|  |
| --- |
| **Lockedme.com**  **(Sprint work and Project Specification)** |

**Version History:**

|  |  |
| --- | --- |
| Author | Barkha Kaur |
| Purpose | Screenshots of the application |
| Date | 14thAugust 2021 |
| Version | 1.0 |

Contents

[1.Modules in the project 3](#_Toc79962781)

[2. Sprint wise Work 4](#_Toc79962782)

[3. Project GITHUBLINK link: 5](#_Toc79962783)

[4.Project Code : 6](#_Toc79962785)

# 1.Modules in the project

1. Display All Files
2. Add File
3. Delete File
4. Search File

# 2. Sprint wise Work

|  |  |
| --- | --- |
| Sprint Number | Modules |
| 1 | Display All Files  Add File |
| 2 | Delete File  Search File  Testing  Deployment (creating a jar file) |

# 3. Project GITHUBLINK link:

|  |
| --- |
| Repository Name : |
| FSD\_Java |
| Github link: |
| https://github.com/BarkhaKaur/FSD\_Java |

# 4.Project Code :

|  |
| --- |
| Folder Structure |
|  |
| FIleManager.java |
| **package** com.lockedme;  **import** java.io.File;  **import** java.io.FileWriter;  **import** java.util.ArrayList;  **import** java.util.Collections;  **import** java.util.List;  **public** **class** FileManager {  /\*\*  \* This method will return file names in ascending order from the folderPath  \* **@param** folderPath  \* **@return** List<String>  \*/  **public** **static** List<String> getAllFiles(String folderPath)  {  //Creating file object  File dir = **new** File(folderPath);    //Getting all the files into file array  File[] listOfFiles = dir.listFiles();    //Declare a list to store file names  List<String> fileNames = **new** ArrayList<String>();    **for**(File f:listOfFiles)  fileNames.add(f.getName());    //sort the files  Collections.*sort*(fileNames);    **return** fileNames;  }    /\*\*  \* This method will add or append the content into the specified file  \* **@param** folderPath  \* **@param** fileName  \* **@param** content  \* **@return** boolean  \*/    **public** **static** **boolean** addFiles(String folderPath,String fileName,List<String> content)  {  File file = **new** File(folderPath,fileName);    //To add or append content to File  **try**(FileWriter fileWriter = **new** FileWriter(file,**true**))  {  **for**(String s:content)  {  //Write content into the file  fileWriter.write(s + System.*lineSeparator*());  }  **return** **true**;  }  **catch**(Exception ex)  {  **return** **false**;  }  }    /\*\*  \* This method will delete the file if it exists  \* **@param** folderPath  \* **@param** fileName  \* **@return** boolean  \*/  **public** **static** **boolean** deleteFile(String folderPath,String fileName)  {  //adding folderpath with file name and creating file object  File file = **new** File(folderPath + "\\" + fileName);    // return true if delete is successful  **try**  {  **if**(file.delete())  **return** **true**;  **else**  **return** **false**;  }  **catch**(Exception ex)  {  **return** **false**;  }    }    /\*\*  \* This method will search file from the folder.  \* **@param** folderPath  \* **@param** fileName  \* **@return**  \*/  **public** **static** **boolean** searchFile(String folderPath,String fileName)  {  //adding folderPath with file name and creating file object  File file = **new** File(folderPath + "\\" + fileName);  //return true if file exists  **try**  {  **if**(file.exists())  **return** **true**;  **else**  **return** **false**;  }  **catch**(Exception ex)  {  **return** **false**;  }    }      } |
| LockedMeMain.java |
| package com.lockedme;  import java.util.ArrayList;  import java.util.List;  import java.util.Scanner;  public class LockedMeMain {  static final String FOLDER\_PATH = "C:\\Users\\Barkha\\Study\\Java\\simplilearn\_First\_Reddy\\Repo\\FSD\_Java\\MyPhase1Project\\LockedMeFiles";  public static void main(String[] args)  {  //Variable Declaration  Scanner scannerObj = new Scanner(System.in);  int choice = 0; //Variable for Main Menu choice  String seeChoice; // Variable to hold Y/N if user wants to see main menu again  int proceed = 1; //Variable to check if user wants to do more operations  int mainMenuProceed = 1; //variable to see if user wants to see the main menu again.  int invalidOuter = 0; //if user chooses other than numbers 1 to 5,set it to 1 to repeat the choices.    //Display menu for the first time.  displayMenu();  do  {  System.out.println("Enter your choice:");  /\* If user enters invalid option, the switch case will go to default value  \* If a non-number is entered by user,it will throw NumberFormatException  \*/  try  {  choice=Integer.parseInt(scannerObj.nextLine());  }  catch(NumberFormatException ne)  {  choice = 0;  }  //Anything other than numbers 1-5 is not valid so repeat the loop.  switch(choice)  {  case 1 :  getAllFiles();  //System.out.println("1");  break;  case 2 :  createFile(scannerObj);  //System.out.println("2");  break;  case 3 :  deleteFile(scannerObj);  //System.out.println("3");  break;  case 4 :  searchFile(scannerObj);  //System.out.println("4");  break;  case 5 :  proceed=0; //The outer do-while will only run if proceed==1  System.out.println("GoodBye.Thank you for using our Application.");  System.exit(0);  default :  invalidOuter = 1;  System.out.println("Invalid Option.Choose Numbers between 1 to 5.");    }  /\* If user chooses Invalid option then invalidOuter=1.  \* In that case, do not ask him if he wants to see the Menu again.  \*  \*This inner do-while loop is to cater for the case that user might choose wrong option.  \*Y y N n are all correct options.  \*  \*The inner do while will run till User chooses the correct option  \* \*/  if(invalidOuter==0)  {  do  {  System.out.println("Would you like to see the Menu again? :Y/N");  seeChoice = scannerObj.nextLine();  if(seeChoice.equalsIgnoreCase("Y"))  {  proceed = 1;  mainMenuProceed = 0;  displayMenu();  }  else if(seeChoice.equalsIgnoreCase("N"))  {  proceed = 0;//setting this to 0 will break out of the outer do-while  mainMenuProceed = 0;//setting this to 0 will break out of the inner do-while  System.out.println("GoodBye.Thank you for using our Application.");  }  else  {  System.out.println("Choose 'Y' or 'N'");  mainMenuProceed = 1;  }  }while(mainMenuProceed==1);  }  invalidOuter = 0;  }while(proceed==1);    scannerObj.close();  }    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 file");  System.out.println("3. Delete a file");  System.out.println("4. Search a file");  System.out.println("5. Exit");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");    }    /\*\*  \* This method gets all the files from the FOLDER\_PATH  \*/  public static void getAllFiles()  {  List<String> fileNames = FileManager.getAllFiles(FOLDER\_PATH);  for(String f:fileNames)  System.out.println(f);  }    /\*\*  \* This method takes file name,number of lines and content from user  \* to create the file  \*/  public static void createFile(Scanner scanner)  {  //Variable declaration  //Scanner scanner = 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=scanner.nextLine();    //Read number of lines from user  System.out.println("Enter how many lines in the file");  linesCount = Integer.parseInt(scanner.nextLine());    //Read lines from user  for (int i=1; i<=linesCount;i++)  {  System.out.println("Enter line "+i+":");  content.add(scanner.nextLine());  }    //Save the content into the file  boolean isSaved = FileManager.addFiles(FOLDER\_PATH, fileName, content);    if(isSaved)  System.out.println("File and Data saved successfully");  else  System.out.println("Some error occured.Please contact admin@lockedme.com");  //scanner.close();  }  /\*\*  \* This method is used to get file name from the user to delete that file.  \*/  public static void deleteFile(Scanner scannerObj)  {  //Code for deleting a file  String fileNameDel;  // Scanner scannerObj = new Scanner(System.in);  System.out.println("Enter file name to be deleted");  fileNameDel = scannerObj.nextLine();  boolean isDeleted = FileManager.deleteFile(FOLDER\_PATH, fileNameDel);  if(isDeleted)  System.out.println("File deleted successfully");  else  System.out.println("Unable to delete.Either file not there or some access issue.");  //scannerObj.close();  }    /\*\*  \* This method takes in a file name and lets user know if its present  \*/  public static void searchFile(Scanner scannerObj1)  {  //Code for searching a file  String fileNameSearch;  //Scanner scannerObj1 = new Scanner(System.in);  System.out.println("Enter file name to be searched");  fileNameSearch = scannerObj1.nextLine();  boolean isFound = FileManager.searchFile(FOLDER\_PATH, fileNameSearch);  if(isFound)  System.out.println("File is present in the folder ");  else  System.out.println("File is not present in the folder.");  //scannerObj1.close();  }      } |