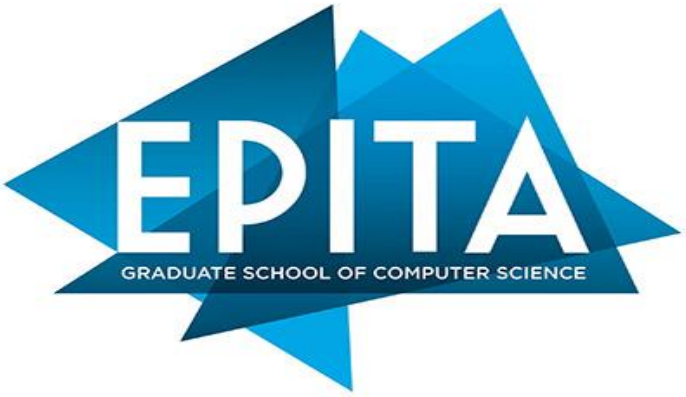


Identity Management System

Fundamental Java Project

Technical Specification

Submitted By,

Shruthi Chandrasekaran

Information Systems Management.

EPITA, International Master.

Submitted To,

Prof. Thomas Broussard

EPITA, International Master.

July 13, 2018.

MSc Spring 2018

Subject description:

The Identity Access Management System is fashioned to handle identities and its attributes. The objective of this application is to render a platform to the user to perform operations like **Create, Read, Update and Delete** operations on the database by an ADMIN.

Subject Analysis:

In this console application the ADMIN can perform the following functions on the various identities that is stored in the database

- Create an Identity
- Search/List an Identities
- Modify an Identity
- Delete an Identity
- Search all the Identities
- Delete all the Identities

An authentication is performed again to check whether it is the Admin using the application to perform "List all the Identities" and "Delete all the Identities" operation. The security measure enforced where the ADMIN has to login first using ADMIN_ID and ADMIN_PASSWORD to perform the listed operations.

Application Feasibility:

The prototype of this application created in a mode for employing a highly secured environment of Identity and access Management. The development platform, servers and database are open source devising the features quite feasible and secured with authentication enabled.

Expected Results:

The result of the application is to authenticate the ADMIN through the ADMIN_ID and ADMIN_PASSWORD, establish connection to the Application, and perform below functions:

- Create an Identity
- Search an Identities
- Modify an Identity
- Delete an Identity
- Search all the Identities
- Delete all the Identities

Scope of the Application:

Scope:

- Privacy support feature for online transaction using IDM solution focusing on privacy.
- Defined permissions enabled for Admin and other users.
- Centralization of ADMIN.

Limitations:

- Console application with no GUI
- Only the ADMIN

Evolutions:

- To create multiple admins and users to access the IAM Core project.
- To make a complete web based project with proper bootstraps and css.

Chosen Algorithm:

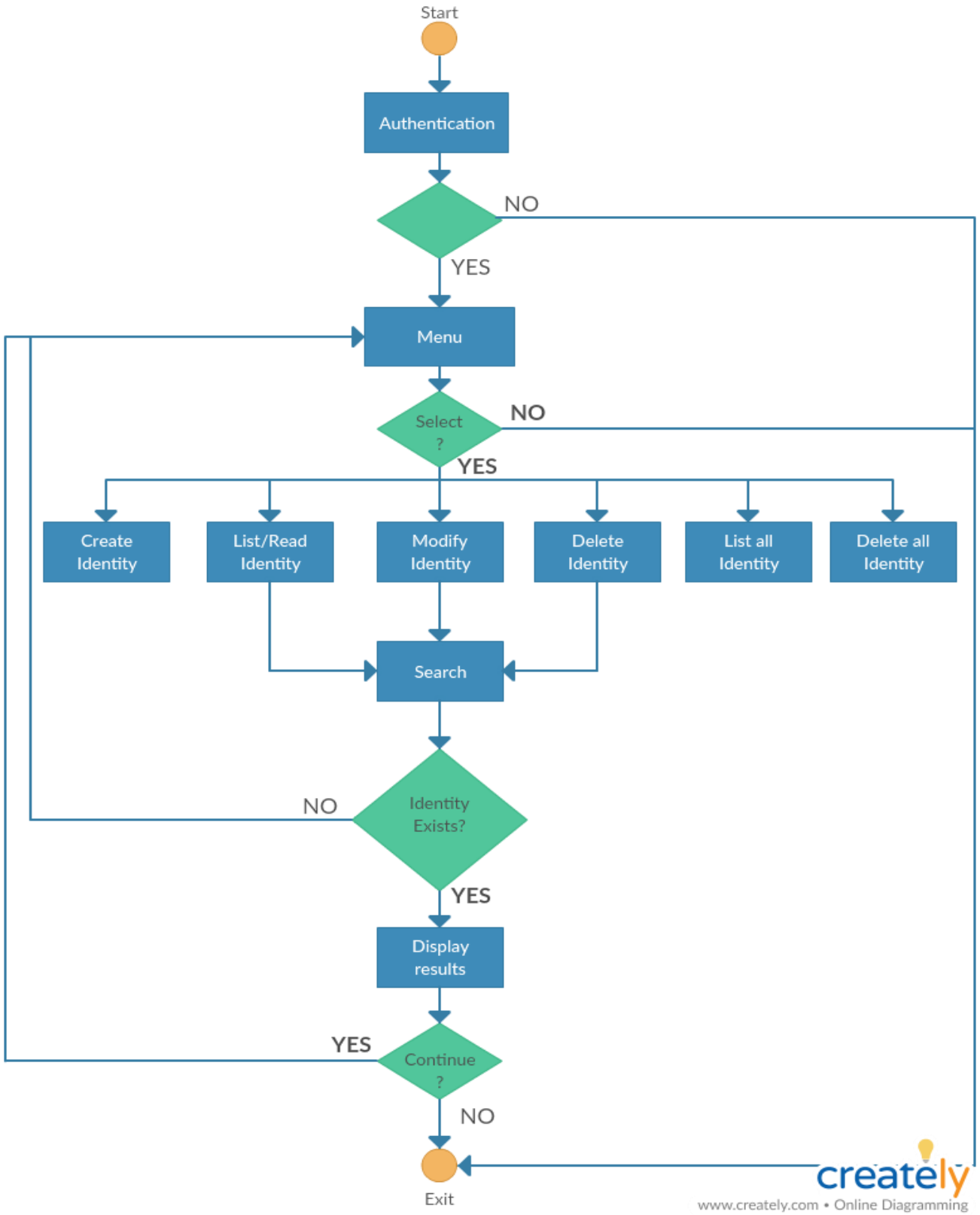
Data Access pattern or DAO pattern is used to separate high level business services from low level data accessing API. Data can be persisted by reading/writing data from/into the database using a JDBC connector

Conception:

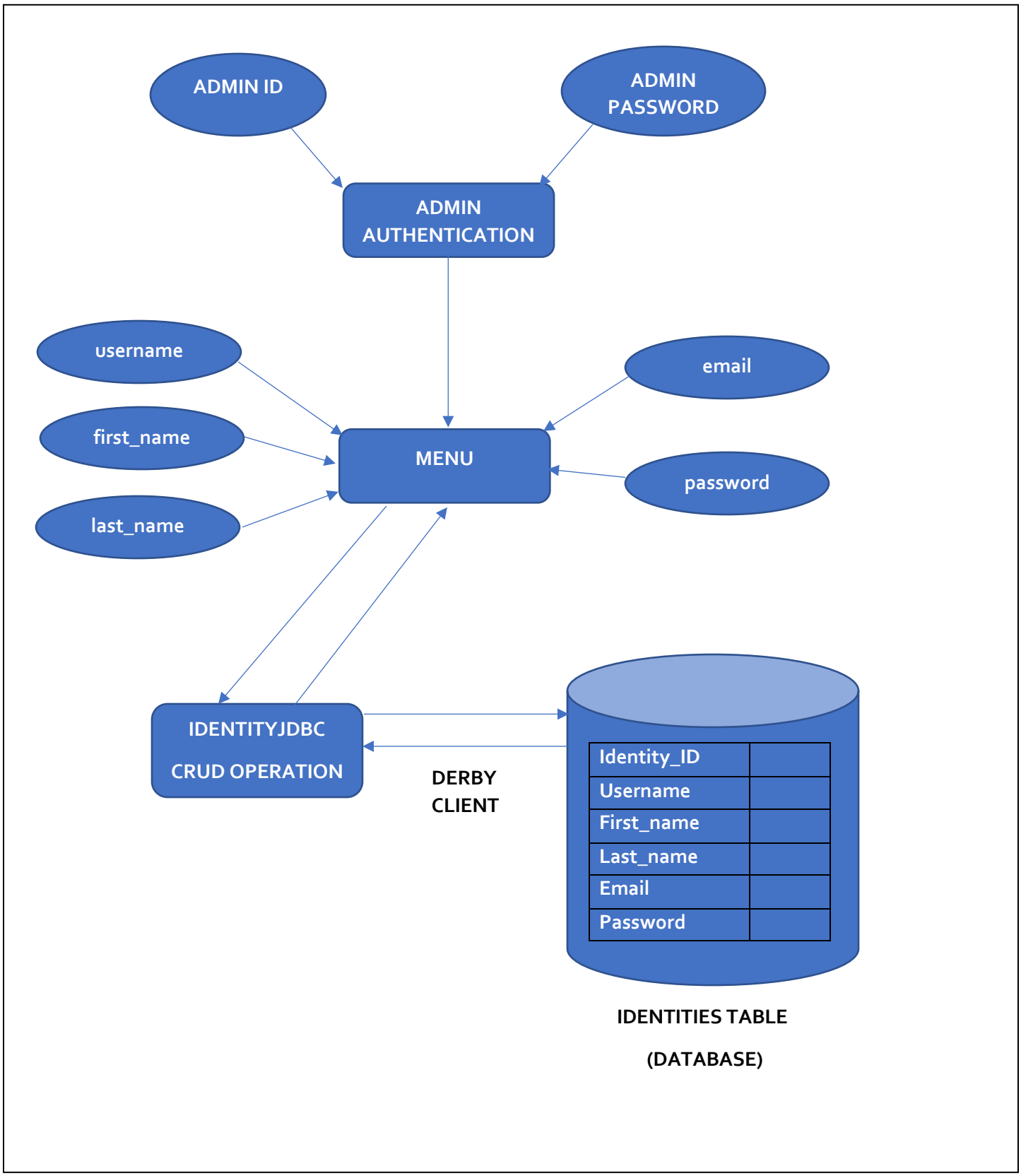
Data Structures:

The data structures provided by the java utility package are very powerful and perform a wide range of functions. The data structure consists of Array List, Boolean, String, interface and classes.

Global Application Flow:



Global Schema and Major Feature Schema:



Console Operation Description and Screenshots:

The identity management application is a console-based application; the ADMIN has to select one of options available.

Authentication:

ADMIN will be prompted to login as to authenticate him and has to submit his ADMIN_ID and ADMIN_PASSWORD.

Menu:

After authentication, a menu will be displayed and the following functionalities can be performed.

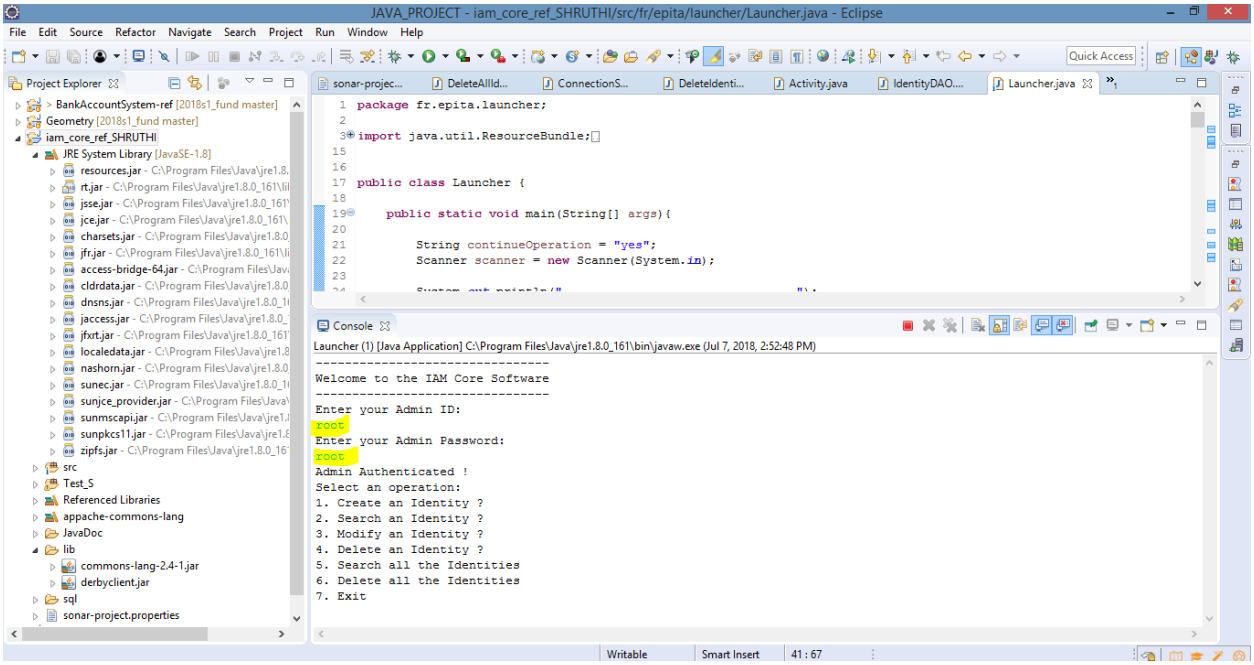
Console Operations implemented on the system are:

- Create an Identity
- Search an Identity
- Modify an Identity
- Delete an Identity
- Search all the Identities
- Delete all the Identities
- Exit

Each operations are listed below:

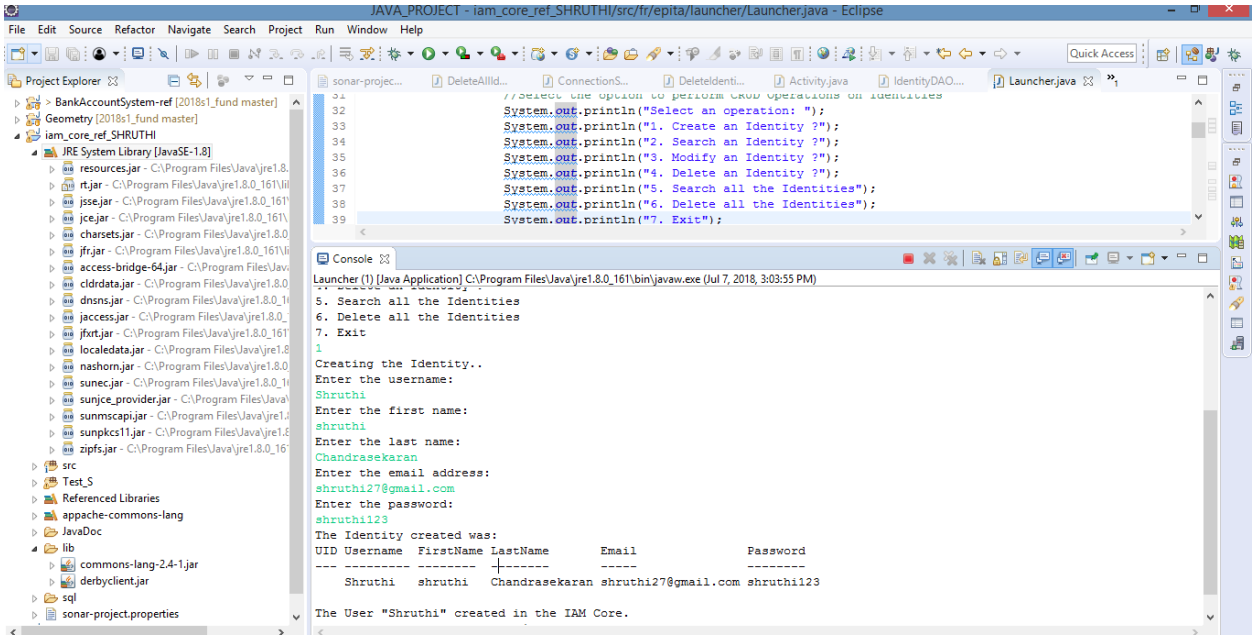
Authentication:

ADMIN is authenticated in the Admin DAO that takes input as ADMIN_ID and ADMIN_PASSWORD, and connect to the application.



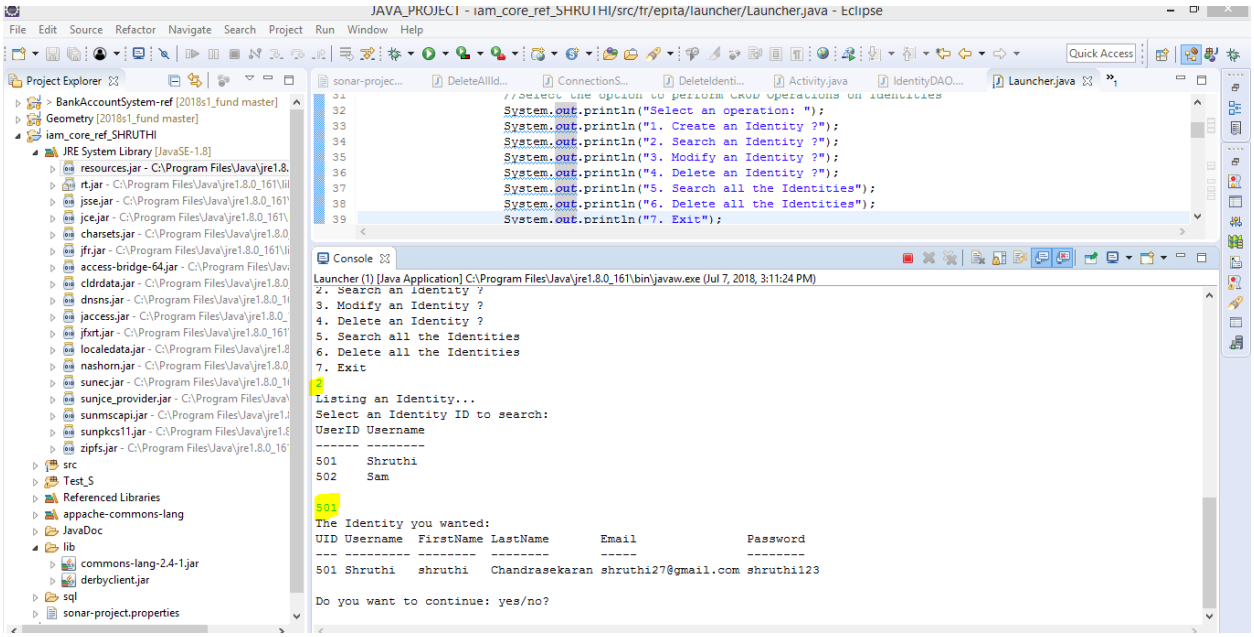
1. Create an Identity:

By selecting option 1, ADMIN is authenticated to create a new identity. Username, firstname, lastname, email and password for the user is requested to enter. The entered details for the new identity created is displayed.



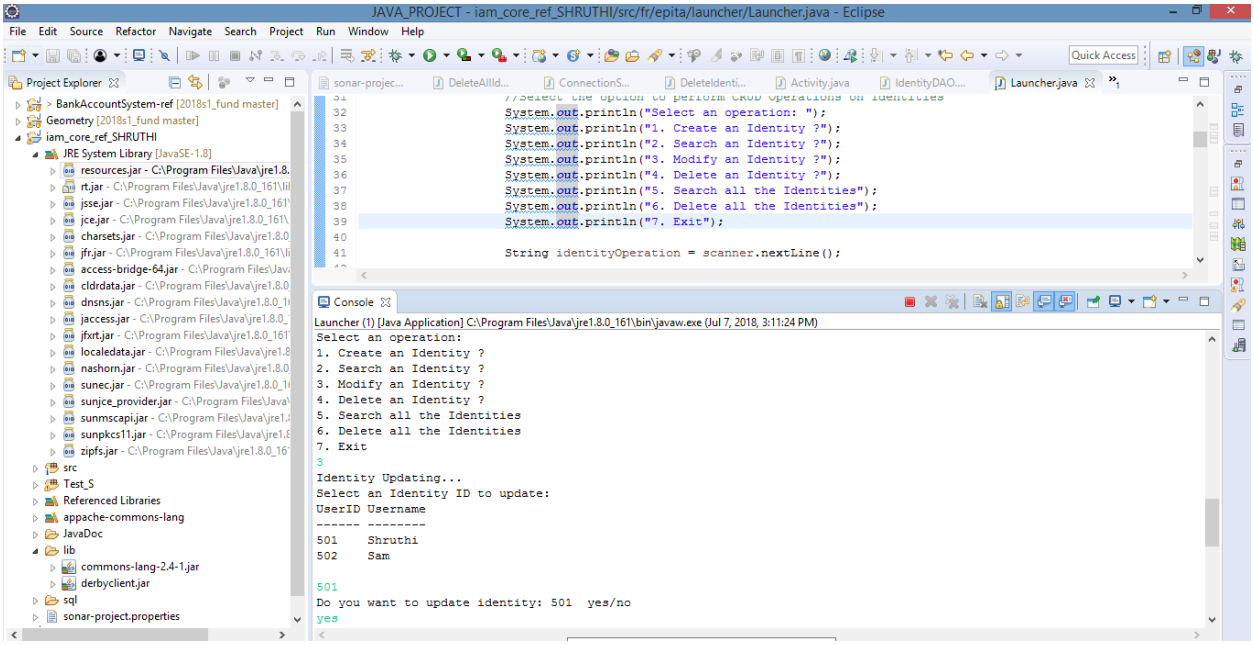
2. Search Identity:

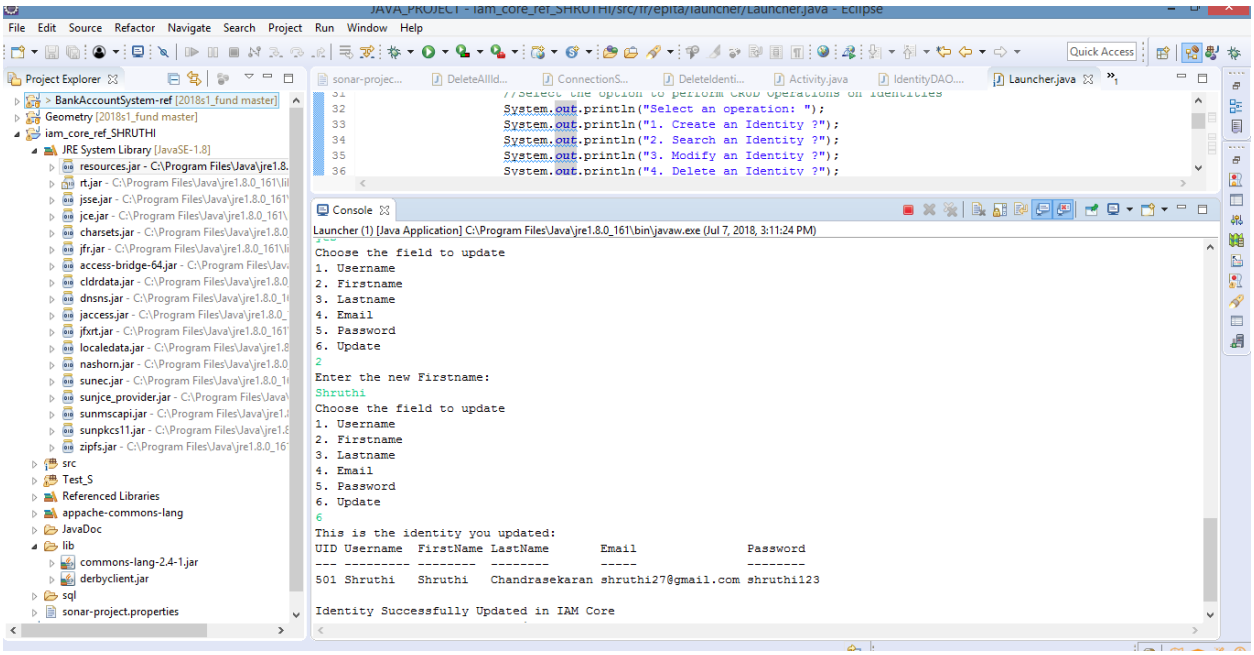
By selecting option 2, ADMIN is provided with a list of identity containing UID and Username, user can select the identity with the respective UID and corresponding details of the user is displayed.



3. Modify an Identity:

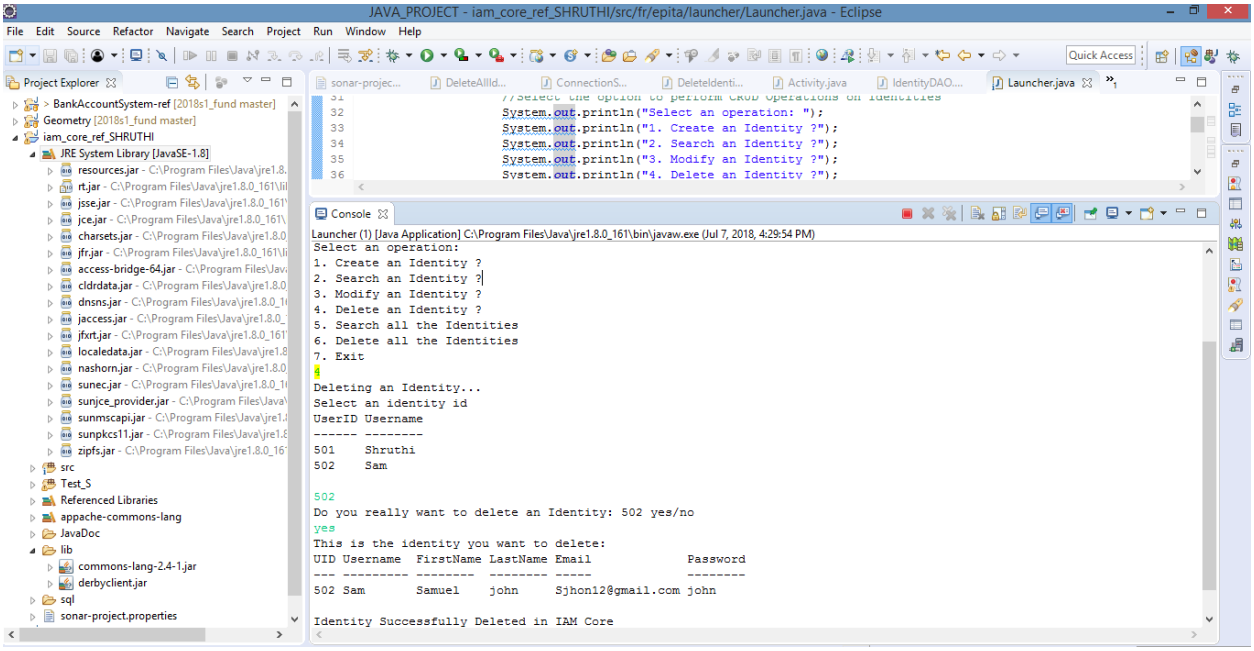
By selecting option 3, ADMIN is provided with a list of identity containing UID and Username and ADMIN can update Username, Firstname, Lastname, Email and Password and update the field with modified values.





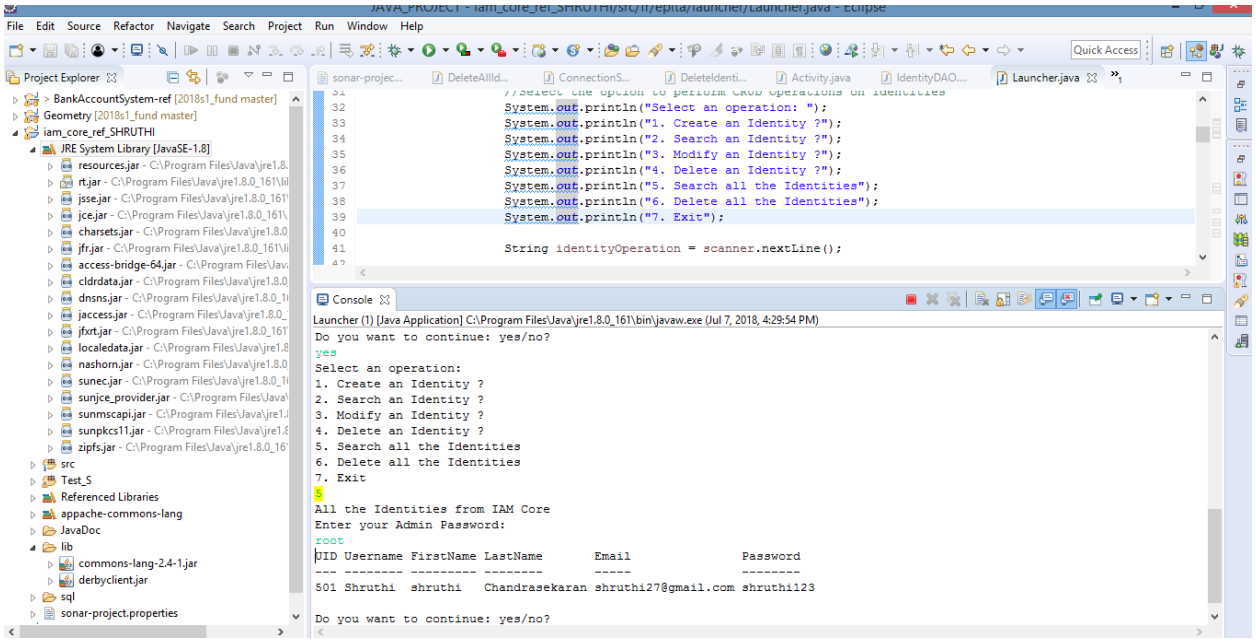
4. Delete an Identity:

By selecting option 4, ADMIN can delete the identity by choosing the UID.



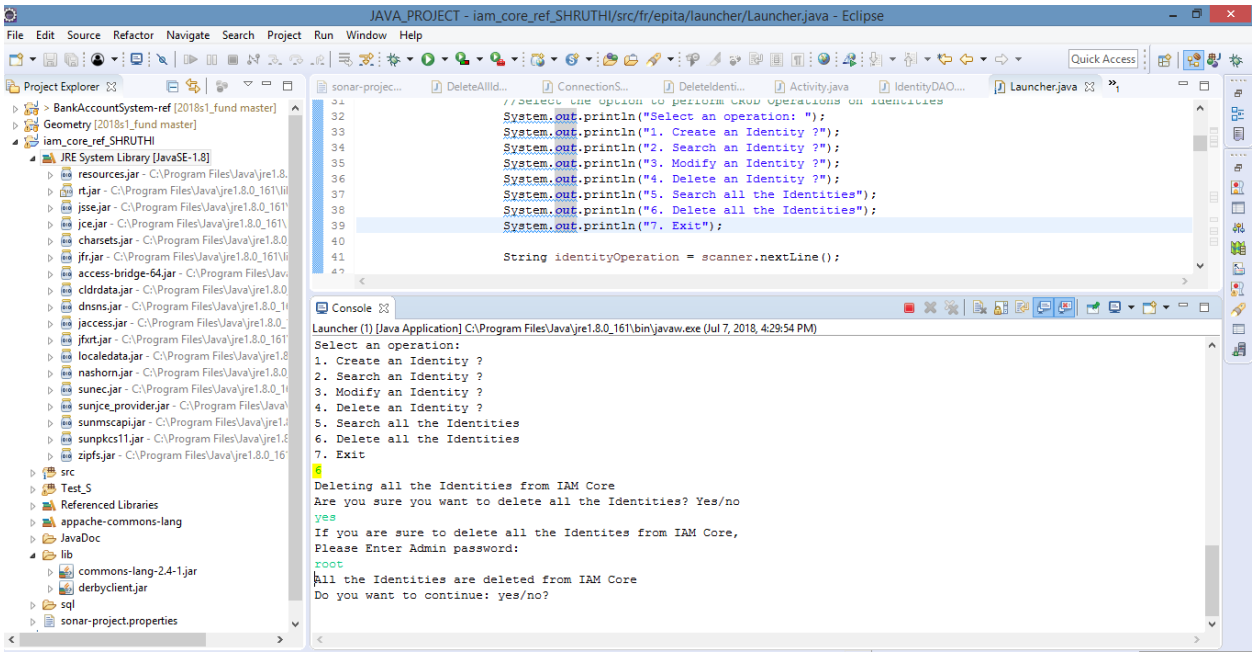
5. Search all the Identities:

By selecting option 5, Admin can view all the Identities in the database by providing the Admin password again.



6. Delete all Identity:

By selecting option 6, Admin can delete all the Identities in the database by providing the Admin password.



7. Exit:

By selection option 7, we exit from the application.

Configuration Instruction:

This application was built using:

Platform: Platform Independent
IDE: Eclipse Latest version: Oxygen 4.7.1a
JDK: JDK 1.8
Database: Derby DBM

Steps to run the IAMCore application

1. **Setup the database:**

Download apache derby client.
Create an instance of derby, instance name=Identities with DbUser value and DbPassword value.

To start the server:

db-derby-10.14.2.0-bin/bin/startNetworkServer.bat

Go to /iam_core_ref_SHRUTHI/src/fr/epita/config/config.properties edit DbUser value and DbPassword value.

Go to iam_core_ref_SHRUTHI/sql run Create_Identity_Table.sql on your instance Identities.

2. **Run the application:**

Go to /iam_core_ref_SHRUTHI/src/fr/epita/launcher/Launcher.java and execute Launcher.java**.**

Generate JAVADOC:

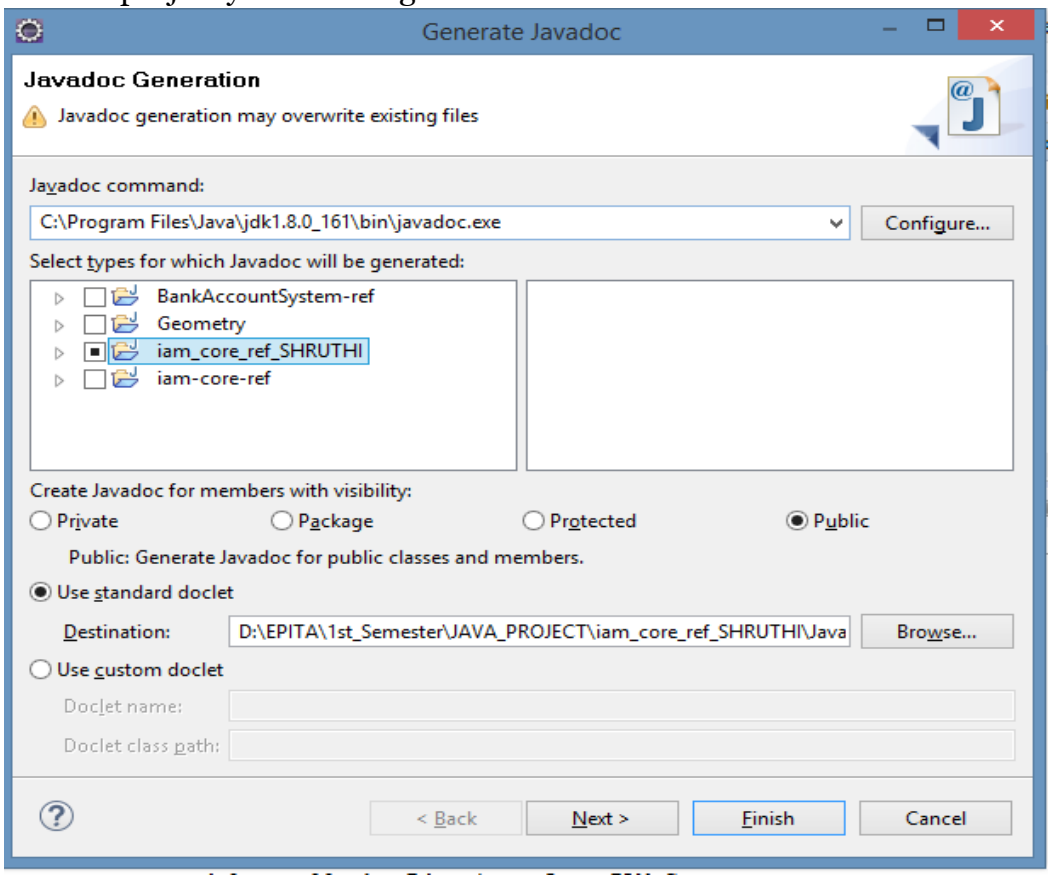
To Generate the Javadoc for the application, try to comment all the needed functions in all the files using the comment line:

```
/**
* Description about your Function
* @param <<parametername>> Description about your parameters
*/
```

Then click on “Project” and Select “Generate Javadoc”.

The select the Javadoc command from the JDK folder you installed. Normally it stores in C:/Program Files/Java/jdk1.8.0_161/bin/javadoc.exe.

In addition, select the project you want to generate the Javadoc as shown below and click on Finish.



SonarLint Testing:

SonarLint for Eclipse is an extension that provides on-the-fly feedback to developers on new bugs and quality issues injected into their code. Few options like:

1. Analyze set of files.
2. Mute specific rules.
3. Exclude specific files and issues.
4. Find logs if needed.

SonarLint plugin can be found in Eclipse Marketplace by searching. Help -> Eclipse MarketPlace.

Search for SonarLint and Install the plugin.

Configure Sonar in your Eclipse:

Download the related software:

1. sonarqube-6.7
2. sonar-scanner-3.0.3.778-windows

Unzip both the files and set up the environment variable for sonar-scanner-3.0.3.778-windows.

Name: Sonar Scanner

Path: Desktop:\sonar-scanner-3.0.3.778-windows\bin

Create a file under your project and name it as “sonar-project.properties”.

```
# must be unique in a given SonarQube instance
sonar.projectKey= <<Unique Key to be given>>
# This is the name and version displayed in the SonarQube UI.
sonar.projectName=<<Project name to be given>>
sonar.projectVersion=1.0
sonar.host.url=http://localhost:9000
# Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.
# This property is optional if sonar.modules is set.
sonar.sources=<<Give the path of your “src” folder>>
# encoding of the source code. Default is default system encoding
#sonar.sourceEncoding=UTF-8
```

Run the sonrqube-6.7 server. \sonarqube-6.7\bin\windows-x86-64\StartSonar.bat and

Run the sonar-scanner-3.0.3.778-windows. \sonar-scanner-3.0.3.778-windows\bin\sonar-scanner.bat

Bind project:

Once the set is done, the next step is to link Eclipse project with projects defined and analyzed on this Sonar server.

To do so, right-click on the project in the Project Explorer, then SonarLint ->Bind to a SonarQube project.

Analyze project:

Once the setup is finished. Analyze your code by right click on the project in Project Explorer, then SonarLint -> Analyze.

Bibliography:

https://www.sonarlint.org/eclipse/

http://thomas-broussard.fr/

https://dzone.com/articles/sonarqube-67-amp-sonarlint-3-eclipse-plug-in-insta

https://db.apache.org/derby/integrate/plugin_help/derby_app.html

https://www.jetbrains.com/help/idea/generating-javadoc-reference-for-a-project.html

https://www.javacodeexamples.com/

https://www.codecademy.com/learn/learn-java