Aadhar-Based Citizen Registration and KYC Services

An Object-Oriented Programming in Java Project by

Name	Ronit Ray, Abhirup Patra
Roll Numbers	19, 80
Section	CSE-3A
Semester	5th

University of Engineering and Management, Kolkata

ACKNOWLEDGEMENTS

Computer science is a field where you can know it all but still know nothing at all- you spend a semester studying a subject and believe you have a firm grasp on its concepts, but when you are plunged into the industry and expected to work in the area, you realize things aren't working the way you'd like them to, and your knowledge is severely lacking in several ideas. This is why project work is so important- it backs up theoretical knowledge with solid practical implementation skills and helps a student build strong foundations.

We really enjoyed building this project and would like to thank Prof. Souvik Chatterjee, Prof. Moumita Basu, and Prof. Sudeshna Kundu for their guidance throughout the semester, and for introducing OOP concepts and Java in an engaging and simple manner. We hope this project shows how far we have come this semester, and is an acceptable one.

-Ronit Ray, Abhirup Patra

October, 2017.

INTRODUCTION

Introduced by the Unique Identification Authority of India (UIDAI), Aadhar is a citizen registration platform and biometric database that serves as a singular, unified record for citizen data for the entire country. The project was launched in 2009 and over the next 8 years has grown to become the world's largest citizen ID system with over 1.171 billion unique records. It started out initially as a proof of residence and citizenship for Indian citizens of all ages, but since the NDA government took power in 2014, Aadhar has taken centre-stage as the go-to citizen identification database for a number of applications in all walks of life.

Some uses of Aadhar include:

- Creating bank accounts
- Issuing ration supplies
- Biometric attendance systems
- Registration of phone numbers
- Know-Your-Customer or KYC Records for all shopkeepers

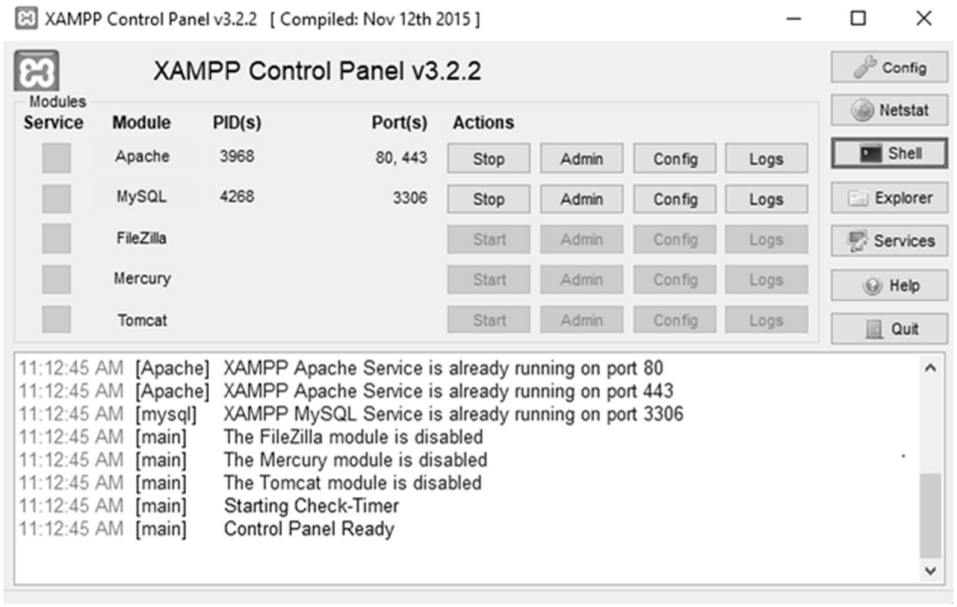
We live in a digital world and it is hence important for our citizenship verification methods to move forward from the days of paper-based PAN cards or Voter ID cards. Aadhar is a fully digital platform that securely stores a citizen's data post registration in the cloud, and is accessible at any time through a secure portal by the person himself or a registered user.

This project is an attempt to simulate a console for registration for the Aadhar platform and another console for the KYC procedure for any shopkeeper. For obvious reasons, we have had to simplify the Aadhar code by eliminating fingerprint and iris scans, QR codes and reducing the 12-digit code to a 5 digit ID. The project serves as a proof of concept that this sort of a platform can be built with relative ease, and the extension to a real-life Aadhar database is only a matter of implementation that can be performed with relative ease given the time and resources, computing, monetary, or otherwise.

The project has been implemented using Core Java, Java Swing and AWT for the GUI, and MySQL and JDBC for the database.

The Database:

MySQL was used for the database server, hosted on localhost (127.0.0.1) using XAMPP and phpmyadmin.



The database structure is simple enough, with a 5-digit ID set as an auto-increment field (meaning it will be generated by itself, starting with 10000), name, gender, address, city, and state- strings of various lengths, pin code- a 6-digit integer and phone number- a 10-digit integer.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**id**	int(5)			No	*None*		AUTO_INCREMENT
2	**name**	varchar(20)	latin1_swedish_ci		No	*None*		
3	**gender**	varchar(1)	latin1_swedish_ci		No	*None*		
4	**address**	varchar(20)	latin1_swedish_ci		No	*None*		
5	**city**	varchar(10)	latin1_swedish_ci		No	*None*		
6	**state**	varchar(15)	latin1_swedish_ci		No	*None*		
7	**pin**	int(6)			No	*None*		
8	**phone**	bigint(10)			No	*None*		

The ID has also been used as a primary key indexed using a B-TREE by MySQL.

Connecting to MySQL using Java

MySQL provide a database driver in the form of a JAR file available for free on dev.mysql.com.

Further, Java has an API known as JDBC for transactions on databases. The JDBC API is written entirely on Java, which is why it is preferred over the old standard ODBC which was written in C and is unsecured and platform dependent. There are four kinds of JDBC drivers:

1. JDBC-ODBC Bridge Driver
2. Native API Driver
3. Network Protocol Driver
4. Thin Driver

We use the thin driver which is written in Java for convenience and best performance.

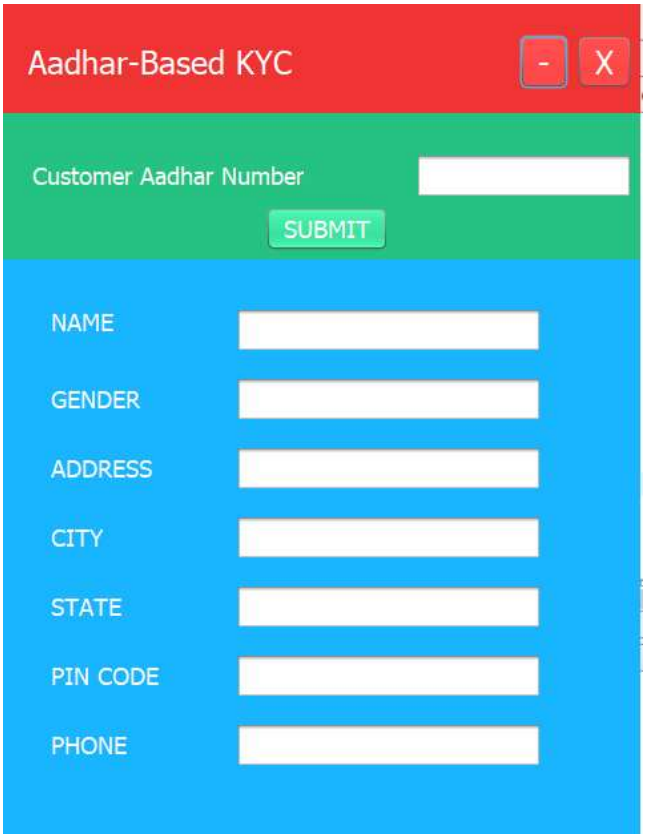
The DBConnection class is created to be used throughout the project to establish a connection to the database.

```
import java.sql.*;

public class DBConnection

{

    private static final String USERNAME="dba";

    private static final String PASSWORD="dbadmin";

    private static final String CON=
"jdbc:mysql://localhost:3306/aadhar_db";


    public static Connection getConnection() throws SQLException

    {return DriverManager.getConnection(CON,USERNAME,PASSWORD);}

}
```

The usage of this is `Connection c= DBConnection.getConnection();`

This is a good OOP practice because it keeps the login and connection details of the database separate from the application itself.

Form Design



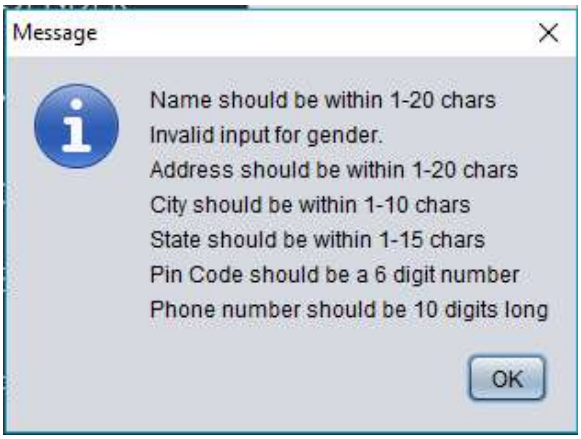
There are two forms, one for registration and one for the KYC. The registration form will take the fields as input, and after validation, will insert them into the database. It is meant for government officials to use for registering citizens at authorized venues.

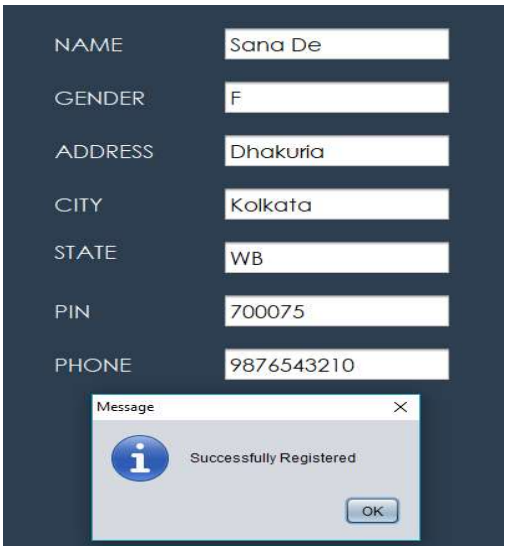
The KYC is for shopkeepers and service providers. All they need to do is enter a valid Aadhar number, after which all relevant details for the citizen are fetched and displayed in the fields. The provider can then use this data as needed.

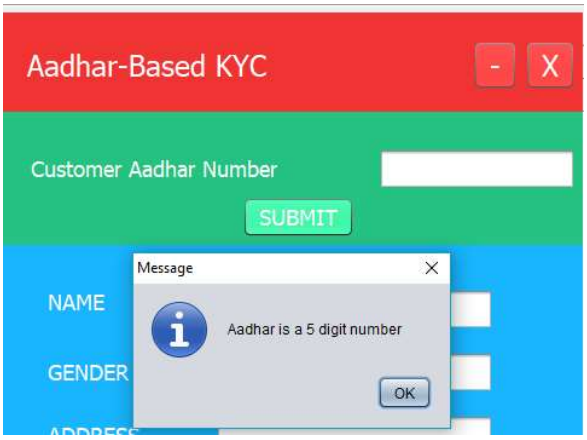
The forms are designed entirely on Java Swing on NetBeans using a Visual Studio-like palette without writing any code. Validation for event-listeners had to be coded separately and inserted into the classes.

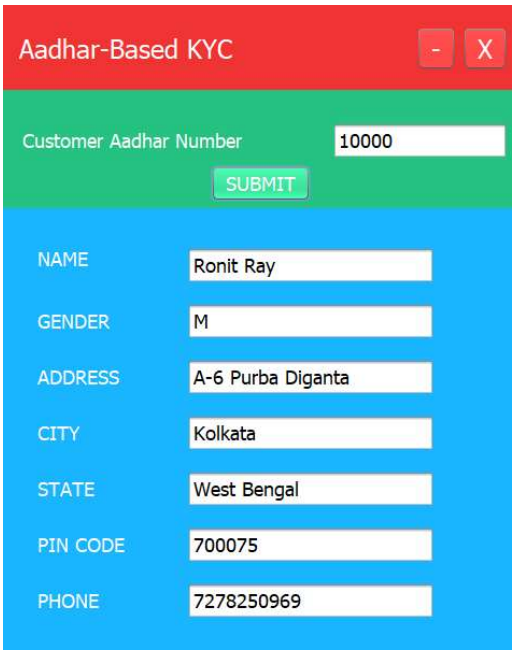
OUTPUT

The validation and success screens for the forms are given below. Each validation is performed separately in an if-else block and will be thrown as an error message using a JOptionPane showMessageDialog instance.









CONCLUSION

In this project, a MySQL-based database was linked to a Java application and two JFrame based forms were implemented. The first form takes citizen data as input, validates it and inserts it into the database. The second form is for KYC. It searches an input Aadhar ID in the database and fetches relevant data. The project was implemented successfully and is fully functional. Source code is available upon request and will be uploaded to github.com/RonitRay.

REFERENCES

- **https://www.javatpoint.com/java-jdbc**
- **Awais Mirza's YouTube tutorials for JDBC**
- **https://en.wikipedia.org/wiki/Aadhaar**
- **https://dev.mysql.com/downloads/connector/**
- **https://netbeans.org/kb/docs/java/quickstart-gui.html**
- **KeepToo & VertexDigitalArt tutorials for Flat and Material Design on YouTube.**