SAFE STORE:

One Place Solution For Managing Your Passwords

Idea:

Nowadays it is becoming increasingly difficult to effectively maintain Usernames and passwords for all registered websites. This compelled users to set the same pattern of passwords for all websites or to provide the same password for all registered websites which is a compromise of his own Privacy and security.

SAFE STORE is one such solution to solve this problem. A registered user can save his credential once and the application will take complete responsibility for storing the password safely and retrieving them anytime. As Internet connection is not a prerequisite for storing/Retrieving credentials, Application is implicitly secure. The password file resides locally in the user's computer so he need not to worry about the compromise of trust.

Specification:

Overall capabilities of the application can be divided into 6 basic operations.

1. Registration
2. Login
3. Logout
4. Delete Account
5. Store/Retrieve credentials for a website
6. Search Credentials for a website

1. Registration

A User needs to register first before logging in and start using the application by providing the below his details.

First Name, Last Name, Email ID, user provided password.

All these details are captured via an interactive console, and they are validated against a regular Expression designed for each field. Users will be forced to enter data in repeatedly until they enter it in the correct format. Which can be configured anytime by updating **com.safeStore.UserOperations.UserDefaults.java file.**

Once all the details are entered in required format, they are stored in a file "**UserData.data**" file. Email ID can be used to uniquely identify a user(Primary Key).

2. Login

   Once a user registers, he can login by selecting the appropriate option displayed in the console. User must use an EMail ID as username and if his password matches with the stored password, a login request will be accepted. Last login time of the user is stored each time a user logs out of the application. When a user logs in again, his last login time would be displayed in the system default time zone.

3. Store/Retrieve Passwords:

   Users can perform Store credentials for any website and retrieve them any time. Only logged in users are able to perform this. Application will not allow users to store more than one credential for a website with the same username.

   All credentials are sorted using Selection sort Algorithm in Ascending order during Retrieval.

4. Delete My account:

   User can delete his account, Application would search and delete the unique data file corresponding to the user and delete all details captured during registration.

5. Logout:

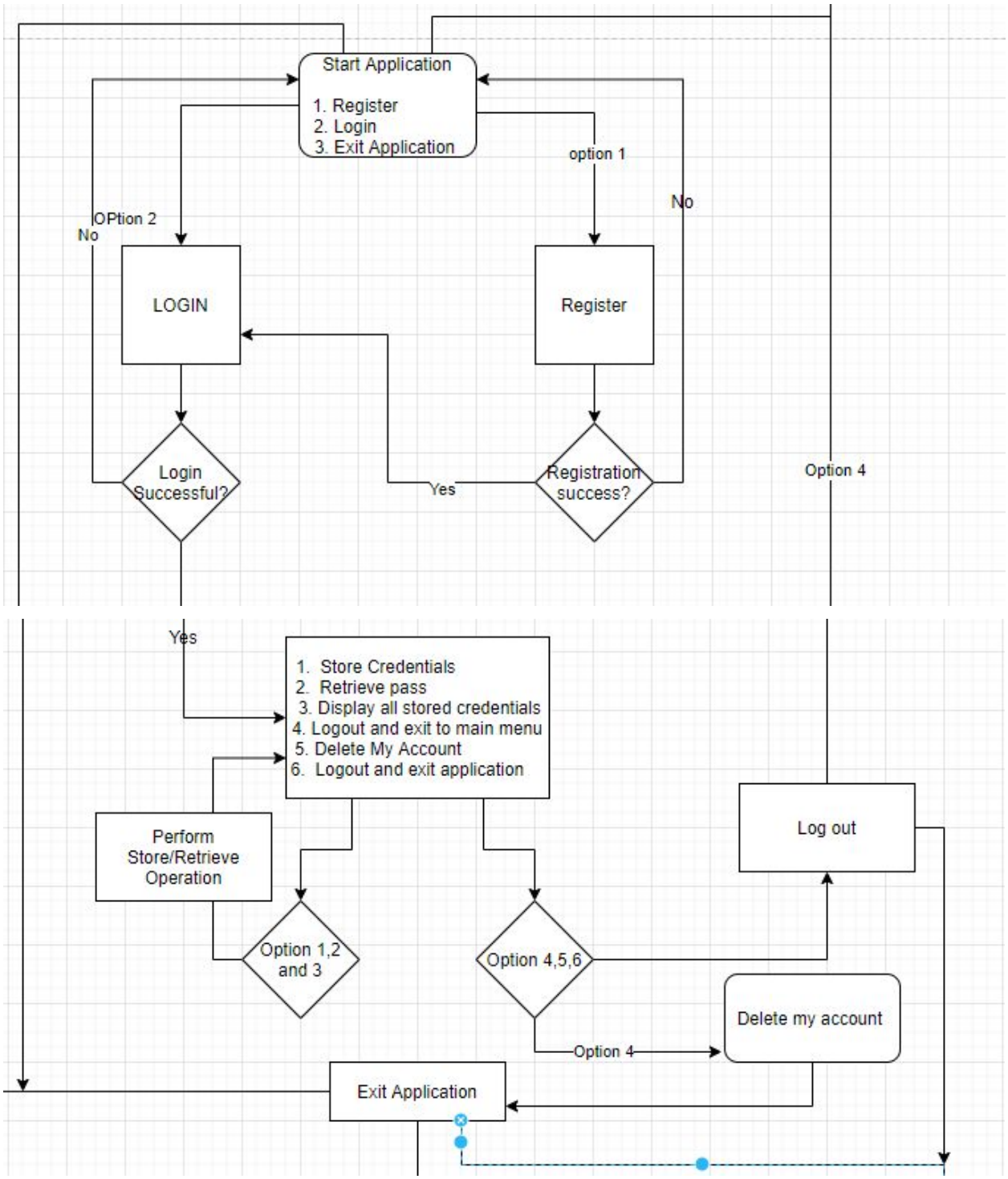
   Users can log out of the application any time by selecting the application. By doing so current system time would be captured as his last login time.

6. Search Credentials: Using this option, users can retrieve credentials for a particular website. If no Websites found for a website, an error message would be displayed along with appropriate error message.

Below are the text Files the application uses for persistence:

1. **UserDetails.data:** used for storing user details such as Name, email ID, password, last login etc. This file is common for all registered users.
2. **AppData\userEmailID_storedPasswords.txt:** These files are created for each user with a unique name during registration. It stores the user stored credentials for a website.

Overall flow of the application is shown va below diagram:



GitHub link for codes: https://github.com/sandeshMS1996/Safe_Store.git

Important Packages:

1. **com.SafeStore.UserManager** : contain codes related to user registration, Login,Logout, Delete user Account
2. **com.SafeStore.DataManager**: Contain codes for storing and retrieving user stored credentials for websites.
3. **com.SafeStore.MainApp**: contains the main method for the application and handles all user interactions.

Run the application:

To run the application(Minimum java version required: **Java 1.8** or later):

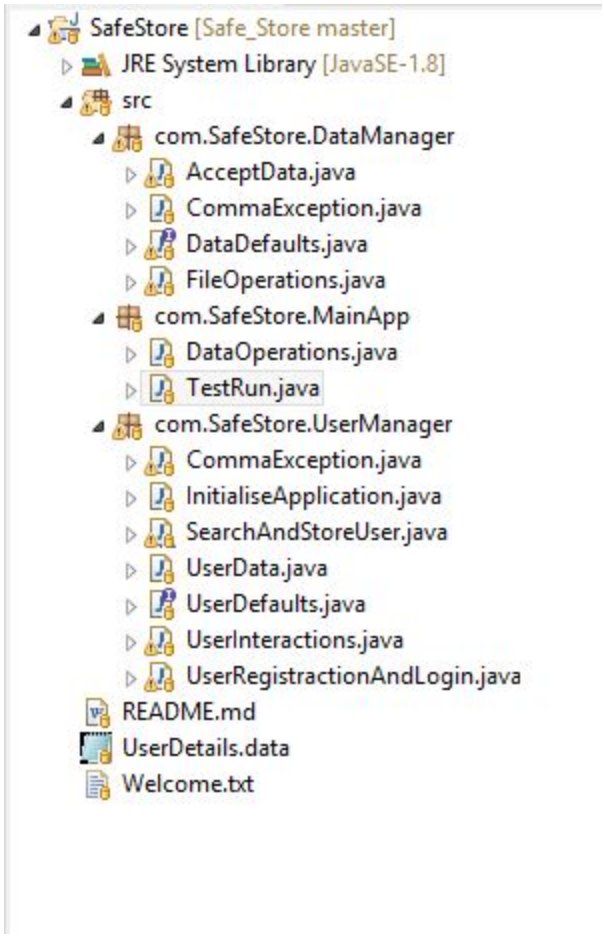
Step 1: go to the directory where you would want to download the project in command prompt and type the below.

```
git clone https://github.com/sandeshMS1996/Safe_Store.git
```

```
Step2: open the project in eclipse IDE and navigate to
com/SafeStore/MainApp/TestRun.java
```

```
Step3: Right click and select Run as -> java application
```

```
Directory structure of the project
```



OR

Open the below location and enter **java -jar SafeStore.jar** in command prompt

Downloaded folder->Docs->SafeStore.jar

Software development details:

Software Development Model: AJILE

Number of Sprints : 3

Duration of each Sprint : 1 Week(5 Days)

Details of each Sprint:

Sprint 1(Duration 1week):

User story 1:

As a new user, I need to Register for SAFE STORE so that I can login and use the application.

Description:

Accept below details from the user using console and store details into User Data file: First name, Last Name, Email ID , Password

Acceptance criteria:

a. Validate each field using regular expressions and all details are mandatory.
b. Make Email ID as user name for further login
c. Store the current system time as last login time for the user
d. All details except Password should be shored in lowercase except Password
e. should reject the registration if the user already registered using entered email ID

User story 2:

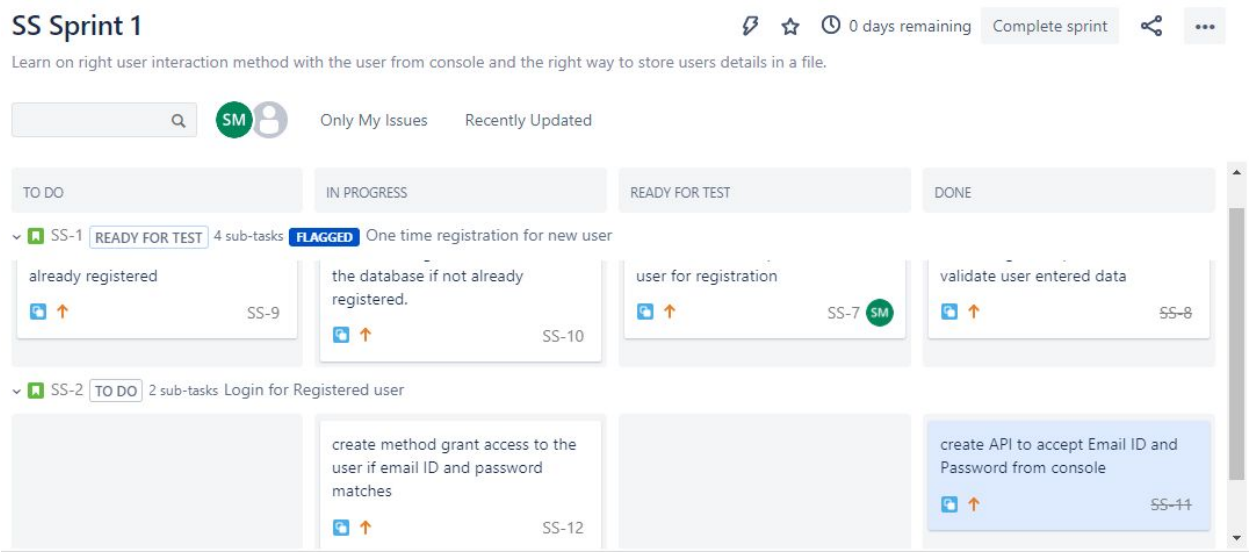
As a registered user, I need to Login Application using my registered email Id and password.

Description : Accept Email and password from user, and login to to the database if his credentials matches

Acceptance criteria:

a. Only registered user should be able to login
b. Users should be able to login only if their password matches else reject login request with appropriate error message.
c. Display last login time if login successful.

JIRA screenshot for sprint 1:



Sprint 2(Duration 1week):

User story 1:

As a user, I need to logout from the application so that others cannot retrieve/store passwords on my account.

Description: Users must have the option to log off from the application

Acceptance criteria:

a. Only logged in user must be able to logout
b. once user Logs out current system time should be stored as his last login time.

User story 2:

As a User, i need delete my account from SAFESTORE so that all my saved details would be deleted

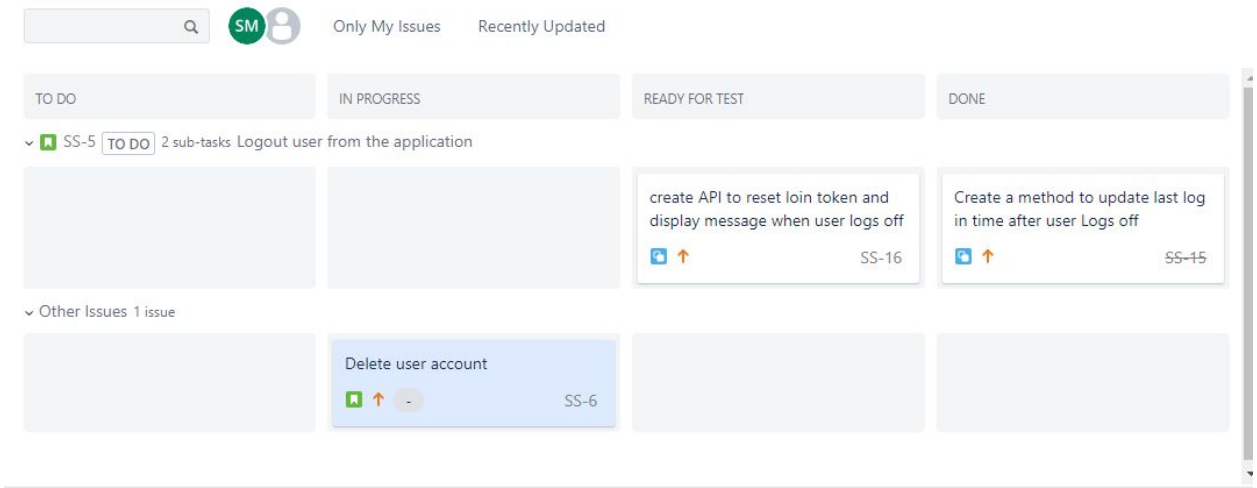
Description: User must be able to delete his account details along with his stored credentials.

Acceptance criteria:

a. Only logged in user must be able to delete only his account

b. when a user opts to delete his account,all his stored credentials for all websites should be deleted.
c. when a user deletes his account, entry for the user from the user file should be deleted.

Screenshot of scrum board for sprint 2 from JIRA



Sprint 3(Duration 1week):

User story 1:

As a registered user, i want to store my credentials for website so that i can retrieve them later

Description: Accept below details from user and store these details in a user specific data file

website, username for the website, password for the user for a website

Acceptance criteria:

a. only a logged in user must be able to store his credentials
b. must avoid duplicates, i.e. if there already exists an entry with the same username for a website, application must reject such store requests.

User story 2:

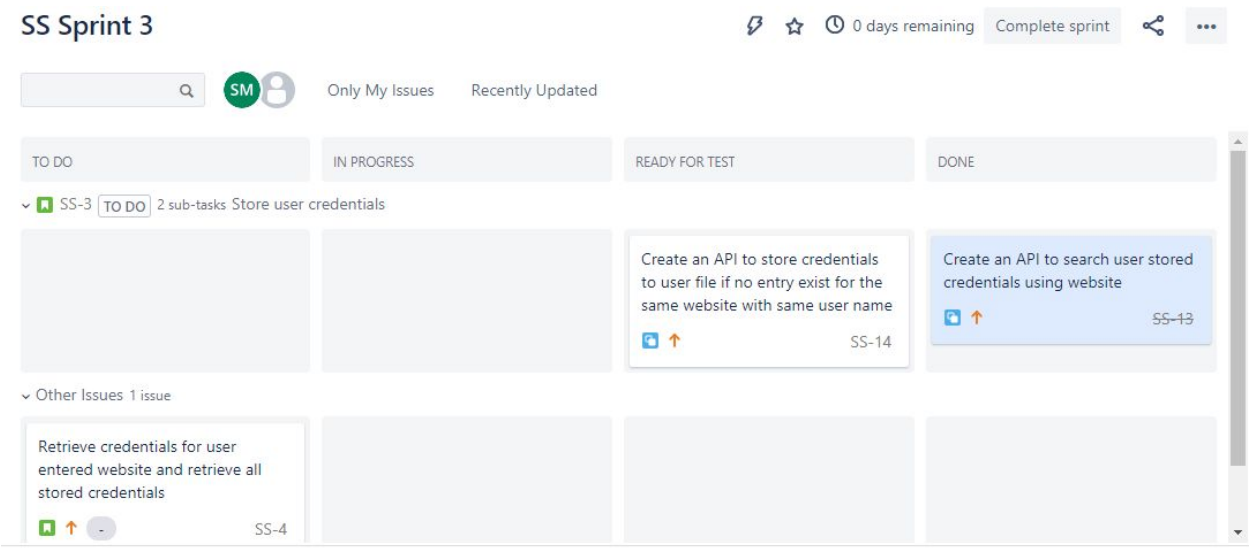
As a user, i need to search credentials for a website or display all my credentials so that i can login to the website

Description: Accept a search string from the user and display username and password for the same. if the user opts to retrieve his all credentials, display all his credentials.

Acceptance criteria:

a. Only logged in users must be able to retrieve his credentials.
b. If no website found for the search string entered, display the appropriate error message and if so, give the appropriate suggestions from the stored data.

Sprint 3 Jira scrum board:



Important Java concepts Used.

1.Handeling user and data files using **NIO**

2. Handling new date and time API(java.util.time)

3. Handling **Exceptions** via try, catch and finally blocks, creating user defined Exception(CommaException), and Throw an Exception.

4. Using **Lambda** expressions for small operations.

5. Defining **Static Methods in Interfaces**

6. Using encapsulation, Inheritance, and polymorphism(all these features are being used here).

Conclusion:

Overall its a useful application for Storing and Retrieving credentials for any website. It is designed to be self driven where users need not to know anything about the application to use it. Self Explanatory error messages, Options to choose from, Meaningful Success or failure Message makes it easy to use.

Below are some of the some of Enhancement that would be applied in future.

1. USing Encryption techniques for storing user data safely.
2. Using Hash function to store user registration details safely.
3. Enable users to store/Retrieve data at any place by integrating to WEB