****

**Submitted by**

**Rahul Kumar Thai Valappil**

**IDENTITY MANAGEMENT SYSTEM**

**Table of Contents**

**1 Subject description...................................................................................**

**2 Subject Analyses.....................................................................................**

2.1 Major features..........................................................................

2.2 Application Feasibility................................................................

2.3 Data description........................................................................

2.4 Expected results.........................................................................

2.5 Scope of the application (limits, evolutions)..............................

**3 Conception………………………………………………………………………………………..**

3.1 Data structures........................................................................

3.2 Global application flow........................................................

3.3 Global schema and major features schema................

**4 GUI descriptions.................................................................**

**5 Configuration instructions.........................................................................**

**6 Commented Screenshots..........................................................................**

**7 Bibliography……………………………………………………………………………………….**

**1. Subject description:**

This Project Identity Management System is taken to develop an application to manage the information of users in a database. The term **“User”** is used as “the person who will manage the information” and the term **“Identity”** is used to signify the “information of user in the system”.

**2. Subject Analysis:**

**2.1 Major features:**

1. it’s a **Java (Swing) GUI application** where user can perform below functions related to his/her information:

* Create an Identity
* Update an Identity
* Delete an Identity
* View if the identity exists in the database

2. Application provides the user facility to store data in **XML or SQL data base** format. User can configure the data type in **configApp.xml**, which should be located in **“C:\iamcore” path**.

**2.2 Application Feasibility:**

The development of this application with the given features is quite feasible and hence, the application has been developed.

**2.3 Data description:**

Below are the types of data we created to make this application achieve its given features:

* **Class Identity:**

We have defined the variables (features of identity) in this class. And also, basic methods like getters and setters for these features are defined here.

**Variables:**

* Private int uid;
* private String email;
* private String displayName;
* private Date Birthdate;

**Methods:**

* Identity()
* int getUid()
* void setUid(int)
* String getEmail()
* void setEmail(String )
* String getDisplayName()
* void setDisplayName(String )
* String toString()
* **Interface IdentityDAOInterface:**

This interface defines the behaviour of Identity and we have declared and defined the outline of this behaviour is form of methods here in this interface:

**Methods:**

* + public void create(Identity identity);
  + public List<Identity> readAll();
  + public List<Identity> search(Identity criteria);
  + public void update(Identity identity) throws DaoUpdateException;
  + public void delete(Identity identity);
  + public void close()
* **Class IdentityXmlDAO:**
* Class IdentityXmlDAO implements the interface IdentityDAOInterface by overriding all the methods.
* This class handles the identity data in xml format to achieve the functionalities of the application i.e Creation, Update and Deletion of an identity as well as Viewing the identity.
* **Class IdentityJdbcDAO:**
* This class implements the interface IdentityDAOInterface by overriding all the methods.
* This class handles the identity data in sql format to achieve the functionalities of the application i.e Creation, Update and Deletion of an identity as well as Viewing the identity.
* To perform all these functions, a connection is setup with the database i.e. **Apache Derby**, which is configurable through configApp.xml
* **Class Configurations:**
* This class handling the basic configuration data for the application. It will store information for creating XML or Database data access object in map structure.

Map<String, String> xmlDAOConfMAp

Map<String, String> dbDAOConfMAp

**2.4 Expected results:**

The application is supposed to perform following four functions by choosing the appropriate data model from configuration file (XML or Derby database).

* Create an Identity
* Update an Identity
* Delete an Identity
* View an Identity

**2.5 Scope of the application (limits, evolutions):**

* **Limits**
* Currently developed as GUI Dialog based application so that remote user cannot access the application.
* Deletion or update can perform only one identity at a time.
* No user authentication procedure is provided
* **Evolutions:**
* Application can be enhanced as a web application.
* Multiple Identities can be deleted or updated at a time.
* User authentication can be implement for security measures.

**3. Conception:**

**3.1 Data Structures**

Application mainly using Java List and Map for storing or accessing Identity or configuration data’s. Also application handling identity data through Identity Class. Application mainly using following patterns.

* Single pattern:

Used mainly for handling configuration details and Identity management.

* Data Access Object patterns

Patterns will choose how to handle the identity data. Currently application management the data in XML as well as SQL.

**3.2** **Global application flow**

E:\EPITA_StudyMaterials\Java\Project\FinalProject\Docs\Java Logic.png

* 1. **Global schema and major features schema**
  + **Identity Management Sequence Diagram with XML DAO**

**E:\EPITA_StudyMaterials\Java\Project\FinalProject\Docs\XML Sequence.png**

* + **Identity Management Sequence Diagram with Database DAO**

**D:\Java\Rahul Varun\DAO Sequence.png**

* + **UML Diagram for Identity Management System**

**E:\EPITA_StudyMaterials\Java\Project\FinalProject\Docs\UML Diag.png**

**4. GUI Operations Description:**

As this is a GUI application, user has to choose corresponding operations from the IAM Dialog based GUI application. User can perform the following functionalities:

1. **Create an Identity**
2. **Search an Identity**
   1. Update an Identity
   2. Delete an Identity
3. **Close**

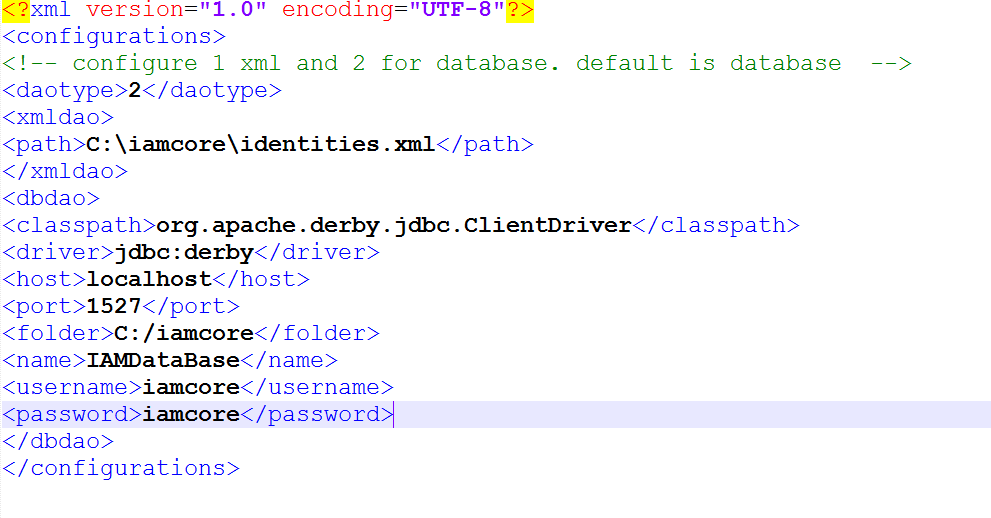
**Option 1**: If user inputs 1, he will be prompted to give Display Name, Uid, Birthdate and Email for identity one by one.

**Option 2:** If he chooses option 2, he will be prompted to enter Display Name and email id .Search result will show in table format to the user and user can select the identity entry which he wants to update or delete. User cannot change Uid of an Identity.

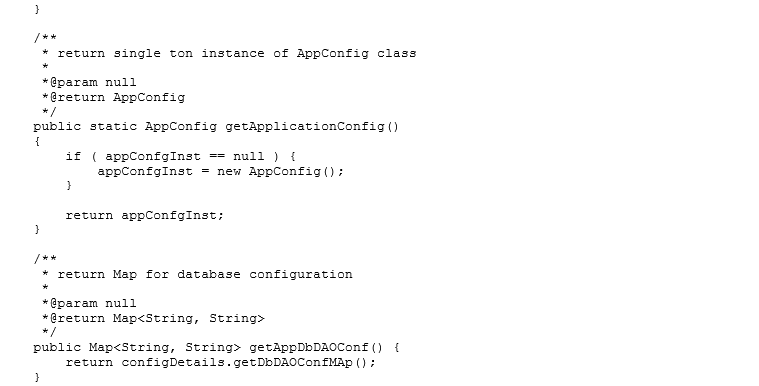
**Option 3:** This will close the application.

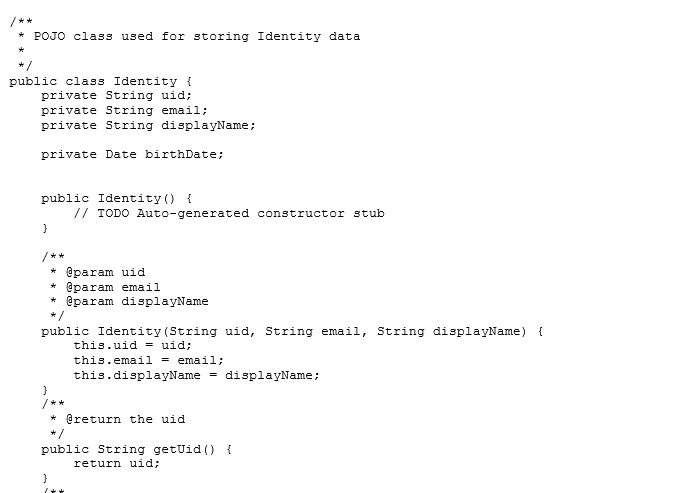
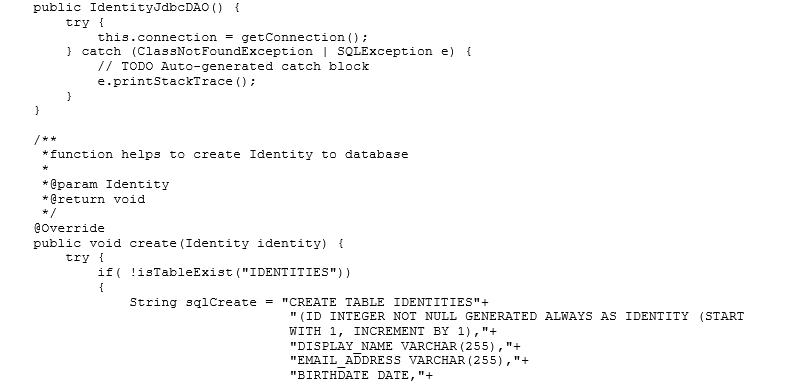
**5 Configuration instructions**

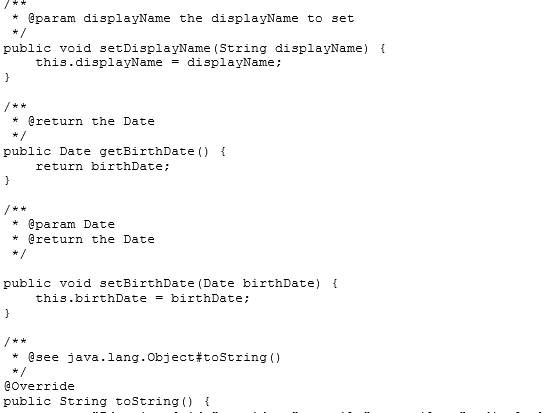
1. Install latest Java run time environment.
2. User should install the Apache Derby database before doing any operation and start the derby Server by executing startNetworkServer.bat from Derby installation folder. Please visit following links for more details <https://db.apache.org/derby/papers/DerbyTut/install_software.html>
3. User needs to configure the Application from **C:/iamcore/configApp.xml** path before starting the Identity management system. Following things can be configurable through this file.

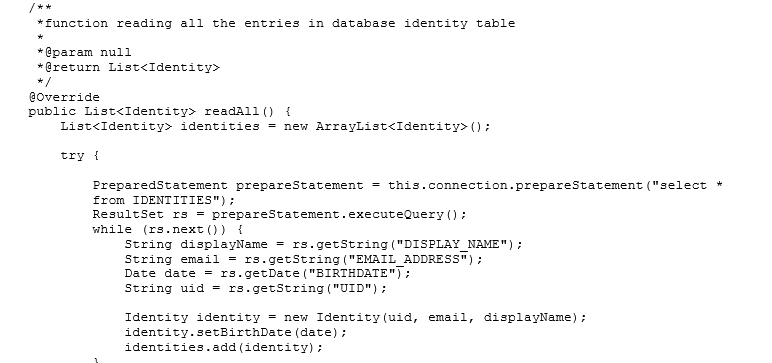


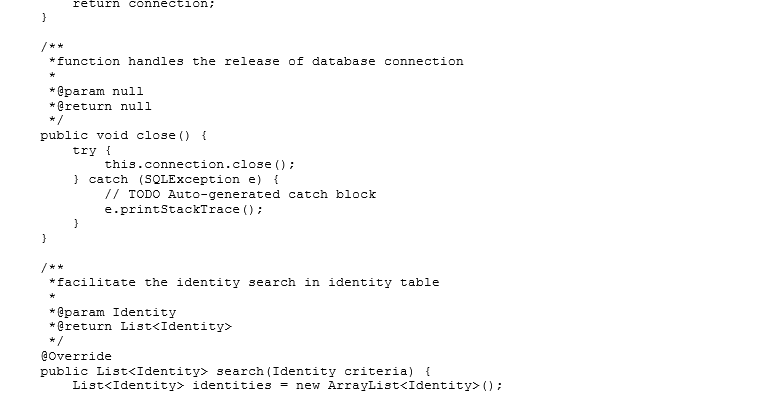
**6. Commented Screenshots**

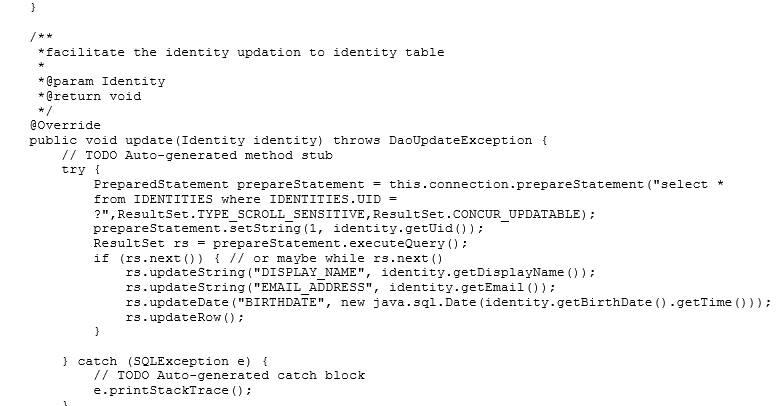
****

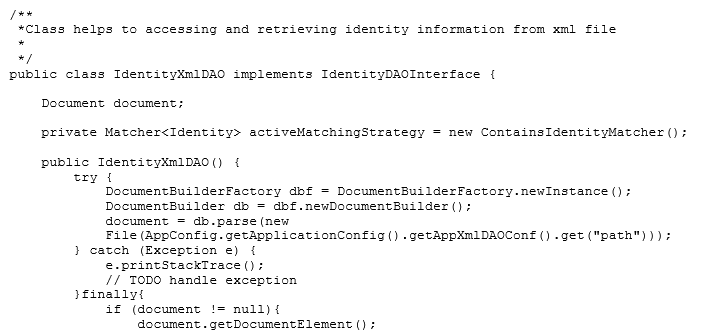
****

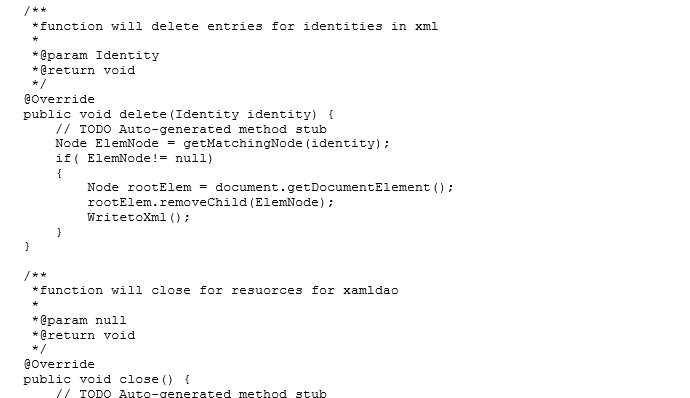
****

****

****

****

****

****

**7. Bibliography**

1. <http://thomas-broussard.fr/work/java/courses/index.xhtml>

2. <https://db.apache.org/derby/papers/>

3. <https://docs.oracle.com/javase/tutorial/>

4. <http://www.w3schools.com/>