Specification Document

We are building a command line application for LockedMe.com. This release will serve as the prototype of the File Store application.

The following capabilities will be present in the application:

1. A user will be able to list all stored text files in their current working directory in alphabetical order
2. A user will be able to add a text file in their current working directory
3. A user will be able to delete a text file in their current working directory
4. A user will be search whether a particular text file exists in their working directory

In order to complete these user stories, a user will interact with a command line interface that initially presents them with a Menu, the user can either enter the Sub-Menu to be presented with options they can choose or they can exit the program.

If they choose to enter the Sub-Menu, they will be able to select options that are detailed within the user stories. They can continue to operate the application until they choose to exit, where they will return to the main Menu and then decide to exit the application.

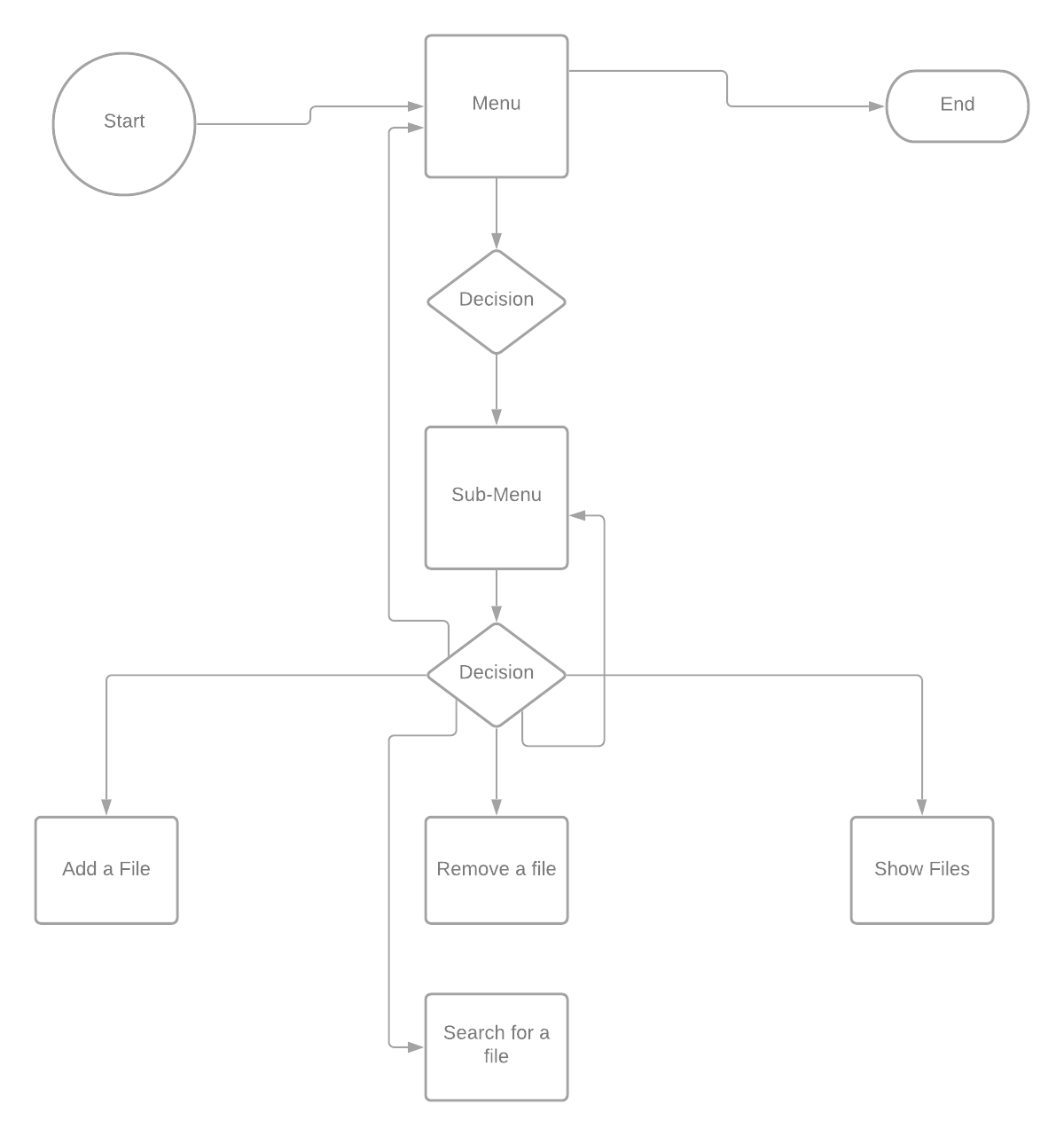
# Sprints

A total of three sprints will be involved in rolling out the product

1. First Sprint
   1. We will develop the Menu class
      1. We will display a greeting
      2. We will display the choices
   2. We will develop the FileClass class
      1. Here we will stub out the functions of dealing with Strings rather than Files just to get a blueprint of the functionality around dealing with File objects
         1. Stub out an add method
         2. Stub out a remove method
         3. Hard code a list of static files
         4. Use the Collections API to sort, and use the comparator method String.CASE\_INSENSITIVE\_ORDER to retrieve the placeholder strings to retrieve them in ascending order
      2. Test the Menu and FileClass methods
2. Second Sprint
   1. We will further develop the Menu class by introducing a sub-menu context and have the functions hold the logic for operational switching between different menu contexts
   2. In the FileClass we will introduce File Objects into the program so that we can deal with real files
   3. Added a method to sync content of directory with our ArrayList data structure
   4. Test the added functionality
3. Third Sprint
   1. Add logic to enforce only files with “.txt” can be added and validations of what can be returned from the current directory
   2. Have both Menu and FileClass implemented from interfaces to keep methodology the same for future projects
   3. Create custom exceptions to handle invalid inputs, duplicate files, or file not in current directory as to not interrupt program flow
   4. Test the added validations and exceptions

In summary, the concepts used in this project are core Java, Collections API, Exceptions, and package structure.

# Flow Diagram



# Screen Shots

To demonstrate the above flow diagrams. We will walk through screenshots of the program. You will be greeted by the main menu.

|  |
| --- |
|  |

You can type ‘Y’ to continue to the submenu context, or ‘N' to exit the application. Continuing with the application you will see a list of choices:

|  |
| --- |
|  |

We can add a file by selecting option 2:

|  |
| --- |
|  |

Once you’ve entered a valid text file, it will be added to your current working directory and a confirmation message will be on screen. We will add another file selecting option 2 again and we will add an employees.txt file. We will now select option 1 to see the files in the current working directory:

|  |
| --- |
|  |

You will notice that your files appear in ascending order. Let’s now delete a file by selecting option 3:

|  |
| --- |
|  |

Once you provide a valid text file to delete you will receive a confirmation as per the above. To exit out of the current context select option 4 and to exit the application select the ‘N’ option:

|  |
| --- |
|  |

# Source Code

The application was divided into four packages: application; business.interfaces; business.logic; and exceptions.

|  |
| --- |
| **package** application;  **import** business.logic.Menu;  **import** exceptions.InvalidInputException;  **public** **class** FileStore {  **public** **static** **void** main(String[] args) {  Menu menu = **new** Menu();  **try** {  menu.showMainMenu();  } **catch** (InvalidInputException e) {  System.***err***.println(e.getMessage());  }  }  } |

|  |
| --- |
| **package** business.interfaces;  **import** exceptions.InvalidInputException;  **public** **interface** MenuInterface {    **public** **void** showMainMenu() **throws** InvalidInputException;  **public** **void** showSubMenu();  } |

|  |
| --- |
| package business.interfaces;  import java.io.IOException;  import exceptions.DuplicateFileException;  import exceptions.FileNotInDirectoryException;  public interface FileInterface {  public void showFileList();  public void remove() throws FileNotInDirectoryException;  public void add() throws DuplicateFileException, IOException;  } |

|  |
| --- |
| package business.logic;  import java.io.IOException;  import java.util.Scanner;  import business.interfaces.\*;  import exceptions.DuplicateFileException;  import exceptions.FileNotInDirectoryException;  import exceptions.InvalidInputException;  public class Menu implements MenuInterface {  public void showMainMenu() throws InvalidInputException {  Scanner scan = new Scanner(System.in);  System.out.println("\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\tWelcome to Company Locker Pvt Ltd...\n\n");  System.out.println("\tThis program allows you to safely store your text documents."  + "\n\tYou'll be able to list out your files in alphabetical order,\n"  + "\tand delete any text files that are no longer needed...\n\n");  System.out.println("\tWould you like to enter the application Y/N?");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\t\t\t\t\t--developed by Altaf Quadri");  System.out.println("\n Please enter your selection:\t");  String enter = scan.nextLine().toLowerCase();  try {  if (enter.equals("y")) {  showSubMenu();  } else if (enter.equals("n")) {  System.out.println("Thanks for visiting Company Locker Pvt Ltd...");  } else {  throw new InvalidInputException("Please a select valid choice!");  }  } catch (InvalidInputException e) {  System.err.println(e.getMessage());  showMainMenu();  }  }  public void showSubMenu() {  Scanner scan = new Scanner(System.in);  int counter = 0;  while (counter == 0) {  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\n\tPlease press (1) to show files for the system\n");  System.out.println("\tPlease press (2) to add a file\n");  System.out.println("\tPlease press (3) to delete a file\n");  System.out.println("\tPlease press (4) to return to main menu\n");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\nPlease provide your entry:\t");  String choice = scan.nextLine();  FileClass myFile = new FileClass();  try {  if (choice.equals("1")) {  myFile.showFileList();  }  else if (choice.equals("2")) {  try {  myFile.add();  } catch (DuplicateFileException e) {  System.err.println(e.getMessage());  } catch (IOException e) {  System.err.println("You need to input a filename!");  }  }  else if (choice.equals("3")) {  try {  myFile.remove();  } catch (FileNotInDirectoryException e) {  System.err.println(e.getMessage());  }  }  else if (choice.equals("4")) {  counter++;  } else {  throw new InvalidInputException("Please select a valid option from 1-4");  }  } catch (InvalidInputException e) {  System.err.println(e.getMessage());  }  } // end while  try {  showMainMenu();  } catch (InvalidInputException e) {  System.err.println(e.getMessage());  e.printStackTrace();  }  }  } |

|  |
| --- |
| package business.logic;  import java.io.File;  import java.io.IOException;  import java.util.ArrayList;  import java.util.Arrays;  import java.util.Collections;  import java.util.List;  import java.util.Scanner;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  import java.util.stream.Collectors;  import business.interfaces.\*;  import exceptions.DuplicateFileException;  import exceptions.FileNotInDirectoryException;  import exceptions.InvalidInputException;  public class FileClass implements FileInterface {  public static List<String> fileList = new ArrayList<>();  public void syncLists() {  List<String> pathnames = new ArrayList<>();  File f = new File("./");  pathnames = Arrays.asList(f.list());  pathnames = pathnames.stream().filter(p -> p.endsWith(".txt")).collect(Collectors.toList());  fileList.clear();  fileList.addAll(pathnames);  }  public void showFileList() {  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  syncLists();  if (fileList.isEmpty())  System.out.println("The current directory does not contain any files");  Collections.sort(fileList, String.CASE\_INSENSITIVE\_ORDER);  System.out.println("###################################################################");  if (!fileList.isEmpty()) System.out.println("Your files:");  System.out.println("-------------------------------------------------------------------");  fileList.forEach(file -> System.out.println(file));  System.out.println("###################################################################");  }  public void add() throws DuplicateFileException, IOException {  Scanner scan = new Scanner(System.in);  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\nPlease enter the name of the text file you would like to add:");      String fileName = scan.nextLine();    String regex = ".\*\\.txt$";  Pattern p = Pattern.compile(regex);  Matcher matcher = p.matcher(fileName);  boolean found = matcher.matches();    try {  if (!found) {  throw new InvalidInputException("File names must end in .txt");  }  } catch (InvalidInputException e) {  System.err.println(e.getMessage());  while (!found) {  System.out.println("Please enter a file ending with (.txt)!");  fileName = scan.nextLine();  matcher = p.matcher(fileName);  found = matcher.matches();  }  }    File myFile = new File(fileName);  //catch  if (myFile.createNewFile()) {  fileList.add(fileName);  System.out.println("The " + fileName + " file has successfully been added");  } else {  throw new DuplicateFileException("This file already exists!");  }  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  }  public void remove() throws FileNotInDirectoryException {  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  Scanner scan = new Scanner(System.in);  System.out.println("Please enter the name of the text file you would like to delete:");  String fileName = scan.nextLine();    String regex = ".\*\\.txt$";  Pattern p = Pattern.compile(regex);  Matcher matcher = p.matcher(fileName);  boolean found = matcher.matches();    try {  if (!found) {  throw new InvalidInputException("File names must end in .txt");  }  } catch (InvalidInputException e) {  System.err.println(e.getMessage());  while (!found) {  System.out.println("Please enter a file ending with.txt");  fileName = scan.nextLine();  matcher = p.matcher(fileName);  found = matcher.matches();  }  }    File myFile = new File(fileName);  if (myFile.delete()) {  syncLists();  System.out.println("The " + fileName + " was deleted!");  } else {  throw new FileNotInDirectoryException("The file does not exist in the current directory!");  }  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  }  } |

|  |
| --- |
| **package** exceptions;  **public** **class** InvalidInputException **extends** Exception {  **public** InvalidInputException(String message) {  **super**(message);  }  } |

|  |
| --- |
| **package** exceptions;  **public** **class** FileNotInDirectoryException **extends** Exception {  **public** FileNotInDirectoryException(String message) {  **super**(message);  }  } |

|  |
| --- |
| **package** exceptions;  **public** **class** DuplicateFileException **extends** Exception{  **public** DuplicateFileException(String message) {  **super**(message);  }  } |

This command line project can be further enhanced by adding functionality to edit the contents of the text files that we have stored. At a later phase this can be developed into a web application that can be accessed from anywhere where the files are stored in a database for retrieval rather than the user directory.