

LAPORAN UJIAN PRAKTIKUM

Mata Praktikum : Rekayasa Perangkat Lunak 2
Kelas : 4IA18
Praktikum ke- : Ujian
Tanggal : Jumat, 8 Desember 2023
Materi : Manual Book Project rentalMobil
NPM : 50420614
Nama : Joandri Alkahfi Keandrea
Ketua Asisten : Marcell
Paraf Asisten :
Nama Asisten :

Jumlah Lembar : 31 Lembar

LABORATORIUM TEKNIK INFORMATIKA UNIVERSITAS

GUNADARMA

2023

REKAYASA PERANGKAT LUNAK 2

(Manual Book Pembuatan Proyek rental)



Nama : Joandri Alkahfi Keandrea
NPM : 50420614
Kelas : 4IA18

LABORATORIUM TEKNIK INFORMATIKA
UNIVERSITAS GUNADARMA

2023
MANUAL BOOK

A. Deskripsi Aplikasi

Aplikasi Sistem Pemesanan dan Rental Mobil adalah sebuah solusi perangkat lunak yang dirancang untuk memudahkan proses pemesanan dan penyewaan mobil. Aplikasi ini memungkinkan pengguna untuk mencari, memilih, dan memesan mobil sesuai dengan kebutuhan mereka. Pengguna juga dapat mengelola detail reservasi, seperti waktu pengambilan dan pengembalian mobil, serta pembayaran melalui aplikasi. Aplikasi ini dirancang untuk memberikan kemudahan dan kenyamanan dalam proses rental mobil.

B. Tujuan Aplikasi

Meningkatkan kenyamanan pelanggan: Aplikasi ini bertujuan untuk memberikan pengalaman pemesanan dan penyewaan mobil yang lebih mudah dan nyaman bagi pelanggan. Mereka dapat melihat pilihan mobil, memilih waktu yang sesuai, dan membayar secara online. Efisiensi operasional: Aplikasi ini membantu penyedia layanan rental mobil dalam mengelola reservasi, pembaruan inventaris, dan melacak status pengembalian mobil. Hal ini dapat meningkatkan efisiensi operasional. Peningkatan pendapatan: Aplikasi ini memungkinkan penyedia layanan untuk menjangkau lebih banyak pelanggan dan meningkatkan pendapatan mereka melalui sistem pemesanan online.

C. Batasan Masalah

1. Aplikasi ini terbatas pada penyewaan mobil pribadi dan non-komersial. Tidak mencakup penyewaan mobil untuk tujuan komersial seperti angkutan barang atau jasa taksi.
2. Aplikasi ini hanya berlaku untuk penyewaan mobil dalam suatu wilayah geografis tertentu. Wilayah ini akan ditentukan oleh penyedia layanan.
3. Aplikasi ini tidak mencakup fitur perawatan dan perbaikan mobil. Fokus utamanya adalah pada proses pemesanan dan penyewaan.

D. Kebutuhan Spesifik

No	Fitur	Kebutuhan Fungsional	Kebutuhan Antarmuka	Kebutuhan Unjuk Kerja
----	-------	----------------------	---------------------	-----------------------

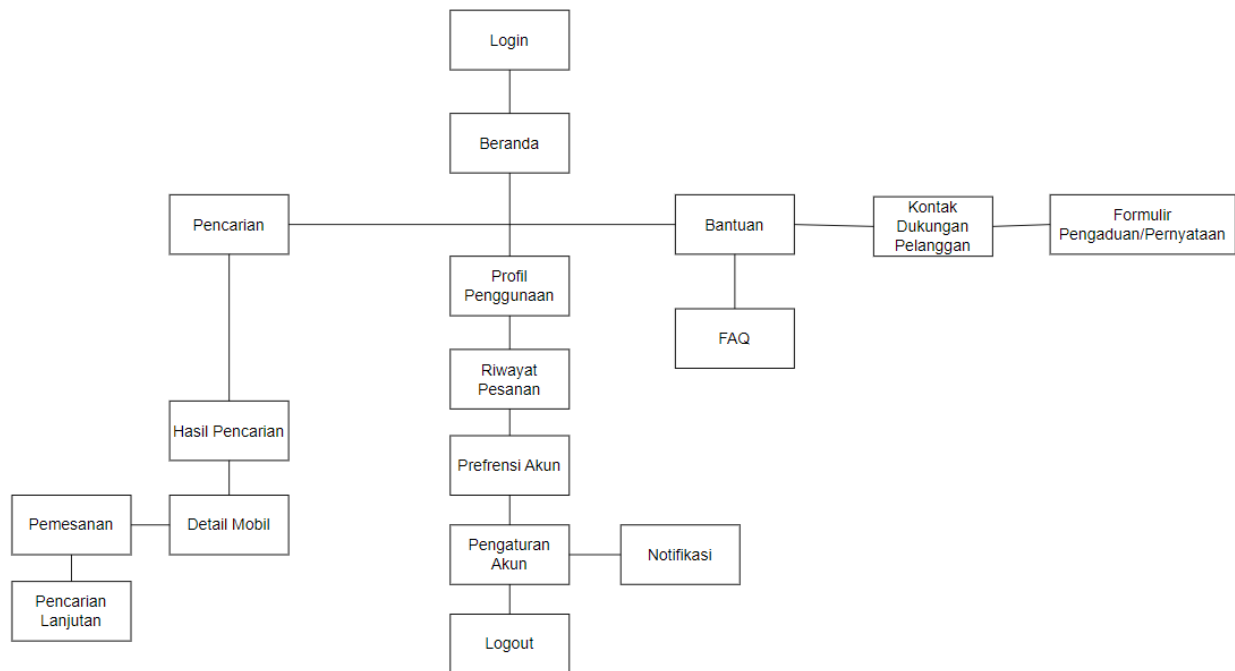
1	Tambah Nama Merk	Fitur ini digunakan untuk melakukan tambah data mobil yang di rental	<ul style="list-style-type: none"> - Satu form untuk menaruh atribut lain - Satu edit text untuk memasukan nama merk - Satu textBox - Dua buah button untuk tambah dan batal 	<ul style="list-style-type: none"> - Dapat dimasukan Nama Merk
2	Daftar Merk	Menampilkan Daftar Merk mobil yang di rental	<ul style="list-style-type: none"> - Terdiri dari satu form - Satu edit text - Beberapa textbox - Beberapa button 	<ul style="list-style-type: none"> - daftar yang ditampilkan dapat di ubah dan dihapus sesuai kebutuhan pemilik rental

E. Kebutuhan Perangkat

Aplikasi ini memerlukan perangkat dan infrastruktur berikut:

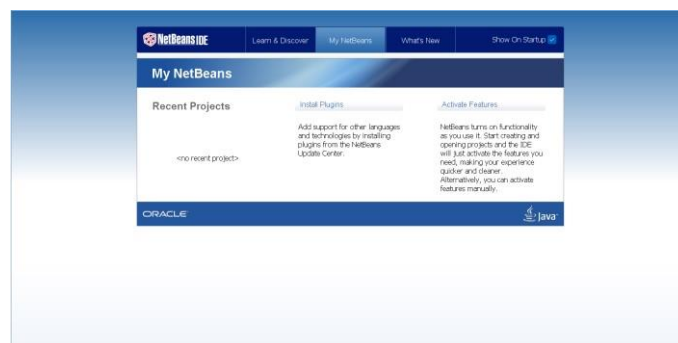
1. Platform Mobile: Aplikasi dapat diinstal dan digunakan pada perangkat seluler, seperti smartphone dan tablet, yang menjalankan sistem operasi Android dan iOS.
2. Koneksi Internet: Pengguna memerlukan koneksi internet untuk mengakses aplikasi, mencari mobil yang tersedia, melakukan reservasi, dan melakukan pembayaran.
3. Metode Pembayaran Online: Aplikasi ini memerlukan dukungan untuk berbagai metode pembayaran online, seperti kartu kredit, transfer bank, atau dompet digital, untuk melakukan pembayaran reservasi.
4. Antarmuka Admin: Penyedia layanan memerlukan antarmuka admin berbasis web atau aplikasi terpisah untuk mengelola inventaris mobil, melacak reservasi, dan mengelola informasi pelanggan.

D. Struktur Navigasi

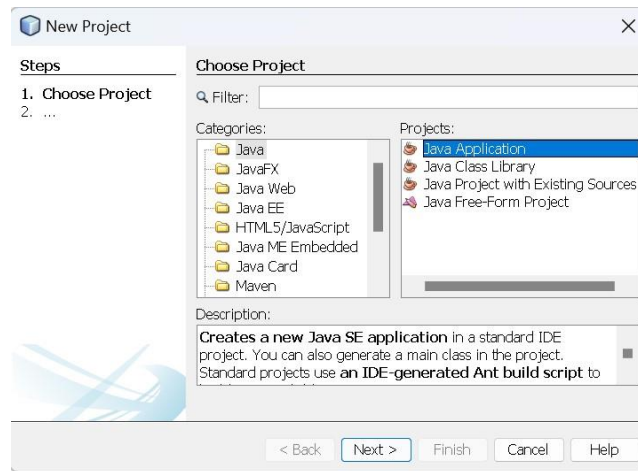


Project DepokApp dengan Spring Hibernate

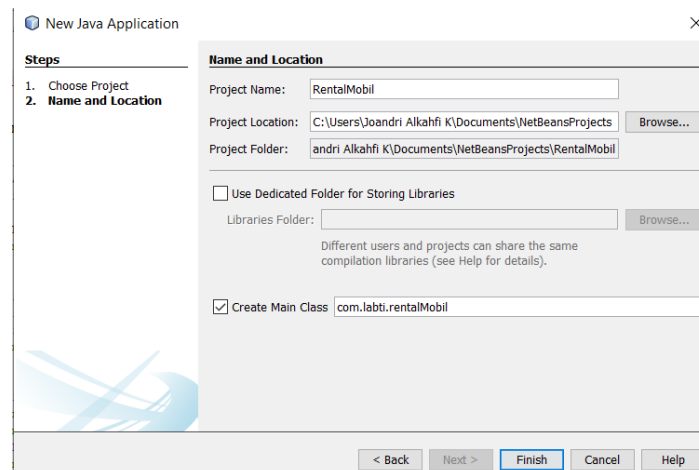
Tampilan awal netbeans, dengan versi yang digunakan adalah netbeans 8.2 dengan jdk 8 untuk menunjang projek ini.



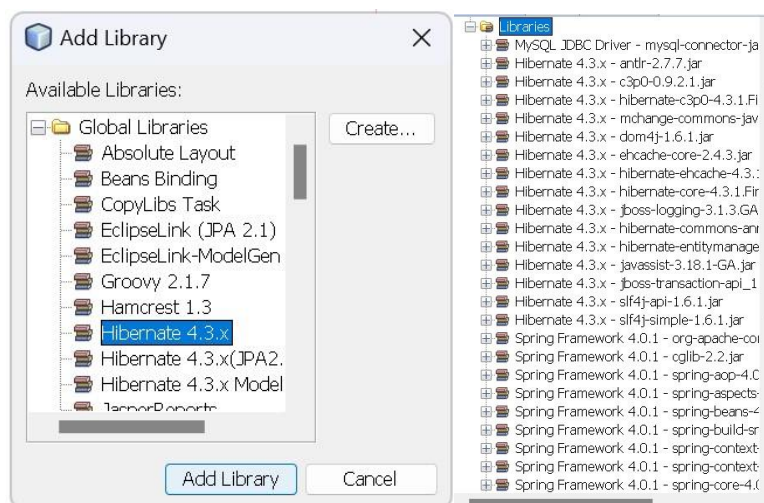
Membuat projek baru dengan klik “ new project “ pada bagian kiri atas pada netbeans, dan memilih kategori yaitu java kemudian projects java application.



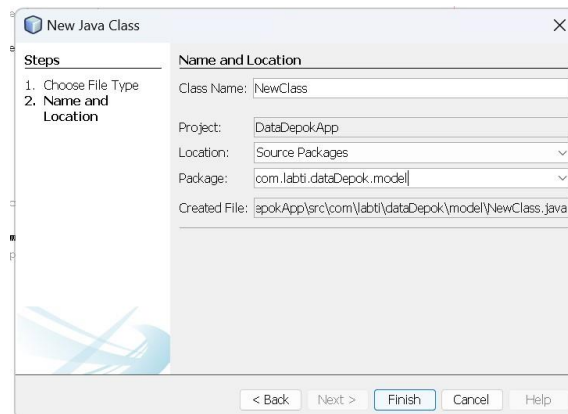
Memerikan nama proyek yaitu RentalMobil dengan nama package com.labti.rentalMobil



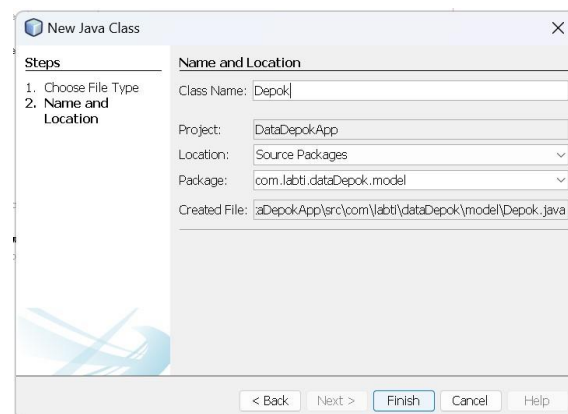
Menambahkan library yang diperlukan antara lain, yaitu hibernate, MySql JDBC driver, Spring framework dan lainnya.



Kemudian menambahkan package class yang diberi nama “com.labti.rentalMobil.model”, lalu klik finish.



Membuat file Rental.java pada package class “com.labti.rentalMobil.model”, kemudian membuat kode program berikut.

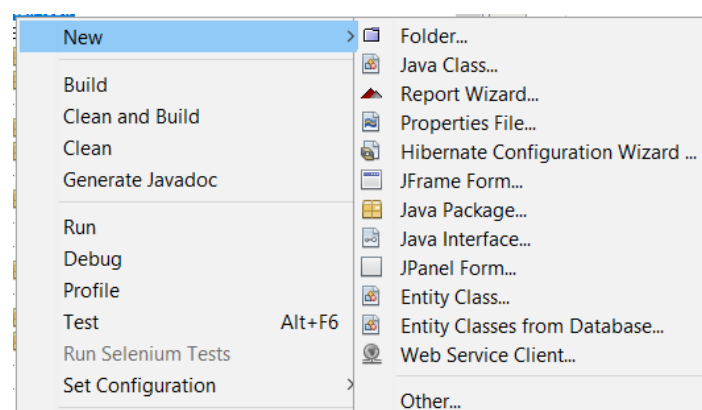


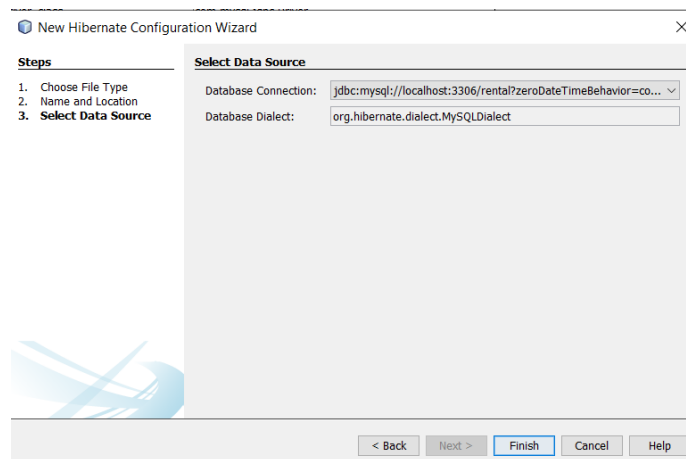
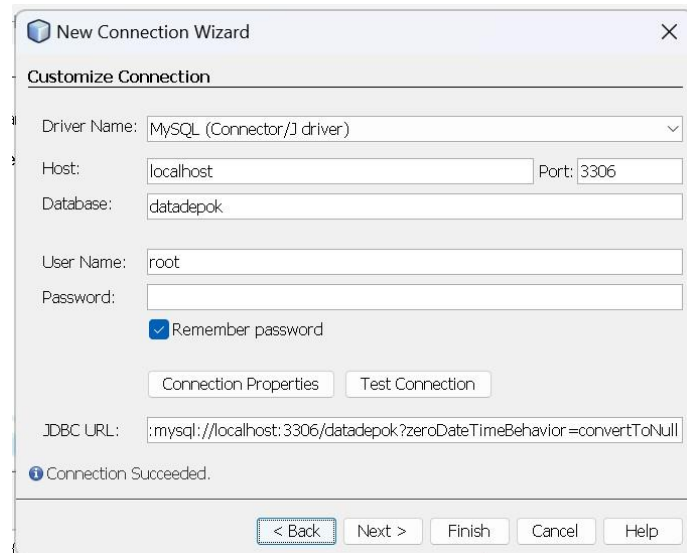

```

Start Page x Rental.java x
Source History
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6   package com.labti.rentalMobil.model;
7
8   import java.io.Serializable;
9   import javax.persistence.Column;
10  import javax.persistence.Entity;
11  import javax.persistence.Id;
12  import javax.persistence.Table;
13
14  @Table(name = "rental")
15  @Entity
16  public class Rental {
17      @Id
18      @Column(name = "tahun", length = 8)
19      private String tahun;
20
21      @Column(name = "nama", length = 50)
22      private String nama;
23
24      @Column(name = "merk", length = 50)
25      private String merk;
26
27      @Column(name = "deskripsi", length = 50)
28      private String deskripsi;
29
30      public String getTahun() {
31          return tahun;
32      }
33
34      public void setTahun(String tahun) {
35          this.tahun = tahun;
36      }
37
38      public String getNama() {
39          return nama;
40      }
41
42      public void setNama(String nama) {
43          this.nama = nama;
44      }
45
46      public String getMerk() {
47          return merk;
48      }
49
50      public void setMerk(String merk) {
51          this.merk = merk;
52      }
53
54      public String getDeskripsi() {
55          return deskripsi;
56      }
57
58      public void setDeskripsi(String deskripsi) {
59          this.deskripsi = deskripsi;
60      }
61  }

```

Menambahkan hibernate pada proyek yang kita buat, kemudian menambahkan database dari tabel untuk dihubungkan ke prjek netbeans.





Tabel pada database “ rental” mysql seperti pada gambar dibawah, yang berrisi 4 kolom (tahun, nama, merk dan deskripsi)

localhost/phpmyadmin/index.php?route=/table/structure&db=rental&table=rental

Server: 127.0.0.1 » Database: rental » Tabel: rental

Jelajahi Struktur SQL Cari Tambahkan Ekspor Impor Hak Akses Operasi Trigger

Struktur tabel Tampilan hubungan

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
<input type="checkbox"/> 1	tahun	int(8)			Tidak	Tidak ada			Ubah Hapus Lainnya
<input type="checkbox"/> 2	nama	varchar(50)	utf8mb4_general_ci	Ya	NULL				Ubah Hapus Lainnya
<input type="checkbox"/> 3	merk	varchar(50)	utf8mb4_general_ci	Ya	NULL				Ubah Hapus Lainnya
<input type="checkbox"/> 4	deskripsi	varchar(50)	utf8mb4_general_ci	Ya	NULL				Ubah Hapus Lainnya

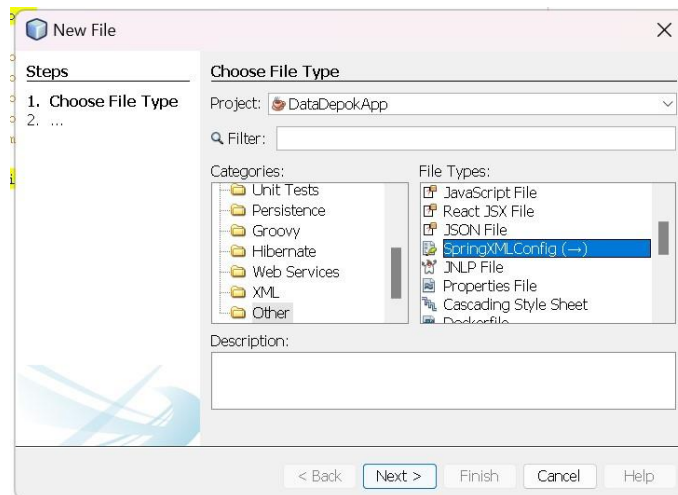
Melakukan konfigurasi pada hibernate dengan database pada mysql, kodingan seperti dibawah.

```

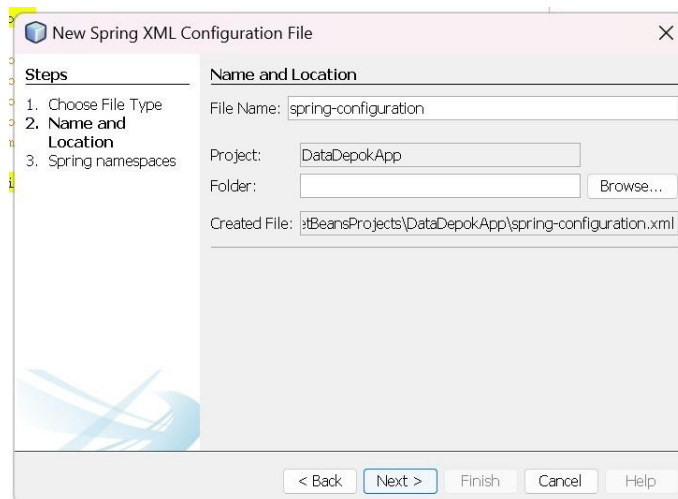
42 <hibernate-configuration>
43 <session-factory>
44 <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
45 <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
46 <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/rental?zeroDateTimeBehavior=convertToNull</property>
47 <property name="hibernate.connection.username">root</property>
48 <mapping class="com.labti.rentalMobil.model.Rental"/>
49 </session-factory>
50 </hibernate-configuration>

```

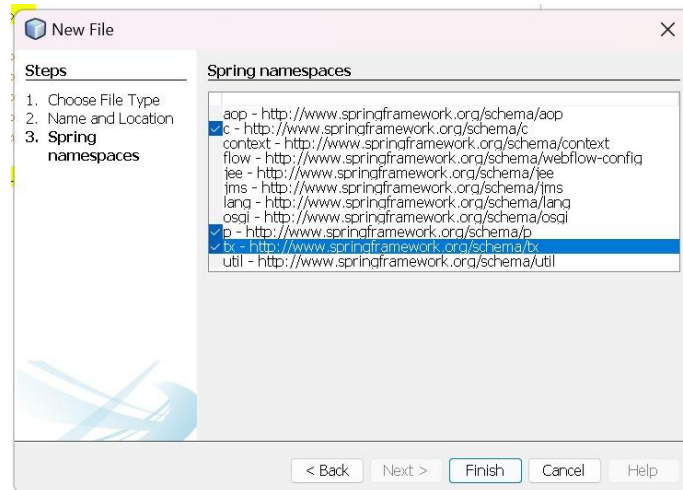
Melakukan konfigurasi pada spring dengan database pada mysql, langkahnya seperti pada dibawah.



Menambahkan framework spring



Menamai dengan spring-configuration



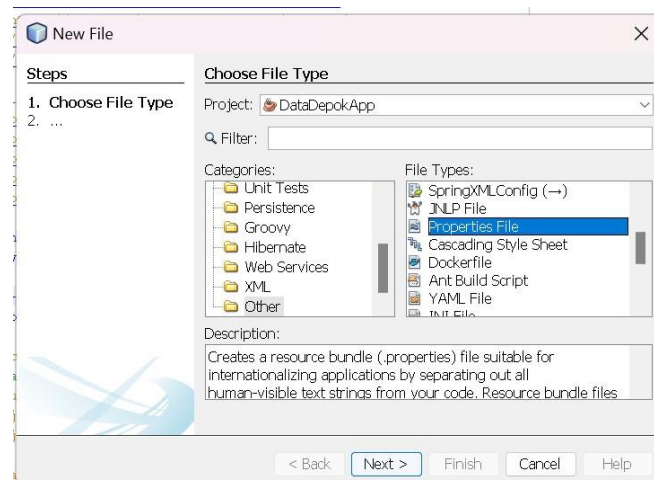
Menceklis bagian c, p dan tx

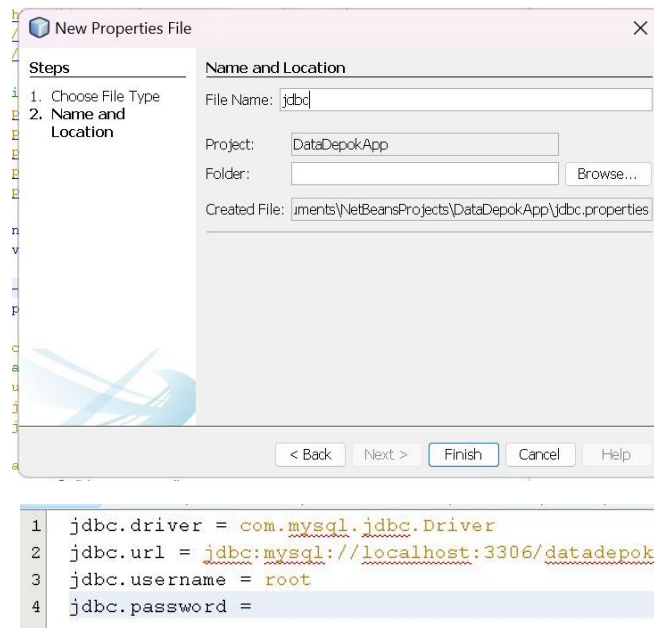
Melakukan konfigurasi pada jdbc dengan database pada mysql, langkahnya seperti pada dibawah.

```

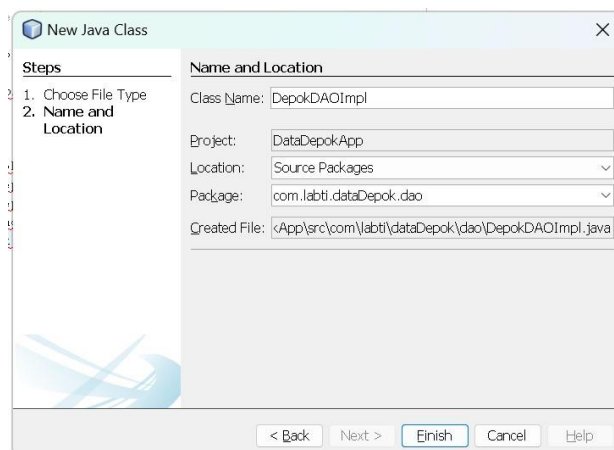
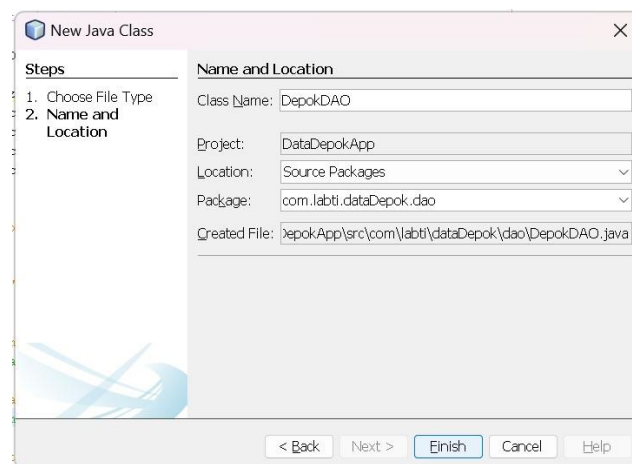
1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xmlns:context="http://www.springframework.org/schema/context"
5     xmlns:p="http://www.springframework.org/schema/p"
6     xmlns:tx="http://www.springframework.org/schema/tx"
7     xsi:schemaLocation="http://www.springframework.org/schema/beans
8         http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
9         http://www.springframework.org/schema/context
10        http://www.springframework.org/schema/context/spring-context-4.0.xsd
11        http://www.springframework.org/schema/tx
12        http://www.springframework.org/schema/tx/spring-tx-4.0.xsd">
13
14    <context:annotation-config />
15    <tx:annotation-driven transaction-manager="transactionManager" />
16
17    <context:component-scan base-package="com.lahti.databepok" />
18    <context:property-placeholder location="classpath:jdbc.properties" />
19
20    <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"
21        p:driverClassName="${jdbc.driver}"
22        p:url="${jdbc.url}"
23        p:username="${jdbc.username}"
24        p:password="${jdbc.password}" />
25
26    <bean id="sessionFactory" class="org.springframework.orm.hibernate4.LocalSessionFactoryBean"
27        p:dataSource-ref="dataSource"
28        p:configLocation="classpath:hibernate.cfg.xml" />
29
30    <bean id="transactionManager" class="org.springframework.orm.hibernate4.HibernateTransactionManager"
31        p:sessionFactory-ref="sessionFactory" />
32
33

```





Kemudian menambahkan package class yang diberi nama “ com.labti.rentalMobil.dao ”, lalu klik finish. Lalu membuat file class RentalDAO.java



Kodingan RentalDAO.java

```
6 package com.labti.dataDepok.dao;
7
8 import com.labti.dataDepok.model.Depok;
9 import java.util.List;
10
11 public interface DepokDAO {
12     public void save(Depok depok);
13     public void update(Depok depok);
14     public void delete(Depok depok);
15     public Depok getDepok(String id);
16     public List<Depok> getDepoks();
17 }
18
```

Kodingan RentalDAOImpl.java

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

@Repository
public class DepokDAOImpl implements DepokDAO {
    @Autowired
    private SessionFactory sessionFactory;

    @Override
    public void save(Depok depok) {
        sessionFactory.getCurrentSession().save(depok);
    }

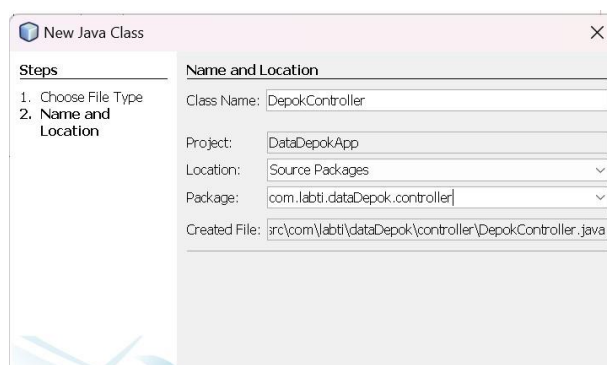
    @Override
    public void update(Depok depok) {
        sessionFactory.getCurrentSession().update(depok);
    }

    @Override
    public void delete(Depok depok) {
        sessionFactory.getCurrentSession().delete(depok);
    }

    @Override
    public Depok getDepok(String id) {
        return (Depok) sessionFactory.getCurrentSession().get(Depok.class, id);
    }

    @Override
    public List<Depok> getDepoks() {
        return sessionFactory.getCurrentSession().createCriteria(Depok.class).list();
    }
}
```

Kemudian menambahkan package class yang diberi nama “com.labti.dataRental.controller”, lalu klik finish. Lalu membuat file class RentalController.java



Kodingan sebagai berikut:

```
package com.labti.dataDepok.controller;

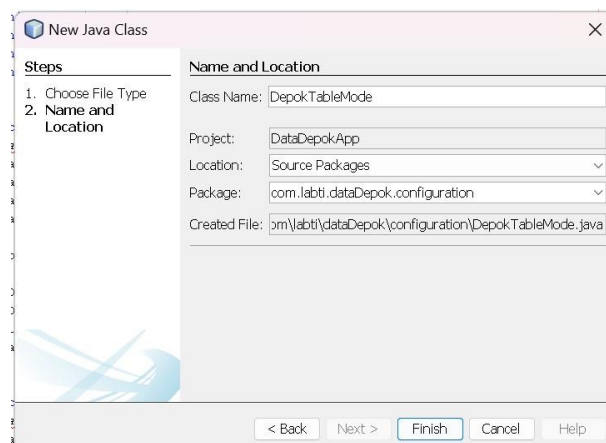
import com.labti.dataDepok.app;
import com.labti.dataDepok.config.DepokTableModel;
import com.labti.dataDepok.model.Depok;
import com.labti.dataDepok.view.DepokView;
import java.util.List;
import javax.swing.JOptionPane;
import javax.swing.table.TableModel;
```

```

16 public class DepokController {
17     private final DepokView depokView;
18     private DepokTableModel depokTableModel;
19     private List<Depok> depoks;
20
21     public DepokController (DepokView depokView) {
22         this.depokView = depokView;
23     }
24
25     public void tampilData() {
26         depoks = app.getDepokService().getDepoks();
27         depokTableModel = new DepokTableModel(depoks);
28         this.depokView.getTabel().setModel((TableModel) depokTableModel);
29     }
30
31     public void show() {
32         int index = this.depokView.getTabel().getSelectedRow();
33         this.depokView.getNpm().setText(String.valueOf(
34             this.depokView.getTabel().getValueAt(index, 0)));
35         this.depokView.getNama().setText(String.valueOf(
36             this.depokView.getTabel().getValueAt(index, 1)));
37         this.depokView.getAlamat().setText(String.valueOf(
38             this.depokView.getTabel().getValueAt(index, 2)));
39         this.depokView.getRating().setText(String.valueOf(
40             this.depokView.getTabel().getValueAt(index, 3)));
41     }
42
43     public void clear() {
44         this.depokView.getNpm().setText("");
45         this.depokView.getNama().setText("");
46         this.depokView.getAlamat().setText("");
47         this.depokView.getRating().setText("");
48     }
49
50     public void saveDepok() {
51         Depok depok = new Depok();
52         depok.setId(this.depokView.getNpm().getText());
53         depok.setNama(this.depokView.getNama().getText());
54         depok.setAlamat(this.depokView.getAlamat().getText());
55         depok.setRating(this.depokView.getRating().getText());
56         app.getDepokService().save(depok);
57         JOptionPane.showMessageDialog(null, "Data Berhasil di simpan", "info",
58             JOptionPane.INFORMATION_MESSAGE);
59         clear();
60         tampilData();
61     }
62
63     public void updateDepok() {
64         Depok depok = new Depok();
65         depok.setId(this.depokView.getNpm().getText());
66         depok.setNama(this.depokView.getNama().getText());
67         depok.setAlamat(this.depokView.getAlamat().getText());
68         depok.setRating(this.depokView.getRating().getText());
69         app.getDepokService().update(depok);
70         JOptionPane.showMessageDialog(null, "Data berhasil di Edit", "info",
71             JOptionPane.INFORMATION_MESSAGE);
72         clear();
73         tampilData();
74     }
75
76     public void deleteDepok() {
77         if(this.depokView.getNpm().getText() == null){
78             JOptionPane.showMessageDialog(null, "Silahkan pilih id", "error", JOptionPane.ERROR_MESSAGE);
79         }
80         Depok depok = new Depok();
81         depok.setId(this.depokView.getNpm().getText());
82         int pilih = JOptionPane.showConfirmDialog(null, "Apakah data ingin dihapus ?", "Warning", JOptionPane.YES_NO_OPTION,
83             JOptionPane.WARNING_MESSAGE);
84         if (pilih == JOptionPane.YES_OPTION) {
85             app.getDepokService().delete(depok);
86             JOptionPane.showMessageDialog(null, "Data Berhasil di Hapus", "info", JOptionPane.INFORMATION_MESSAGE);
87             clear();
88             tampilData();
89         }
90     }
91 }

```

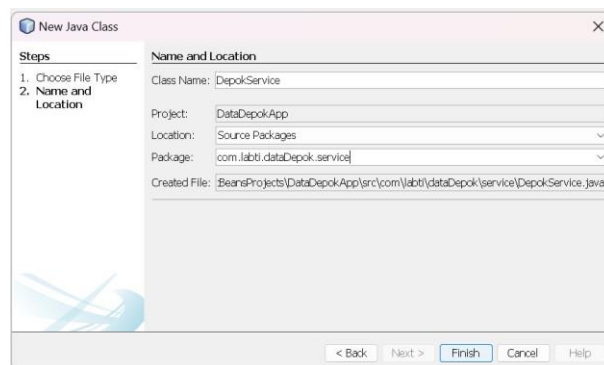
Kemudian menambahkan package class yang diberi nama “com.labti.dataRental.config”, lalu klik finish. Lalu membuat file class RentalTableModel.java



Kodingan sebagai berikut:

```
6 package com.labti.dataDepok.config;
7
8 import com.labti.dataDepok.model.Depok;
9 import java.util.ArrayList;
10 import java.util.List;
11 import javax.swing.table.AbstractTableModel;
12
13 public class DepokTableModel extends AbstractTableModel {
14     private List<Depok> depoks = new ArrayList<>();
15     private final String HEADER[] = {"id", "nama", "alamat", "rating"};
16
17     public DepokTableModel(List<Depok> mahasiswa) {
18         this.depoks = mahasiswa;
19     }
20
21     @Override
22     public int getRowCount() {
23         return depoks.size();
24     }
25
26     @Override
27     public int getColumnCount() {
28         return HEADER.length;
29     }
30
31     @Override
32     public String getColumnName(int columnIndex) {
33         return HEADER[columnIndex];
34     }
35
36     @Override
37     public Object getValueAt(int rowIndex, int columnIndex) {
38         Depok depok = depoks.get(rowIndex);
39         switch (columnIndex) {
40             case 0:
41                 return depok.getId();
42             case 1:
43                 return depok.getNama();
44             case 2:
45                 return depok.getAlamat();
46             case 3:
47                 return depok.getRating();
48             default:
49                 return null;
50         }
51     }
52 }
```

Kemudian menambahkan package class yang diberi nama “com.labti.dataRental.service”, lalu klik finish. Lalu membuat file class RentalService.java dan RentalServiceImpl.java



Kodingan RentalService.java


```

6 package com.labti.dataDepok.service;
7
8 import com.labti.dataDepok.model.Depok;
9 import java.util.List;
10
11 public interface DepokService {
12     public void save(Depok depok);
13     public void update(Depok depok);
14     public void delete(Depok depok);
15     public Depok getDepok(String npm);
16     public List<Depok> getDepoks();
17 }

```

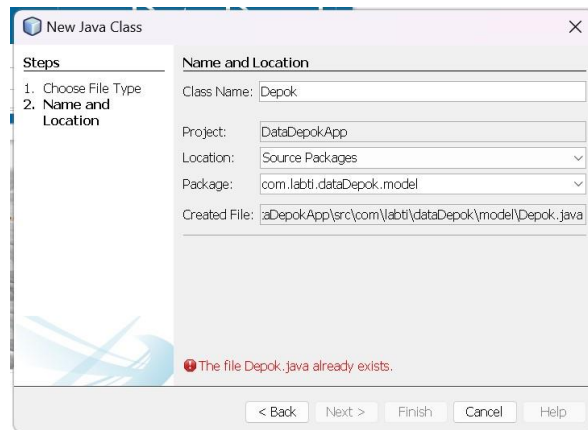
Kodingannya sebagai berikut:

```

18 public class DepokServiceImpl implements DepokService {
19     @Autowired
20     private DepokDAO depokDao;
21
22     @Transactional
23     @Override
24     public void save(Depok depok) {
25         depokDao.save(depok);
26     }
27
28     @Transactional
29     @Override
30     public void update(Depok depok) {
31         depokDao.update(depok);
32     }
33
34     @Transactional
35     @Override
36     public void delete(Depok depok) {
37         depokDao.delete(depok);
38     }
39
40     @Override
41     public Depok getDepok(String npm) {
42         return depokDao.getDepok(npm);
43     }
44
45     @Override
46     public List<Depok> getDepoks() {
47         return depokDao.getDepoks();
48     }
49 }

```

Kemudian menambahkan package class yang diberi nama “com.labti.dataRental.model”, klik finish. Lalu membuat file class Rental.java



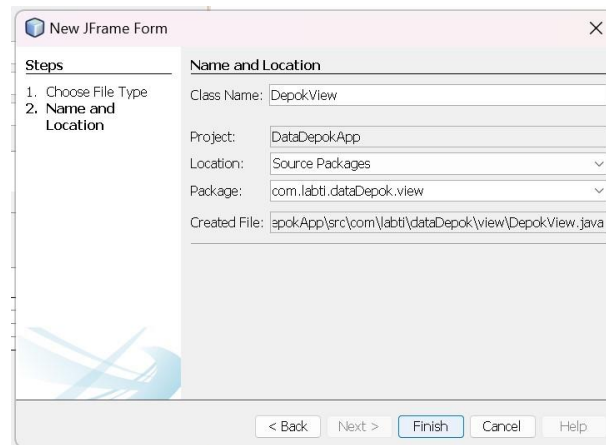
Pada bagian app.java, dengan file ini project akan di eksekusi dan akan menampilkan output. Kodingan pada app.java adalah sebagai berikut:

```

6 package com.labti.dataDepok;
7
8 import com.labti.dataDepok.view.DepokView;
9 import org.springframework.context.ApplicationContext;
10 import org.springframework.context.support.ClassPathXmlApplicationContext;
11 import com.labti.dataDepok.service.DepokService;
12
13 public class app {
14
15     private static ApplicationContext applicationContext;
16
17     public static void main(String[] args){
18         applicationContext = new ClassPathXmlApplicationContext("classpath:spring-configuration.xml");
19         new DepokView().setVisible(true);
20     }
21
22     public static DepokService getDepokService(){
23         return (DepokService)applicationContext.getBean("DepokService");
24     }
25 }
26
27

```

Kemudian menambahkan package class yang diberi nama “com.labti.rentalMobil.view”, lalu klik finish. Lalu membuat file JFrame dengan nama RentalView.java



Desain JFrame RentalView.java

Data Depok

Id

Nama

Alamat

Rating

Title 1	Title 2	Title 3	Title 4

Kodingan untuk RentalView.java adalah sebagai berikut:

```

6      package com.labti.dataDepok.view;
7
8      import com.labti.dataDepok.controller.DepokController;
9      import javax.swing.JOptionPane;
10     import javax.swing.JTable;
11     import javax.swing.JTextField;
12     import javax.swing.JTextArea;
13     import net.sf.jasperreports.engine.JasperFillManager;
14     import net.sf.jasperreports.engine.JasperPrint;
15     import net.sf.jasperreports.view.JasperViewer;
16
17     public class DepokView extends javax.swing.JFrame {
18         private final DepokController depokController = new DepokController(this);
19
20         /**
21          * Creates new form DepokView
22          */
23         public DepokView() {
24             initComponents();
25             depokController.tampilData();
26         }

```

Mengimport semua package yang dibutuhkan dan juga dengan package ReportJaspers, yang akan dibahas berikut ini.

```

177     private void simpanActionPerformed(java.awt.event.ActionEvent evt) {
178         depokController.saveDepok();
179         try {
180             JasperPrint jp = JasperFillManager.fillReport(getClass().getResourceAsStream("reportDepok.jasper"), null, Konakai.getConnection());
181             JasperViewer.viewReport(jp, false);
182         } catch (Exception e) {
183             JOptionPane.showMessageDialog(rootPane, e);
184         }
185     }
186
187     private void updateActionPerformed(java.awt.event.ActionEvent evt) {
188         depokController.updateDepok();
189     }
190
191     private void hapusActionPerformed(java.awt.event.ActionEvent evt) {
192         depokController.deleteDepok();
193     }
194
195     private void kembalikanMouseClicked(java.awt.event.MouseEvent evt) {
196         depokController.show();
197     }
198
199     /**
200      * @param args the command line arguments
201      */
202     public static void main(String args[]) {
203         /* Set the Window look and feel */
204         /* Look and feel setting code (optional) */
205
206         /* Create and display the form */
207         java.awt.EventQueue.invokeLater(new Runnable() {
208             public void run() {
209                 new DepokView().setVisible(true);
210             }
211         });
212     }

```

Berikut adalah source code pada setiap button yaitu simpan, update dan delete. Termasuk pada bagian button simpan terdapat kodingan untuk iReport.

```

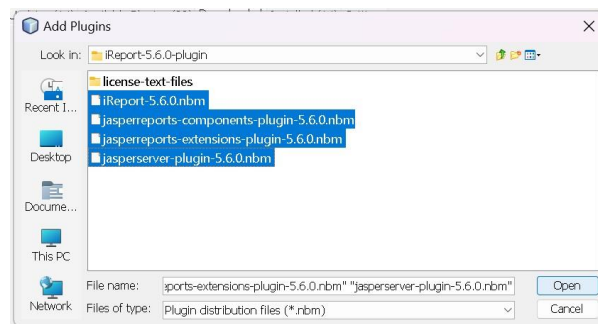
234 public JTextArea getAlamat() {
235     return alamat;
236 }
237
238 public void setAlamat(JTextArea alamat) {
239     this.alamat = alamat;
240 }
241
242 public JTextField getRating() {
243     return rating;
244 }
245
246 public void setRating(JTextField harga) {
247     this.rating = harga;
248 }
249
250 public JTextField getName() {
251     return nama;
252 }
253
254 public void setName(JTextField Nama) {
255     this.nama = Nama;
256 }
257
258 public JTextField getId() {
259     return id;
260 }
261
262 public void setId(JTextField id) {
263     this.id = id;
264 }
265 public JTable getTable() {
266     return tabel;
267 }

```

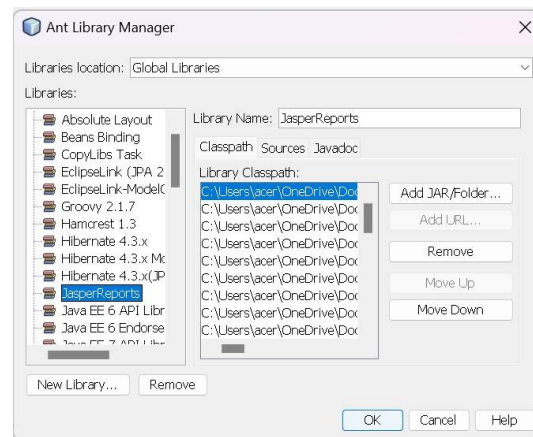
Memasukan fungsi set dan get untuk setiap textfield dari 4 data yang ada (tahun, nama/place, merk/text area, dan deskripsi) dan juga pada tabel.

Menambahkan iReport pada Project DepokApp

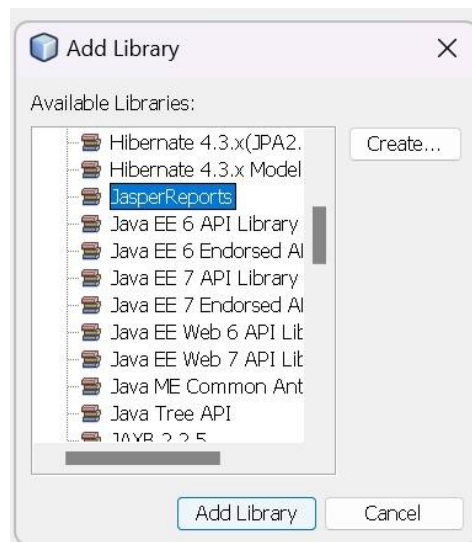
Menambahkan plugin iReport yang diperlukan untuk ditambahkan dalam project RentalApp dengan Spring Hibernate sebelumnya.



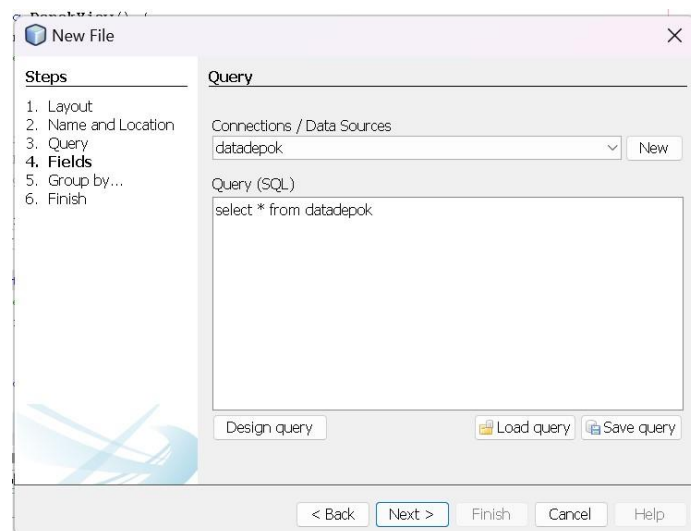
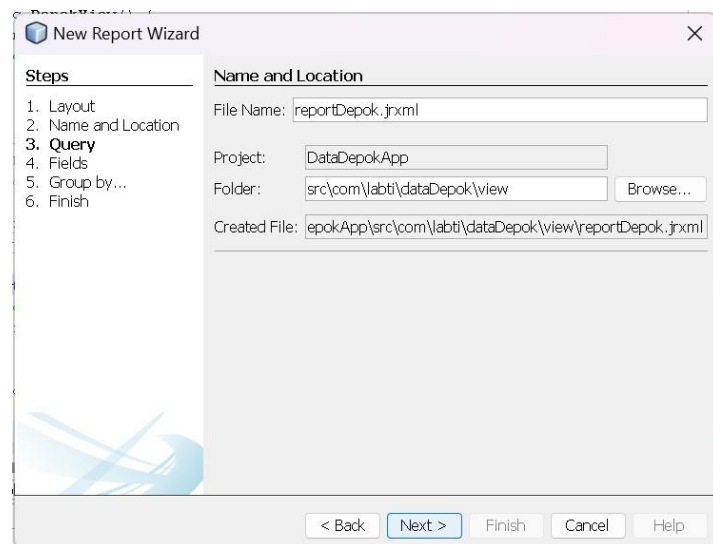
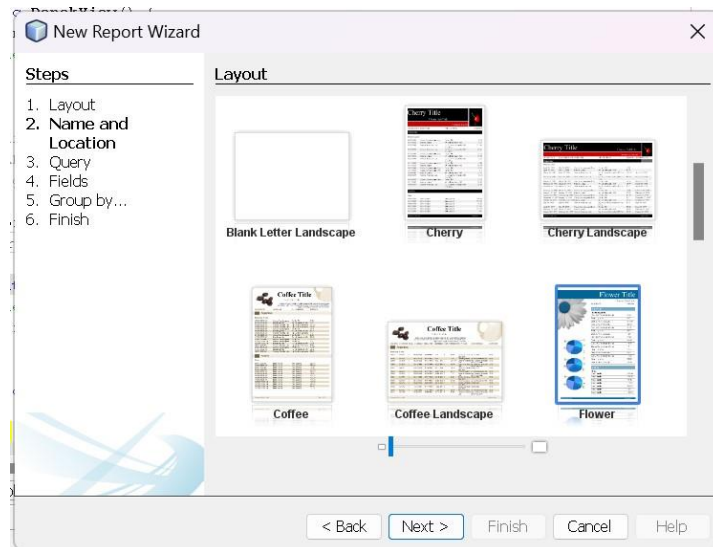
Kemudian menambahkan libraries JasperReports



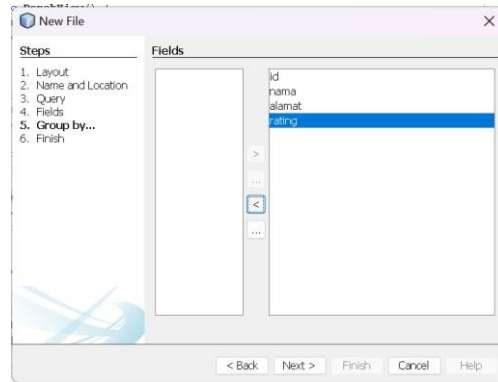
Menambahkan libraries JasperReports pada project



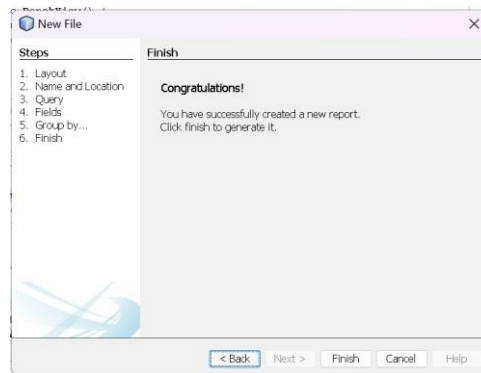
Menambahkan file dengan Report Wizard untuk membuat step-step seperti membuat layout, name and location, query, fields dan sebagainya.



Dapat dilihat setiap langkah, mulai dari layout, nama dan juga connection dengan database.



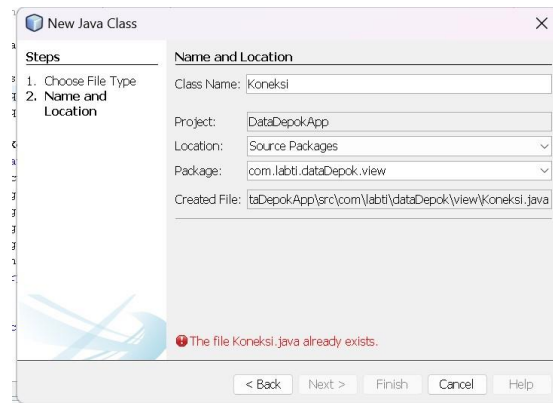
#	Nama	Jenis	Penyortiran	Atribut Kosong	Bawaan Ekstra
<input type="checkbox"/> 1	id	int(8)		Ya	NULL
<input type="checkbox"/> 2	nama	varchar(50) latin1_swedish_ci		Ya	NULL
<input type="checkbox"/> 3	alamat	varchar(50) latin1_swedish_ci		Ya	NULL
<input type="checkbox"/> 4	rating	int(8)		Ya	NULL



Berikut file layout design untuk menampilkan output hasil report dari project yang menampilkan rental mobil

Rental Mobil				
Murah Meriah				
Page Header				
tahun	nama	merk	deskripsi	
\$F{tahun}	\$F{nama}	\$F{merk}	\$F{deskripsi}	
new java.util.Date()				
"Page "+\$V{PAGE_NUMBER}+" of" " + \$V				

Kemudian menambahkan class pada package com.labti.rentalMobil.view yaitu Koneksi.java

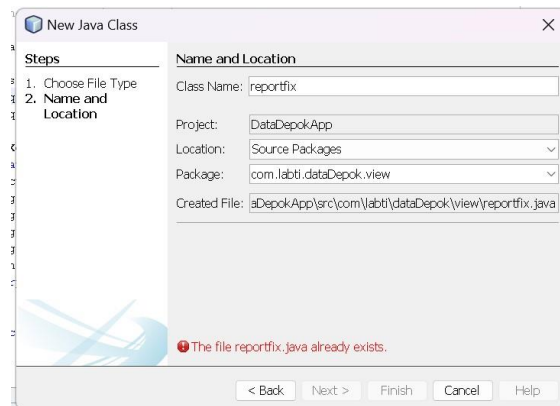


```

6 package com.labti.dataDepok.view;
7
8 import com.mysql.jdbc.Connection;
9 import java.sql.DriverManager;
10 import java.sql.SQLException;
11
12 public class Koneksi {
13     public static Connection getConnection() {
14         Connection connection = null;
15         String driver = "com.mysql.jdbc.Driver";
16         String url = "jdbc:mysql://localhost:3306/datadepok";
17         String user = "root";
18         String password = "";
19         if (connection == null) {
20             try {
21                 Class.forName(driver);
22                 connection = (Connection) DriverManager.getConnection(url, user, password);
23             } catch (ClassNotFoundException | SQLException error) {
24                 System.exit(0);
25             }
26         }
27         return connection;
28     }
29 }

```

Kemudian menambahkan class pada package com.labti.rentalMobil.view yaitu reportffix.java



```

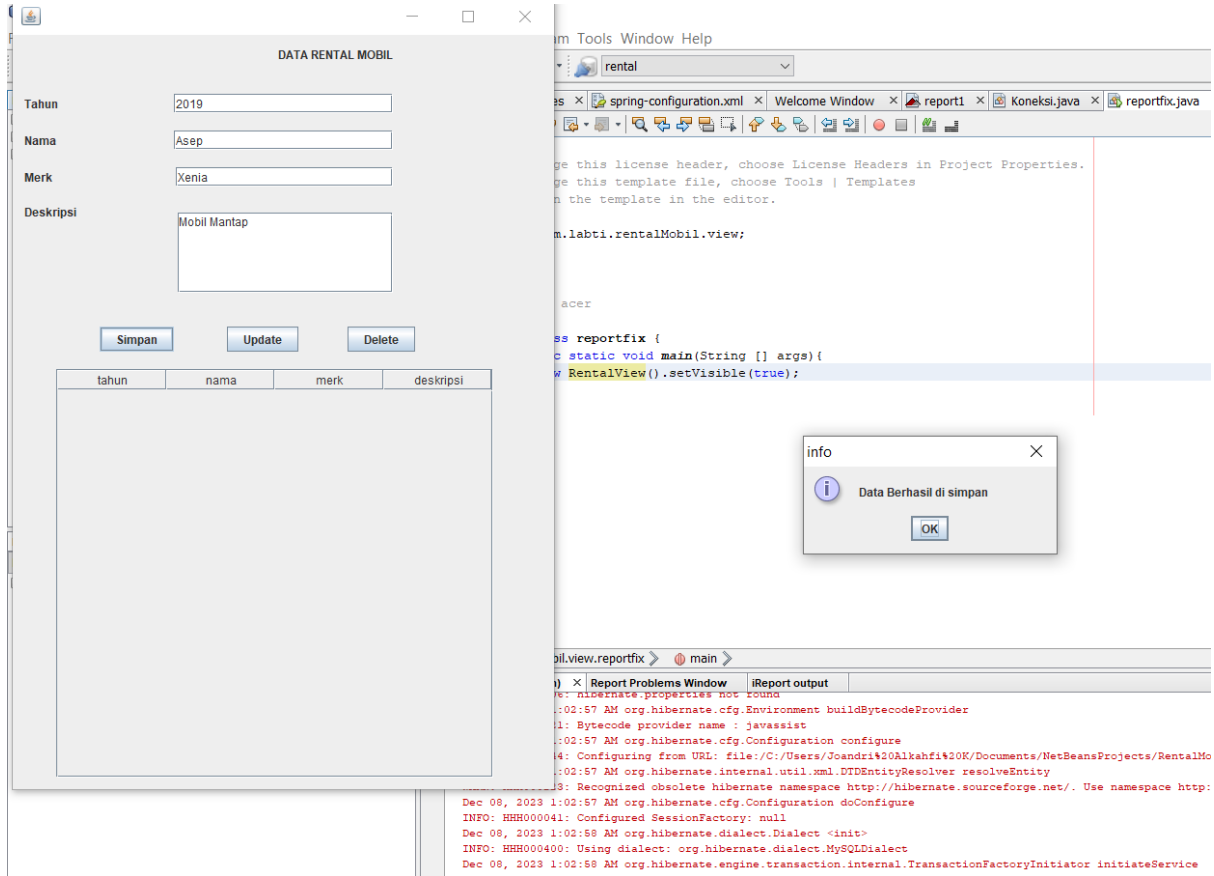
6 package com.labti.dataDepok.view;
7
8 /**
9  *
10  * @author acer
11  */
12 public class reportfix {
13     public static void main(String [] args) {
14         new DepokView().setVisible(true);
15     }
16 }
17

```

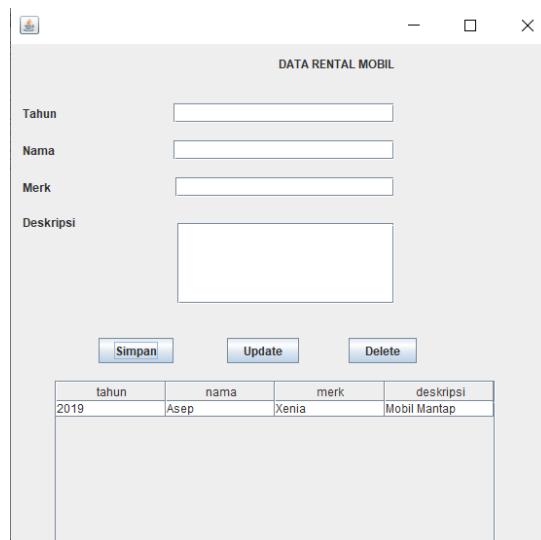
Output Project DepokApp

“Data Depok”







Menjalankan (run) file app.java



Memasukan 1 data seperti pada gambar diatas.



Data akan tampil pada tabel.

				tahun	nama	merk	deskripsi
<input type="checkbox"/>		Ubah		Salin		Hapus	2019 Asep Xenia Mobil Mantap
<input type="checkbox"/>		Ubah		Salin		Hapus	2020 Joandri Avanza Mobil siapa

Memasukan Project ke dalam GitHub

Karena terdapat database, maka file database nya akan di ekspor dulu

Mengekspor tabel dari basis data "rental"

Metode ekspor:

☒ Cepat - menampilkan opsi minimum


☐ Kustom - menampilkan semua opsi

Format:

SQL


Ekspor

Membuka GitHub


GwGgDong

+
-
@
#

Overview
Repositories
Projects
Packages
Stars



Joandri Alkahfi K
 GwGgDong
[Edit profile](#)
 0 followers · 1 following

Popular repositories

pweb

gatau

Hack

Public

PTJONI

Public

LabIRPL2

Java

Public

7 contributions in the last year

Dec
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov

Mon
Wed
Fri



Contribution settings
2023
2022
2021

Membuat repository baru, dan login pada Git Bash

Create a new repository

A repository contains all project files, including the revision history. Already have : [Import a repository.](#)


Required fields are marked with an asterisk (*).


Owner * /

Repository name *

Great repository names are short and memorable. Need inspiration? How about s

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

```
Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$ git init
Initialized empty Git repository in C:/Users/Joandri Alkahfi K/Downloads/Rental/.git/

Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$ git add .
warning: in the working copy of 'RentalMobil/build/classes/com/labti/rentalMobil/view/RentalView.form', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'RentalMobil/build/classes/com/labti/rentalMobil/view/report1.jrxml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'RentalMobil/build/classes/hibernate.cfg.xml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'RentalMobil/build/classes/jdbc.properties', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'RentalMobil/build/classes/spring-configuration.xml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'RentalMobil/nbproject/private/retriever/www.spr
```

Melakukan perintah perintah untuk mengupload projek, git init git add README.md

git commit -m "first commit" git branch -M main git remote add origin <https://github.com/GwGgDong/rentalMobil.git> git push -u origin main

```
Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$ git push
fatal: The current branch master has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin master

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$ ^C

Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$ git push --set-upstream origin master
Enumerating objects: 80, done.
Counting objects: 100% (80/80), done.
Delta compression using up to 12 threads
Compressing objects: 100% (66/66), done.
Writing objects: 100% (80/80), 66.21 KiB | 4.41 MiB/s, done.
Total 80 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (7/7), done.
To https://github.com/GwGgDong/LabtiRPL2.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$
```

Pada git status sudah terlihat sudah ter-add folder Rental, yang terdapat project nya yaitu (folder project netbeans) dan file rental.sql yaitu file database nya, tempat menyimpan data dari project.

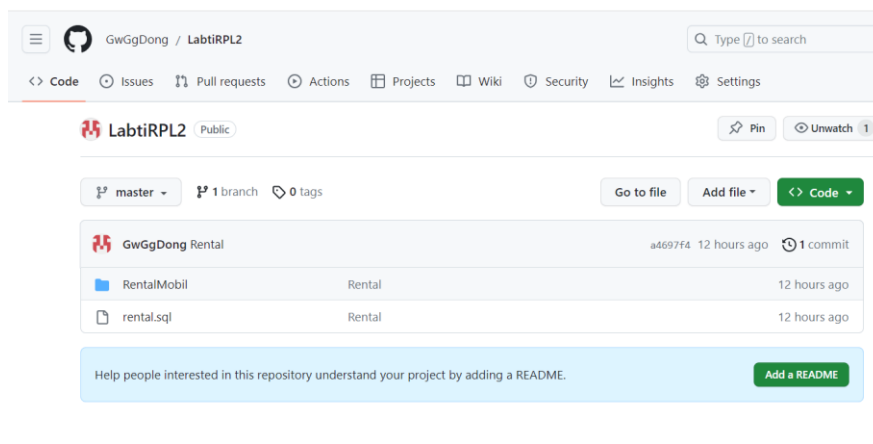
Melakukan first commit

```

Joandri Alkahfi K@HP-PAVILION-GAMING-15-JAK36 MINGW64 ~/Downloads/Rental (master)
$ git commit -m "Rental"
[master (root-commit) a4697f4] Rental
55 files changed, 5331 insertions(+)
create mode 100644 RentalMobil/build.xml
create mode 100644 RentalMobil/build/classes/.netbeans_automatic_build
create mode 100644 RentalMobil/build/classes/.netbeans_update_resources
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/app.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/config/RentalTableModel.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/controller/RentalController.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/dao/RentalDAO.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/dao/RentalDAOImpl.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/model/Rental.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/service/RentalService.class
create mode 100644 RentalMobil/build/classes/com/labti/rentalMobil/service/RentalServiceImpl.class

```

Setelah itu, melakukan git remote untuk menambahkan sebuah track ke remote repository berada yaitu di <https://github.com/GwGgDong/rentalMobil.git>



Maka sudah terlihat project sudah terupload ke dalam repository Project berhasil ter upload ke dalam repository

Link Reposiroty project depokapp :

<https://github.com/GwGgDong/LabtiRPL2>

link github :

<https://github.com/GwGgDong>

=====TERIMAKASIH=====