



FEBRUARY 21, 2018

# TECHNICAL DOCUMENT


IDENTITY MANAGEMENT APPLICATION

UDHAYASHANKAR PALANIVEL

ROSHNEE MEENA

EPITA

Kremlin Bicetre



## Contents

1	Subject Description:.....	2
2	Subject Analysis: .....	2
2.1	Major Features.....	2
2.2	Application feasibility .....	2
2.3	Data Description:.....	2
2.4	Expected Results: .....	3
2.5	Algorithm Study: .....	3
2.6	Scope of the Application: .....	4
3	CONCEPTION .....	4
3.1	Data Structures .....	4
3.2	Global Application Flow .....	4
3.2.1	Authentication:.....	6
3.2.2	Menu to choose the functionality.....	7
3.2.3	Create an identity .....	8
3.2.4	Search an identity .....	8
3.2.5	Update an identity .....	9
3.2.6	Delete an Identity .....	9
3.2.7	Closing the program.....	10
3.3	Configuration Instructions.....	12

# **1 Subject Description:**

The Iam project is an Identity Management software developed to manage the identities and users dynamically.

## **2 Subject Analysis:**

### **2.1 Major Features**

- Authentication: A user should be authenticated to manage the identities
- The major features of this Iam project is to INSERT the identities, SEARCH the identities, UPDATE the identities and DELETE the identities of the users using this management software by an authenticated person.

### **2.2 Application feasibility**

- The application developed can be feasible to achieve the above features dynamically by using derby connection for database and JFrame and interface to dynamically manage the identity system.

### **2.3 Data Description**

- Here we use the database to store and retrieve the user details
- The Derby database can be used for this purpose. The configuration method can be developed for the purpose of global configuration, which helps to read the property file to get the database details
- The program can have an interface which is implemented by with the methods of all the functionalities required.

- We can have a class that implements the interface and has the methods to insert the user details, search the user details, update the user details, delete the user details and the connection method to connect to the database.
- Then another method which designed using JFrame for dynamic execution. It gets the details from the user and do the above functionalities dynamically.
- The program has a login class for the authentication. Only if the username and password match with the one in the database he/she can manage the identities.
- The program can have exception classes to throw exception if some problem occurs during any functionalities.
- It can have a logger system which logs the program details in the local file in our system. Which helps us to find when and where the error has occurred.

## 2.4 Expected Results

- When we run the program, it will display a page which asks for the User credentials. If the user credentials are correct it will take to next frame where it shows the options create, delete, update and search.
- When the user selects the option, it will take to the respective pages. Once the required details are filled and done button is pressed the operation is performed in the database.

## 2.5 Algorithm Study

- Table is created in the database to add the user details and update.
- The database is connected to the java using JDBC
- The create, search, update and delete functionality are designed based on the linear algorithm.

- The authentication page is designed so that only the authenticated users can perform the above functionality
- Logger systems and exception classes are developed for improving the program standards.

## 2.6 Scope of the Application

- The application management is restricted to the authenticated users. It manages both the identities and the users. This is helpful for the dynamic management.

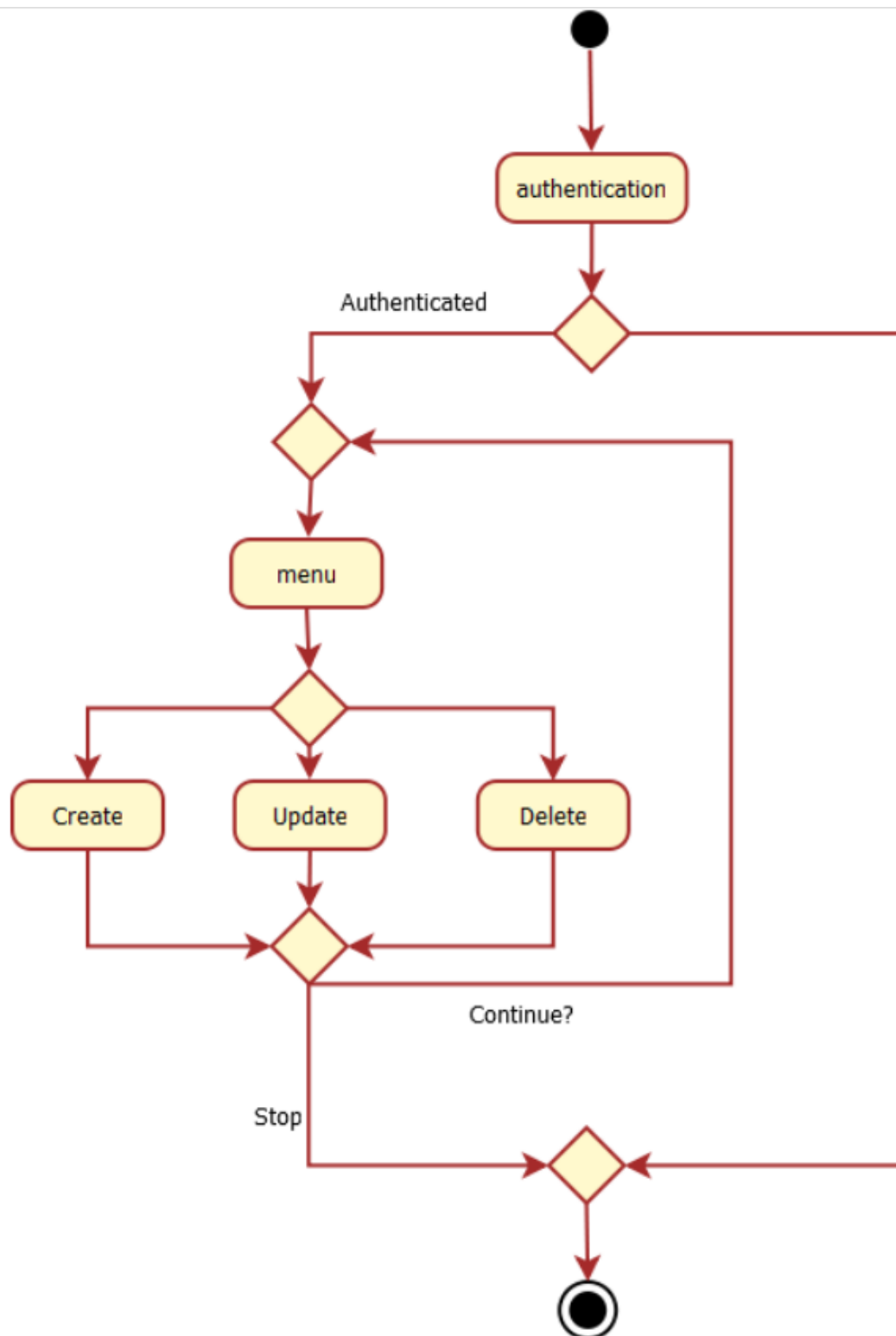
# 3 CONCEPTION

## 3.1 Data Structures

- Identities: The identities are the entities which is will managed by the application. It has the field display name, email id and the uid All the three field are String as per the requirement.
- Users: The authorized user details will be stored in the database. The database will be having username and the corresponding password. Both are of string type.

## 3.2 Global Application Flow

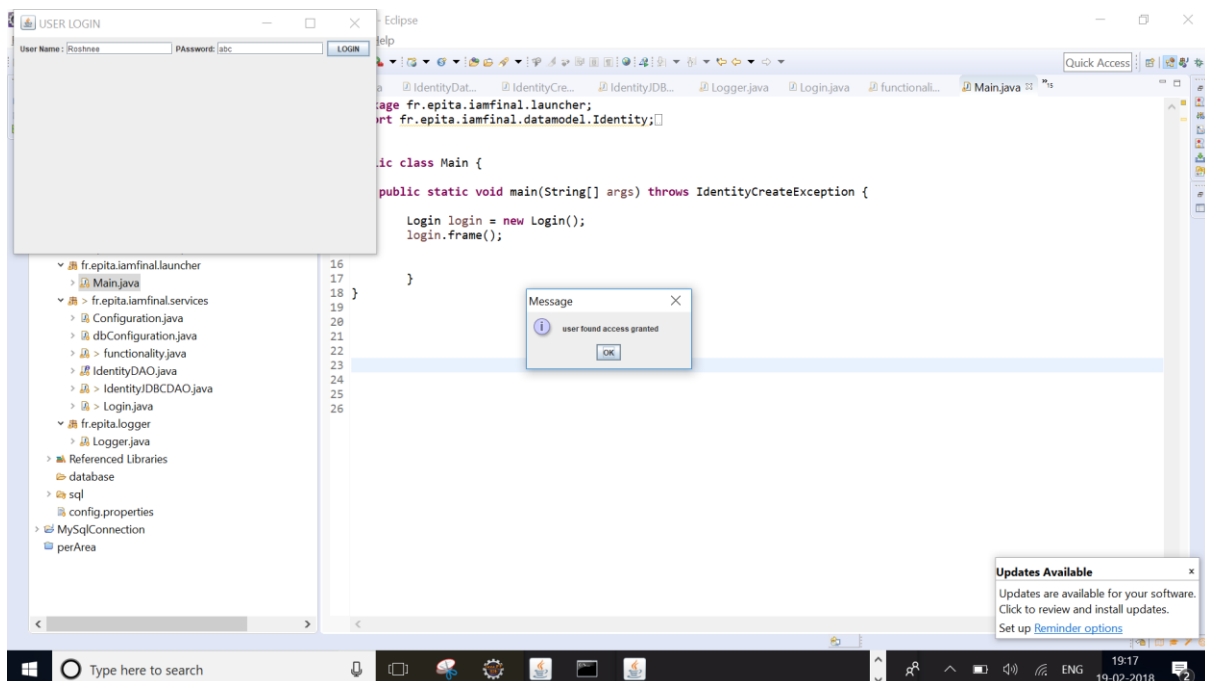
The expected flow given in the requirement is



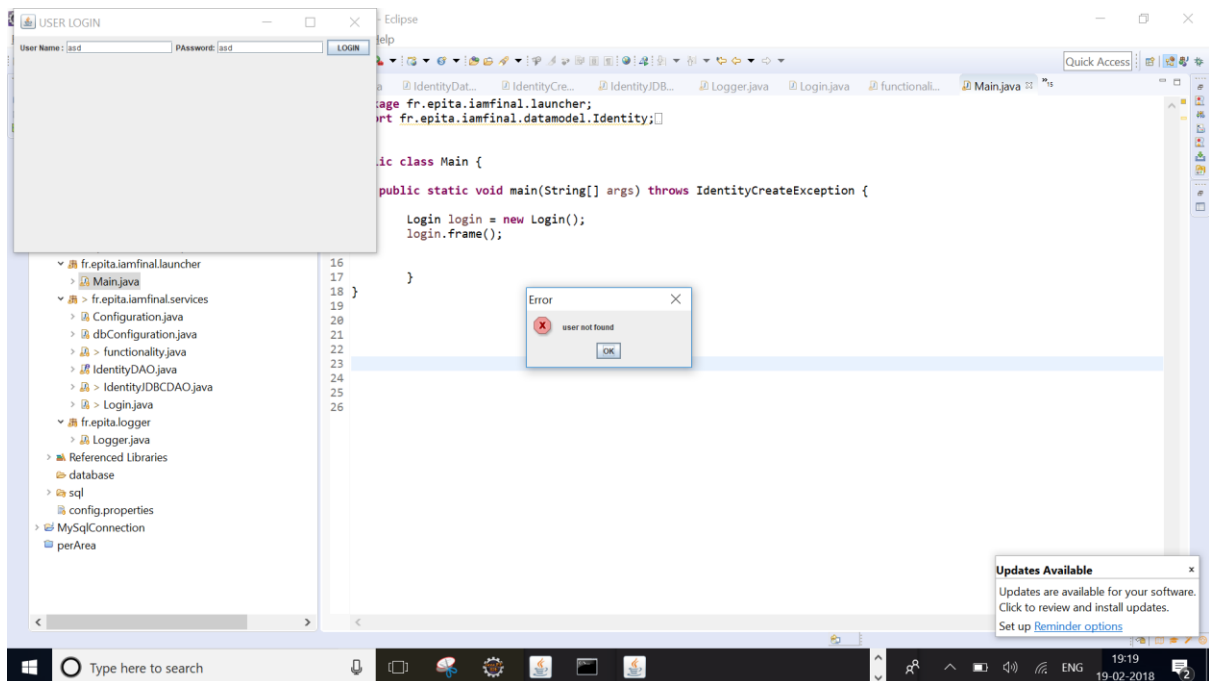
The way in which the project is designed is explained below.

### 3.2.1 Authentication:

- In the login class is created, when the application is made to run. It asks for the user credentials. Only if the details match the user can manage the identities.
- If login details are correct

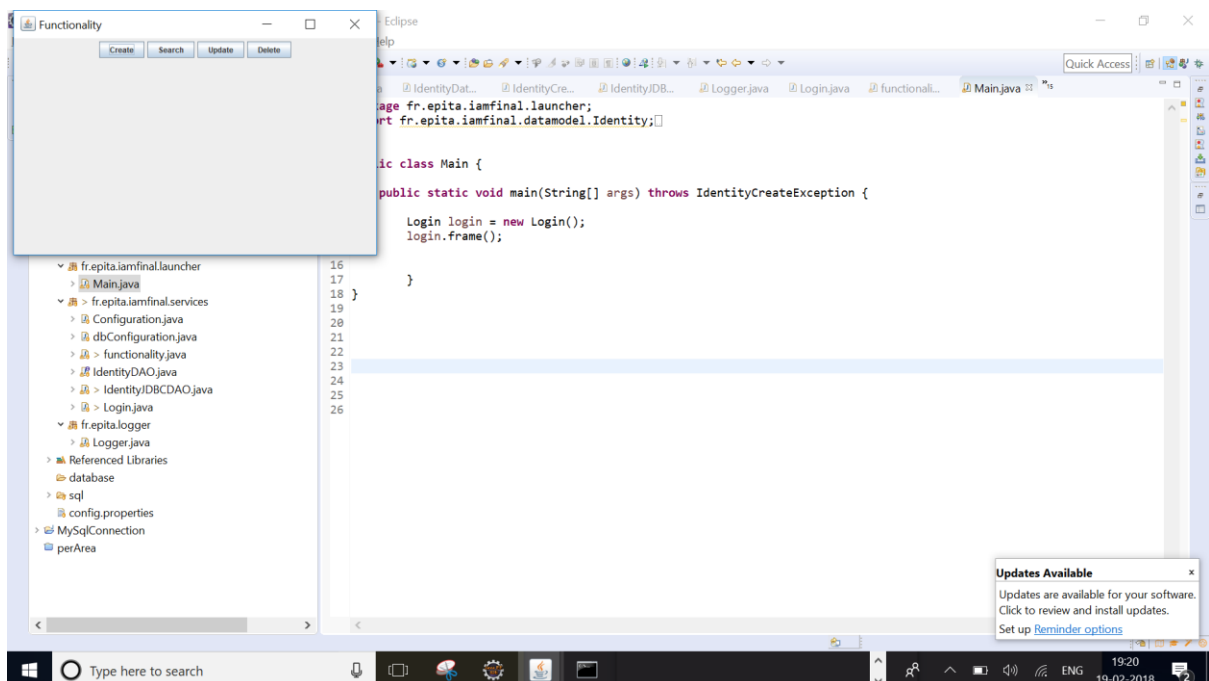


- If login details are incorrect



### 3.2.2 Menu to choose the functionality

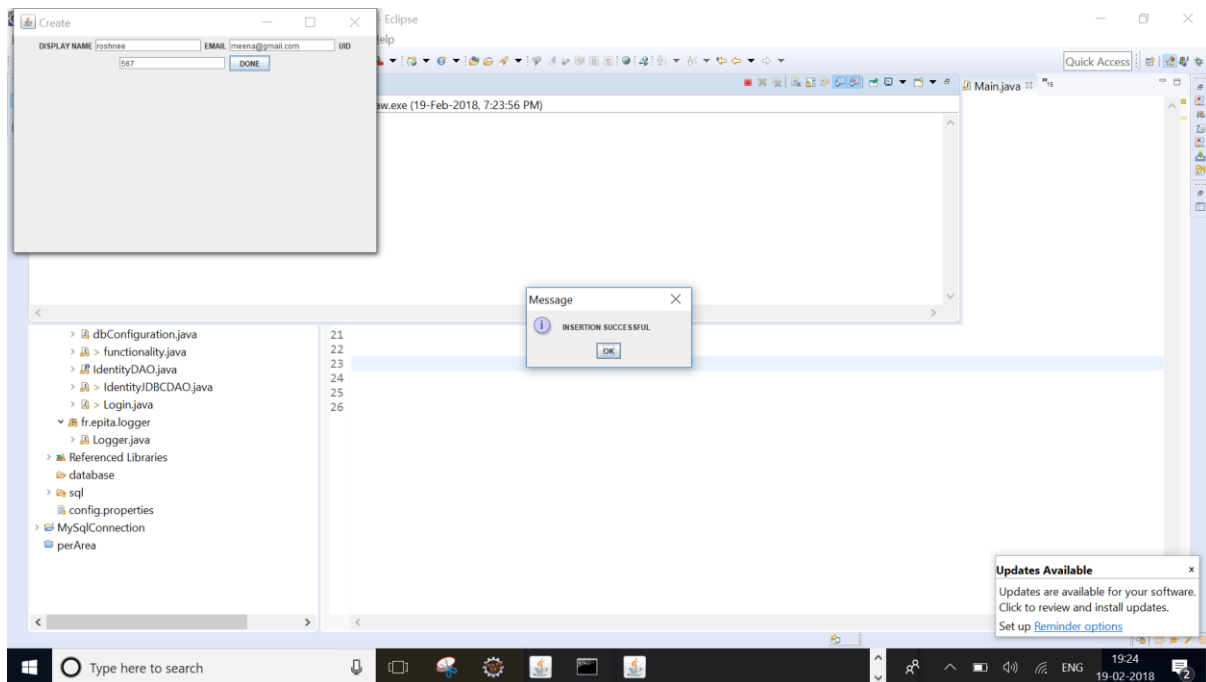
After login it displays a menu to choose create, search, update and delete. By clicking on it the corresponding functionalities are performed.





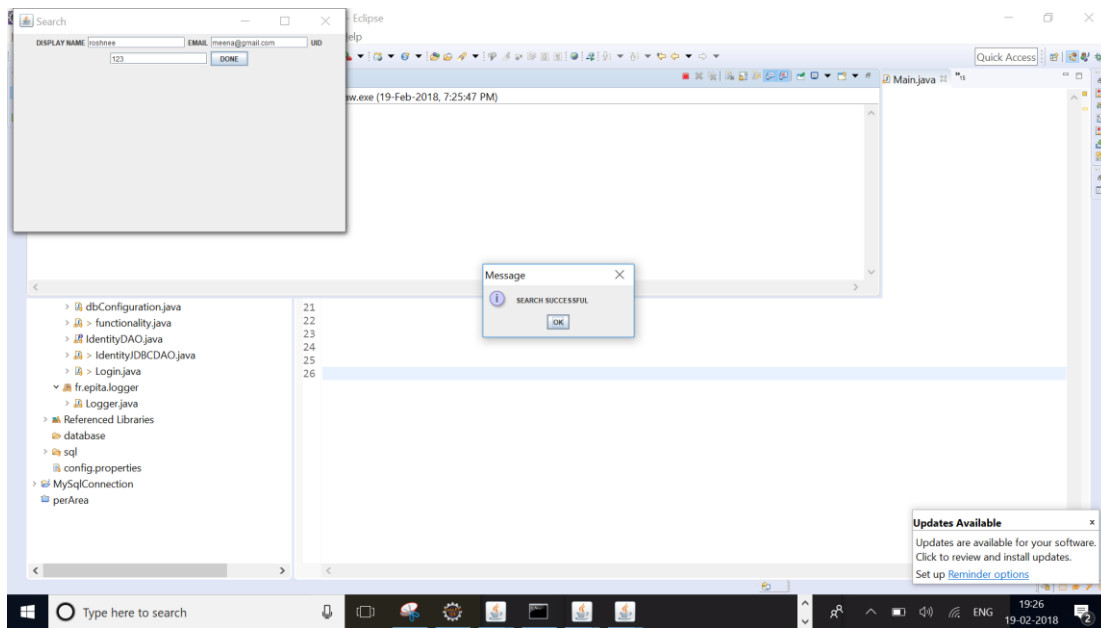
### 3.2.3 Create an identity

The identities are created when the user enters the display name, email id and uid. And if the identities are inserted it will display “identities are created” in the console.



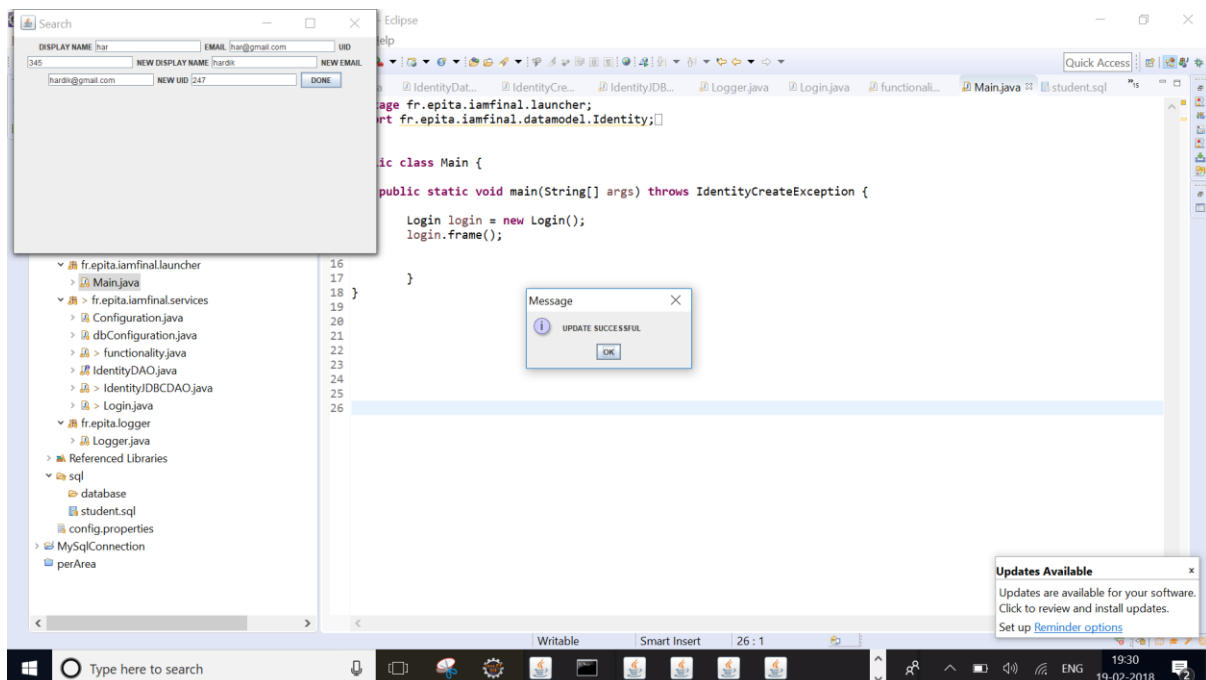
### 3.2.4 Search an identity

The user enters the identities to searched and the searched identities are returned as list. If the identities are not available it displays “identity not available” in the console.



### 3.2.5 Update an identity

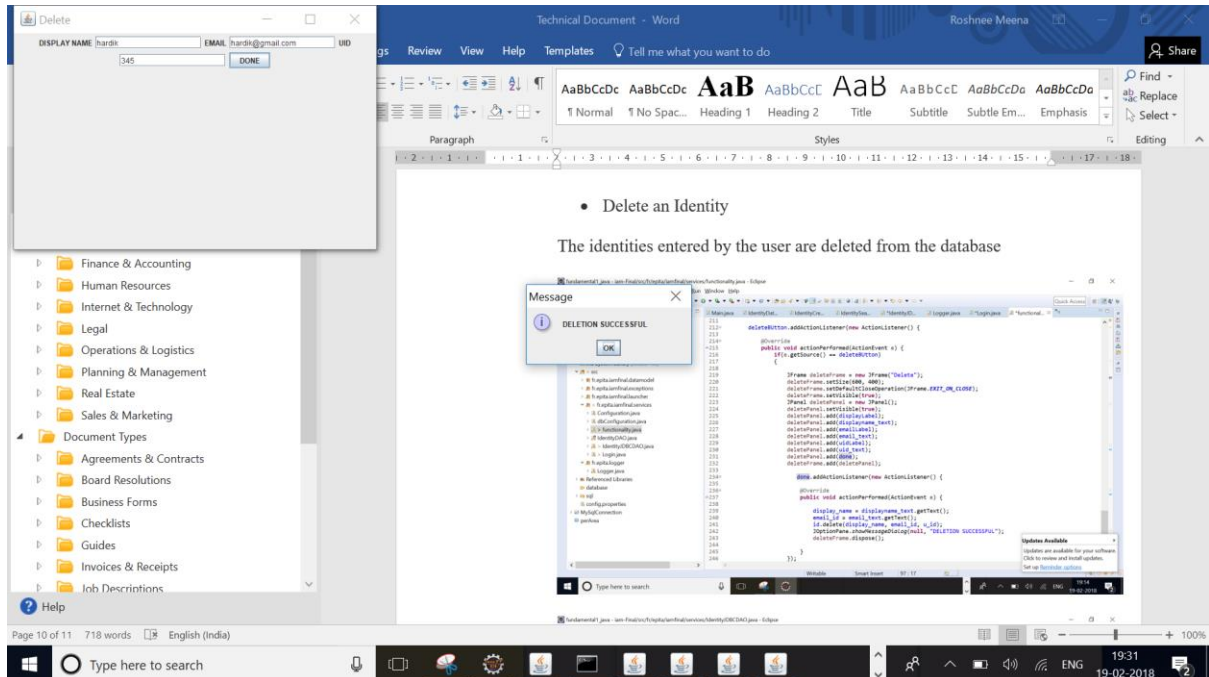
If the user enters the identities to be updated and the new identity. The old one gets updated with the new one in the database. If the identities are not available it displays “identity not available” in the console



### 3.2.6 Delete an Identity

The identities entered by the user are deleted from the database.

If the identities are not available it displays “identity not available” in the console

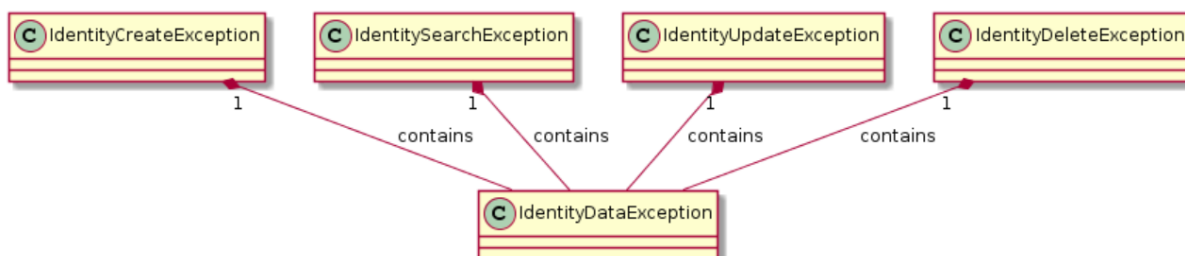
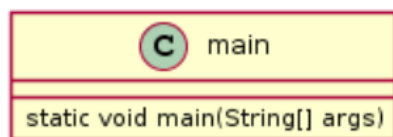
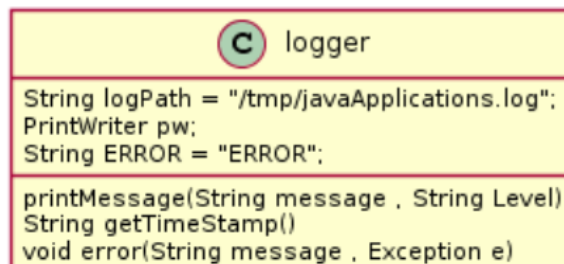
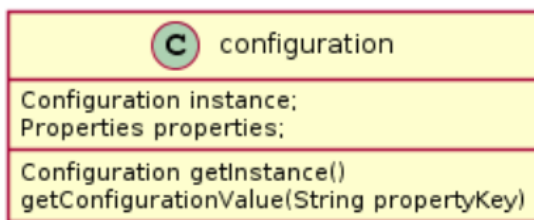
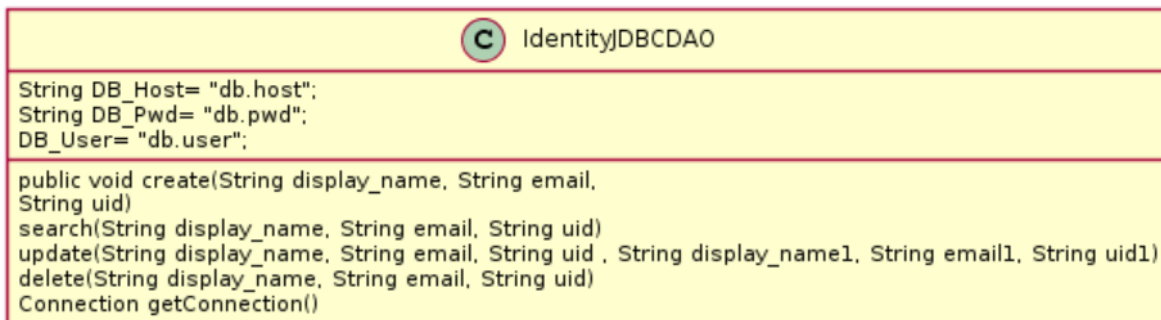
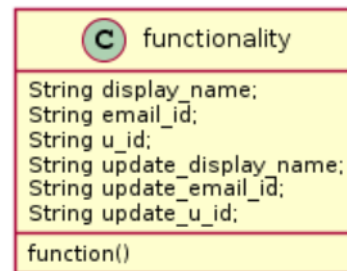
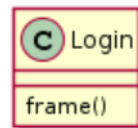


### 3.2.7 Closing the program

The “X” button at the top of the panel will close the program.

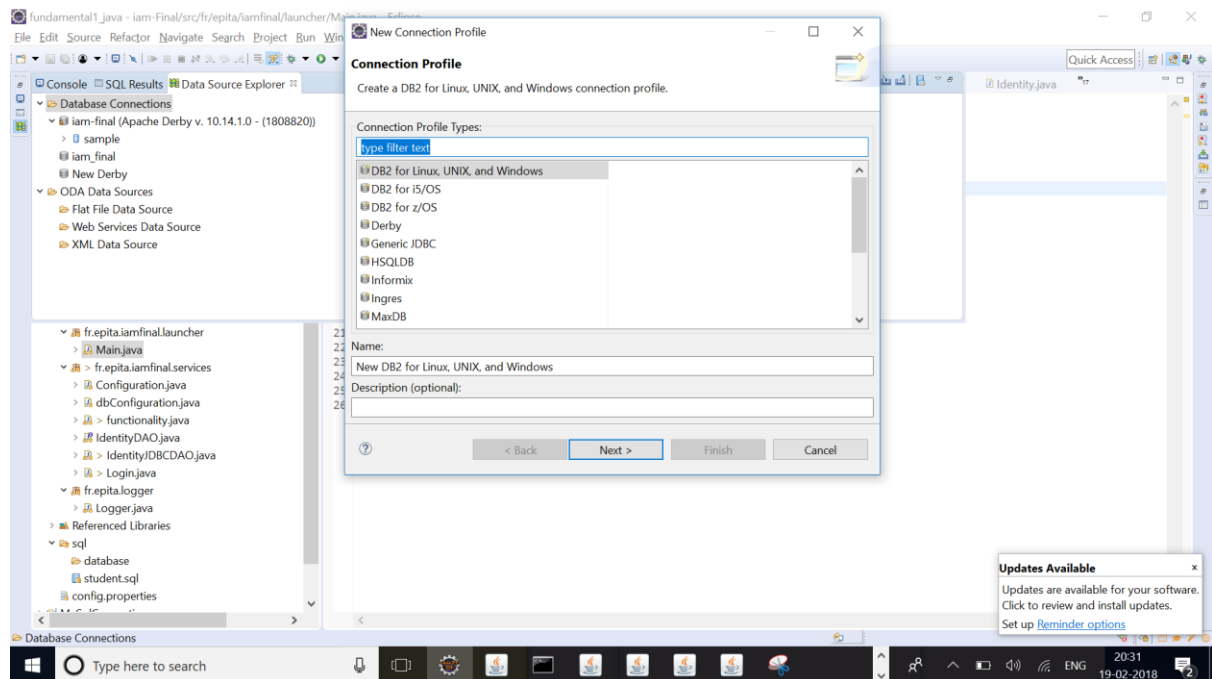
## 3.3 Global schema and Major features schema

This is the class diagram of all the classes involved in the program



### 3.4 Configuration Instructions

We should have installed the Java JDK, Eclipse oxygen Java EE and the Derby



1. In the contextual window select Derby. Click Next.
2. In the next window click on the “New Driver definition button”
3. Select any Definition on the Name/Type tab and then move to the JAR List tab.
4. On the JAR List tab click on the Add and navigate to the lib folder inside your Derby database. select the derbyclient.jar file
5. Make sure the Derby Database is running by clicking on Test Connection button, it should succeed.
6. Click on Finish button.
7. Schema creation
8. Once the above process is done, within the Data Source Explorer window you should be able to expand and see the schemas
9. Right click over the New Derby and select: Open SQL Scrapbook.

10. Once the new window opens, in the Project Explorer navigate: iam-final/SQL/identity table creation and open the file. Copy the file contents to the Scrapbook window and execute.
11. Select the query and right click “Execute selected text”.
12. Execution should be successful.
13. Go back to the Data Source Explorer window and right click on the Schemas folder, select Refresh. You should now see a Schema with a name that matches your user name, inside it the Identities table should have been created.

Once all of this is done close the Scrapbook and the Identity table creation sq. windows