Functional Specification Document

# 1.Project background:

# 

The Company Lockers Pvt.Ltd aim to digitize their products and have choosed LockedMe.com as their first project. So a prototype of MVP model has been made to demonstrate them and further the business with them.

# 2. Author details:

Name: Dhruva Kumar Renuka

Email:dhruvaveeresh12@gmail.com

# 3. Project description:

3.1 As per the requirements a welcome message and author’s name, a menu driven application is provided to use the service of LockedMe.com.

|  |
| --- |
| 1: Returns current file names in ascending order  2: Takes you to File management service  3: Exits the application |

It promotes the user to enter a choice. Until the user enters the specified choice, the application keeps promoting for correct choice.

When the user choice is ‘1’ the files are returned in ascending order.

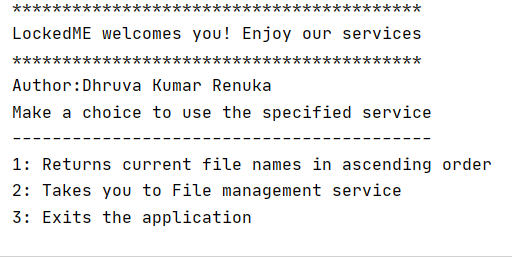
3.2 When user choice is ‘2’, application will take you to File Management Service. Where you can choose for a file management service among the 4 mentioned options.

|  |
| --- |
| ====================================================  You just entered a File Management Service. Happy to serve you!!  ====================================================  Press 1: To add a new file  Press 2: To delete a file  Press 3: To search for a file  Press 4: To go back to the previous Menu |

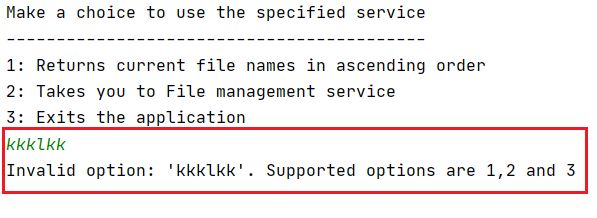
3.3 When user choice is ‘3’ application exits.

# 4. Workflow:

4.1 When application starts a welcome message, the author name and Navigation menu is displayed as shown in *fig 1.1*.

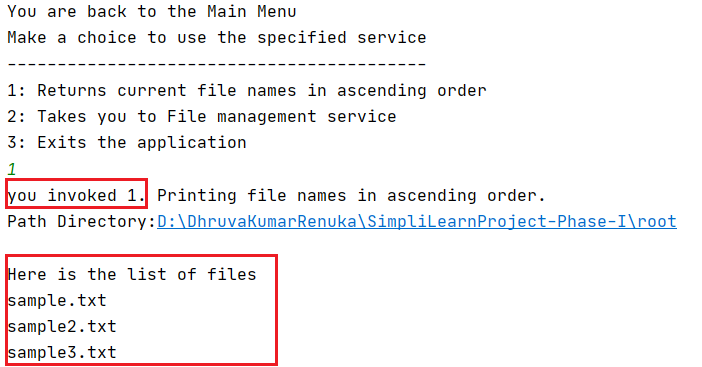
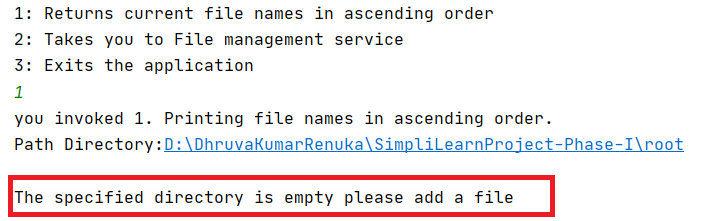


*fig1.1*

If an invalid option is pressed other than 1,2 or 3 it’ll prompt for input again as shown in *fig1.2*.

*fig1*.*2*

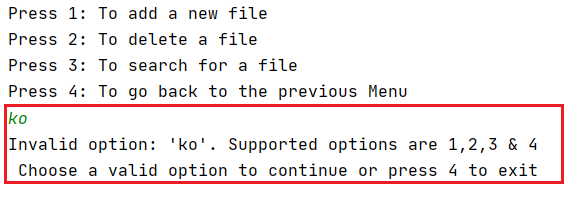
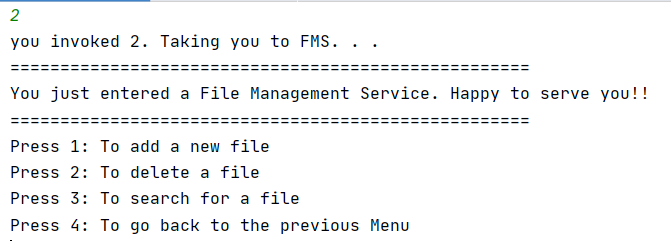
4.2 After getting valid input say ‘1’ the files will be printed in Ascending order. If the folder is empty then a message %%% ***The specified directory is empty please add a file***%%% as shown in *fig 1.3*. If there are files then it will be displayed in Alphabetical order. Here ***quicksort*** is used to sort the files *fig1.4*.



*fig 1.3* *fig 1.4*

Now again the application will ask for an input. If the user choice is ‘2’ It’ll take him to FileManagementService.

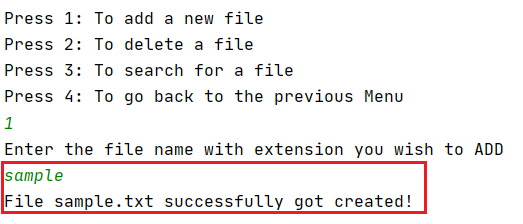
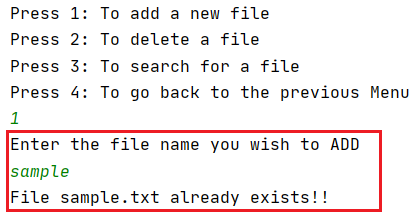
4.3 File Management Service

Below *fig1.5* shows the File Management Service and *fig1.6* shows that the user will be prompted for a choice until his choice is either 1,2,3 or 4.

*fig1.5* *fig1.6*

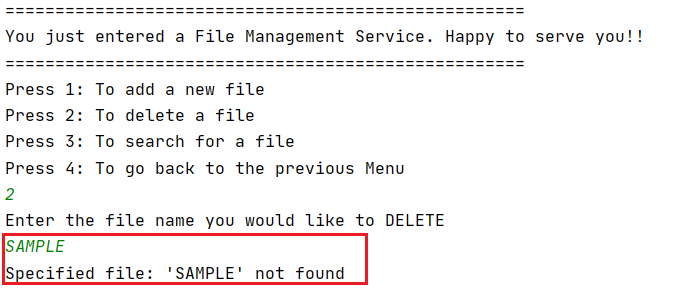
4.3.1 Adding of a file:

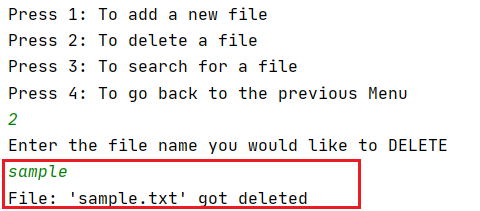
In the current version only .txt files can be added. The user can specify the filename after choosing the option ‘1’ as shown in *fig 1.7*. Once the user specifies the filename a file with .txt extension will be created in the ‘**root**’ folder if at all the file is not existing in the folder. A success message appears ***\*\*File <filename.txt> successfully got created\*\****. If user tries to create a filename whose file already exists then creation of new file with the same name is forbidden and a following message appears ***\*\*File <filename.txt> already exists!!\*\**** *fig1.8*.



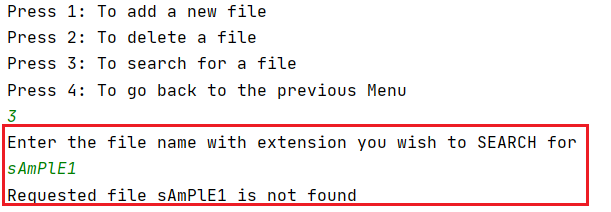
*fig1.7*  *fig1.8*

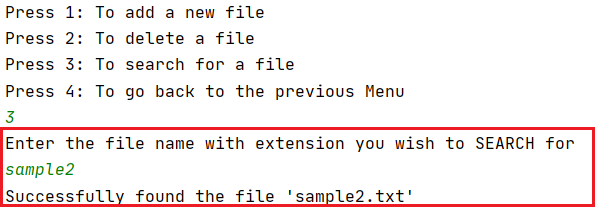
4.3.2 Delete a file:

If a user wishes to delete a file, he can specify the filename as it is in the folder so that exact file gets deleted. He can does this by choosing the option ‘2’ in FMS. Here the filename is case sensitive. If a file doesn’t exists in the root folder then ***\*\*Specified file: <filename> not found\*\**** message appears *fig 1.10*. If the file exists and case is matched then file gets deleted and a confirmation message ***\*\*File: <filename.txt> got deleted\*\**** *fig 1.9*.

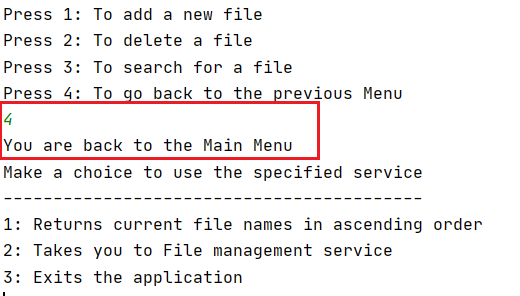
 *fig1.9*  *fig1.10*

4.3.3 Search a file:

The user can search a file using the FMS option ‘3’. When user selects for Search operation, the user has to specify the file name. If the specified filename matches with the files in the root folder, the message appears like ***\*\*successfully found the file <filename.txt>\*\**** *fig1.11*else if there is no file with that name in the root folder an unsuccessful message appears ***\*\*Requested file <filename> not found\*\**** *fig 1.12*.

 *fig1.11* *fig1.12*

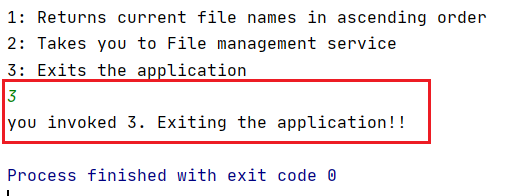
4.3.4 Back to Main Navigation:

If the user choice is ‘4’ application takes the user back to the main navigation menu *fig1.13.*

*fig1.13*

5. Exit application:

When the user is in the main navigation menu, user can choose option ‘3’ to exit the application. Finally once the user wish to exit by choosing the correct exit option an exit message is displayed ***\*\*you invoked 3. Exiting the application\*\**** *fig 1.14*.



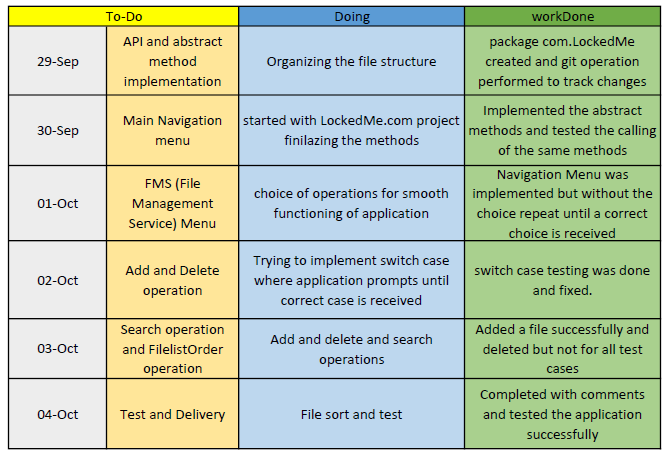
*fig 1.14*

6. Methodology:

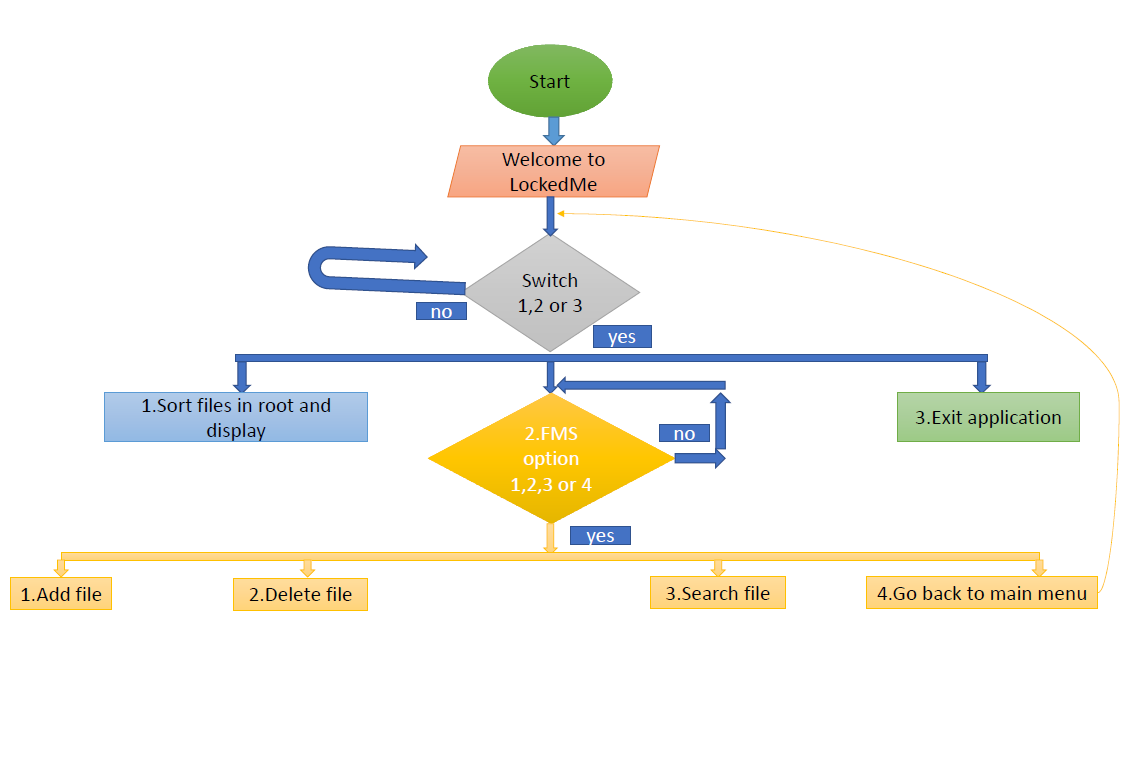
The agile scrum methodology is used. Accordingly the project work was split into 2 sprints each sprint has had 3 days and below is the task board attached.

Number of Sprints------------->2

Duration of Each Sprint------>3days



7. Flowchart:



8. Conclusion of project:

The project source code is in the package com.LockedMe. The businesss logic is implemented in the UserFile class and UserFile implements FileAPI interface where the signature of the methods are declared. The FileManagement class is the one in which has navigation menu logic.

So the projects manages the files in a single folder called root. A user can add .txt file or delete .txt file or search .txt file and sort the files according to their names. Since this is prototype, only limited applications have been added to meet a Minimum Viable Product specifications.

9. USP:

* The application can be extended to all sorts of documents.
* Each user can have separate repository to store his documents.
* Sub classification can be done in the next versions.
* User security can be added by prompting the user to enter password.

10. GitHub Link:

https://github.com/DhruvaRenuka/Phase1-project-simplilearn.git