# Implement OOPS using JAVA with Data Structures and Beyond Project 1 Virtual Key for Your Repository

## Project Objectives:

### Project objective:

As a Full Stack Developer, complete the features of the application by planning the development in terms of sprints and then push the source code to the GitHub repository. As this is a prototyped application, the user interaction will be via a command line.

### Background of the problem statement:

Company Lockers Pvt. Ltd. hired you as a Full Stack Developer. They aim to digitize their products and chose LockedMe.com as their first project to start with. You’re asked to develop a prototype of the application. The prototype of the application will be then presented to the relevant stakeholders for the budget approval. Your manager has set up a meeting where you’re asked to present the following in the next 15 working days (3 weeks):

* Specification document - Product’s capabilities, appearance, and user interactions;
* Number and duration of sprints required;
* Setting up Git and GitHub account to store and track your enhancements of the prototype;
* Java concepts being used in the project;
* Data Structures where sorting and searching techniques are used;
* Generic features and three operations:
  + 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.

The goal of the company is to deliver a high-end quality product as early as possible.

### The flow and features of the application:

* Plan more than two sprints to complete the application;
* Document the flow of the application and prepare a flow chart;
* List the core concepts and algorithms being used to complete this application;
* Code to display the welcome screen. It should display:
  + Application name and the developer details;
  + The details of the user interface such as options displaying the user interaction information;
  + Features to accept the user input to select one of the options listed;
* The first option should return the current file names in ascending order. The root directory can be either empty or contain few files or folders in it;
* The second option should return the details of the user interface such as options displaying the following:
  + Add a file to the existing directory list;
    - You can ignore the case sensitivity of the file names;
  + Delete a user specified file from the existing directory list;
    - You can add the case sensitivity on the file name in order to ensure that the right file is deleted from the directory list;
    - Return a message if FNF (File not found);
  + Search a user specified file from the main directory;
    - You can add the case sensitivity on the file name to retrieve the correct file;
    - Display the result upon successful operation;
    - Display the result upon unsuccessful operation;
  + Option to navigate back to the main context;
* There should be a third option to close the application;
* Implement the appropriate concepts such as exceptions, collections, and sorting techniques for source code optimization and increased performance.

### You must use the following:

* Eclipse/IntelliJ: An IDE to code for the application;
* Java: A programming language to develop the prototype;
* 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: Data structures used for the project;
* Specification document: Any open-source document or Google Docs.

### Following requirements should be met:

* The source code should be pushed to your GitHub repository. You need to document the steps and write the algorithms in it;
* The submission of your GitHub repository link is mandatory. In order to track your task, you need to share the link of the repository. You can add a section in your document;
* Document the step-by-step process starting from sprint planning to the product release;
* Application should not close, exit, or throw an exception if the user specifies an invalid input;
* You need to submit the final specification document which includes:
  + Project and developer details;
  + Sprints planned and the tasks achieved in them;
  + Algorithms and flowcharts of the application;
  + Core concepts used in the project;
  + Links to the GitHub repository to verify the project completion;
  + Your conclusion on enhancing the application and defining the USPs (Unique Selling Points);