LAPORAN PRAKTIKUM
PRAKTIKUM 9:
“PERSISTENT OBJECT”

Disusun Oleh :

Farhan Adka Reynaldi
24060121130045

PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJECT
LAB B2

DEPARTEMEN ILMU KOMPUTER / INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO
SEMARANG
2023

A. Menggunakan Persistent Object sebagai Model Basis Data Relasional

1. PersonDAO.java

```
/*
Nama        : Farhan Adka Reynaldi
NIM             : 24060121130045
Lab             : PBO B2
Waktu       : 31-05-2023
Nama File   : PersonDAO.java
Deskripsi   : interface untuk person access object
*/

public interface PersonDAO {
    public void savePerson(Person p) throws Exception;
```

2. Person.java

```
/*
Nama        : Farhan Adka Reynaldi
NIM             : 24060121130045
Lab             : PBO B2
Waktu       : 31-05-2023
Nama File   : Person.java
Deskripsi   : person database model
*/

public class Person {
    private int id;
    private String name;

    public Person(String n) {
        name = n;
    }

    public Person(int i, String n) {
        id = i;
```

```
        name = n;
    }

    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }
}
```

3. MySQLPersonDAO.java

```
/*
Nama        : Farhan Adka Reynaldi
NIM             : 24060121130045
Lab             : PBO B2
Waktu       : 31-05-2023
Nama File   : MySQLPersonDAO.java
Deskripsi   : implementasi personDAO untuk MySQL
*/

import java.sql.*;

public class MySQLPersonDAO implements PersonDAO {
    public void savePerson(Person person) throws Exception {
        String name = person.getName();

        // Membuat koneksi, nama db, user, password menyesuaikan
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/pbo", "root", "alfa10022010");

        // Kerjakan mysql query
```

```
        String query = "INSERT INTO person(name) VALUES('" + name + "')";
        System.out.println(query);
        Statement s = con.createStatement();
        s.executeUpdate(query);

        // Tutup koneksi database
        con.close();
    }
}
```

4. DAOManager.java

```
/*
Nama        : Farhan Adka Reynaldi
NIM             : 24060121130045
Lab             : PBO B2
Waktu       : 31-05-2023
Nama File   : DAOManager.java
Deskripsi   : pengelola DAO dalam program
*/

public class DAOManager {
    private PersonDAO personDAO;

    public void setPersonDAO(PersonDAO person) {
        personDAO = person;
    }

    public PersonDAO getPersonDAO() {
        return personDAO;
    }
}
```

5. MainDAO.java

```
/*
Nama         : Farhan Adka Reynaldi
NIM                : 24060121130045
Lab                : PBO B2
Waktu        : 31-05-2023
Nama File    : MainDAO.java
Deskripsi    : main program untuk akses DAO
*/

public class MainDAO {
    public static void main(String[] args) {
        Person person = new Person("Indra");
        DAOManager m = new DAOManager();
        m.setPersonDAO(new MySQLPersonDAO());

        try {
            m.getPersonDAO().savePerson(person);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

6. Membuat Basis Data “PBO”

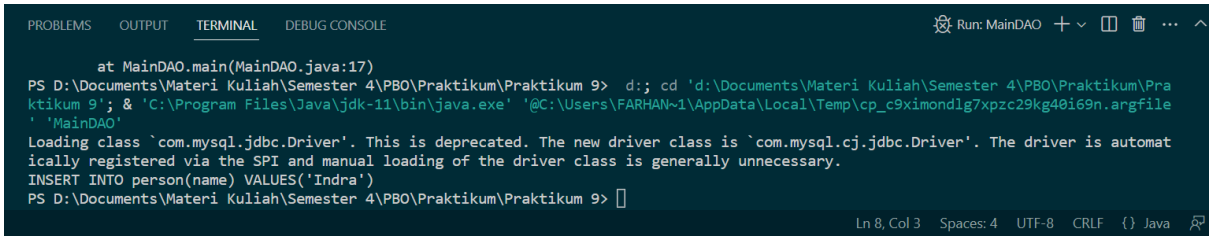
```
CREATE DATABASE PBO;
```

7. Membuat Tabel “Person”

```
CREATE TABLE person(id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,name
VARCHAR(100))
```

8. **Compile dan Run Source Code**

   Untuk meng-compile semua src sekaligus, gunakan command javac *.java pada terminal. Kemudian, run MainDAO beserta connector mysql yang telah di download.



Screenshot di atas merupakan hasil run src dan input person “Indra” ke *database* pbo

9. **Check Tabel “Person”**

```
mysql> prompt 24060121130045>
PROMPT set to '24060121130045>'
24060121130045>create database PBO
    -> ;
Query OK, 1 row affected (0.01 sec)

24060121130045>CREATE TABLE person(id INT PRIMARY KEY AUTO_INCREMENT NOT
    -> NULL,name VARCHAR(100)) ;
ERROR 1046 (3D000): No database selected
24060121130045>use pbo
Database changed
24060121130045>CREATE TABLE person(id INT PRIMARY KEY AUTO_INCREMENT NOT
    -> NULL,name VARCHAR(100)) ;
Query OK, 0 rows affected (0.03 sec)

24060121130045>select * from person;
+----+-------+
| id | name  |
+----+-------+
|  1 | Indra |
+----+-------+
1 row in set (0.00 sec)

24060121130045>
```

Tampak pada screenshot di atas, tabel person telah diisi dengan nama “Indra”. Id menjadi 1 dikarenakan percobaan pemasukan ke 1.

B. Menggunakan Persistent Object Sebagai Objek Terealisasi

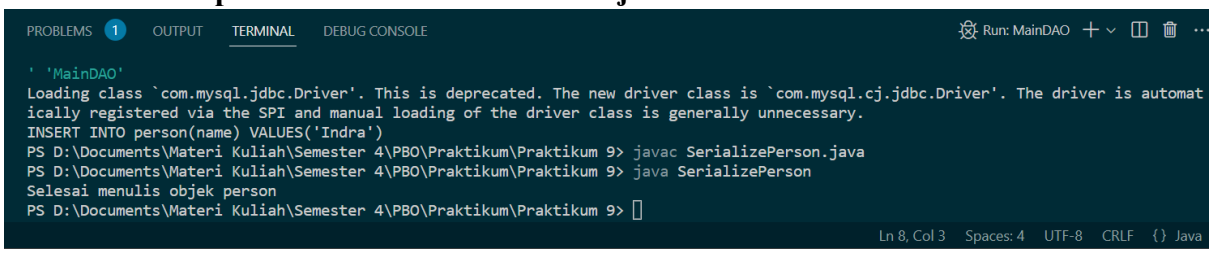
1. SerializePerson.java

```
/*

Nama        : Farhan Adka Reynaldi

NIM             : 24060121130045

Lab             : PBO B2
```

```
Waktu       : 31-05-2023
Nama File   : SerializePerson.java
Deskripsi   : program untuk serialisasi objek Person
*/

import java.io.*;

// class Person
class Person implements Serializable {
    private String name;

    public Person(String n) {
        name = n;
    }

    public String getName() {
        return name;
    }
}

// class SerializePerson
public class SerializePerson {
    public static void main(String[] args) {
        Person person = new Person("Panji");
        try {
            FileOutputStream f = new FileOutputStream("person.ser");
            ObjectOutputStream o = new ObjectOutputStream(f);
            o.writeObject(person);
            System.out.println("Selesai menulis objek person");
            o.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

```
}
```

2. Compile dan Run SerializePerson.java



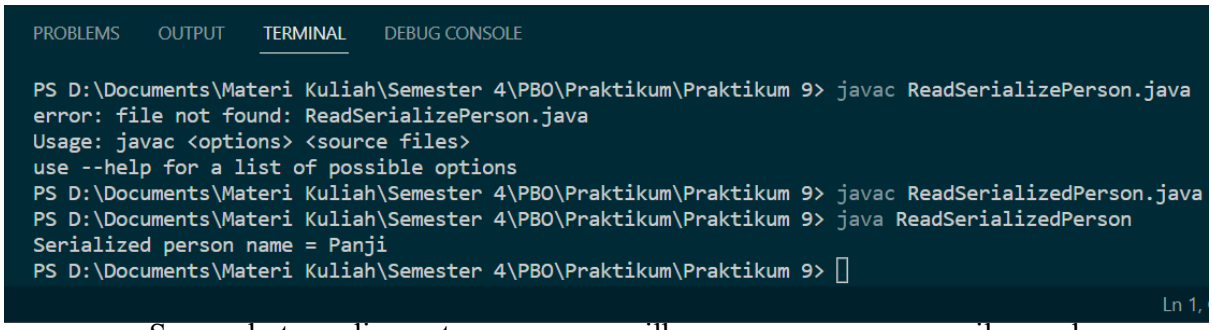
Screenshot di atas menampilkan proses compile dan run SerializePerson.java. Setelah berhasil di run, akan terbuat file person.ser.

3. ReadSerializedPerson.java

```
/*
Nama        : Farhan Adka Reynaldi
NIM             : 24060121130045
Lab             : PBO B2
Waktu       : 31-05-2023
Nama File   : ReadSerializedPerson.java
Deskripsi   : Program untuk serialisasi objek Person
*/

import java.io.*;

public class ReadSerializedPerson {
    public static void main(String[] args) {
        Person person = null;
        try {
            FileInputStream f = new FileInputStream("person.ser");
            ObjectInputStream s = new ObjectInputStream(f);
            person = (Person) s.readObject();
            s.close();
            System.out.println("Serialized person name = " +
person.getName());
        } catch (Exception ioe) {
            ioe.printStackTrace();
        }
```

```
            }
        }
}
```

4. **Compile dan Run ReadSerializedPerson.java**



Screenshot di atas menampilkan proses compile dan run ReadSerializedPerson.java. Setelah berhasil di run, Panji akan masuk ke tabel Person.