NJC LABS SCREENING PROCESS

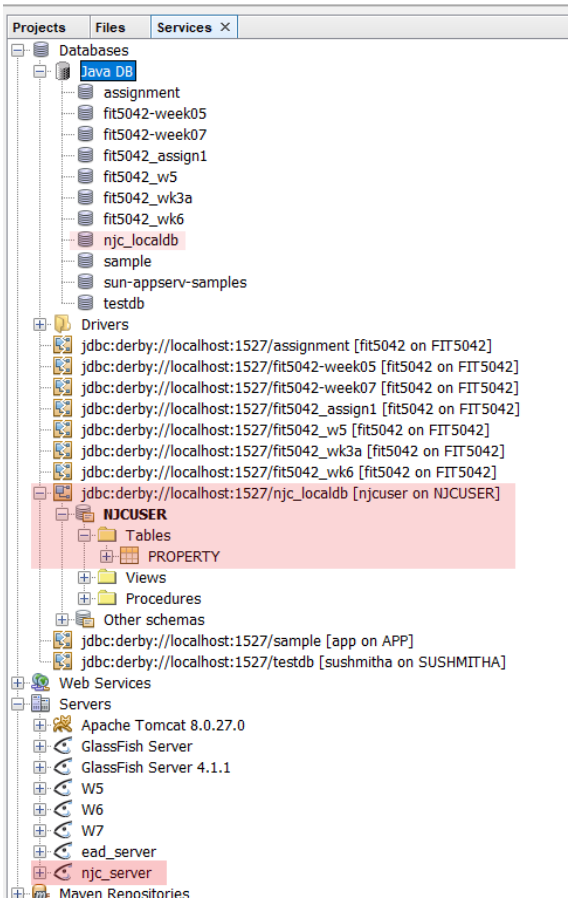
Requirement: Develop a REST API java to retrieve some data from a database table.

TASK 1

Setup a local database, create a table, insert some records using any database client

1. Create server =“njc_server”.
2. Create database =“njc_localdb” with username & password = “njcuser”.
3. Execute following command to create a table:

```
create table "NJCUSER".PROPERTY
(
   PROPERTYID INTEGER not null PRIMARY KEY,
   ADDRESS VARCHAR(255),
   NUMBEROFBEDROOMS INTEGER,
   PRICE DOUBLE
)
```



Sushmitha Siddam

4. Insert data into database by executing following insert commands:

   INSERT INTO NJCUSER.PROPERTY VALUES(51,'22 Boston Ave, Malvern East VIC 3145, Australia', 3, 520000.0);

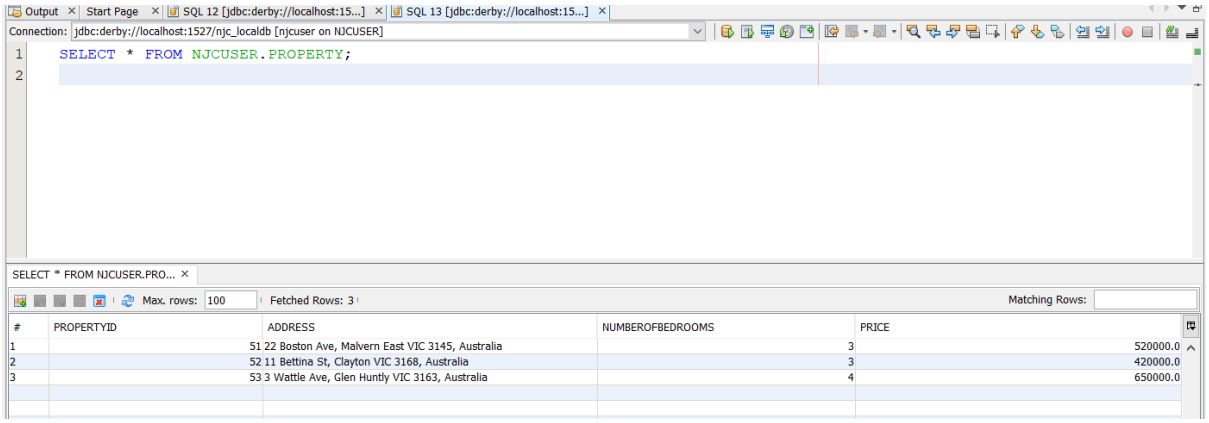
   INSERT INTO NJCUSER.PROPERTY VALUES(52,'11 Bettina St, Clayton VIC 3168, Australia', 3, 420000.0);

   INSERT INTO NJCUSER.PROPERTY VALUES(53,'3 Wattle Ave, Glen Huntly VIC 3163, Australia', 4, 650000.0);

TASK 2

Write SQL to retrieve records from the above table using the same database client

SELECT * FROM NJCUSER.PROPERTY;



TASK 3

Write java code to retrieve the same records

1. Create a JAVA Project = njc-accessDB.
2. Add JAVA DB Driver library to the source package.
3. Create a JAVA Class= NjcAccessDb.java with the following JAVA Code:

```
package njc.accessdb;

/**
 * @author sushmitha
 */
import java.sql.*;

public class NjcAccessDB {

   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {
      // TODO code application logic here
      try {
```

```
      Class.forName("org.apache.derby.jdbc.ClientDriver");
    } catch(ClassNotFoundException e) {
      System.out.println("Class not found "+ e);
    }
    try {
      Connection con = DriverManager.getConnection(
        "jdbc:derby://localhost:1527/njc_localdb","njcuser", "njcuser");

      Statement stmt = con.createStatement();
      ResultSet rs = stmt.executeQuery("SELECT * FROM NJCUSER.PROPERTY");

      while (rs.next()) {
        int propertyid = rs.getInt("PROPERTYID");
        String address = rs.getString("ADDRESS");
        double numberOfBedrooms = rs.getDouble("NUMBEROFBEDROOMS");
        double price = rs.getDouble("PRICE");
        System.out.println("Property ID: "+propertyid);
        System.out.println("Address: "+address);
        System.out.println("Number Of Bedrooms: "+numberOfBedrooms);
        System.out.println("Price: "+price+"\n\n");
      }
    } catch(SQLException e) {
      System.out.println("SQL exception occured" + e);
    }
  }

}
```

Output:

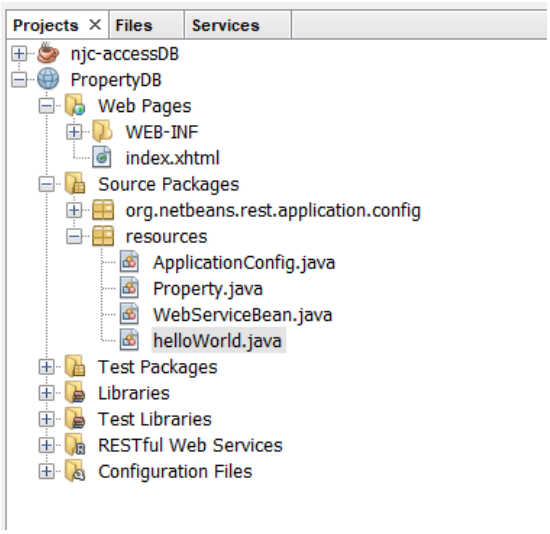
```
run:
Property ID: 51
Address: 22 Boston Ave, Malvern East VIC 3145, Australia
Number Of Bedrooms: 3.0
Price: 520000.0


Property ID: 52
Address: 11 Bettina St, Clayton VIC 3168, Australia
Number Of Bedrooms: 3.0
Price: 420000.0


Property ID: 53
Address: 3 Wattle Ave, Glen Huntly VIC 3163, Australia
Number Of Bedrooms: 4.0
Price: 650000.0
```

TASK 4

Create a REST API to retrieve the same records and build this API using JAVA.

1. Create a JAVA Web Project = “PropertyDB”.
2. Create a package “resources” in Source Packages.
3. Create “Entities from Database” from the created database njc_localdb.
4. The entity class, Property.java is created.
5. Create a “WebServiceBean” JAVA class which is a Session-scoped managed bean.
6. Create a Persistence Unit for Property Entity and initialise it in the managed bean.
7. This Persistence Unit acts as the entity manager which fetches data from the database using a Named Query.
8. Entity Manager fetches all the properties from the data base.

REST API allows to build a web app that provides access to database in a way that it can be reused by multiple applications. REST API generally uses HTTP Requests (like GET and POST methods) to access data.
For this task, helloworld.java is the REST API which ideally manages the entity manager and contains all HTTP Requests. However, in this task the entity manager is created in the managed bean as certain services like fetching all data can be provided by managed in real-time. Since this is not an elaborate project, the persistence unit is created in Managed Bean.



Output:

Property List

Property ID: 51
Property Address: 22 Boston Ave, Malvern East VIC 3145, Australia
Number of Bedrooms: 3
Price: 520000.0

Property ID: 52
Property Address: 11 Bettina St, Clayton VIC 3168, Australia
Number of Bedrooms: 3
Price: 420000.0

Property ID: 53
Property Address: 3 Wattle Ave, Glen Huntly VIC 3163, Australia
Number of Bedrooms: 4
Price: 650000.0