# Virtual Key for Your Repositories (PHASE 1 Project 1 JSD)

# [By ANUSHKA KAMATH]

PROJECT DOCUMENTATION

Github link: <https://github.com/anushkakamath/simplilearn_bnp_project.git>

**The LockedMe App prototype as required by the client completes the following the basic tasks:** (Product’s capabilities, appearance, and user interactions)

1. Displays all the files in the specified directory in ascending order.
2. Adds a user specified file to the root directory with the user’s choice of contents.
3. Deletes a user specified file from the root directory.
4. Searches a user specified file from the root directory and gives it details.
5. Option to exit the app.

(Also contains Validation of user and other operations as a part of non-functional requirements for the project)

**Details on each of the functional operations of the app:**

The first thing we encounter on running this program is a main Menu displaying the following operations that the user may choose from:

1. Retrieve all the files in the directory

2. Business Level Operations Menu

3. Exit from the App

On clicking choice number 2 it takes user to a new Business Level Operations Menu which displays a set of new choices for the user to choose from namely:

1. Add a file

2. Delete an existing file

3. Search file by the name

4. Exit the Business Level Operations Menu

More details on these operations are as given below:

1. **Displays all the files in the specified directory:** The getAllFiles() method of class AppBoImpl returns all the files in the specified directory in ascending order of their file names. Firstly, this method contains the path to the specified directory. This method then adds all the files into a List created by Arrays implementation, which further uses Collections.sort() to sort the files in ascending order to return the list of all the files in that directory. Thus, it retrieves the files in this directory in ascending sorted order.
2. **Business Level Operations Menu:** This option reveals a new menu showing a set of different operations a user may choose from. These being Business Level Operations have been given an additional level of security that allows only admins and resisted users to use these operations. It asks the user to enter the user name and uses regex in AppBOImpl (App Business Objectives Implementation class) to validate users in absence of a database. The authorization is only given to users who have usernames “admin” lower case only followed by at most 3 digits (eg. admin, admin342).
   1. **Add a file to the root directory:** This option provides authorized users to add a file to the root directory with file name of choice and their own contents. In case of failing the validation, an error message is displayed informing the user that he needs access to perform this operation. If the user tries to create a file with the name that already exists, it displays an error message saying “File already exists”. The method createFiles() in AppBOImpl performs this operation. It takes input the User constructor and username entered for validation.
   2. **Delete an existing file from root directory:** This option provides authorized users access to delete files from root directory by specifying the file name. In case of failing the validation, an error message is displayed informing the user that he needs access to perform this operation. In a scenario that user chooses a file to delete that does that exist in the directory it returns a file not found error. The method removeFileByName() in AppBOImpl performs deletion operation. It takes as input username and the file to be deleted
   3. **Search a file by filename from the root directory:** This option allows the authorised users to search a file by specifying the file name from the root directory. In case of failing the validation, an error message is displayed informing the user that he needs access to perform this operation. On successful validation, the user needs to enter the file name to view the size, other file details. The method getFilesByName() in AppBOImpl performs the search operation.
   4. **Exit Business Level Operations:** On clicking this choice user is taken back to the app main menu.
3. **Exit :** This choice exits the app and displays a Thank you message.

**Details on the different packages in the code:**

1. **Package** com.app.model

This package contains a class called User.java that is mainly used to build an app model. It takes the filename, contents in it. This class contains getter, setter methods, constructor, etc.

1. **Package** com.app.bo

This package contains an abstract layer i.e. an AppBO interface that specifies all the business objectives of the app that need to later implemented.

1. **Package** com.app.bo.impl

This package contains a class AppBOImpl that implements the AppBO interface and contains the implementation of all the operation methods mentioned in the AppBO interface. Apart from the implementation of various overridden methods, it contains a isValidName method that checks for the validity of the username. The implementation of the abstract class shows the OOP pillar of inheritance as well. Thus implemented the appropriate concepts such as exceptions, collections, and sorting techniques within the source code.

1. **Package** com.app.exception

This package contains an exception class called BusinessException for handling all business-related exceptions occurring in the app. The BuisnessException handles the various checked expressions occurring in the app showing Exception Handling.

1. **Package** com.app.main

This package contains main class AppMain that takes the required inputs from the user and calls the relevant methods.

This Project uses various Java Concepts like: Inheritance (using interface), Encapsulation (using getter and setter methods), Polymorphism (using overridden methods), Abstraction, Collections, Data structures like Arrays List, etc.

Technologies used in the project:

* Eclipse/Spring Tool Suite: An IDE to code for the application
* Java: A programming language to develop the prototype (Java8)
* Git: To connect and push files from the local system to GitHub
* GitHub: To store the application code and track its versions
* Scrum: An efficient agile framework to deliver the product incrementally
* Search and Sort techniques: Used List, Arrays implementation for searching and storing the files. Collections. Sort used to sort the files in the root directory.
* Specification document: Microsoft Word for documentation

Sprint: Dividing tasks/goals into subtasks called sprints.

* The first sprint consisted of the tasks / achievable like the main functional requirements of the project like adding the file to a directory, deleting a file from the directory, searching a file from the directory by name specified by user, and displaying list of all the files in the directory sorting then in ascending order.
* The second sprint consisted of some performance and code optimisation to add some non-functional requirements like adding validation of user to person Business Level Operations. This is done by validating the username to check if he has access. Due to absence of database, regex is used to find users whose username starts with “admin” followed by zero to three digits at max. Other code optimisations were done in this sprint.

Post completion of the sprints the project was successfully released to its users. The goal of the company to deliver a high-end quality product as early as possible was achieved. 

CONCLUSION

The project was completed by meeting all the functional requirements i.e. all the operations mentioned successfully fulfilling the tasks is the main USP (Unique Selling Point) of the project. The further validation of users by checking username gives security by not allowing everyone access to the file details. The app does not show contents of any file thus maintain privacy and security constraints. The app is thus a first prototype fulfilling the functional requirements having all the characteristics mentioned above.