

Screenshots:



The screenshot shows an IDE with the file `LockedMeMain.java` open. The code defines a `LockedMeMain` class with a `main` method. The `main` method calls `FileOperations.createMainFolderIfNotPresent("main")`, `MenuOptions.printWelcomeScreen("LockedMe", "Vamshi Krishna")`, and `HandleOptions.handleWelcomeScreenInput()`. The console output shows the application's welcome message and instructions.

```
1 package com.lockedme;
2
3 public class LockedMeMain {
4
5     public static void main(String[] args) {
6
7         // Create "main" folder if not present in current folder structure
8         FileOperations.createMainFolderIfNotPresent("main");
9
10        MenuOptions.printWelcomeScreen("LockedMe", "Vamshi Krishna");
11
12        HandleOptions.handleWelcomeScreenInput();
13    }
14
15 }
16
17
```

LockedMeMain [Java Application] C:\Program Files\Java\jdk-14\bin\javaw.exe (Apr 7, 2022, 6:36:05 PM)

** Welcome to LockedMe.com.

** This application was developed by Vamshi Krishna.

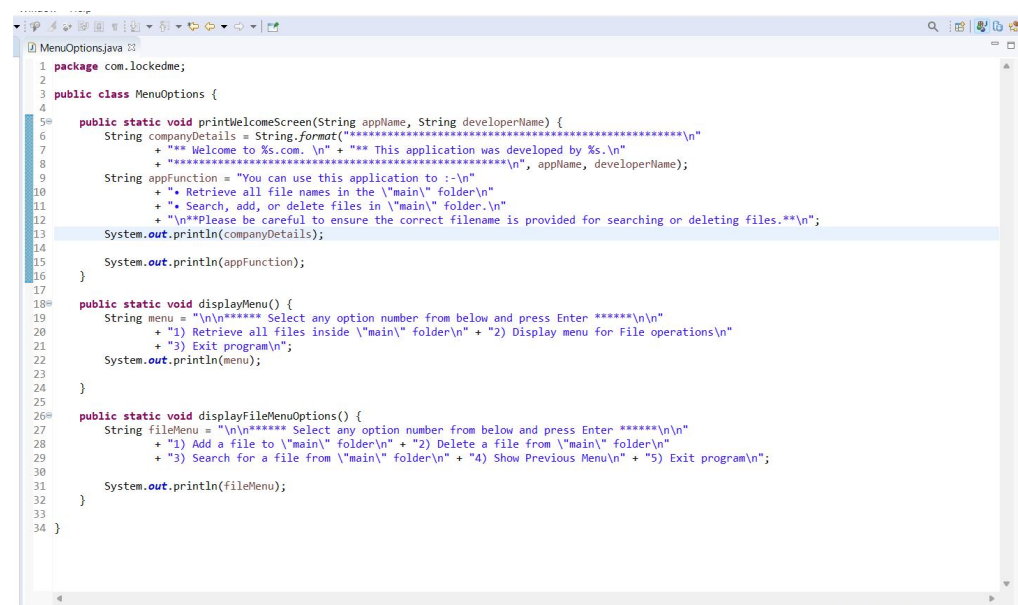
You can use this application to :-

- Retrieve all file names in the "main" folder
- Search, add, or delete files in "main" folder.

Please be careful to ensure the correct filename is provided for searching or deleting files.

***** Select any option number from below and press Enter *****

- 1) Retrieve all files inside "main" folder
- 2) Display menu for File operations
- 3) Exit program



The screenshot shows an IDE with the file `MenuOptions.java` open. The code defines a `MenuOptions` class with three static methods: `printWelcomeScreen`, `displayMenu`, and `displayFileMenuOptions`. The `printWelcomeScreen` method prints a welcome message and instructions. The `displayMenu` method prints a menu of options. The `displayFileMenuOptions` method prints a file menu.

```
1 package com.lockedme;
2
3 public class MenuOptions {
4
5     public static void printWelcomeScreen(String appName, String developerName) {
6         String companyDetails = String.format("*****\n"
7         + "Welcome to %s.com. \n" + "This application was developed by %s.\n"
8         + "*****\n", appName, developerName);
9
10        String appFunction = "You can use this application to :-\n"
11        + "• Retrieve all file names in the \"main\" folder\n"
12        + "• Search, add, or delete files in \"main\" folder.\n"
13        + "\n**Please be careful to ensure the correct filename is provided for searching or deleting files.**\n";
14
15        System.out.println(companyDetails);
16
17        System.out.println(appFunction);
18    }
19
20    public static void displayMenu() {
21        String menu = "\n***** Select any option number from below and press Enter *****\n\n"
22        + "1) Retrieve all files inside \"main\" folder\n" + "2) Display menu for File operations\n"
23        + "3) Exit program\n";
24        System.out.println(menu);
25    }
26
27    public static void displayFileMenuOptions() {
28        String fileMenu = "\n***** Select any option number from below and press Enter *****\n\n"
29        + "1) Add a file to \"main\" folder\n" + "2) Delete a file from \"main\" folder\n"
30        + "3) Search for a file from \"main\" folder\n" + "4) Show Previous Menu\n" + "5) Exit program\n";
31        System.out.println(fileMenu);
32    }
33
34 }
```

```
Window Help
HandleOptions.java
1 package com.lockedme;
2
3 import java.util.List;
4
5
6 public class HandleOptions {
7     public static void handleWelcomeScreenInput() {
8         boolean running = true;
9         Scanner sc = new Scanner(System.in);
10        do {
11            try {
12                MenuOptions.displayMenu();
13                int input = sc.nextInt();
14
15                switch (input) {
16                    case 1:
17                        FileOperations.displayAllFiles("main");
18                        break;
19                    case 2:
20                        HandleOptions.handleFileMenuOptions();
21                        break;
22                    case 3:
23                        System.out.println("Program exited successfully.");
24                        running = false;
25                        sc.close();
26                        System.exit(0);
27                        break;
28                    default:
29                        System.out.println("Please select a valid option from above.");
30                }
31            } catch (Exception e) {
32                System.out.println(e.getClass().getName());
33                handleWelcomeScreenInput();
34            }
35        } while (running == true);
36    }
37
38    public static void handleFileMenuOptions() {
39        boolean running = true;
```

```
HandleOptions.java Speaking: Nikunj [Faculty]
37
38    public static void handleFileMenuOptions() {
39        boolean running = true;
40        Scanner sc = new Scanner(System.in);
41        do {
42            try {
43                MenuOptions.displayFileMenuOptions();
44                FileOperations.createMainFolderIfNotPresent("main");
45
46                int input = sc.nextInt();
47                switch (input) {
48                    case 1:
49                        // File Add
50                        System.out.println("Enter the name of the file to be added to the \"main\" folder");
51                        String fileToAdd = sc.next();
52                        FileOperations.createFile(fileToAdd, sc);
53
54                        break;
55                    case 2:
56                        // File/Folder delete
57                        System.out.println("Enter the name of the file to be deleted from \"main\" folder");
58                        String fileToDelete = sc.next();
59
60                        FileOperations.createMainFolderIfNotPresent("main");
61                        List<String> filesToDelete = FileOperations.displayFileLocations(fileToDelete, "main");
62
63                        String deletionPrompt = "\nSelect index of which file to delete?"
64                            + "\nEnter 0 if you want to delete all elements";
65                        System.out.println(deletionPrompt);
66
67                        int idx = sc.nextInt();
68
69                        if (idx != 0) {
70                            FileOperations.deleteFileRecursively(filesToDelete.get(idx - 1));
71                        } else {
72                            // If idx == 0, delete all files displayed for the name
```

```
HandleOptions.java
74                // If idx == 0, delete all files displayed for the name
75                for (String path : filesToDelete) {
76                    FileOperations.deleteFileRecursively(path);
77                }
78            }
79
80            break;
81        case 3:
82            // file/Folder Search
83            System.out.println("Enter the name of the file to be searched from \"main\" folder");
84            String fileName = sc.next();
85
86            FileOperations.createMainFolderIfNotPresent("main");
87            FileOperations.displayFileLocations(fileName, "main");
88
89            break;
90        case 4:
91            // Go to Previous menu
92            return;
93        case 5:
94            // Exit
95            System.out.println("Program exited successfully.");
96            running = false;
97            sc.close();
98            System.exit(0);
99
100        default:
101            System.out.println("Please select a valid option from above.");
102        }
103    } catch (Exception e) {
104        System.out.println(e.getClass().getName());
105        handleFileMenuOptions();
106    }
107 } while (running == true);
108 }
109 }
110 }
111 }
```

```

1 package com.lockedme;
2
3 import java.io.File;
4
5 public class FileOperations {
6
7     public static void createMainFolderIfNotPresent(String folderName) {
8         File file = new File(folderName);
9
10        // If file doesn't exist, create the main folder
11        if (!file.exists()) {
12            file.mkdirs();
13        }
14    }
15
16    public static void displayAllFiles(String path) {
17        FileOperations.createMainFolderIfNotPresent("main");
18        // All required files and folders inside "main" folder relative to current
19        // folder
20        System.out.println("Displaying all files with directory structure in ascending order\n");
21
22        // listFilesInDirectory displays files along with folder structure
23        List<String> fileListNames = FileOperations.listFilesInDirectory(path, 0, new ArrayList<>());
24
25        System.out.println("Displaying all files in ascending order\n");
26        Collections.sort(fileListNames);
27
28        fileListNames.stream().forEach(System.out::println);
29    }
30
31    public static List<String> listFilesInDirectory(String path, int indentationCount, List<String> fileListNames) {
32        File dir = new File(path);
33        File[] files = dir.listFiles();
34        List<File> fileList = Arrays.asList(files);
35
36        Collections.sort(fileList);
37
38        if (files != null && files.length > 0) {
39
40

```

```

41
42            for (File file : fileList) {
43
44                System.out.print(" ".repeat(indentationCount * 2));
45
46                if (file.isDirectory()) {
47                    System.out.println("-- " + file.getName());
48
49                    // Recursively indent and display the files
50                    fileListNames.add(file.getName());
51                    listFilesInDirectory(file.getAbsolutePath(), indentationCount + 1, fileListNames);
52                } else {
53                    System.out.println("-- " + file.getName());
54                    fileListNames.add(file.getName());
55                }
56            }
57        } else {
58            System.out.print(" ".repeat(indentationCount * 2));
59            System.out.println("|-- Empty Directory");
60        }
61        System.out.println();
62        return fileListNames;
63    }
64
65    public static void createFile(String fileToAdd, Scanner sc) {
66        FileOperations.createMainFolderIfNotPresent("main");
67        Path pathToFile = Paths.get("./main/" + fileToAdd);
68        try {
69            Files.createDirectories(pathToFile.getParent());
70            Files.createFile(pathToFile);
71            System.out.println(fileToAdd + " created successfully");
72
73            System.out.println("Would you like to add some content to the file? (Y/N)");
74            String choice = sc.next().toLowerCase();
75
76            sc.nextLine();
77            if (choice.equals("y")) {
78                System.out.println("\n\nInput content and press enter\n");
79
80

```

```

81
82        public static void searchFileRecursively(String path, String fileName, List<String> fileListNames) {
83            File dir = new File(path);
84            File[] files = dir.listFiles();
85            List<File> fileList = Arrays.asList(files);
86
87            if (files != null && files.length > 0) {
88                for (File file : fileList) {
89
90                    if (file.getName().startsWith(fileName)) {
91                        fileListNames.add(file.getAbsolutePath());
92                    }
93
94                    // Need to search in directories separately to ensure all files of required
95                    // fileName are searched
96                    if (file.isDirectory()) {
97                        searchFileRecursively(file.getAbsolutePath(), fileName, fileListNames);
98                    }
99                }
100            }
101        }
102
103        public static void deleteFileRecursively(String path) {
104            File currFile = new File(path);
105            File[] files = currFile.listFiles();
106
107            if (files != null && files.length > 0) {
108                for (File file : files) {
109
110                    String fileName = file.getName() + " at " + file.getParent();
111                    if (file.isDirectory()) {
112                        deleteFileRecursively(file.getAbsolutePath());
113                    }
114
115                    if (file.delete()) {
116                        System.out.println(fileName + " deleted successfully");
117                    } else {
118                        System.out.println("Failed to delete " + fileName);
119                    }
120                }
121            }
122        }
123    }
124 }

```

