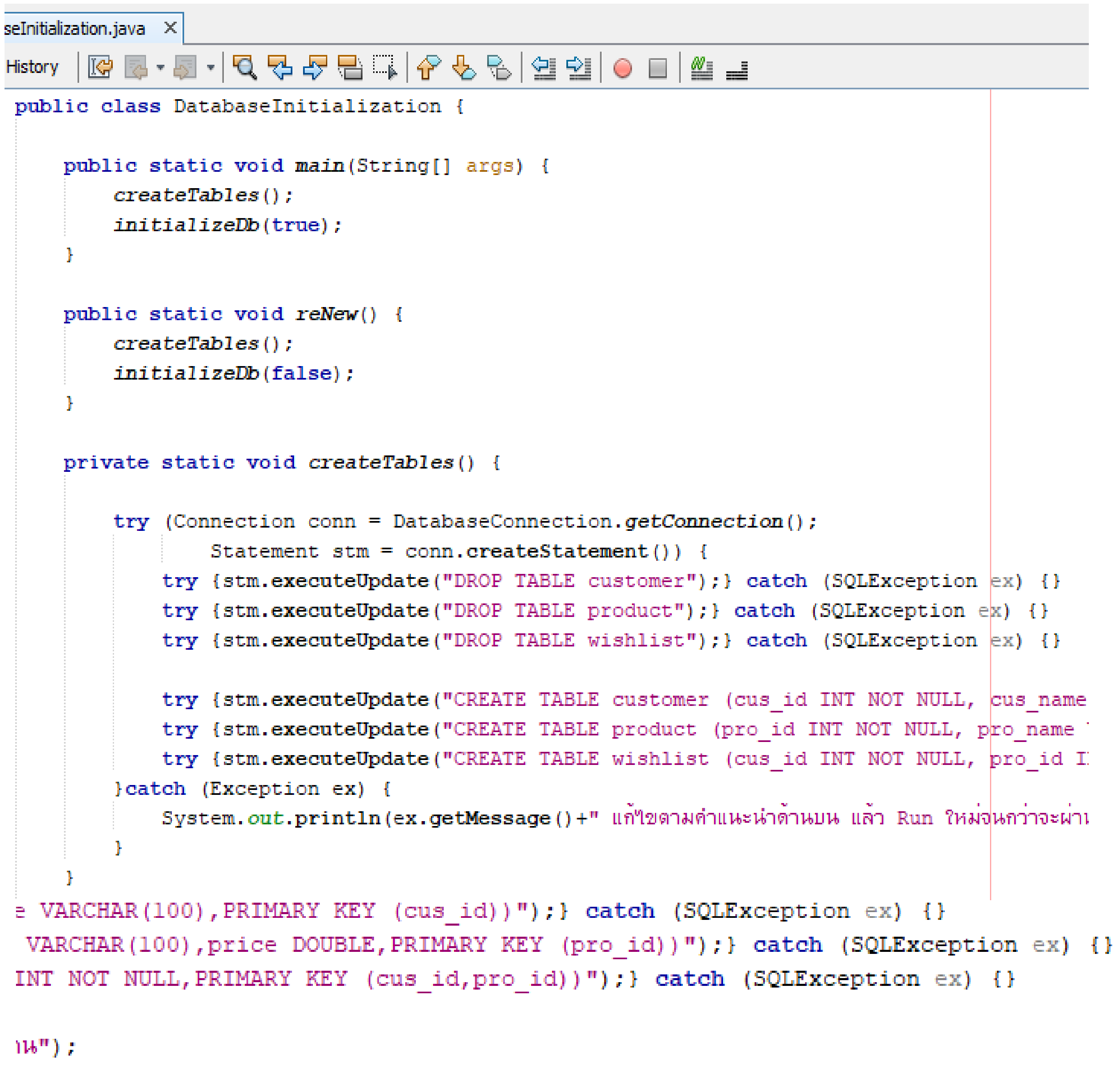
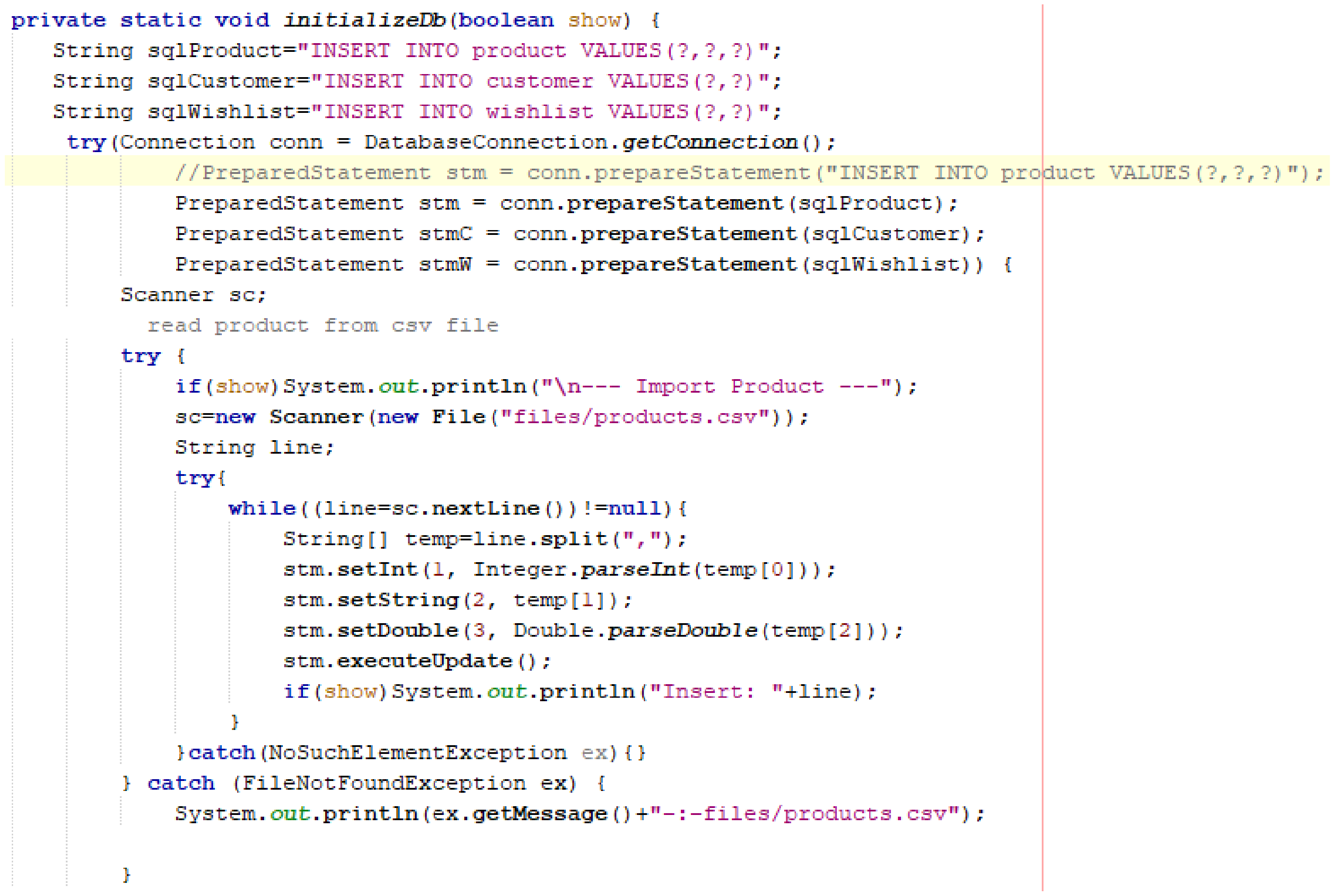
Database Dao (Install)



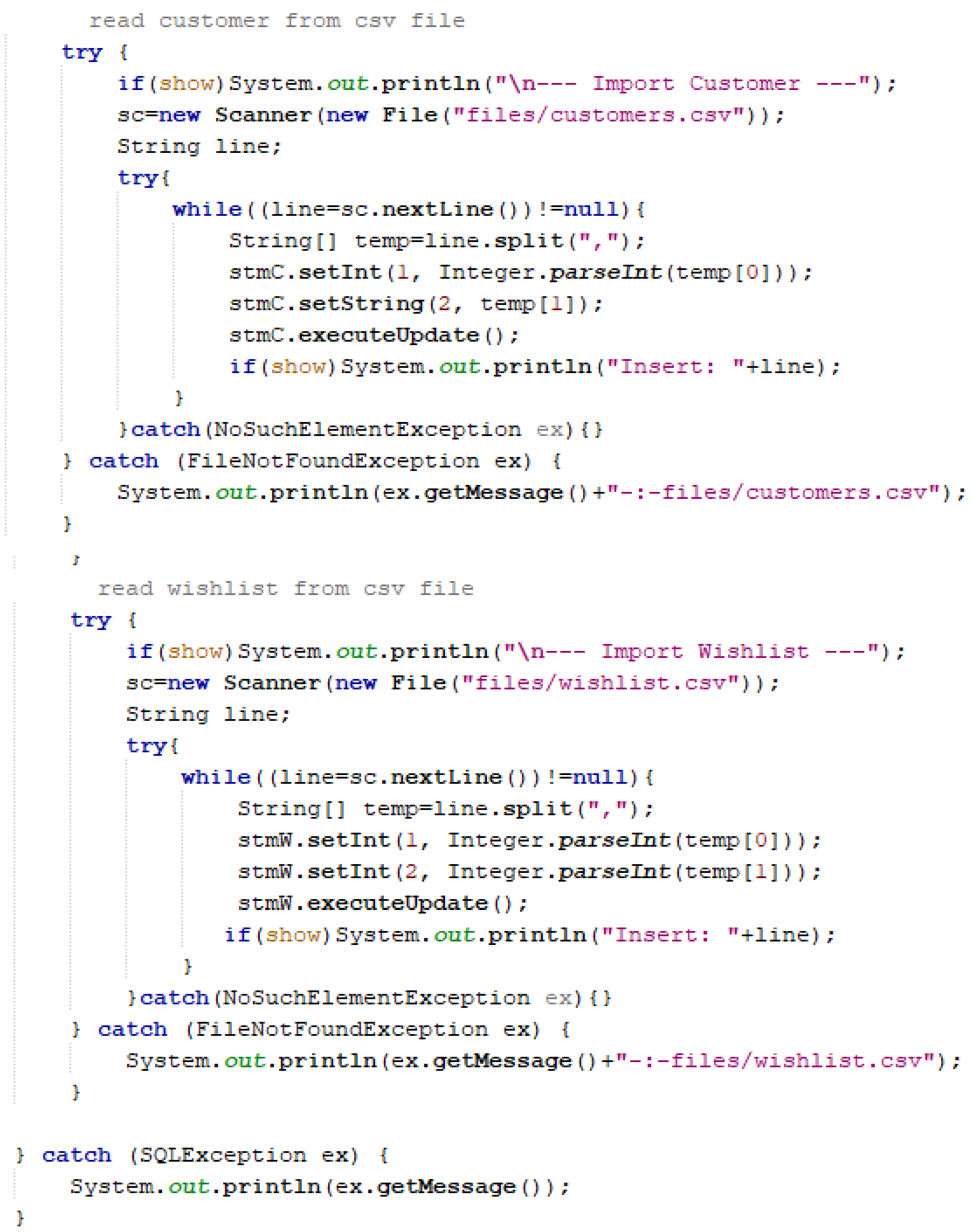
```
public class DatabaseInitialization {

    public static void main(String[] args) {
        createTables();
        initializeDb(true);
    }

    public static void reNew() {
        createTables();
        initializeDb(false);
    }

    private static void createTables() {

        try (Connection conn = DatabaseConnection.getConnection();
                Statement stm = conn.createStatement()) {
            try {stm.executeUpdate("DROP TABLE customer");} catch (SQLException ex) {}
            try {stm.executeUpdate("DROP TABLE product");} catch (SQLException ex) {}
            try {stm.executeUpdate("DROP TABLE wishlist");} catch (SQLException ex) {}

            try {stm.executeUpdate("CREATE TABLE customer (cus_id INT NOT NULL, cus_name
            try {stm.executeUpdate("CREATE TABLE product (pro_id INT NOT NULL, pro_name
            try {stm.executeUpdate("CREATE TABLE wishlist (cus_id INT NOT NULL, pro_id I
        }catch (Exception ex) {
            System.out.println(ex.getMessage()+" แก้ไขตามคำแนะนำด้านบน แล้ว Run ใหม่จนกว่าจะผ่า
        }
    }
e VARCHAR(100),PRIMARY KEY (cus_id))");} catch (SQLException ex) {}
 VARCHAR(100),price DOUBLE,PRIMARY KEY (pro_id))");} catch (SQLException ex) {}
INT NOT NULL,PRIMARY KEY (cus_id,pro_id))");} catch (SQLException ex) {}

าน");
```

Install 2



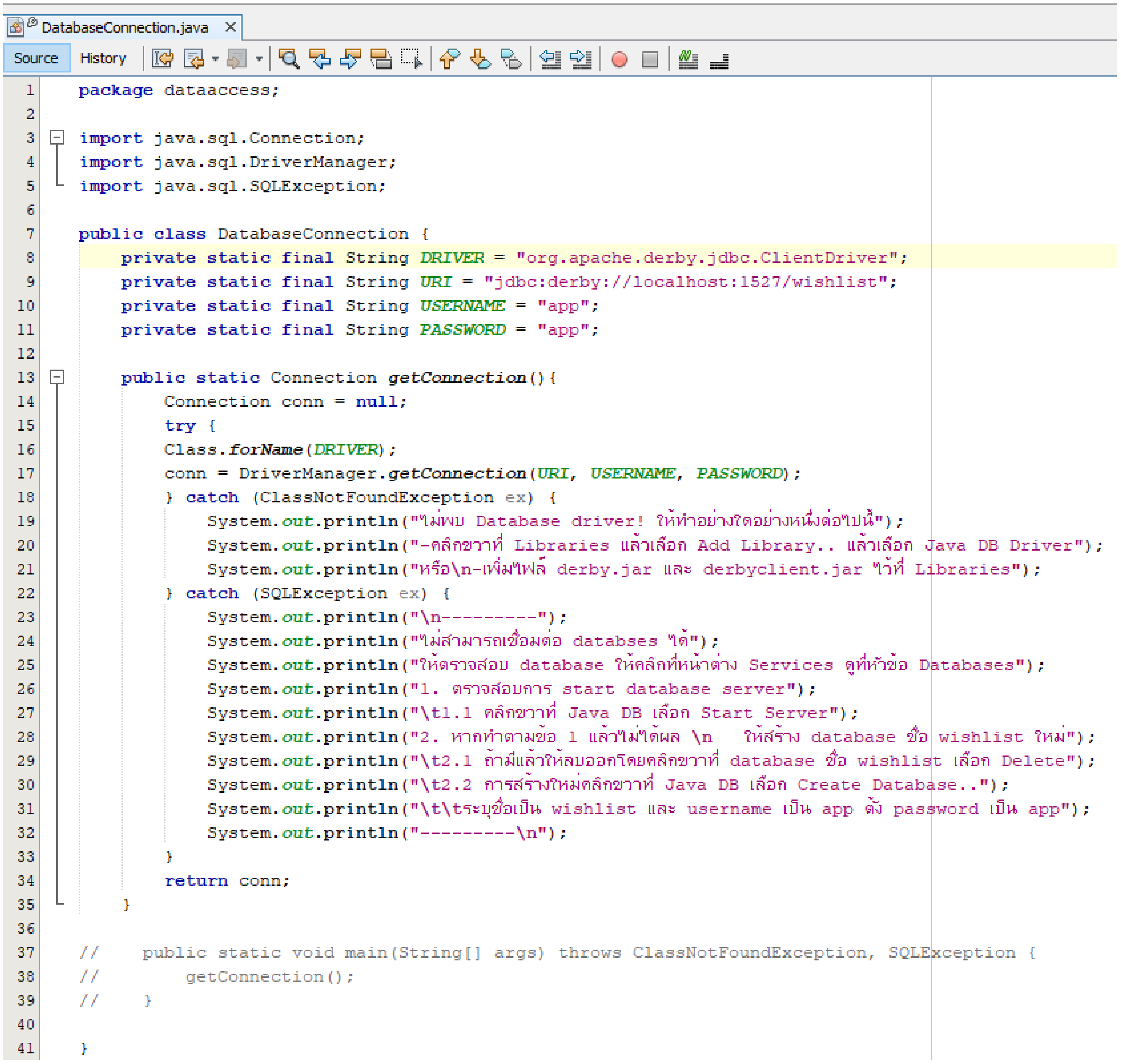
```
private static void initializeDb(boolean show) {
   String sqlProduct="INSERT INTO product VALUES(?,?,?)";
   String sqlCustomer="INSERT INTO customer VALUES(?,?)";
   String sqlWishlist="INSERT INTO wishlist VALUES(?,?)";
    try(Connection conn = DatabaseConnection.getConnection();
            //PreparedStatement stm = conn.prepareStatement("INSERT INTO product VALUES(?,?,?)");
            PreparedStatement stm = conn.prepareStatement(sqlProduct);
            PreparedStatement stmC = conn.prepareStatement(sqlCustomer);
            PreparedStatement stmW = conn.prepareStatement(sqlWishlist)) {
        Scanner sc;
          read product from csv file
        try {
            if(show)System.out.println("\n--- Import Product ---");
            sc=new Scanner(new File("files/products.csv"));
            String line;
            try{
                while((line=sc.nextLine())!=null){
                    String[] temp=line.split(",");
                    stm.setInt(1, Integer.parseInt(temp[0]));
                    stm.setString(2, temp[1]);
                    stm.setDouble(3, Double.parseDouble(temp[2]));
                    stm.executeUpdate();
                    if(show)System.out.println("Insert: "+line);
                }
            }catch(NoSuchElementException ex){}
        } catch (FileNotFoundException ex) {
            System.out.println(ex.getMessage()+"-:-files/products.csv");

        }
```

Install 3



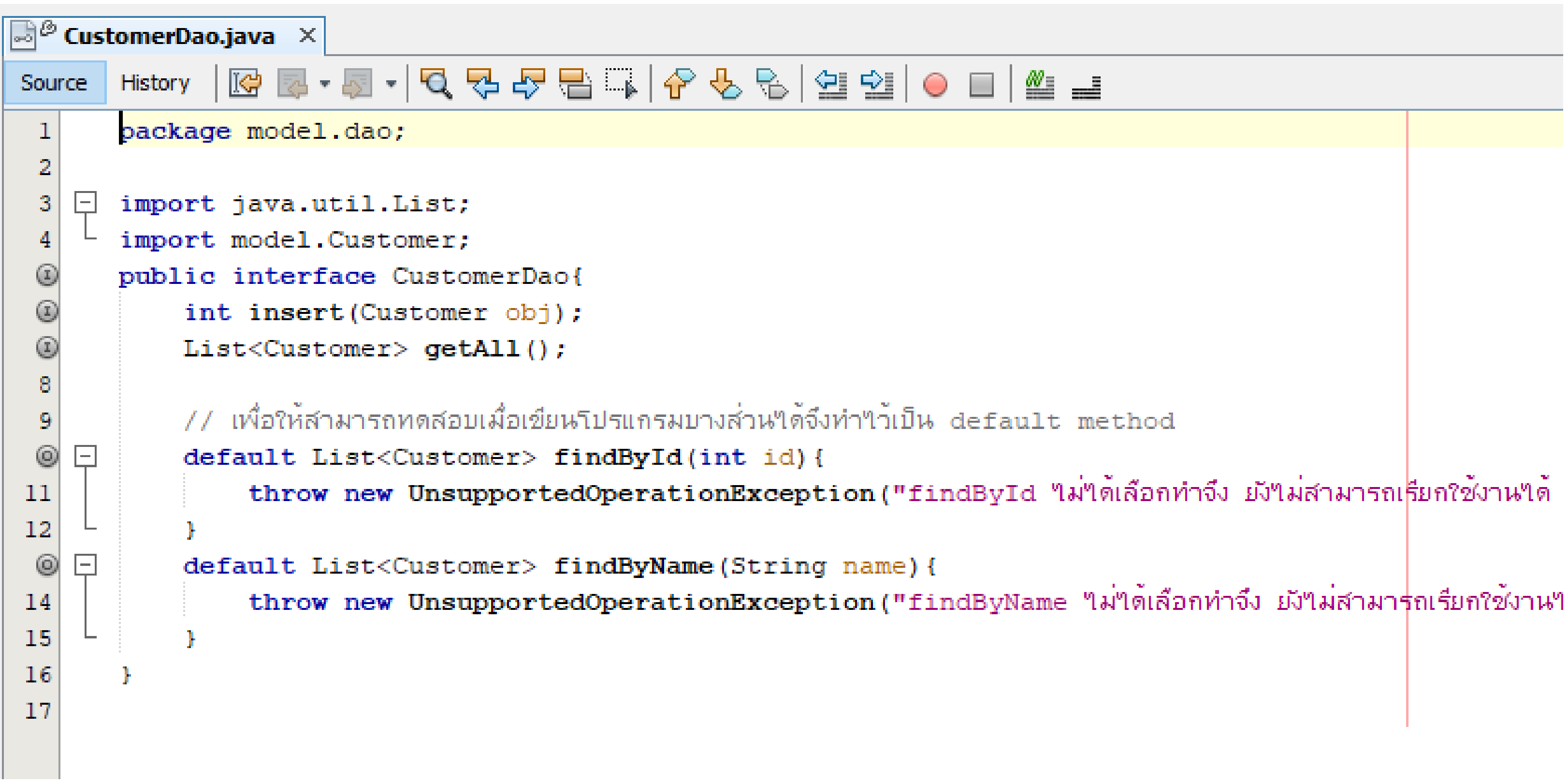
```
          read customer from csv file
        try {
            if(show)System.out.println("\n--- Import Customer ---");
            sc=new Scanner(new File("files/customers.csv"));
            String line;
            try{
                while((line=sc.nextLine())!=null){
                    String[] temp=line.split(",");
                    stmC.setInt(1, Integer.parseInt(temp[0]));
                    stmC.setString(2, temp[1]);
                    stmC.executeUpdate();
                    if(show)System.out.println("Insert: "+line);
                }
            }catch(NoSuchElementException ex){}
        } catch (FileNotFoundException ex) {
            System.out.println(ex.getMessage()+"-:-files/customers.csv");
        }

         ,
          read wishlist from csv file
        try {
            if(show)System.out.println("\n--- Import Wishlist ---");
            sc=new Scanner(new File("files/wishlist.csv"));
            String line;
            try{
                while((line=sc.nextLine())!=null){
                    String[] temp=line.split(",");
                    stmW.setInt(1, Integer.parseInt(temp[0]));
                    stmW.setInt(2, Integer.parseInt(temp[1]));
                    stmW.executeUpdate();
                   if(show)System.out.println("Insert: "+line);
                }
            }catch(NoSuchElementException ex){}
        } catch (FileNotFoundException ex) {
            System.out.println(ex.getMessage()+"-:-files/wishlist.csv");
        }

    } catch (SQLException ex) {
        System.out.println(ex.getMessage());
    }
```

Connection



```java
package dataaccess;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DatabaseConnection {
    private static final String DRIVER = "org.apache.derby.jdbc.ClientDriver";
    private static final String URI = "jdbc:derby://localhost:1527/wishlist";
    private static final String USERNAME = "app";
    private static final String PASSWORD = "app";

    public static Connection getConnection(){
        Connection conn = null;
        try {
        Class.forName(DRIVER);
        conn = DriverManager.getConnection(URI, USERNAME, PASSWORD);
        } catch (ClassNotFoundException ex) {
            System.out.println("ไม่พบ Database driver! ให้ทำอย่างใดอย่างหนึ่งต่อไปนี้");
            System.out.println("-คลิกขวาที่ Libraries แล้วเลือก Add Library.. แล้วเลือก Java DB Driver");
            System.out.println("หรือ\n-เพิ่มไฟล์ derby.jar และ derbyclient.jar ไว้ที่ Libraries");
        } catch (SQLException ex) {
            System.out.println("\n---------");
            System.out.println("ไม่สามารถเชื่อมต่อ databses ได้");
            System.out.println("ให้ตรวจสอบ database ให้คลิกที่หน้าต่าง Services ดูที่หัวข้อ Databases");
            System.out.println("1. ตรวจสอบการ start database server");
            System.out.println("\t1.1 คลิกขวาที่ Java DB เลือก Start Server");
            System.out.println("2. หากทำตามข้อ 1 แล้วไม่ได้ผล \n   ให้สร้าง database ชื่อ wishlist ใหม่");
            System.out.println("\t2.1 ถ้ามีแล้วให้ลบออกโดยคลิกขวาที่ database ชื่อ wishlist เลือก Delete");
            System.out.println("\t2.2 การสร้างใหม่คลิกขวาที่ Java DB เลือก Create Database..");
            System.out.println("\t\tระบุชื่อเป็น wishlist และ username เป็น app ตั้ง password เป็น app");
            System.out.println("---------\n");
        }
        return conn;
    }

//    public static void main(String[] args) throws ClassNotFoundException, SQLException {
//        getConnection();
//    }

}
```

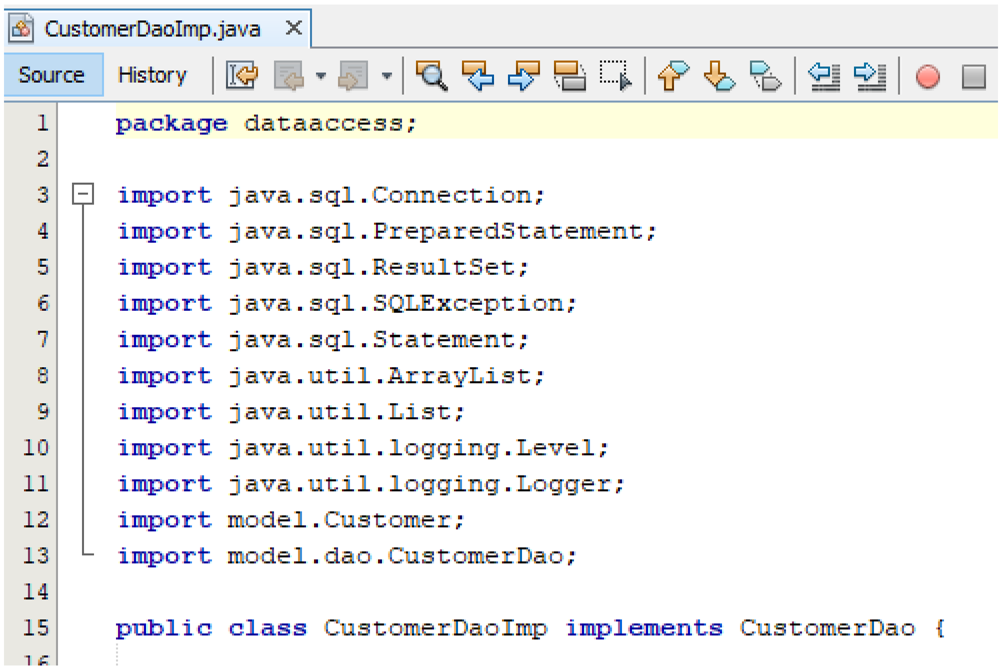
CustomerDao



CustomerDao.java

```java
package model.dao;

import java.util.List;
import model.Customer;
public interface CustomerDao{
    int insert(Customer obj);
    List<Customer> getAll();

    // เพื่อให้สามารถทดสอบเมื่อเขียนโปรแกรมบางส่วนได้จึงทำไว้เป็น default method
    default List<Customer> findById(int id){
        throw new UnsupportedOperationException("findById ไม่ได้เลือกทำจึง ยังไม่สามารถเรียกใช้งานได้
    }
    default List<Customer> findByName(String name){
        throw new UnsupportedOperationException("findByName ไม่ได้เลือกทำจึง ยังไม่สามารถเรียกใช้งานไ
    }
}
```

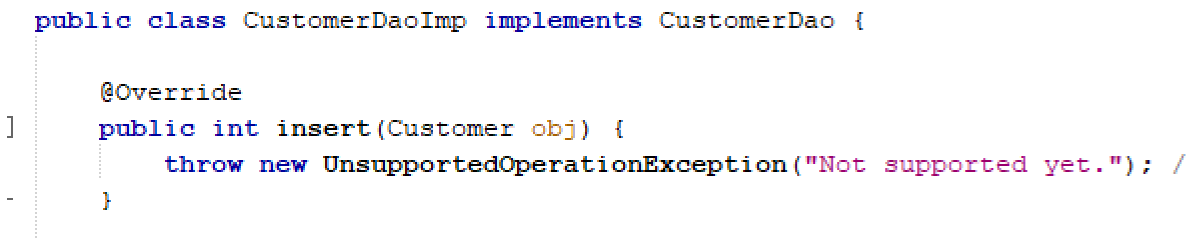
CustomerDaoImp



CustomerDaoImp.java

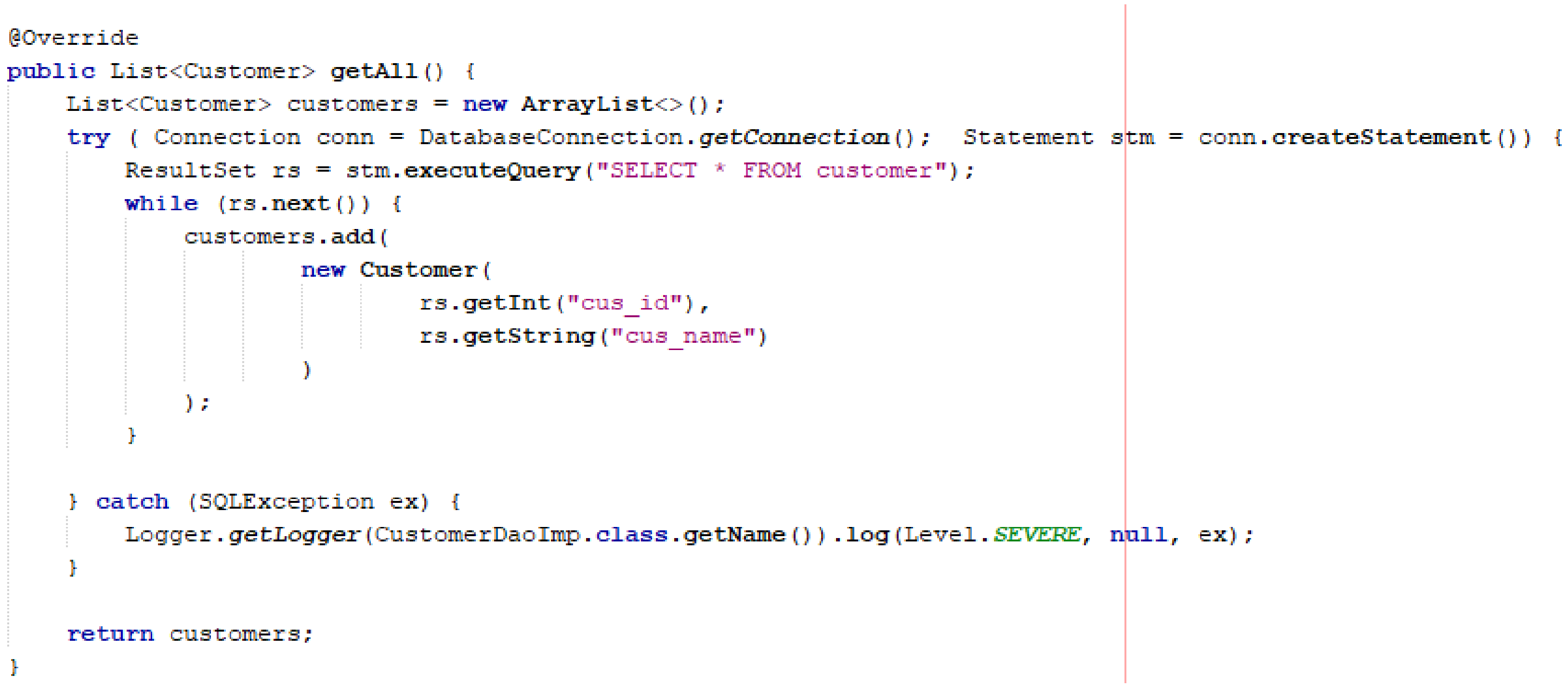
Source History

```java
package dataaccess;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import model.Customer;
import model.dao.CustomerDao;

public class CustomerDaoImp implements CustomerDao {
```

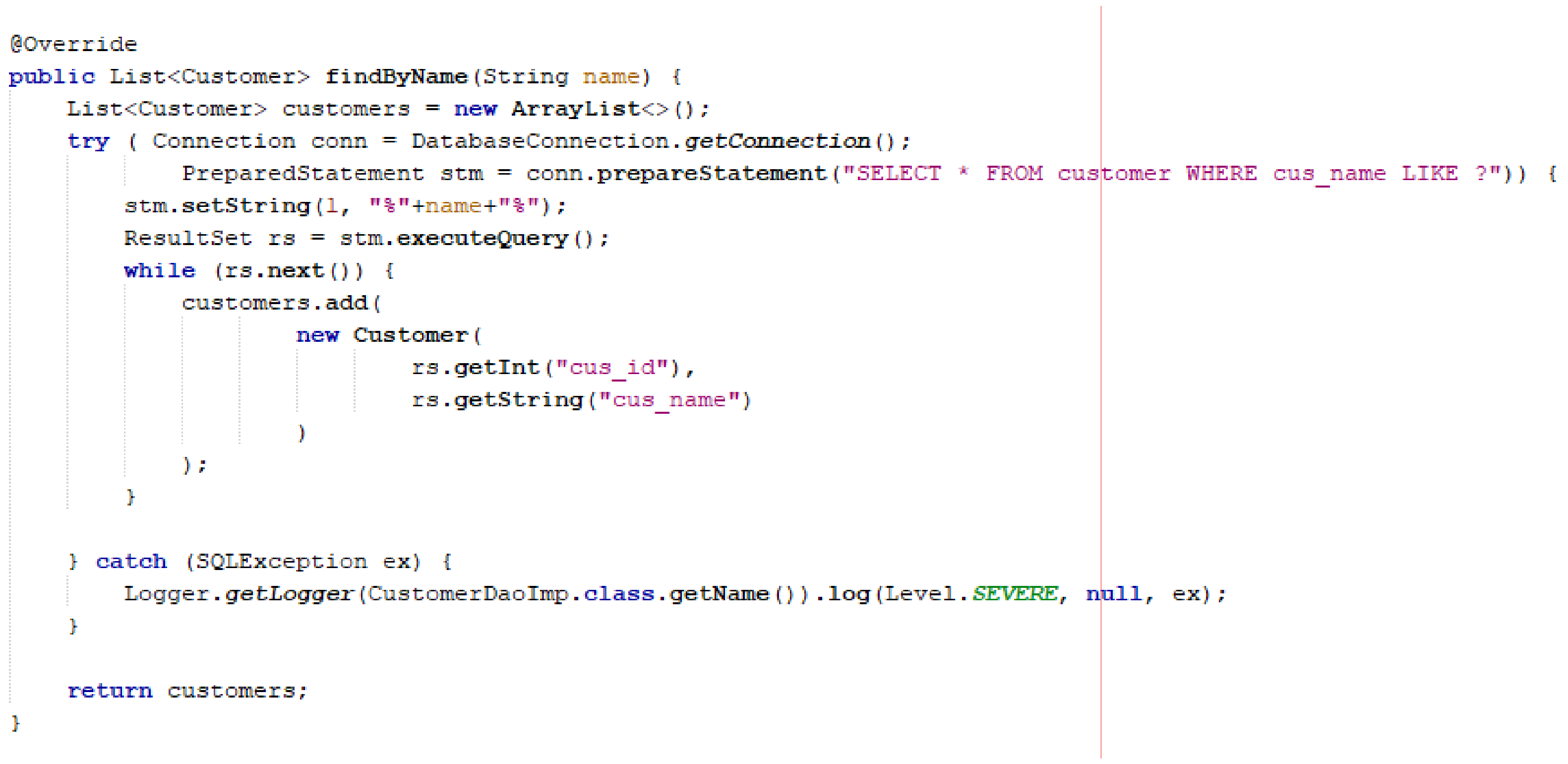


```java
public class CustomerDaoImp implements CustomerDao {

    @Override
    public int insert(Customer obj) {
        throw new UnsupportedOperationException("Not supported yet."); /
    }
```

CustomerDaoImp + getAll + FindName

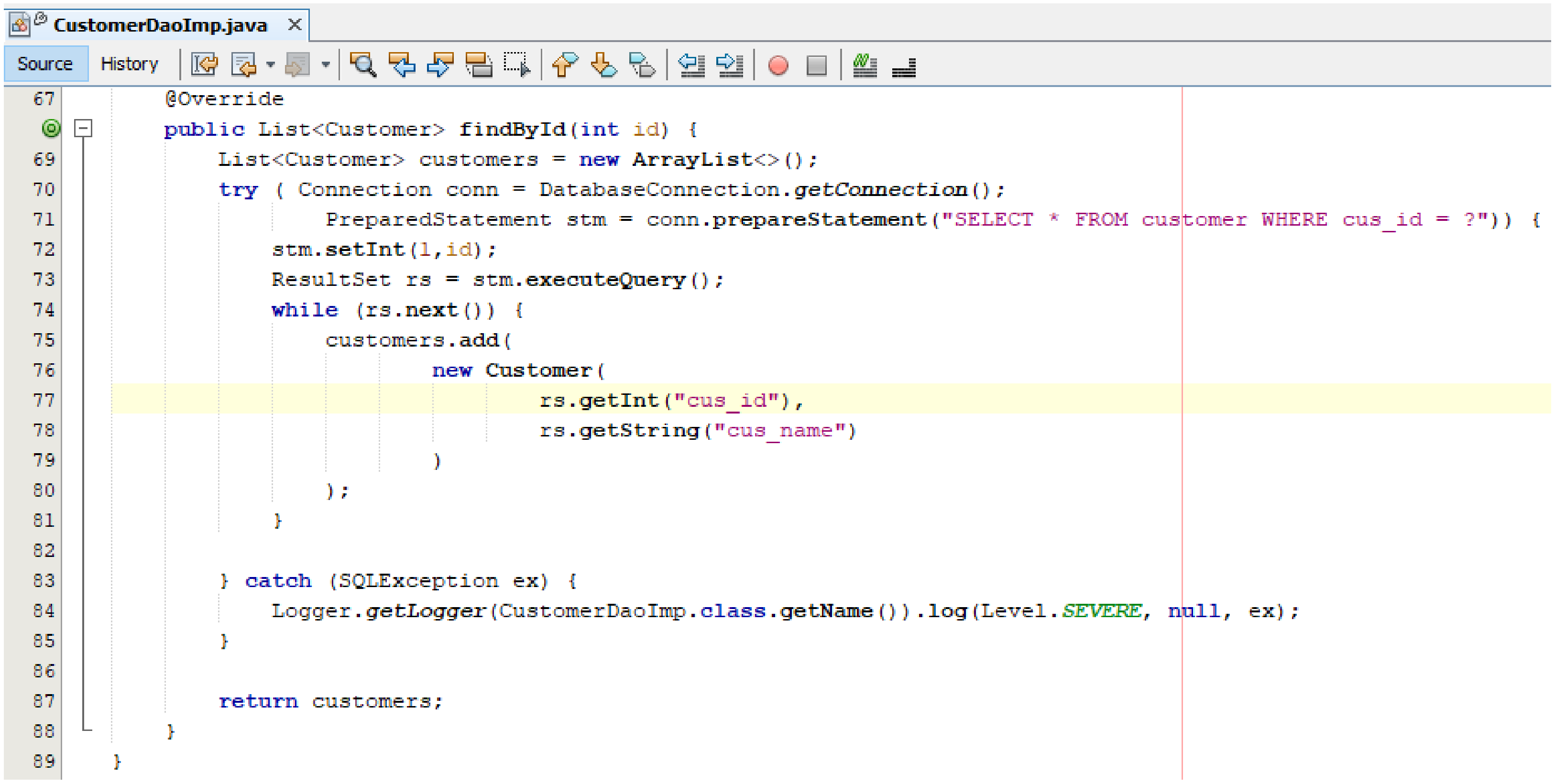


```
@Override
public List<Customer> getAll() {
    List<Customer> customers = new ArrayList<>();
    try ( Connection conn = DatabaseConnection.getConnection();  Statement stm = conn.createStatement()) {
        ResultSet rs = stm.executeQuery("SELECT * FROM customer");
        while (rs.next()) {
            customers.add(
                    new Customer(
                            rs.getInt("cus_id"),
                            rs.getString("cus_name")
                    )
            );
        }

    } catch (SQLException ex) {
        Logger.getLogger(CustomerDaoImp.class.getName()).log(Level.SEVERE, null, ex);
    }

    return customers;
}
```



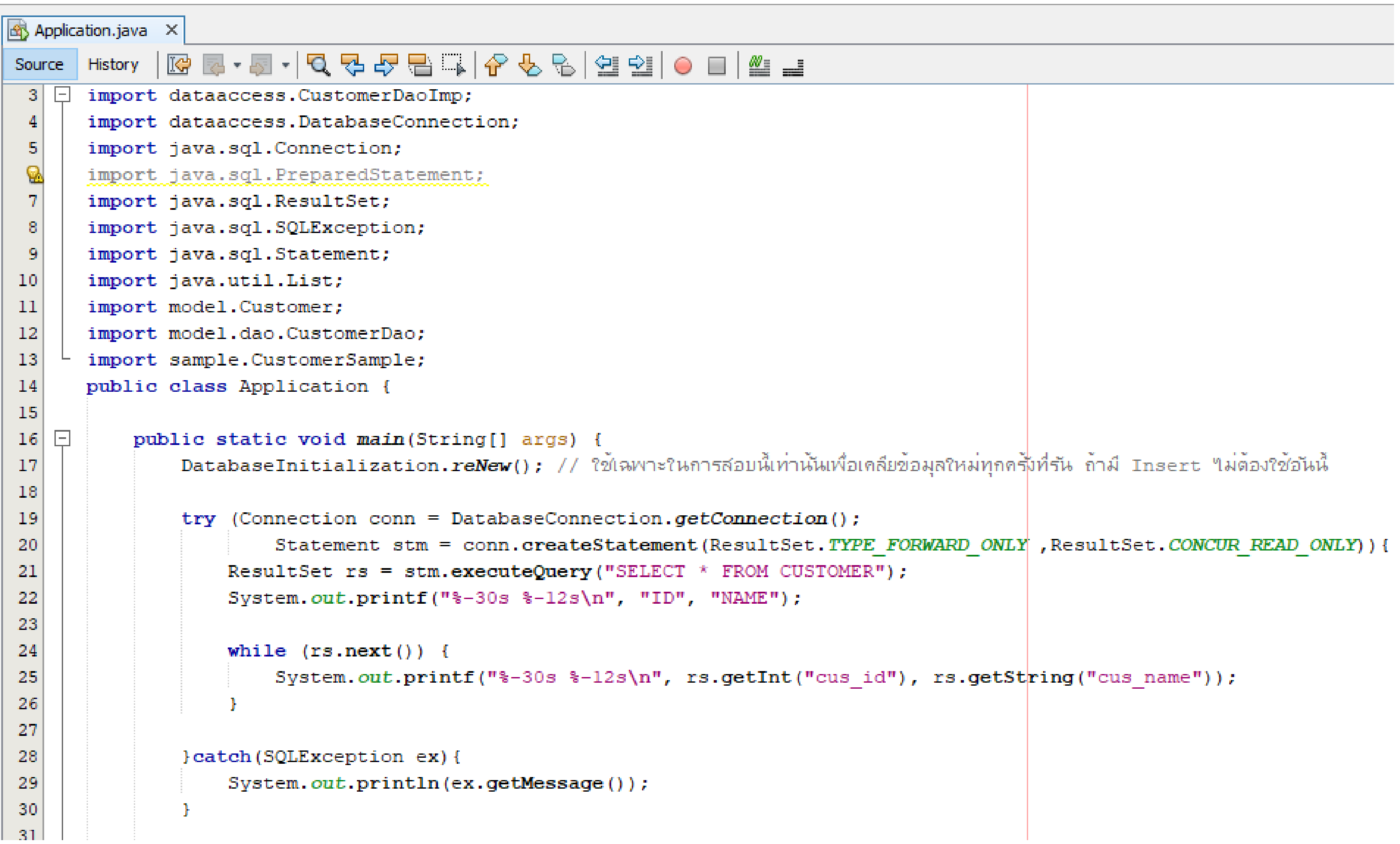
```
@Override
public List<Customer> findByName(String name) {
    List<Customer> customers = new ArrayList<>();
    try ( Connection conn = DatabaseConnection.getConnection();
            PreparedStatement stm = conn.prepareStatement("SELECT * FROM customer WHERE cus_name LIKE ?")) {
        stm.setString(1, "%"+name+"%");
        ResultSet rs = stm.executeQuery();
        while (rs.next()) {
            customers.add(
                    new Customer(
                            rs.getInt("cus_id"),
                            rs.getString("cus_name")
                    )
            );
        }

    } catch (SQLException ex) {
        Logger.getLogger(CustomerDaoImp.class.getName()).log(Level.SEVERE, null, ex);
    }

    return customers;
}
```

CustomerDaoImp + findById



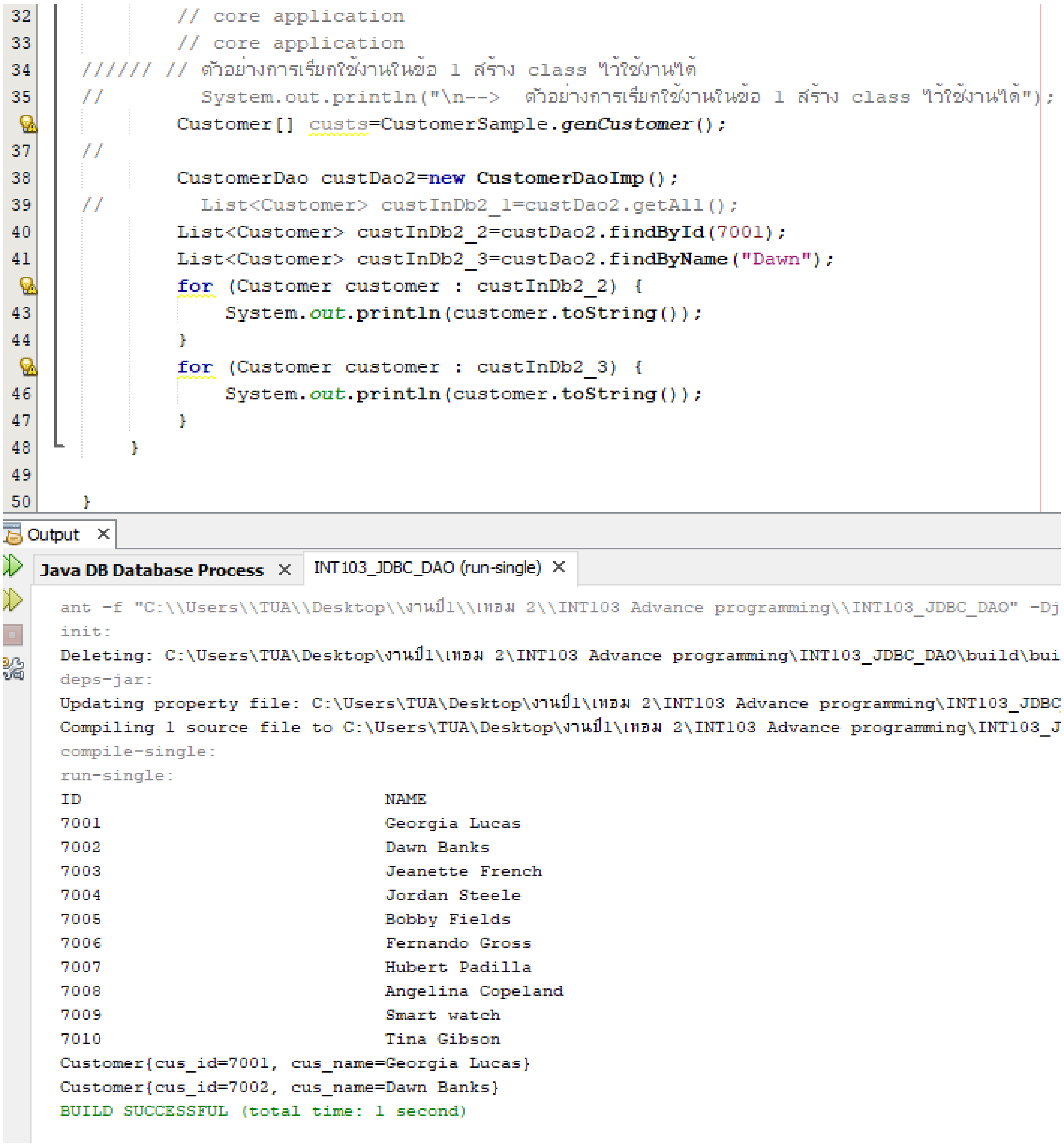
```java
    @Override
    public List<Customer> findById(int id) {
        List<Customer> customers = new ArrayList<>();
        try ( Connection conn = DatabaseConnection.getConnection();
                PreparedStatement stm = conn.prepareStatement("SELECT * FROM customer WHERE cus_id = ?")) {
            stm.setInt(1,id);
            ResultSet rs = stm.executeQuery();
            while (rs.next()) {
                customers.add(
                        new Customer(
                                rs.getInt("cus_id"),
                                rs.getString("cus_name")
                        )
                );
            }

        } catch (SQLException ex) {
            Logger.getLogger(CustomerDaoImp.class.getName()).log(Level.SEVERE, null, ex);
        }

        return customers;
    }
}
```

Application



```java
import dataaccess.CustomerDaoImp;
import dataaccess.DatabaseConnection;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.List;
import model.Customer;
import model.dao.CustomerDao;
import sample.CustomerSample;
public class Application {

    public static void main(String[] args) {
        DatabaseInitialization.reNew(); // ใช้เฉพาะในการสอบนี้เท่านั้นเพื่อเคลียข้อมูลใหม่ทุกครั้งที่รัน ถ้ามี Insert ไม่ต้องใช้อันนี้

        try (Connection conn = DatabaseConnection.getConnection();
                Statement stm = conn.createStatement(ResultSet.TYPE_FORWARD_ONLY ,ResultSet.CONCUR_READ_ONLY)){
            ResultSet rs = stm.executeQuery("SELECT * FROM CUSTOMER");
            System.out.printf("%-30s %-12s\n", "ID", "NAME");

            while (rs.next()) {
                System.out.printf("%-30s %-12s\n", rs.getInt("cus_id"), rs.getString("cus_name"));
            }

        }catch(SQLException ex){
            System.out.println(ex.getMessage());
        }
```

Application 2



```java
            // core application
            // core application
////// // ตัวอย่างการเรียกใช้งานในข้อ 1 สร้าง class ไว้ใช้งานได้
//        System.out.println("\n-->  ตัวอย่างการเรียกใช้งานในข้อ 1 สร้าง class ไว้ใช้งานได้");
            Customer[] custs=CustomerSample.genCustomer();
//
            CustomerDao custDao2=new CustomerDaoImp();
//        List<Customer> custInDb2_1=custDao2.getAll();
            List<Customer> custInDb2_2=custDao2.findById(7001);
            List<Customer> custInDb2_3=custDao2.findByName("Dawn");
            for (Customer customer : custInDb2_2) {
                System.out.println(customer.toString());
            }
            for (Customer customer : custInDb2_3) {
                System.out.println(customer.toString());
            }
        }

    }
```

Output ×

Java DB Database Process × INT103_JDBC_DAO (run-single) ×

```
ant -f "C:\\Users\\TUA\\Desktop\\งานปี1\\เทอม 2\\INT103 Advance programming\\INT103_JDBC_DAO" -Dj
init:
Deleting: C:\Users\TUA\Desktop\งานปี1\เทอม 2\INT103 Advance programming\INT103_JDBC_DAO\build\bui
deps-jar:
Updating property file: C:\Users\TUA\Desktop\งานปี1\เทอม 2\INT103 Advance programming\INT103_JDBC
Compiling 1 source file to C:\Users\TUA\Desktop\งานปี1\เทอม 2\INT103 Advance programming\INT103_J
compile-single:
run-single:
ID                             NAME
7001                           Georgia Lucas
7002                           Dawn Banks
7003                           Jeanette French
7004                           Jordan Steele
7005                           Bobby Fields
7006                           Fernando Gross
7007                           Hubert Padilla
7008                           Angelina Copeland
7009                           Smart watch
7010                           Tina Gibson
Customer{cus_id=7001, cus_name=Georgia Lucas}
Customer{cus_id=7002, cus_name=Dawn Banks}
BUILD SUCCESSFUL (total time: 1 second)
```