|  |
| --- |
| LockedMe  Project Specifications and Sprint works |

|  |  |
| --- | --- |
| Author | Sai Vineeth Tata |
| Purpose | Screenshot of the Application |
| Date | 14th August 2021 |
| Version | 1.0 |

# **Modules in the Project**

1. Display all Files
2. Add a file
3. Delete a file
4. Search a file

# **Sprint Work**

|  |  |
| --- | --- |
| Sprint Number | Modules |
| 1 | Display All Files  Add a new File |
| 2 | Delete a file  Search a file  Testing |

Java Technologies used:

* Exception Handling.
* Working with Files
* Naming Standards
* Modularity
* Oops
* Collections
* Control Structure
* Data Structures

Project link in GitHub: <https://github.com/Sai153793/lockedme-project.git>

**Project Code:**

**Sprint Planning and Task completion:**

**The project is planned to be completed in 2 sprints. Tasks assumed to be completed in the sprint are:**

* **Creating the flow of 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.**

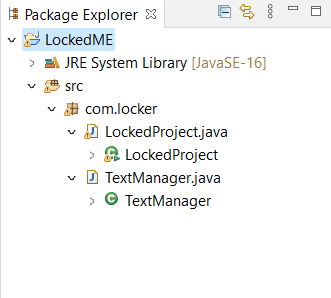
**Process to create new project in the eclipse:**

* **Open Eclipse**
* **Go to File -> New -> Project -> Java Project -> Next.**
* **Type in any project name and click on “Finish.”**
* **Select your project and go to File -> New -> Class.**
* **Enter LockedMeMain in any class name, check the checkbox “public static void main(String[] args)”, and click on “Finish.”**

**Flow of Application:**

****

**Folder Structure:**



**LockedMe code :**

**package com.Lockedme;**

**import java.util.ArrayList;**

**import java.util.Arrays;**

**import java.util.List;**

**import java.util.Scanner;**

**public class LockedMeProject**

**{**

**static final String folderpath="D:\\my java\\Myphase1project\\LockedMeFiles";**

**// private static final FileManager LockedMeMain = null;**

**private static final int case1 = 0;**

**public static void main(String[] args) {**

**int proceed=1;**

**do**

**{**

**//Variable decleration**

**Scanner obj=new Scanner(System.in);**

**int ch;**

**//Menu**

**displayMenu();**

**System.out.println("Enter your choice:");**

**ch=Integer.parseInt(obj.nextLine());**

**switch(ch)**

**{**

**case 1:getAllFiles();**

**break;**

**case 2:addFiles();**

**break;**

**case 3:deleteFiles();**

**break;**

**case 4:searchFiles();**

**break;**

**case 5:System.exit(0);**

**break;**

**default: System.out.println("Invalid Option");**

**break;**

**}**

**}while(proceed>0);**

**}**

**public static void displayMenu(){**

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

**System.out.println("\t\tlockedme.com");**

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

**System.out.println("1.Display all files");**

**System.out.println("2.Add new files");**

**System.out.println("3.Delete a file");**

**System.out.println("4.Search a files");**

**System.out.println("5.Exit");**

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

**}**

**public static void getAllFiles() {**

**// code to get filenames.**

**List<String> fileNames = FileManager.getAllFiles(folderpath);**

**if(fileNames.size()==0)**

**System.out.println("No files in the directory");**

**else**

**{**

**System.out.println("file List is below:/n");**

**for(String f:fileNames)**

**System.out.println(f);**

**}**

**}**

**public static void addFiles() {**

**//code for adding a files.**

**Scanner obj = new Scanner(System.in);**

**String filename;**

**int linesCount;**

**List<String> content = new ArrayList<String>();**

**//Read file name from user**

**System.out.println("Enter file Name:");**

**filename =obj.nextLine();**

**//Read number of lines from user**

**System.out.println("Enter how many lines in the file:");**

**linesCount=Integer.parseInt(obj.nextLine());**

**//Read Lines from user**

**for (int i = 1;i<=linesCount;i++)**

**{**

**System.out.println("Enter line"+i+":");**

**content.add(obj.nextLine());**

**}**

**//save the content into the file**

**boolean isSaved = FileManager.createFiles(folderpath, filename, content);**

**if(isSaved)**

**System.out.println("file and data saved successfully");**

**else**

**System.out.println("some error occured.Please contact admin@sai.com");**

**//Close scanner object**

**}**

**public static void deleteFiles() {**

**//code for deleting a file**

**String fileName;**

**Scanner obj=new Scanner(System.in);**

**System.out.println("Enter file name to be deleted:");**

**fileName=obj.nextLine();**

**boolean isDeleted=FileManager.deleteFile(folderpath,fileName);**

**if(isDeleted)**

**System.out.println("File deleted successfully");**

**else**

**System.out.println("Either file not there or some access issue");**

**}**

**public static void searchFiles() {**

**//code for searching a file**

**String fileName;**

**Scanner obj=new Scanner(System.in);**

**System.out.println("Enter file name to be searched:");**

**fileName=obj.nextLine();**

**boolean isFound=FileManager.searchFile(folderpath,fileName);**

**if(isFound)**

**System.out.println("File is present in the folder");**

**else**

**System.out.println("File is not present in the folder");**

**}**

**}**

**File Manager Code :**

**package com.Lockedme;**

**import java.io.File;**

**import java.io.FileFilter;**

**import java.io.FileWriter;**

**import java.util.ArrayList;**

**import java.util.List;**

**public class FileManager**

**{**

**/\*\***

**\* This method will return the files names from the folder.**

**\* @param folderpath**

**\* @return List<String>**

**\*/**

**public static List<String> getAllFiles(String folderpath)**

**{**

**File fl = new File(folderpath);**

**//getting all the files into FileArray**

**File[] listofFiles = fl.listFiles();**

**//declare a list to store file names**

**List<String> fileNames = new ArrayList<String>();**

**for (File f:listofFiles) //Using ForEach to get the file names**

**fileNames.add(f.getName());**

**return fileNames;**

**}**

**/\*\***

**\* This method will create or append content into the file specified.**

**\* @param folderpath**

**\* @param filename**

**\* @param content**

**\* @return boolean**

**\*/**

**public static boolean createFiles(String folderpath,String filename,List<String> content)**

**{**

**try**

**{**

**File f1=new File(folderpath,filename);**

**FileWriter fw=new FileWriter(f1);**

**for(String s:content)**

**{**

**fw.write(s+"/n");**

**}**

**fw.close();**

**return true;**

**}**

**catch(Exception EX)**

**{**

**return false;**

**}**

**}**

**/\*\***

**\* This method will delete the filename if it exist.**

**\* @param folderpath**

**\* @param fileName**

**\* @return**

**\*/**

**public static boolean deleteFile(String folderpath, String fileName)**

**{**

**File file = new File(folderpath+"//"+fileName);**

**try**

**{**

**if(file.delete())**

**return true;**

**else**

**return false;**

**}**

**catch(Exception Ex)**

**{**

**return false;**

**}**

**}**

**/\*\***

**\* This method will search the file from the folder.**

**\* @param folderpath**

**\* @param fileName**

**\* @return**

**\*/**

**public static boolean searchFile(String folderpath, String fileName)**

**{**

**//adding folderpath with filename and creating file object**

**File file=new File(folderpath+"//"+fileName);**

**if(file.exists())**

**return true;**

**else**

**return false;**

**}**

**}**

**How to push the code to Git Hub 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**

**Conclusion:**

**Further enhancements to the application can be made which may include:**

* **Conditions to check if user is allowed to delete the file or add the file at the specific locations.**
* **Asking user to verify if they really want to delete the selected directory if it’s not empty.**
* **Retrieving files/folders by different criteria like Last Modified, Type, etc.**
* **Allowing user to append data to the file.**