Java Fundamental Project

Title

IAM Project
- The first step in the IAM application: the IAM Core

Submitted by,

Praveen Kumar Selvaraj,
Masters in Data Science and Analytics,
Spring 2018
EPITA

Table of Contents

1. Subject description

The IAM Project is the Fundamental Java Project for the first semester done using Java programming Language

The Project will present the Identity Management Storage process using Java. With the initial implementation of project done during the class sessions by the professor, I have composed the complete programming structure for the Identity Management Storage by making it work. The project will implement the process of Identity Storage system with various functions in it and visualized in the Command line display for the users.

2. Subject analysis

2.1 Major features

Project has four major features available for the users using the application. Starting with the authentication process, user will have the secured way of handling the application. Two Major functions implemented in this project are as follows:

1. Authentication
2. Managing the Identities

- Authentication: This Project made sure that any user who tries to play in managing the Identity has to first clear the login authentication test. This will make sure that only user with known credentials will be using the application avoiding the security threats.

  Implementation for authentication done with common username and password for all the users at the initial stages. Future update of the application will have separate credentials for each users trying to use the application.

- Manage Identities:

  This is the important phase of the project where user would be able to manage the following functions in the application.

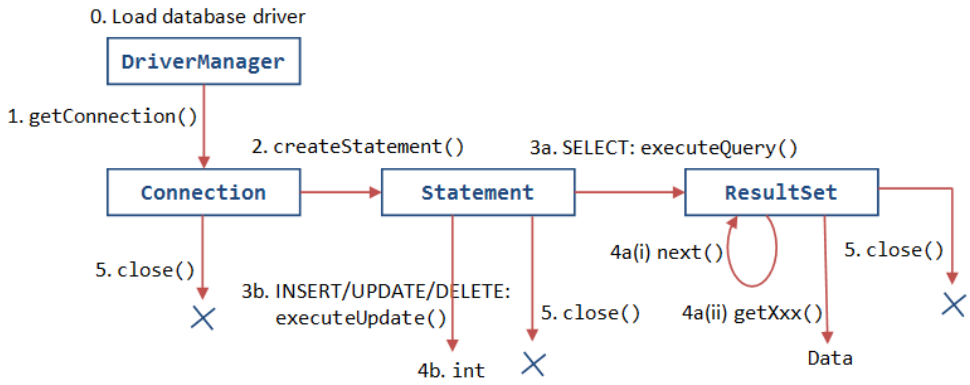
1. Creating the Identity
2. Searching/Reading the information of the Identity
3. Updating the Identity
4. Deleting the identity

We store identities by creating the identity function and perform the various other functions available in the application. Each identity differentiated with the unique user ids and information of the users are stored in the database. Considering the Derby as the lightweight database, I have used the Derby database to store the identity information. Database engine is setup for the process and integrated with the application using the driver Manager class.

2.2 Application Feasibility

Feasibility of the IAM project stands positive since half of the application source code developed during the class session and with the lightweight Database, the Identity Management application is easy to integrate with Database with driver manager classes available in Java. Once the JDBC connection is setup, the user can easily integrate the connection between the application and storage process.

Simple Feasibility illustration of JDBC connection is below:



2.3 Data description

In the IAM project Major functionalities are Create, Update, Search and Delete operations done on an identity. Each Data can be persisted by reading/writing data from the database using a JDBC connector. Derby database considered for the project to store the Identity information.

Identity defined with three following inputs

- UID which is an integer value assigned automatically in the database for each users created. It will be primary key in the table since UID cannot be duplicated for the different users.

- DISPLAYNAME is the String value, which stores the display name of the user in the database.
- EMAIL which is the string value, which stores the email id of the user in the database.

2.4 Expected results

Project is implemented in the command line input/Output since UI part is not done. User will provided the input in the command line of the program and output again is project in the command line only.

Application results in Major functions

1. Creating(Insert) the new user in the Identity management system
2. User can be able to search the identities stored in the database with help of unique user id or name or the email id of the user. It is not mandatory for the user to input all the value to search. Search option created with better user convenience.
3. User can be able to update the information of the identity whenever required. This will help the user to change the ID , name or the email of the existing users.
4. User has advantage over removing the identity when required. Delete option will delete the identity from the database

2.5 Scope of the application

With the help of expected results, user allowed managing the identities. Scope is limited only to the identity in the Database not the users who are controlling it. Update version of IAM project will have the management controlling the users.

Application input/output is limited to Command line since UI not created in the initial part of the project.

3. Conception

3.1 Data structures

Identities

Identities the entities that are managed in the system, within the application we should be able to create, update, search and delete them at user convenience. All identities are going to be stored in a Derby database post setting up the driver engine

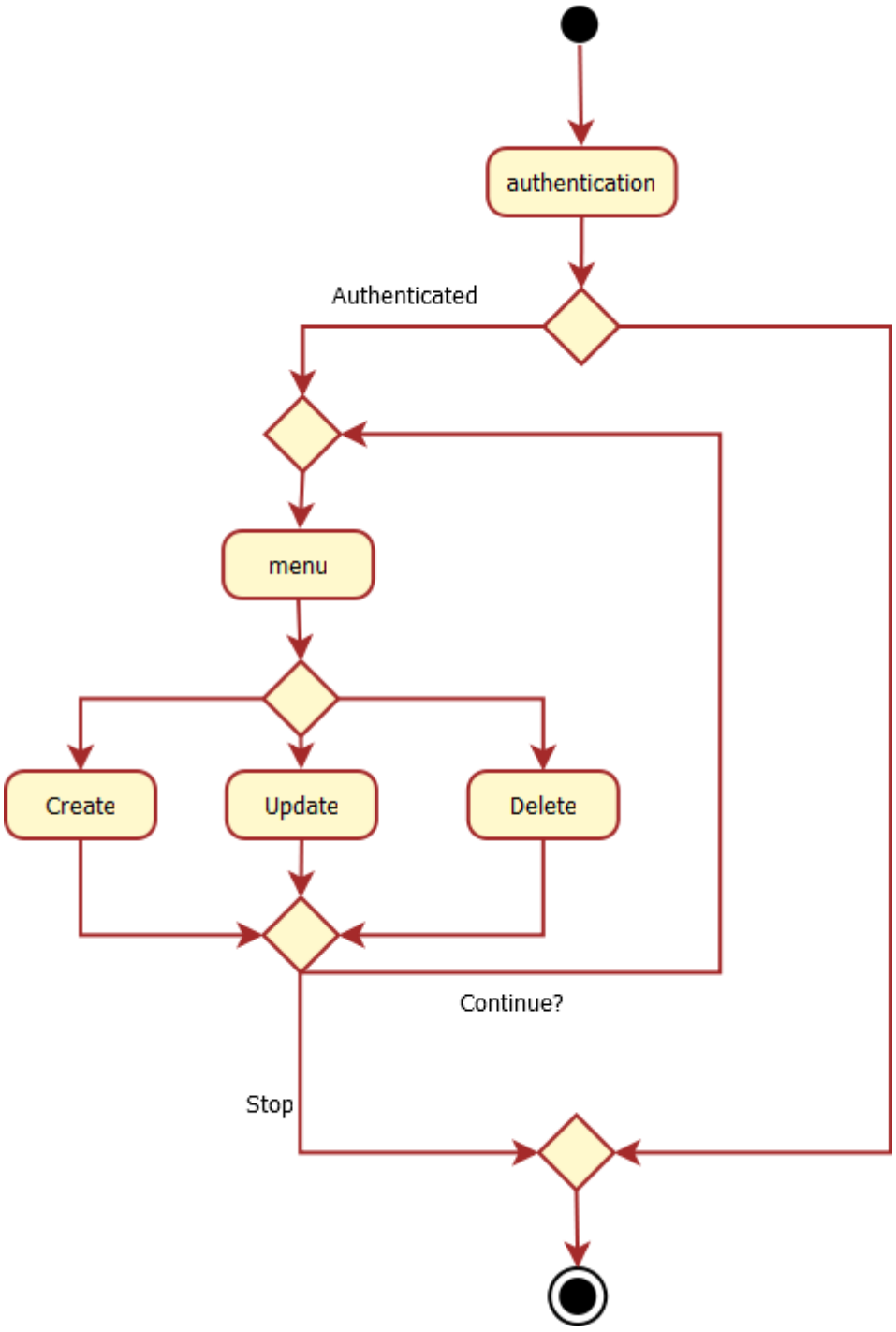
Users

Users are not supposed to be managed by the application, for now only single user is created to use the application. User credential are **hardcoded** in the program at the initial stages. A user will have a name and a password; both are stored as string datatype in the configuration file.

3.2 Global application flow

Global activity flow of the IAM project can be visualized as below:

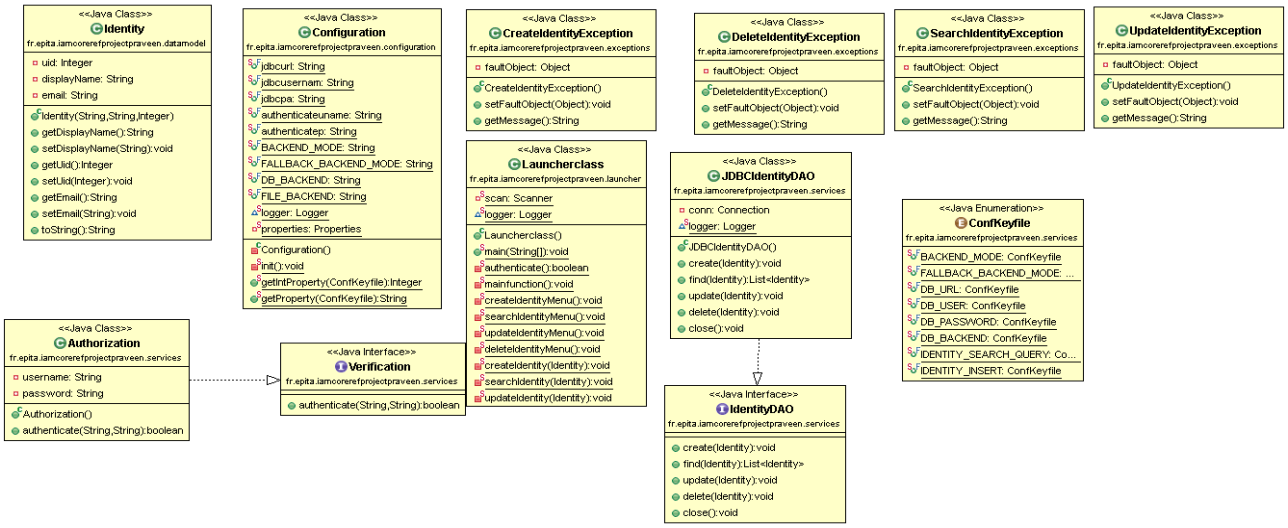
Application starts with the user authentication. User will not be able to go beyond the login screen if failed to input the correct user name and password.



Menu is created with Switch case statement in Java to display different functionalities of the project as options. User opts to choice within the choices for performing various functions on the identity.

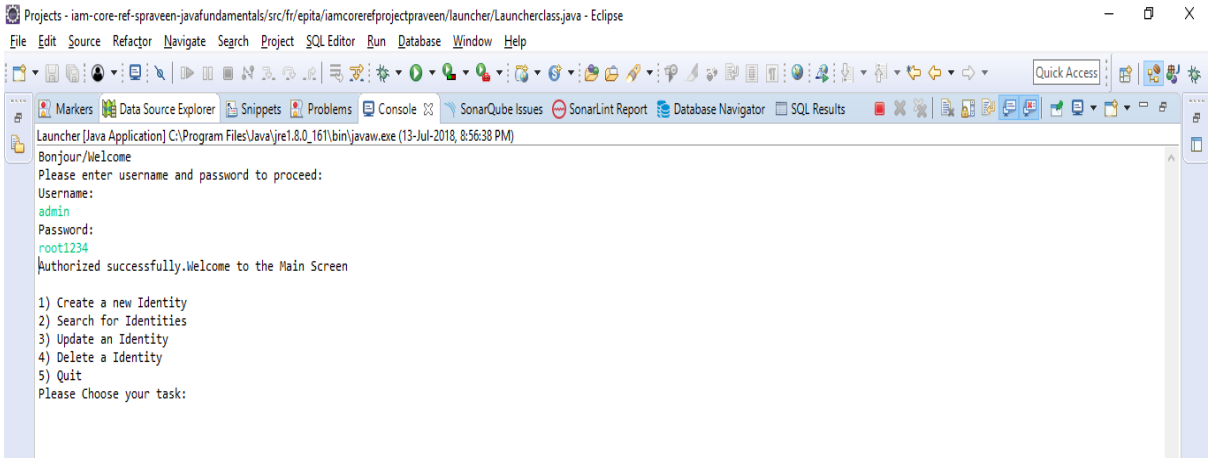
3.3 Global schema and major features schema

With the global scheme being designed



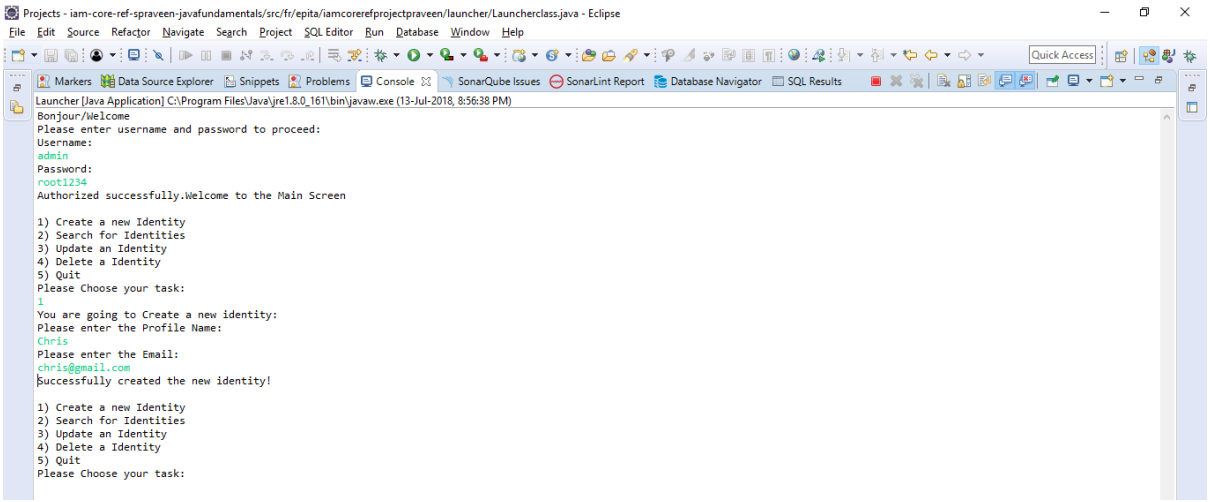
4. Console operations description

Menu

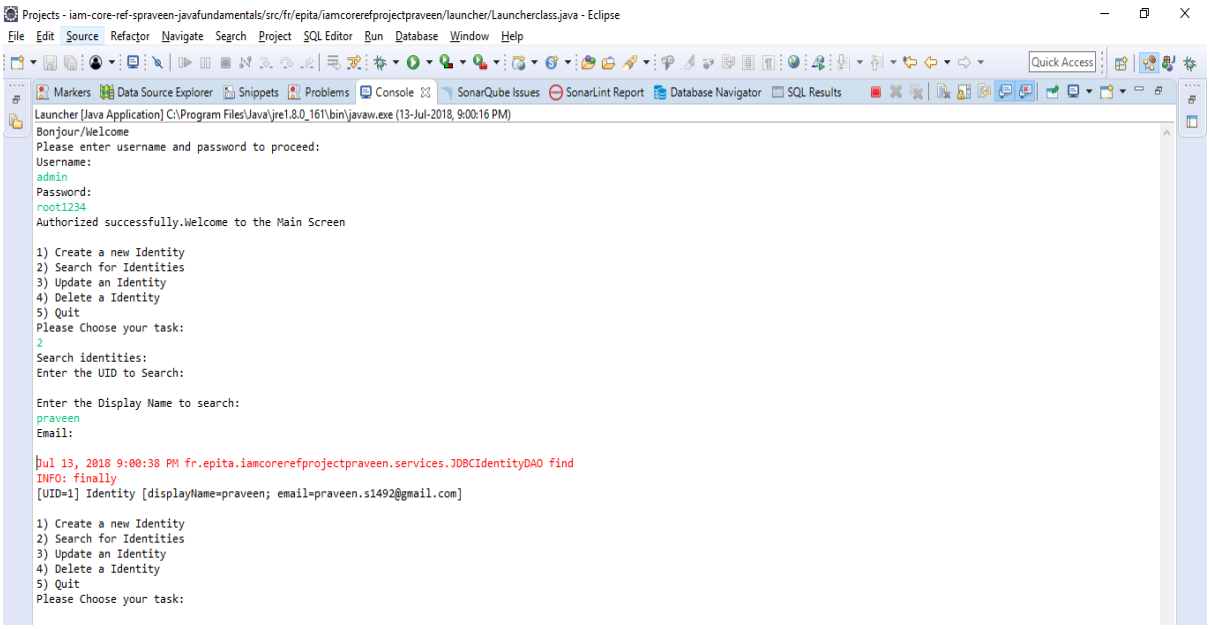


Options Available.

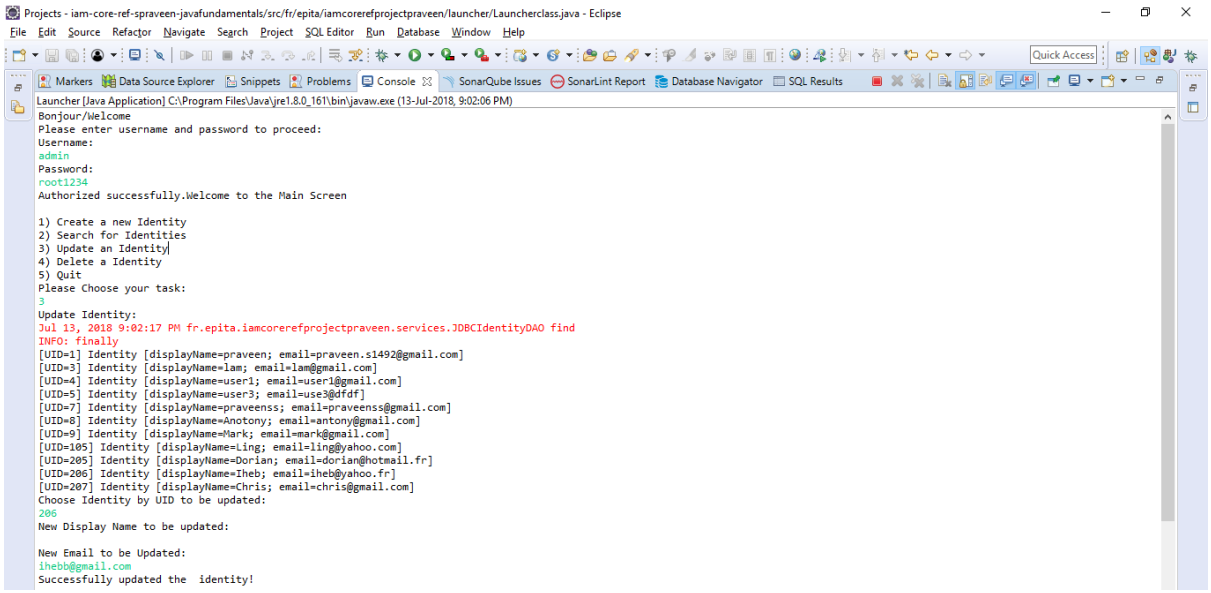
When user gives 1) Create a new Identity



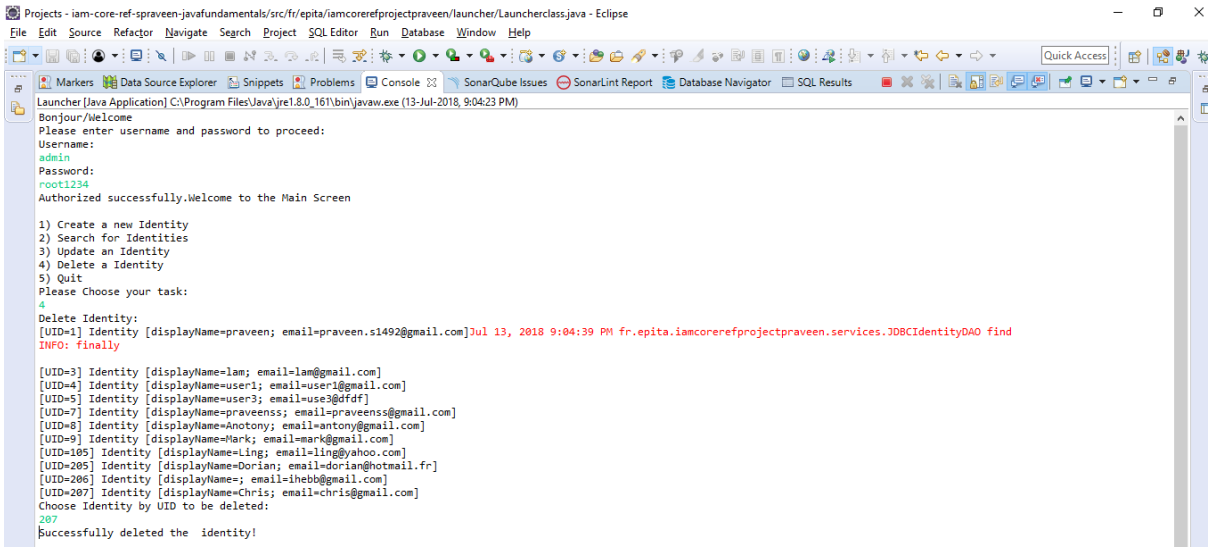
When user give option 2) Search for Identities



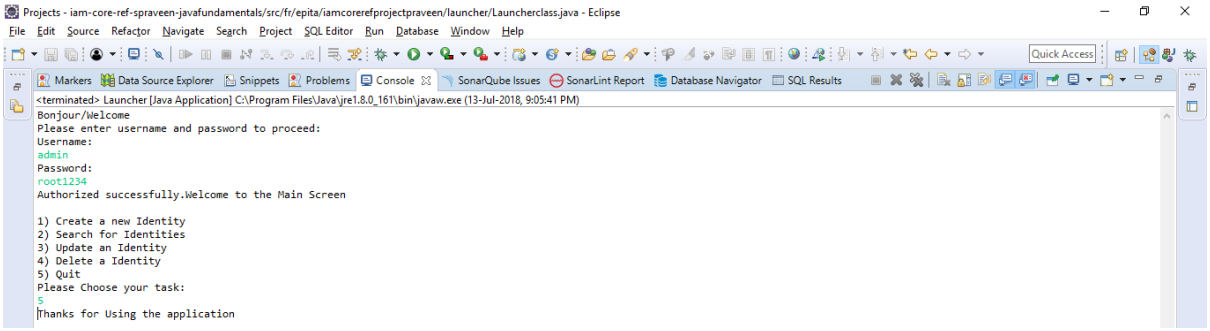
When user gives the option, 3) update an Identity



When user gives the option, 4) Delete a Identity



When the user decides to quit, Option 5) quit



5. Configuration instructions/ New User Pre requisites

1. User needs to make sure that Java is installed in the system.
2. Common application Eclipse should be Installed in the system to run the application
3. Please set up the derby database by downloading from the below website https://db.apache.org/derby/releases/release-10.14.2.0.cgi

4. Once downloaded, extract the zip file into a new directory. Open the directory bin and please run the StartServer.bat file.
5. Please establish the derby connection in the Eclipse by following the below steps

    a. In the data source explorer, please click new under the Database connections.
    b. Select Derby and click next
    c. In the window, click “new driver definition button to point the derbyclient.jar file.
    d. Move the derbyclient.jar file to new folder in the project.
    e. Once the driver is selected, create the database **“testconnection”**, copy the JDBC url created in the connection and use in the JDBC class for the connection establishment
6. Make sure the derby database is running by clicking on “Test connection”. It should display “succeeded” when its connected

Schema and Table Creation

1. Right click the New Derby option under Database connection and Open “SQL Scrapbook”
2. Create schema TEST under testconnection database using the sql code below

    Write the query `CREATE SCHEMA TEST;` and press Alt+X to execute the query

3. Open the **praveensql.**sql file in sql folder of the project and execute the create table query in that.
4. Table “Identities” is now created

Running the Launcher Class:

Outside the source folder of the project, there is folder named test folder which test.properties file.

It has all the configuration and SQL queries in it. This file will be used to call all the values the parameters defined in the launcher class of the program.

How to run the program:

1. Since the external file test.properties holds all the configuration files, running the launcher class would not yield any results.

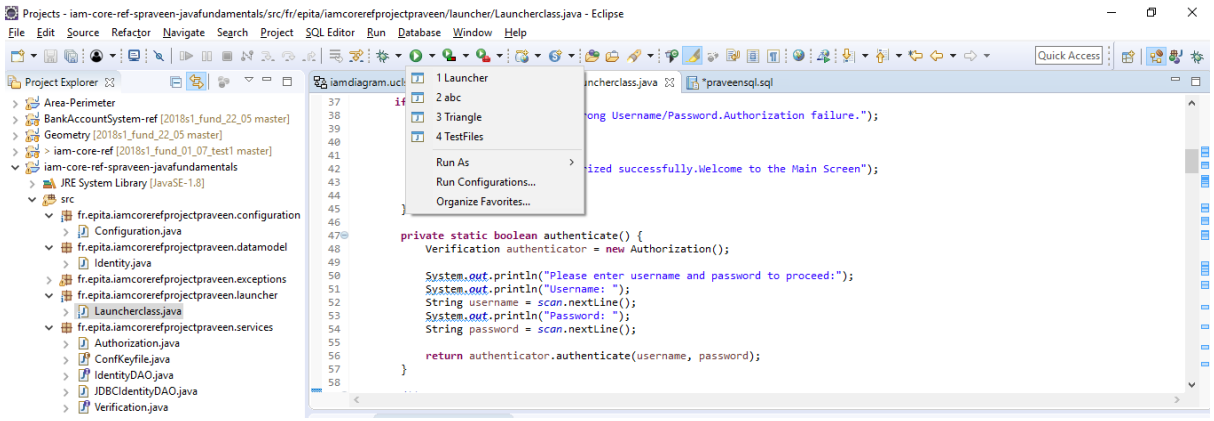
2. User has to run the launcher file by running the configuration. Configuration will have the location of the properties files passed as arguemtns.

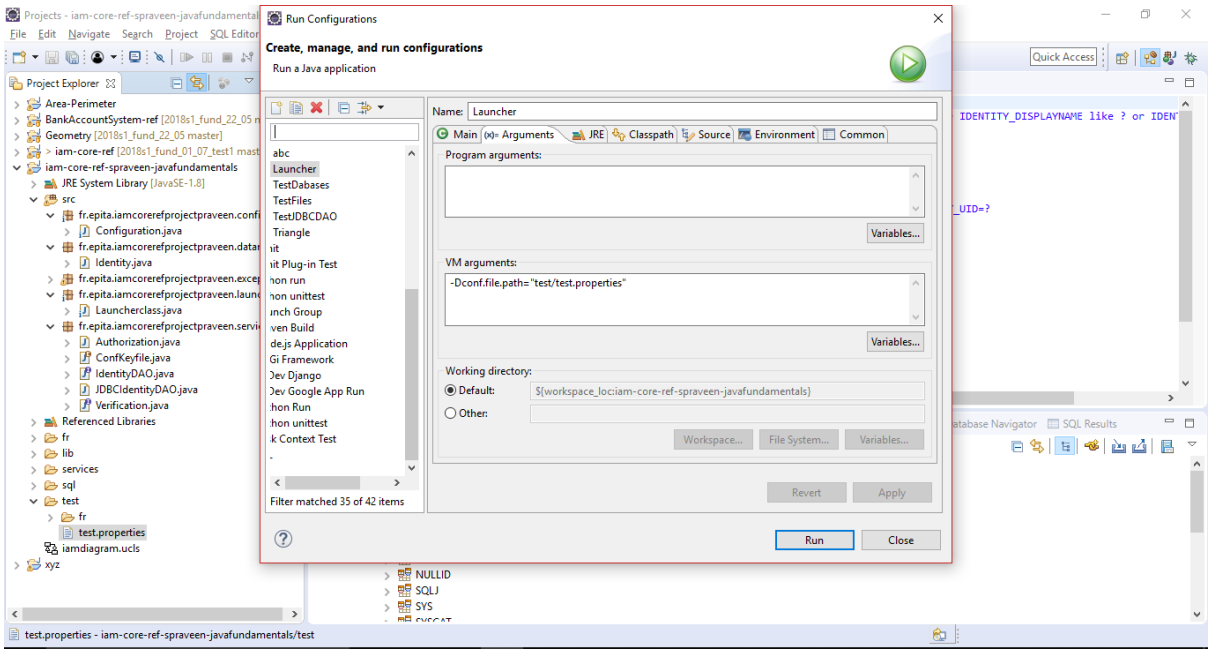
3. Click “**Run Configuration”** under the run launcher option.

4. In the window click “**Arguments”** and in the “**VM arguments Box”** enter the location of the properties files “**-Dconf.file.path="test/test.properties**" and then click run.

“ **Java properties file is used to store project configuration data or settings**”

Please use the below screenshots to run the launcher file with configurations (location of properties files passed as arguments)




Run Configurations
Create, manage, and run configurations
Run a Java application
Name: Launcher
Main
Arguments
JRE
Classpath
Source
Environment
Common
Program arguments:
Variables...
VM arguments:
-Dconf.file.path="test/test.properties"
Variables...
Working directory:
Default:
${workspace_loc:iam-core-ref-spraveen-javafundamentals}
Other:
Workspace...
File System...
Variables...
Revert
Apply
Run
Close
Launcher
TestDabases
TestFiles
TestJDBCDAO
Triangle
Filter matched 35 of 42 items
Project Explorer
Area-Perimeter
BankAccountSystem-ref
Geometry
iam-core-ref
iam-core-ref-spraveen-javafundamentals
JRE System Library [JavaSE-1.8]
src
Configuration.java
Identity.java
Launcherclass.java
Authorization.java
ConfKeyfile.java
IdentityDAO.java
JDBCIdentityDAO.java
Verification.java
Referenced Libraries
fr
lib
services
sql
test
test.properties
iamdiagram.ucls
xyz
Quick Access
NULLID
SQLJ
SYS
test.properties - iam-core-ref-spraveen-javafundamentals/test