**LockedMe – Virtual Key for Repositories**

CONTENT -

* [Project And Developer Details](#Project_And_Developer_Details)
* [Sprint planning and Task completion](#Sprint_plan)
* [Core concepts used in project](#Core_concepts)
* [Algorithm used for the Application](#Algorithm_used_for_the_Application)
* [Flow of the Application](#Flow).
* [Pushing Code to GitHub repository](#Pushing_code_to_GitHub_repository)
* [Unique Selling Points of the Application](#USP)
* [Conclusions](#Conclusions)

## Project And Developer Details –

This project is developed by Richa Agrawal. The project is hosted on GitHub - <https://github.com/rookie-coderr/virtualKeyRepo>

LockedMe.Com is a virtual repository to store, retrieve and delete your files from a directory seamlessly. Below are the details of the developed prototype of the application. The prototype of the application will be having following features –

* Retrieving the file names in an ascending order
* Business-level operations:
  + Option to add a user specified file to the application
  + Option to delete a user specified file from the application
  + Option to search a user specified file from the application
  + Navigation option to close the current execution context and return to the main context
* Option to close the application

## Sprints planning and Task completion –

The project is planned to be completed in 1 sprint. 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.

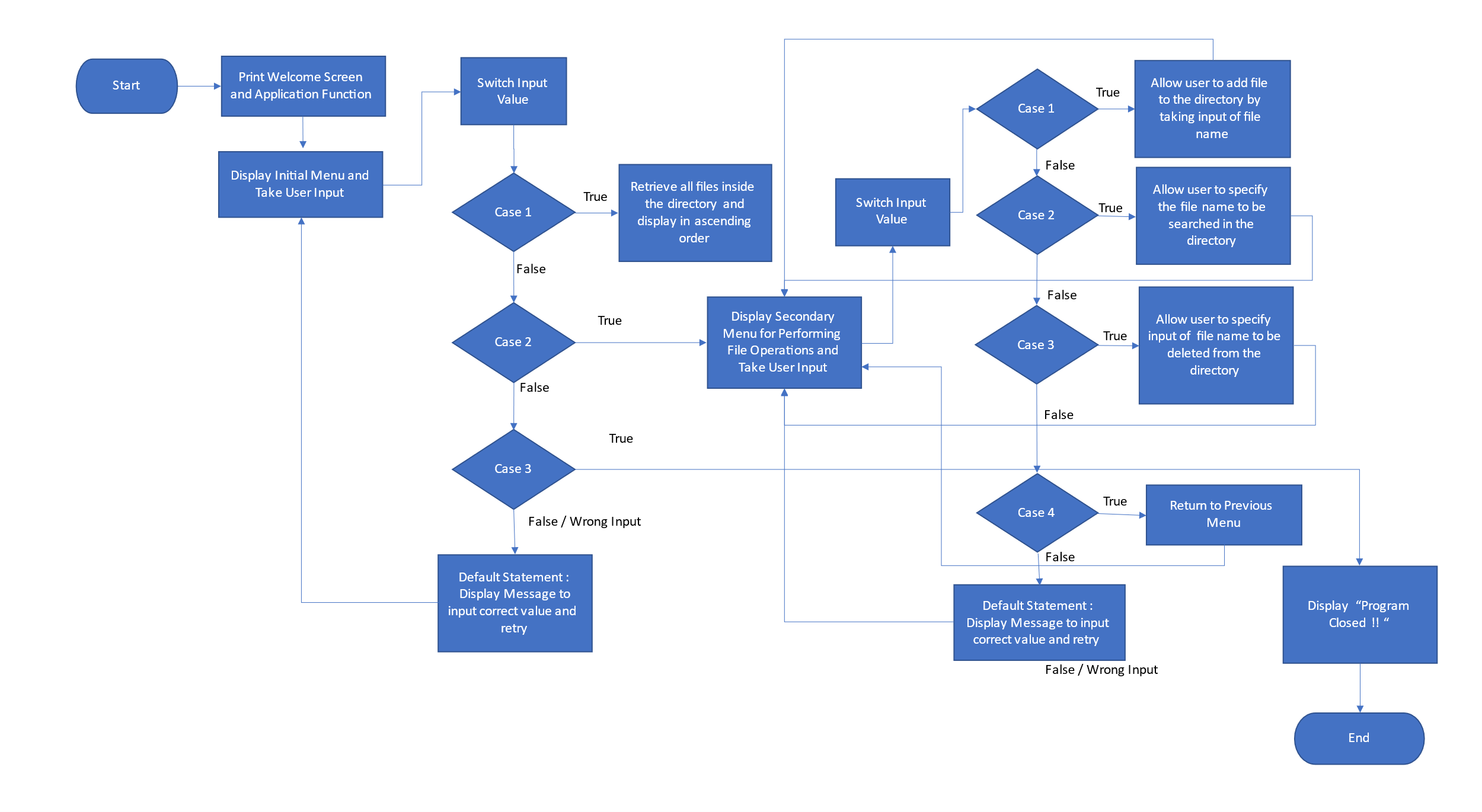
## Core concepts used in project -

* Collection Framework
* File Handling
* Searching & Sorting
* Flow Control
* Exception Handling
* Streams API

ALGORITHM USED FOR THE APPLICTION–

* Step 1: Start
* Step 2: Display the main Screen and take user input.
* Step 3: Switch input value
  + Case 1: Call showFiles method to display files in ascending order.
  + Case 2: Call moreOptions method from FileOperations class.
    - Step 1: Call displayOptions to display submenu , take user input and return the value to moreOptions.
    - Step 2: Switch input values.
      * Case 1: Call addFiles method to allow user to add files in the directory .
        + If addition successful print “File added” Else print “enter valid input”
        + Call moreOptions.
      * Case 2: Call searchFiles method to allow the user to search file in the directory
        + If file found – print the location of file Else print “File not found
        + Call moreOptions.
      * Case 3: Call deleteFiles method to allow user to delet file from the directory
        + If file deletion successful – print “File deleted successfully else print “File not found”
        + Call moreOptions
      * Case 4: Call welcomeScreen and break;
  + Case 3: Print “Program closed !!” and terminate.

## Flow of the Application



## 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 origin main**

## Unique Selling Points of the Application

1. The application is designed to keep on running and taking user inputs even after exceptions occur. To terminate the application, appropriate option needs to be selected.
2. The user can easily change the source directory of searching just by changing a variable value in the source code.
3. User is also provided the option to write content if they want into the newly created file.
4. The application also allows user to delete files which are not empty.
5. The user can seamlessly switch between options or return to previous menu even after any required operation like adding, searching, deleting or retrieving of files is performed.
6. The application is designed with modularity in mind. Even if one wants to update the path, they can change it through the source code. Application has been developed keeping in mind that there should be very less “hardcoding” of data.

## Conclusions

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.