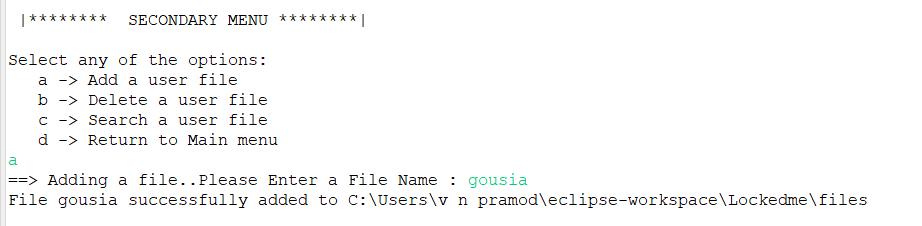
**}**

**filepath.createNewFile();**

**System.out.println("File "+filename+" successfully added to "+ folder\_name);**

**}**

**Output:**

****

**Step 5.4:**  Writing method to search for all files as specified by user input in “main” folder and it’s subfolders.

**void searchFile(String filename) {**

**String[] list = folder\_name.list();**

**for (String file: list) {**

**if (filename.equals(file)) {**

**System.out.println("FILE FOUND : File " + filename + " exists at " + folder\_name);**

**return;**

**}**

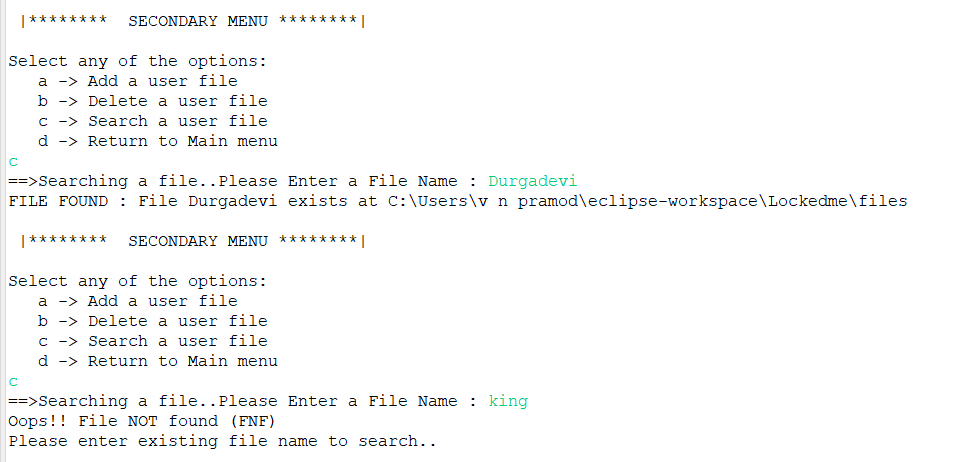
**}**

**System.out.println("Oops!! File NOT found (FNF)");**

**System.out.println("Please enter existing file name to search..");**

**}**

**Output:**



**Step 5.5:**  Writing method to delete file/folder specified by user input in “main” folder and it’s subfolders. It uses the searchFilesRecursively method and prompts user to specify which index to delete. If folder selected, all it’s child files and folder will be deleted recursively. If user wants to delete all the files specified after the search, they can input value 0.

void deleteFile(String filename) {

File filepath = new File(folder\_name +"/"+filename);

String[] list = folder\_name.list();

for (String file: list) {

if (filename.equals(file) && filepath.delete()) {

System.out.println("File " + filename + " successfully deleted from " + folder\_name);

return;

}

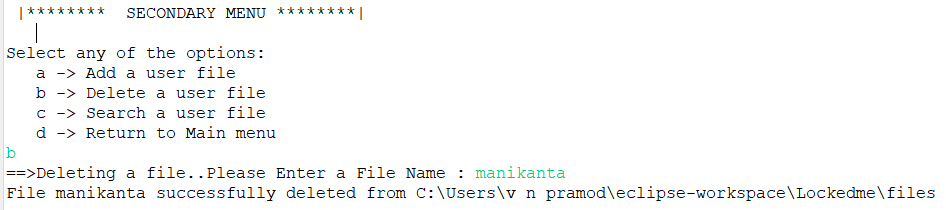
}

System.out.println("Error occurred while Deleting File..");

System.out.println("Please enter existing file name to delete...");

}

**Output:**



## **Step 6:** Pushing the code to GitHub repository

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add**

* Commit the changes using the following command:

**git commit . -m <commit message>**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**

## Conclusions

1. Gained knowledge on file handling properties in JAVA
2. A working model of LockedMe project used for file handling is created
3. This project can be made more interactive using frontend technologies like HTML.
4. For further enhancements we can add Yes/No option when user exiting from the application.