

LAPORAN
TUGAS BESAR

PEMROGRAMAN BERORIENTASI OBJEK

APLIKASI PEMESANAN TIKET



Kelas : IF 09 K
Kelompok : 5

No	NIM	Nama Mahasiswa
1	21102010	Bunga Laelatul Muna
2	21102015	Nazwa Aulia Rakhma
3	21102020	Aisyah Hasna Aulia
4	21102027	Sukhaenah Tri Utami

PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
SEMESTER GENAP 2023/2024

DAFTAR ISI

COVER	1
DAFTAR ISI.....	1
1. PENDAHULUAN	2
2. RANCANGAN APLIKASI.....	3
2.1 FUNGSIONALITAS PRODUK	3
2.2 RANCANGAN DATA	4
3. IMPLEMENTASI APLIKASI.....	6
3.1 IMPLEMENTASI ANTARMUKA	6
3.1.1 Login	6
3.1.2 Kereta	10
3.1.3 Bus.....	21
3.1.4 Pesawat	30
3.1 DIAGRAM KELAS	39

1. Pendahuluan

Dalam pembuatan tugas besar ini, kami mengembangkan sebuah Java GUI dalam bentuk desktop untuk pemesanan tiket yang menggabungkan penggunaan MySQL sebagai basis data dan Java GUI sebagai antarmuka pengguna. Java GUI ini bertujuan untuk menyediakan platform yang mudah digunakan bagi pengguna dalam memesan tiket perjalanan baik tiket bus, kereta maupun pesawat.

Dengan adanya platform tersebut memberikan dampak dan manfaat yang baik bagi masyarakat seperti kemudahan pemesanan tiket karena dalam platform yang kami buat menyediakan antarmuka pengguna yang mudah digunakan bagi khalayak umum sehingga pengguna dapat dengan mudah melakukan pemesanan tiket tanpa kesulitan. Mereka dapat memilih jenis transportasi yang diinginkan (kereta, pesawat atau bus) dan memilih jalur perjalanan yang diinginkan.

Kemudian manfaat lainnya adalah dalam bidang manajemen data penumpang platform ini juga memungkinkan pengguna untuk mengelola data penumpang secara efisien. Pengguna dapat memasukkan informasi penting seperti nama, nomor identitas (NIK), nomor handphone dan alamat penumpang. Data ini akan disimpan dalam database MySQL sehingga dapat diakses dan dikelola dengan mudah.

Selanjutnya adalah dalam mengelola harga dan jurusan platform ini menyediakan fitur pengelolaan harga dan jurusan transportasi. User dapat mengubah harga tiket, menambahkan atau menghapus jurusan yang sudah ditampilkan di halaman awal. Dalam Program ini menggunakan exception user defined pada bagian pengisian NIK dan exception syntax error untuk bagian button beli serta exception number untuk bagian harga.

Dengan menggabungkan MySQL sebagai basis data dan Java GUI sebagai antarmuka pengguna, platform ini memberikan solusi yang tanggap dalam pemesanan tiket transportasi. Dengan kemudahan penggunaan, pengelolaan data yang efisien dan memberikan pengalaman yang nyaman dan praktis bagi pengguna yang ingin memesan tiket transportasi dengan cepat dan mudah.

2. Rancangan Aplikasi

2.1 Fungsionalitas Produk

Nama	:	Login
Deskripsi	:	Fungsionalitas ini memungkinkan pengguna untuk masuk ke dalam aplikasi dengan menggunakan akun yang telah terdaftar. Pengguna akan diminta untuk memasukkan username dan password yang valid untuk mengakses fitur-fitur lain dalam aplikasi.

Nama	:	Pilihan Pembelian
Deskripsi	:	Fungsionalitas ini menyediakan pengguna dengan pilihan untuk membeli tiket bus, tiket kereta, atau tiket pesawat. Pengguna akan dapat memilih jenis transportasi yang diinginkan dan melanjutkan proses pembelian tiket.

Nama	:	Tampilan Tiket Bus; Tampilan Tiket Kereta; Tampilan Tiket Pesawat
Deskripsi	:	Fungsionalitas ini menampilkan pemesanan tiket bus, kereta, dan pesawat yang tersedia. Pengguna dapat menginputkan rincian tiket, seperti kota asal, tujuan, tanggal keberangkatan, harga, dan identitas penumpang.

Nama	:	Mencari Penumpang
Deskripsi	:	Fungsionalitas ini memungkinkan pengguna untuk mencari data penumpang yang telah terdaftar dalam sistem. Pengguna dapat memasukkan kriteria pencarian, seperti nama penumpang atau identitas lain, dan sistem akan menampilkan hasil pencarian yang sesuai.

Nama	:	Edit Data Penumpang
Deskripsi	:	Fungsionalitas ini memungkinkan user admin untuk mengedit data penumpang yang telah terdaftar dalam sistem. Admin dapat mengubah informasi mengenai data penumpang serta detail lainnya tergantung pada persyaratan aplikasi.

Nama	:	Hapus Data Penumpang
Deskripsi	:	Fungsionalitas ini memungkinkan user admin untuk menghapus data penumpang yang telah terdaftar dalam sistem. Admin dapat menghapus data penumpang serta detail lainnya tergantung pada aplikasi.

2.2 Rancangan Data

Nama Tabel 01	:	login
No		Nama Atribut
1		username
2		password

Nama Tabel 02	:	bus
No		Nama Atribut
1		NIK (PK)
2		nama
3		kotaAsal
4		tujuan
5		tanggalBerangkat
6		harga

Nama Tabel 03	:	penumpang_bus
No	Nama Atribut	Tipe Data
1	NIK (FK)	varchar(20)
2	nama	varchar(50)
3	kotaAsal	varchar(50)
4	tujuan	varchar(50)
5	tanggalBerangkat	date
6	alamat	varchar(100)
7	noHp	varchar(15)

Nama Tabel 04	:	kereta
No	Nama Atribut	Tipe Data
1	NIK (PK)	varchar(20)
2	nama	varchar(50)
3	kotaAsal	varchar(50)
4	tujuan	varchar(50)
5	tanggalBerangkat	date
6	harga	int(11)

Nama Tabel 05	:	penumpang_kereta
No	Nama Atribut	Tipe Data
1	NIK (FK)	varchar(20)
2	nama	varchar(50)
3	kotaAsal	varchar(50)
4	tujuan	varchar(50)
5	tanggalBerangkat	date
6	alamat	varchar(100)
7	noHp	varchar(15)

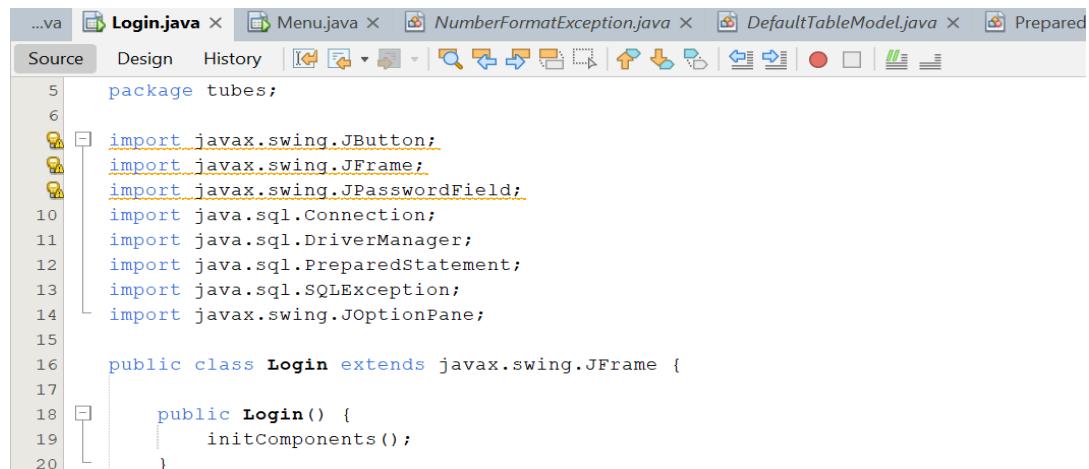
Nama Tabel 06	:	pesawat
No	Nama Atribut	Tipe Data
1	NIK (PK)	varchar(20)
2	nama	varchar(50)
3	kotaAsal	varchar(50)
4	tujuan	varchar(50)
5	tanggalBerangkat	date
6	harga	int(11)

Nama Tabel 07	:	penumpang
No	Nama Atribut	Tipe Data
1	NIK (FK)	varchar(20)
2	nama	varchar(50)
3	kotaAsal	varchar(50)
4	tujuan	varchar(50)
5	tanggalBerangkat	date
6	alamat	varchar(100)
7	noHp	varchar(15)

3. Implementasi Aplikasi

3.1 Implementasi Antarmuka

3.1.1 Login

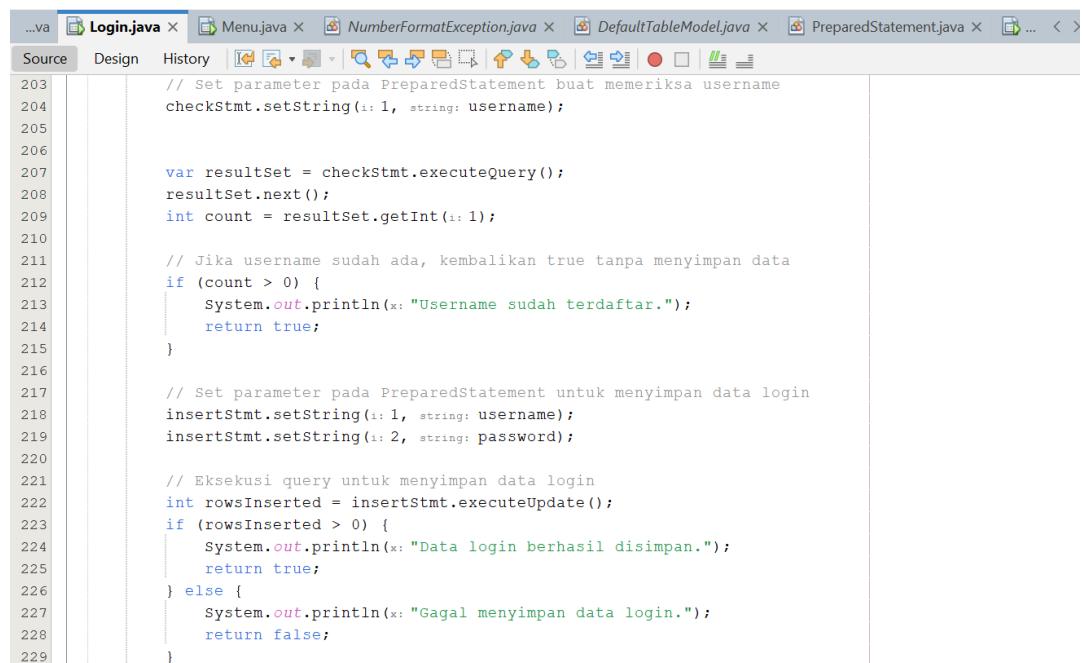


The screenshot shows a Java IDE interface with the Login.java file open. The code is as follows:

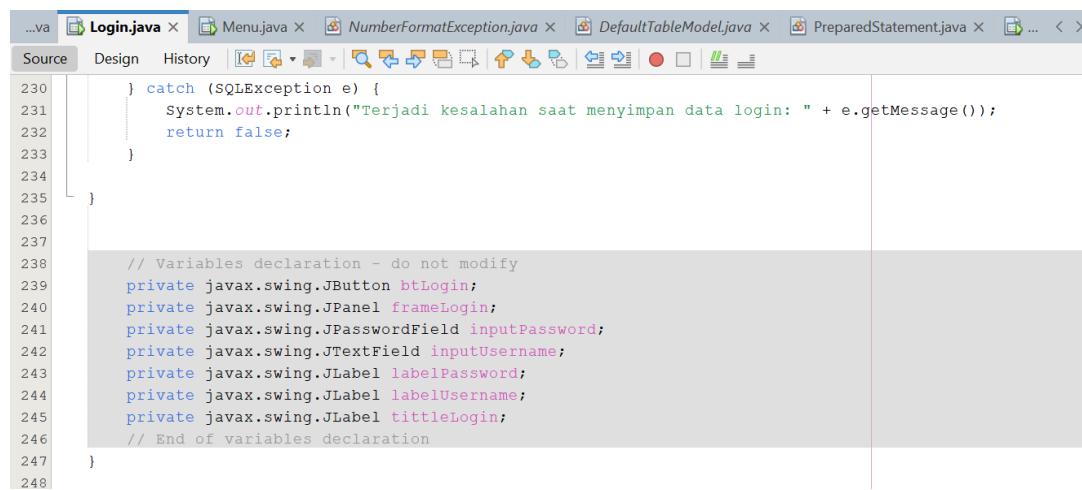
```

5 package tubes;
6
7 import javax.swing.JButton;
8 import javax.swing.JFrame;
9 import javax.swing.JPasswordField;
10 import java.sql.Connection;
11 import java.sql.DriverManager;
12 import java.sql.PreparedStatement;
13 import java.sql.SQLException;
14 import javax.swing.JOptionPane;
15
16 public class Login extends javax.swing.JFrame {
17
18     public Login() {
19         initComponents();
20     }

```

```
203     // Set parameter pada PreparedStatement buat memeriksa username
204     checkStmt.setString(1, string: username);
205
206
207     var resultSet = checkStmt.executeQuery();
208     resultSet.next();
209     int count = resultSet.getInt(1);
210
211
212     // Jika username sudah ada, kembalikan true tanpa menyimpan data
213     if (count > 0) {
214         System.out.println("Username sudah terdaftar.");
215         return true;
216     }
217
218     // Set parameter pada PreparedStatement untuk menyimpan data login
219     insertStmt.setString(1, string: username);
220     insertStmt.setString(1, string: password);
221
222     // Eksekusi query untuk menyimpan data login
223     int rowsInserted = insertStmt.executeUpdate();
224     if (rowsInserted > 0) {
225         System.out.println("Data login berhasil disimpan.");
226         return true;
227     } else {
228         System.out.println("Gagal menyimpan data login.");
229         return false;
229 }
```



```
230     } catch (SQLException e) {
231         System.out.println("Terjadi kesalahan saat menyimpan data login: " + e.getMessage());
232         return false;
233     }
234
235 }
236
237
238 // Variables declaration - do not modify
239 private javax.swing.JButton btLogin;
240 private javax.swing.JPanel frameLogin;
241 private javax.swing.JPasswordField inputPassword;
242 private javax.swing.JTextField inputUsername;
243 private javax.swing.JLabel labelPassword;
244 private javax.swing.JLabel labelUsername;
245 private javax.swing.JLabel tittleLogin;
246 // End of variables declaration
247
248 }
```

LOGIN

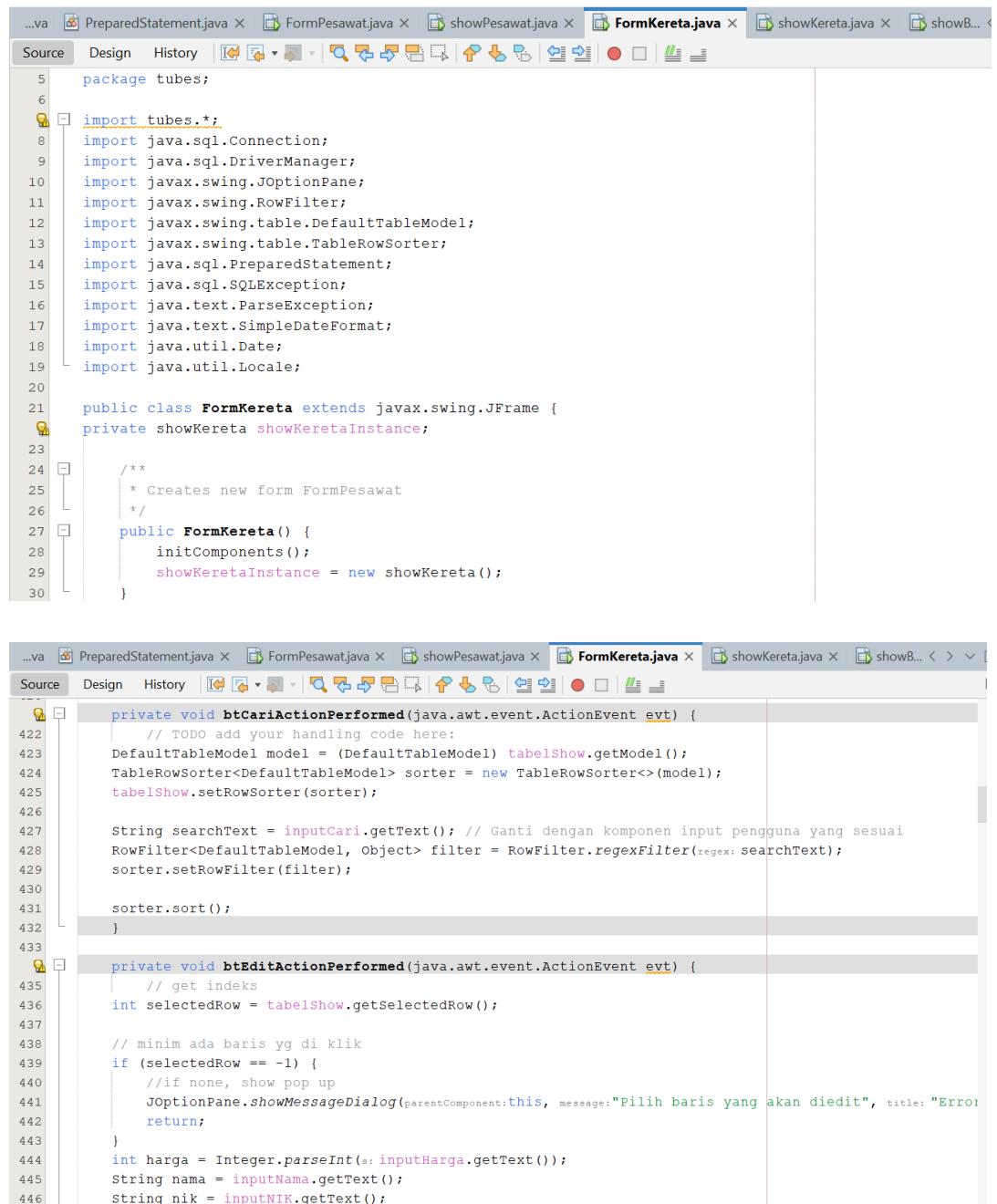
Username

Password

Deskripsi Program:

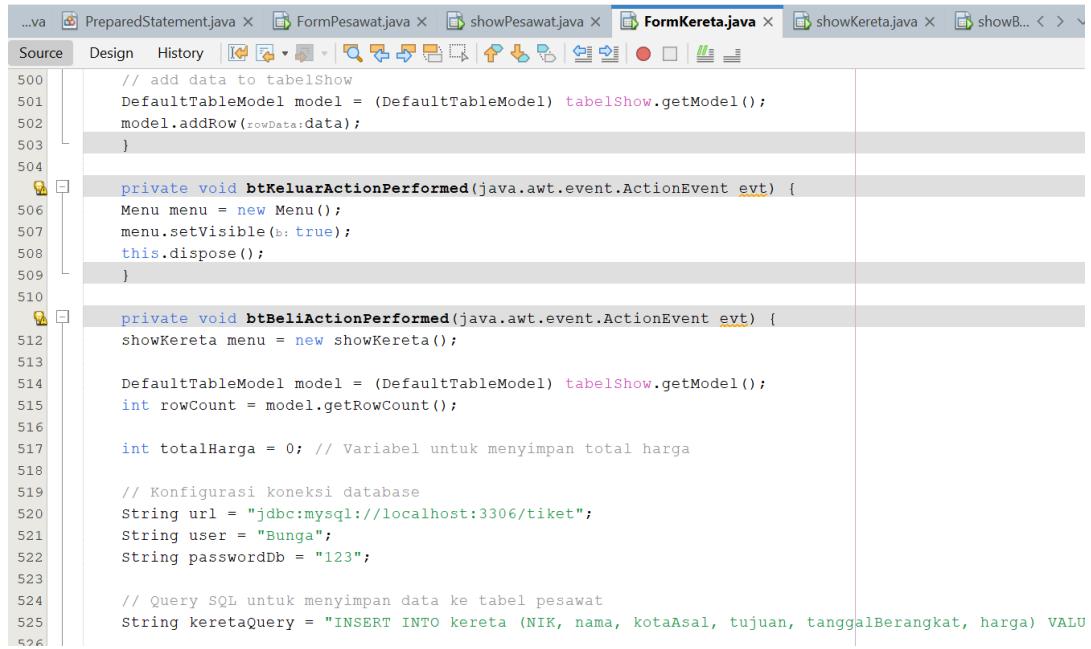
Bagian login akan menggunakan JDBC (Java Database Connectivity) untuk menjalin koneksi antara program Java dengan database MySQL. Koneksi ini memungkinkan program untuk mengambil dan membandingkan data pengguna yang tersimpan di database. Bagian login dalam program pembelian tiket dengan Java GUI dan MySQL akan memeriksa kecocokan antara data yang dimasukkan pengguna dengan data yang tersimpan di database. Sehingga ketika user sudah pernah login, maka pada mysql yang tertulis hanya akan satu data karena data sama.

3.1.2 Kereta

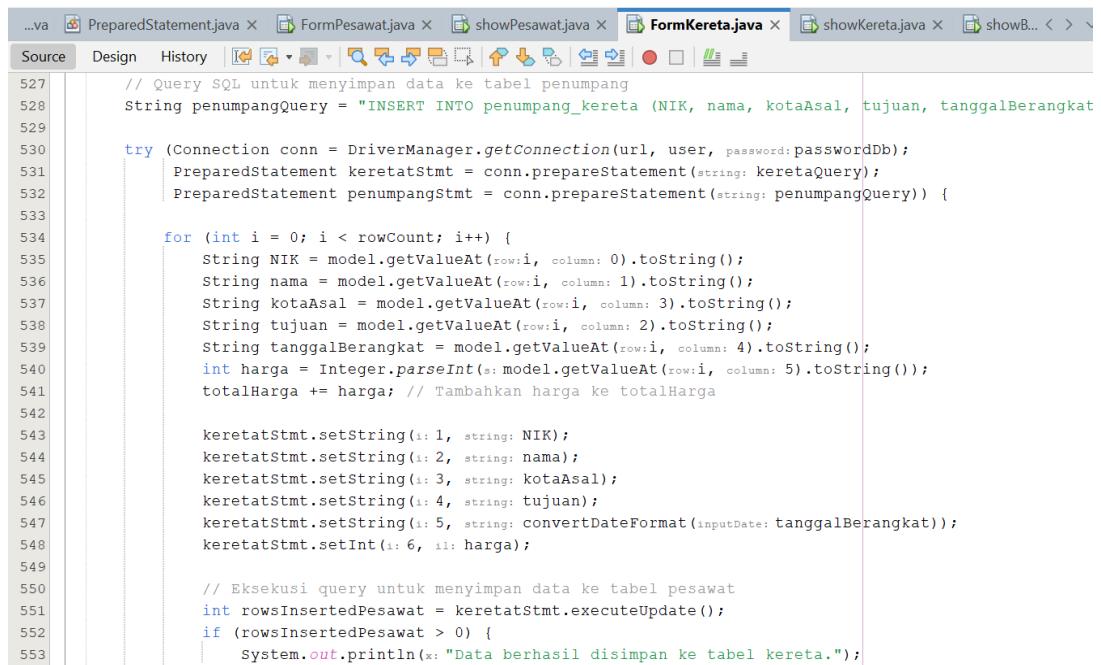


```
5 package tubes;
6
7 import tubes.*;
8 import java.sql.Connection;
9 import java.sql.DriverManager;
10 import javax.swing.JOptionPane;
11 import javax.swing.RowFilter;
12 import javax.swing.table.DefaultTableModel;
13 import javax.swing.table.TableRowSorter;
14 import java.sql.PreparedStatement;
15 import java.sql.SQLException;
16 import java.text.ParseException;
17 import java.text.SimpleDateFormat;
18 import java.util.Date;
19 import java.util.Locale;
20
21 public class FormKereta extends javax.swing.JFrame {
22     private showKereta showKeretaInstance;
23
24     /**
25      * Creates new form FormPesawat
26      */
27     public FormKereta() {
28         initComponents();
29         showKeretaInstance = new showKereta();
30     }
31
32     private void btCariActionPerformed(java.awt.event.ActionEvent evt) {
33         // TODO add your handling code here:
34         DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
35         TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);
36         tabelShow.setRowSorter(sorter);
37
38         String searchText = inputCari.getText(); // Ganti dengan komponen input pengguna yang sesuai
39         RowFilter<DefaultTableModel, Object> filter = RowFilter.regexFilter(regex: searchText);
40         sorter.setRowFilter(filter);
41
42         sorter.sort();
43     }
44
45     private void btEditActionPerformed(java.awt.event.ActionEvent evt) {
46         // get indeks
47         int selectedRow = tabelShow.getSelectedRow();
48
49         // minim ada baris yg di klik
50         if (selectedRow == -1) {
51             //if none, show pop up
52             JOptionPane.showMessageDialog(parentComponent:this, message:"Pilih baris yang akan diedit", title: "Error")
53             return;
54         }
55         int harga = Integer.parseInt(: inputHarga.getText());
56         String nama = inputNama.getText();
57         String nik = inputNIK.getText();
58     }
59 }
```

```
...va PreparedStatement.java x FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showB... < > Source Design History |  447     String kotaTujuan = inputTujuan.getText();  
448     String kotaAsal = inputAsal.getText();  
449     String tanggal = comboTanggal.getSelectedItem().toString();  
450     String bulan = comboBulan.getSelectedItem().toString();  
451     String tahun = comboTahun.getSelectedItem().toString();  
452  
453     //merge jdi string  
454     String tanggalBerangkat = tanggal + " " + bulan + " " + tahun;  
455  
456     // Menyimpan data  
457     Object[] data = {nama, nik, kotaTujuan, kotaAsal, tanggalBerangkat, harga};  
458  
459     DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();  
460     model.setValueAt(aValue: nik, row:SelectedRow, column: 0);  
461     model.setValueAt(aValue: nama, row:SelectedRow, column: 1);  
462     model.setValueAt(aValue: kotaTujuan, row:SelectedRow, column: 3);  
463     model.setValueAt(aValue: kotaAsal, row:SelectedRow, column: 2);  
464     model.setValueAt(aValue: tanggalBerangkat, row:SelectedRow, column: 4);  
465     model.setValueAt(aValue: harga, row:SelectedRow, column: 5);  
466  
467     // Menampilkan pesan bahwa perubahan berhasil disimpan  
468     JOptionPane.showMessageDialog(parentComponent:this, message:"Perubahan berhasil disimpan", title: "Informasi  
469     showKeretaInstance.updateData(showPesawatInstance:showKeretaInstance, model);  
470 }
```



```
500 // add data to tabelShow
501 DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
502 model.addRow(rowData);
503 }
504
505 private void btKeluarActionPerformed(java.awt.event.ActionEvent evt) {
506     Menu menu = new Menu();
507     menu.setVisible(true);
508     this.dispose();
509 }
510
511 private void btBeliActionPerformed(java.awt.event.ActionEvent evt) {
512     showKereta menu = new showKereta();
513
514     DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
515     int rowCount = model.getRowCount();
516
517     int totalHarga = 0; // Variabel untuk menyimpan total harga
518
519     // Konfigurasi koneksi database
520     String url = "jdbc:mysql://localhost:3306/tiket";
521     String user = "Bunga";
522     String passwordDb = "123";
523
524     // Query SQL untuk menyimpan data ke tabel pesawat
525     String keretaQuery = "INSERT INTO kereta (NIK, nama, kotaAsal, tujuan, tanggalBerangkat, harga) VALU
526
```



```
527 // Query SQL untuk menyimpan data ke tabel penumpang
528 String penumpangQuery = "INSERT INTO penumpang_kereta (NIK, nama, kotaAsal, tujuan, tanggalBerangkat
529
530 try (Connection conn = DriverManager.getConnection(url, user, password:passwordDb);
531      PreparedStatement keretatStmt = conn.prepareStatement(string: keretaQuery);
532      PreparedStatement penumpangStmt = conn.prepareStatement(string: penumpangQuery)) {
533
534     for (int i = 0; i < rowCount; i++) {
535         String NIK = model.getValueAt(rowi, column: 0).toString();
536         String nama = model.getValueAt(rowi, column: 1).toString();
537         String kotaAsal = model.getValueAt(rowi, column: 3).toString();
538         String tujuan = model.getValueAt(rowi, column: 2).toString();
539         String tanggalBerangkat = model.getValueAt(rowi, column: 4).toString();
540         int harga = Integer.parseInt(model.getValueAt(rowi, column: 5).toString());
541         totalHarga += harga; // Tambahkan harga ke totalHarga
542
543         keretatStmt.setString(1, string: NIK);
544         keretatStmt.setString(2, string: nama);
545         keretatStmt.setString(3, string: kotaAsal);
546         keretatStmt.setString(4, string: tujuan);
547         keretatStmt.setString(5, string: convertDateFormat(inputDate: tanggalBerangkat));
548         keretatStmt.setInt(6, il: harga);
549
550     // Eksekusi query untuk menyimpan data ke tabel pesawat
551     int rowsInsertedPesawat = keretatStmt.executeUpdate();
552     if (rowsInsertedPesawat > 0) {
553         System.out.println("Data berhasil disimpan ke tabel kereta.");
554     }
555 }
```

```
...va FormPreparedStatement.java X FormPesawat.java X showPesawat.java X FormKereta.java X showKereta.java X showB... < > ✓
Source Design History | 


```

554 } else {
555 System.out.println("Gagal menyimpan data ke tabel kereta.");
556 }
557
558 // Mengambil data alamat dan no hp dari input pengguna
559 String alamat = inputAlamat.getText();
560 String noHp = inputHp.getText();
561
562 penumpangStmt.setString(1, string: NIK);
563 penumpangStmt.setString(2, string: nama);
564 penumpangStmt.setString(3, string: kotaAsal);
565 penumpangStmt.setString(4, string: tujuan);
566 penumpangStmt.setString(5, string: convertDateFormat(inputDate: tanggalBerangkat));
567 penumpangStmt.setString(6, string: alamat);
568 penumpangStmt.setString(7, string: noHp);
569
570 // Eksekusi query untuk menyimpan data ke tabel penumpang
571 int rowsInsertedPenumpang = penumpangStmt.executeUpdate();
572 if (rowsInsertedPenumpang > 0) {
573 System.out.println("Data berhasil disimpan ke tabel penumpang.");
574 } else {
575 System.out.println("Gagal menyimpan data ke tabel penumpang.");
576 }
577 // Memanggil buat tampil ke tabel tiket - penumpang
578 menu.tambahDataTabelTiket(nama:NIK, NIK:nama, kotaAsal, tujuan, tanggalBerangkat, harga);
579 menu.tambahDataTabelPenumpang(nama, NIK, kotaAsal, tujuan, tanggalBerangkat, alamat, noHp);
580

```


```

```
...va PreparedStatement.java X FormPesawat.java X showPesawat.java X FormKereta.java X showKereta.java X showB... < > ✓
Source Design History | 

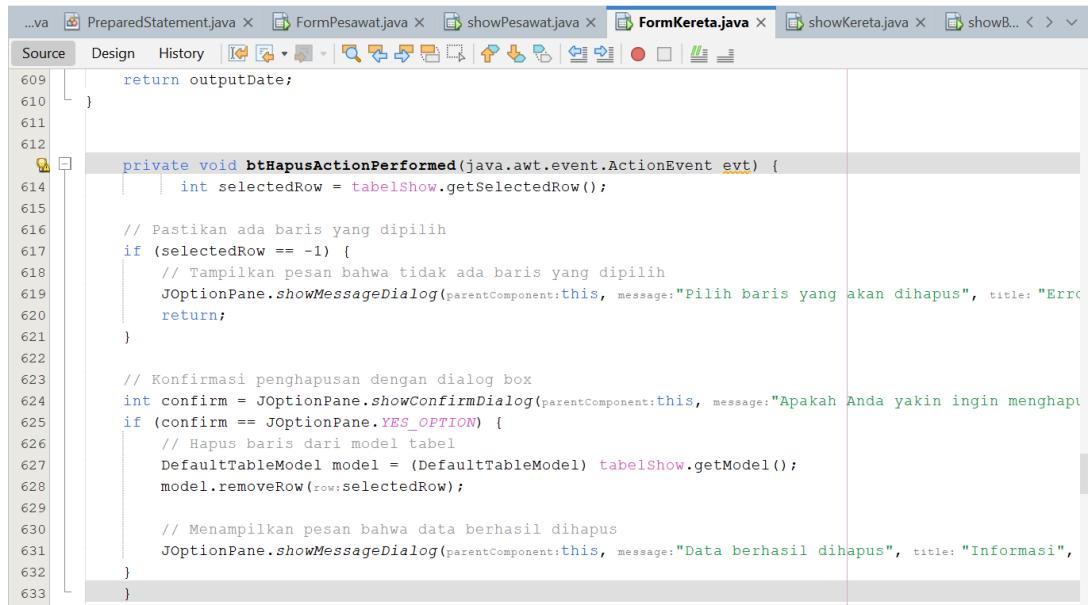

```

581 }
582 //syntax eror except
583 } catch (SQLException e) {
584 System.out.println("Terjadi kesalahan saat menyimpan data: " + e.getMessage());
585 }
586
587 // Set total harga ke showPesawat
588 menu.setTotalHarga(totalHarga);
589
590 menu.setVisible(b: true);
591 this.dispose();
592 }

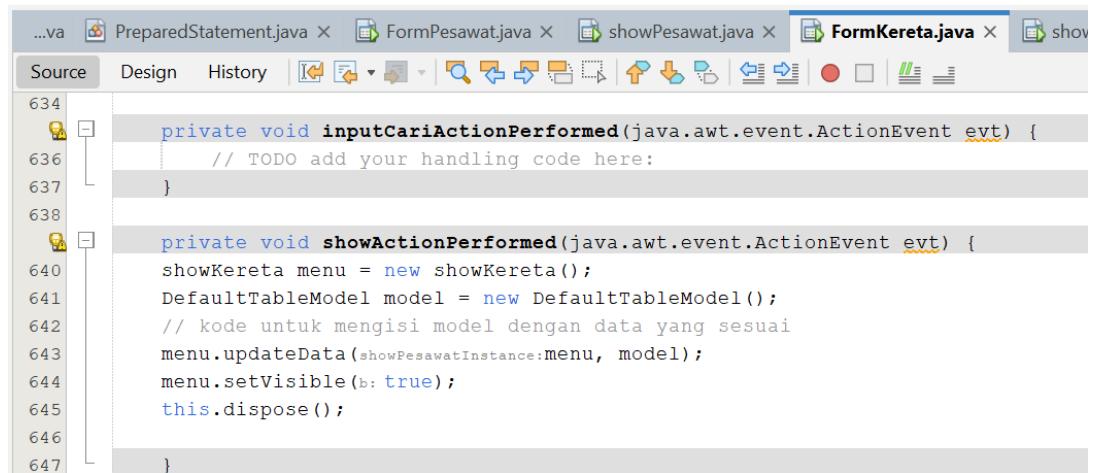
593
594 private String convertDateFormat(String inputDate) {
595 // Assuming the input date format is 'd MMMM yyyy'
596 SimpleDateFormat inputFormat = new SimpleDateFormat(pattern:"d MMMM yyyy", new Locale(language: "id"));
597 SimpleDateFormat outputFormat = new SimpleDateFormat(pattern:"yyyy-MM-dd");
598 String outputDate = null;
599
600 try {
601 Date date = inputFormat.parse(source: inputDate);
602 outputDate = outputFormat.format(date);
603 } catch (ParseException e) {
604 e.printStackTrace();
605 // Set outputDate to null if parsing fails
606 outputDate = null;
607 }

```


```



```
609     return outputDate;
610 }
611
612 private void btHapusActionPerformed(java.awt.event.ActionEvent evt) {
613     int selectedRow = tabelShow.getSelectedRow();
614
615     // Pastikan ada baris yang dipilih
616     if (selectedRow == -1) {
617         // Tampilkan pesan bahwa tidak ada baris yang dipilih
618         JOptionPane.showMessageDialog(parentComponent:this, message:"Pilih baris yang akan dihapus", title: "Error");
619         return;
620     }
621
622     // Konfirmasi penghapusan dengan dialog box
623     int confirm = JOptionPane.showConfirmDialog(parentComponent:this, message:"Apakah Anda yakin ingin menghapus?", title: "Konfirmasi");
624     if (confirm == JOptionPane.YES_OPTION) {
625         // Hapus baris dari model tabel
626         DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
627         model.removeRow(row:selectedRow);
628
629         // Menampilkan pesan bahwa data berhasil dihapus
630         JOptionPane.showMessageDialog(parentComponent:this, message:"Data berhasil dihapus", title: "Informasi");
631     }
632 }
633 }
```



```
634 private void inputCariActionPerformed(java.awt.event.ActionEvent evt) {
635     // TODO add your handling code here:
636 }
637
638 private void showActionPerformed(java.awt.event.ActionEvent evt) {
639     showKereta menu = new showKereta();
640     DefaultTableModel model = new DefaultTableModel();
641     // kode untuk mengisi model dengan data yang sesuai
642     menu.updateData(showPesawatInstance.menu, model);
643     menu.setVisible(b: true);
644     this.dispose();
645
646 }
647 }
```

Source Design History

```

686     java.awt.EventQueue.invokeLater(new Runnable() {
687         public void run() {
688             new FormKereta().setVisible(b: true);
689         }
690     }
691
692     // Variables declaration - do not modify
693     private javax.swing.JButton btAdd;
694     private javax.swing.JButton btBeli;
695     private javax.swing.JButton btCari;
696     private javax.swing.JButton btEdit;
697     private javax.swing.JButton btHapus;
698     private javax.swing.JButton btKeluar;
699     private javax.swing.JComboBox<String> comboBulan;
700     private javax.swing.JComboBox<String> comboJam;
701     private javax.swing.JComboBox<String> comboTahun;
702     private javax.swing.JComboBox<String> comboTanggal;
703     private javax.swing.JTextField inputAlamat;
704     private javax.swing.JTextField inputAsal;
705     private javax.swing.JTextField inputCari;
706     private javax.swing.JTextField inputHarga;
707     private javax.swing.JTextField inputHp;
708     private javax.swing.JTextField inputNIK;
709     private javax.swing.JTextField inputNama;
710     private javax.swing.JTextField inputTujuan;
711     private javax.swing.JPanel jPanell;

```

Design Preview [FormKereta]

Pemesanan Tiket Kereta

Kota Asal		Nama Penumpang	
Kota Tujuan		NIK	
Tanggal Berangkat	1	Januari	2023
Jam Berangkat	12.00 WIB	No HP	
Harga		Add Data	

Cari Cari Data

NIK	Nama	Kota Tujuan	Kota Asal	Tanggal Berangkat	Harga

Hapus Edit Bell Show Keluar

Deskripsi Program:

- **btEditActionPerformed**

Aksi ini digunakan untuk mengedit data pemesanan tiket kereta yang sudah ada. Pengguna akan diminta untuk memasukkan informasi tiket yang ingin diedit seperti nama penumpang, tanggal keberangkatan dan lain-lain. Setelah itu, data yang sudah diedit akan diperbarui.

- **btAddActionPerformed**

Aksi ini digunakan untuk menambahkan data pemesanan tiket kereta baru. Pengguna akan diminta untuk memasukkan informasi tiket seperti kota asal, kota tujuan, nama penumpang, tanggal keberangkatan dan lain-lain. Setelah itu, tiket akan terdaftar ketika user menekan tombol add data.

- **btKeluarActionPerformed**

Aksi ini digunakan untuk keluar dari program pemesanan tiket kereta. Program kemudian akan kembali menuju menu utama.

- **btBeliActionPerformed**

Aksi ini digunakan untuk menampilkan daftar pemesanan tiket kereta yang telah dilakukan. Program akan menampilkan informasi pemesanan tiket kereta seperti NIK, nama penumpang, tanggal keberangkatan, harga dan lain-lain.

- **btHapusActionPerformed**

Aksi ini digunakan untuk menghapus data pemesanan tiket kereta yang telah ditambah oleh pengguna. Pengguna akan memilih baris data yang ingin dihapus, kemudian menekan tombol hapus pada samping tabel.


```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    FormKereta menu = new FormKereta();
    menu.setVisible(b: true);
    this.dispose();           // TODO add your handling code here:
}

private void btCariActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    DefaultTableModel model = (DefaultTableModel) tablePenumpang.getModel();
    TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);
    tablePenumpang.setRowSorter(sorter);

    String searchText = inputCari.getText(); // Ganti dengan komponen input pengguna yang sesuai
    RowFilter<DefaultTableModel, Object> filter = RowFilter.regexFilter(regex: searchText);
    sorter.setRowFilter(filter);

    sorter.sort();
}

private void inputCariActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
```

```
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new showKereta().setVisible(b: true);
        }
    });
}
```

Design Preview [showKereta]

PEMESANAN TIKET KERETA

Nama	NIK	Kota Asal	Kota Tujuan	Tanggal Berangkat	Harga

Total Total

DATA PENUMPANG KERETA

Cari

Cari Data

Nama	NIK	Kota Asal	Kota Tujuan	Tanggal Berangkat	Alamat	No Hp

Kembali

Deskripsi Program:

- SetTotalHarga

Aksi ini digunakan untuk menghitung dan menampilkan total harga dari pemesanan tiket kereta yang telah diinputkan pengguna.

- **UpdateDataPenumpang**

Aksi ini digunakan untuk mengubah atau memperbarui informasi penumpang dalam pemesanan tiket kereta. Ketika pengguna menambah data pemesanan tiket, maka program akan mengupdate data penumpang sesuai dengan data yang diinputkan..

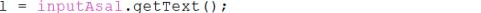
- **btCariActionPerformed**

Aksi ini digunakan untuk melakukan pencarian dalam daftar pemesanan tiket kereta. Pengguna akan diminta untuk memasukkan kriteria pencarian seperti NIK, nama penumpang atau tanggal keberangkatan. Kemudian program akan memfilter data hingga ditemukan data yang sesuai.

3.1.3 Bus

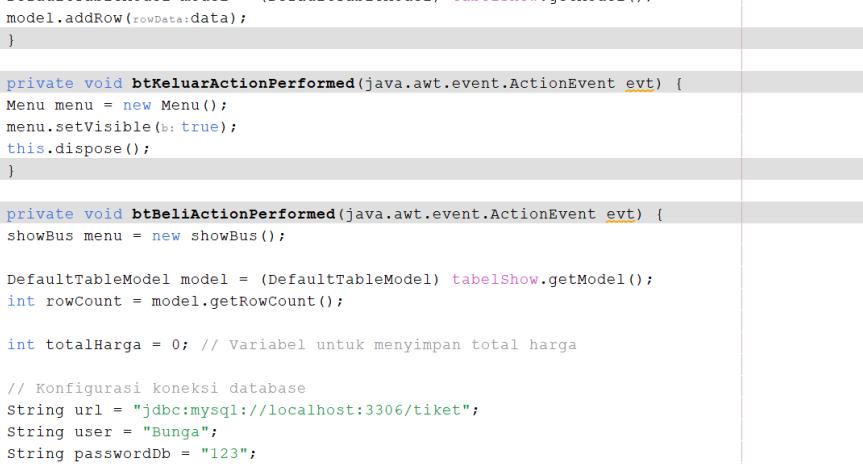
```
...va showPesawat.java x FormKereta.java x showKereta.java x showBusjava x FormBus.java x JPasswordField.java... < > Source Design History 
```

```
private void btCariActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();  
    TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);  
    tabelShow.setRowSorter(sorter);  
  
    String searchText = inputCari.getText(); // Ganti dengan komponen input pengguna yang sesuai  
    RowFilter<DefaultTableModel, Object> filter = RowFilter.regexFilter(regex: searchText);  
    sorter.setRowFilter(filter);  
  
    sorter.sort();  
}  
  
private void btEditActionPerformed(java.awt.event.ActionEvent evt) {  
    // get indeks  
    int selectedRow = tabelShow.getSelectedRow();  
  
    // minim ada baris yg di klik  
    if (selectedRow == -1) {  
        //if none, show pop up  
        JOptionPane.showMessageDialog(parentComponent:this, message:"Pilih baris yang akan diedit", title: "Error")  
        return;  
    }  
    int harga = Integer.parseInt(: inputHarga.getText());  
    String nama = inputNama.getText();  
    String nik = inputNIK.getText();  
    String kotaTujuan = inputTujuan.getText();  
}
```

```
...va showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java x JPasswordField.java... < > Source Design History 
```

451 String kotaTujuan = inputTujuan.getText();
452 String kotaAsal = inputAsal.getText();
453 String tanggal = comboTanggal.getSelectedItem().toString();
454 String bulan = comboBulan.getSelectedItem().toString();
455 String tahun = comboTahun.getSelectedItem().toString();
456
457 //merge jdi string
458 String tanggalBerangkat = tanggal + " " + bulan + " " + tahun;
459
460 // Menyimpan data
461 Object[] data = {nama, nik, kotaTujuan, kotaAsal, tanggalBerangkat, harga};
462
463 DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
464 model.setValueAt(aValue: nik, row:selectedRow, column: 0);
465 model.setValueAt(aValue: nama, row:selectedRow, column: 1);
466 model.setValueAt(aValue: kotaTujuan, row:selectedRow, column: 3);
467 model.setValueAt(aValue: kotaAsal, row:selectedRow, column: 2);
468 model.setValueAt(aValue: tanggalBerangkat, row:selectedRow, column: 4);
469 model.setValueAt(aValue: harga, row:selectedRow, column: 5);
470
471 // Menampilkan pesan bahwa perubahan berhasil disimpan
472 JOptionPane.showMessageDialog(parentComponent:this, message:"Perubahan berhasil disimpan", title: "Informasi"
473 showBusInstance.updateData(showPesawatInstance:showBusInstance, model);
474 }

```
...va showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java x JPasswordField.java... < >
Source Design History |             
475
476     private void btAddActionPerformed(java.awt.event.ActionEvent evt) {
477         String hargaText = inputHarga.getText();
478         int harga;
479         try {
480             if (hargaText.isEmpty()) {
481                 throw new Exception(message:"Harga harus diisi");
482             }
483             harga = Integer.parseInt(hargaText);
484         } catch (NumberFormatException e) {
485             JOptionPane.showMessageDialog(parentComponent:this, message:"Harga tidak valid", title: "Error", messageType:JOptionPane.ERROR_MESSAGE);
486             return;
487         } catch (Exception e) {
488             JOptionPane.showMessageDialog(parentComponent:this, message:e.getMessage(), title: "Error", messageType:JOptionPane.ERROR_MESSAGE);
489             return;
490         }
491
492         String nama = inputNama.getText();
493         String nik = inputNIK.getText();
494         String kotaTujuan = inputTujuan.getText();
495         String kotaAsal = inputAsal.getText();
496         String tanggal = comboTanggal.getSelectedItem().toString();
497         String bulan = comboBulan.getSelectedItem().toString();
498         String tahun = comboTahun.getSelectedItem().toString();
499         String tanggalBerangkat = tanggal + " " + bulan + " " + tahun;
500         // Menyimpan data
501         Object[] data = {nik,nama,kotaTujuan, kotaAsal, tanggalBerangkat, harga};
```

```
...va showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java x JPasswordField.java... < > 

Source Design History



503 // add data to tabelShow  
504 DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();  
505 model.addRow(rowData:data);  
506 }  
507  
508 private void btKeluarActionPerformed(java.awt.event.ActionEvent evt) {  
509 Menu menu = new Menu();  
510 menu.setVisible(b: true);  
511 this.dispose();  
512 }  
513  
514 private void btBeliActionPerfomed(java.awt.event.ActionEvent evt) {  
515 showBus menu = new showBus();  
516  
517 DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();  
518 int rowCount = model.getRowCount();  
519  
520 int totalHarga = 0; // Variabel untuk menyimpan total harga  
521  
522 // Konfigurasi koneksi database  
523 String url = "jdbc:mysql://localhost:3306/tiket";  
524 String user = "Bunga";  
525 String passwordDb = "123";  
526  
527 // Query SQL untuk menyimpan data ke tabel pesawat  
528 String keretaQuery = "INSERT INTO bus (NIK, nama, kotaAsal, tujuan, tanggalBerangkat, harga) VALUES  
529


```

```
...va showPesawat.java X FormKereta.java X showKereta.java X showBus.java X FormBus.java X JPasswordField.java... < >
Source Design History |  |             
557 } else {
558     System.out.println("Gagal menyimpan data ke tabel Bus.");
559 }
560
561 // Mengambil data alamat dan no hp dari input pengguna
562 String alamat = inputAlamat.getText();
563 String noHp = inputHp.getText();
564
565 penumpangStmt.setString(1, string: NIK);
566 penumpangStmt.setString(2, string: nama);
567 penumpangStmt.setString(3, string: kotaAsal);
568 penumpangStmt.setString(4, string: tujuan);
569 penumpangStmt.setString(5, string: convertDateFormat(inputDate: tanggalBerangkat));
570 penumpangStmt.setString(6, string: alamat);
571 penumpangStmt.setString(7, string: noHp);
572
573 // Eksekusi query untuk menyimpan data ke tabel penumpang
574 int rowsInsertedPenumpang = penumpangStmt.executeUpdate();
575 if (rowsInsertedPenumpang > 0) {
576     System.out.println("Data berhasil disimpan ke tabel penumpang.");
577 } else {
578     System.out.println("Gagal menyimpan data ke tabel penumpang.");
579 }
580
581 menu.tambahDataTabelTiket(nama:NIK, NIK:nama, kotaAsal, tujuan, tanggalBerangkat, harga);
582 menu.tambahDataTabelPenumpang(nama, NIK, kotaAsal, tujuan, tanggalBerangkat, alamat, noHp);
```

```
...va showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java x JPasswordField.java < > v
Source Design History | 
611     return outputDate;
612 }
613
614  private void btHapusActionPerformed(java.awt.event.ActionEvent evt) {
615     int selectedRow = tabelShow.getSelectedRow();
616
617     // Pastikan ada baris yang dipilih
618     if (selectedRow == -1) {
619         // Tampilkan pesan bahwa tidak ada baris yang dipilih
620         JOptionPane.showMessageDialog(parentComponent:this, message:"Pilih baris yang akan dihapus", title: "Error")
621         return;
622     }
623
624
625     // Konfirmasi penghapusan dengan dialog box
626     int confirm = JOptionPane.showConfirmDialog(parentComponent:this, message:"Apakah Anda yakin ingin menghapus?", title: "Konfirmasi")
627     if (confirm == JOptionPane.YES_OPTION) {
628         // Hapus baris dari model tabel
629         DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
630         model.removeRow(row:selectedRow);
631
632         // Tampilkan pesan bahwa data berhasil dihapus
633         JOptionPane.showMessageDialog(parentComponent:this, message:"Data berhasil dihapus", title: "Informasi")
634     }
635 }
```

```
...va showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java x JPasswordField.j...
Source Design History | 


```

636 private void inputCariActionPerformed(java.awt.event.ActionEvent evt) {
637 // TODO add your handling code here:
638 }
639
640 private void showActionPerformed(java.awt.event.ActionEvent evt) {
641 showBus menu = new showBus();
642 DefaultTableModel model = new DefaultTableModel();
643 // kode untuk mengisi model dengan data yang sesuai
644 menu.updateData(showPesawat.getInstance().menu, model);
645 menu.setVisible(b: true);
646 this.dispose();
647 }
648
649 }
650
651 /**
652 * @param args the command line arguments
653 */
654 public static void main(String args[]) {
655 java.awt.EventQueue.invokeLater(new Runnable() {
 [...5 lines...]
656 });
657 }

```


```

Design Preview [FormBus]

Pemesanan Tiket Bus

Kota Asal	<input type="text"/>	Nama Penumpang	<input type="text"/>												
Kota Tujuan	<input type="text"/>	NIK	<input type="text"/>												
Tanggal Berangkat	<input type="text" value="1"/> <input type="button" value="▼"/> <input type="text" value="Januari"/> <input type="button" value="▼"/> <input type="text" value="2023"/> <input type="button" value="▼"/>	Alamat	<input type="text"/>												
Jam Berangkat	<input type="text" value="12.00 WIB"/> <input type="button" value="▼"/>	No HP	<input type="text"/>												
Harga	<input type="text"/>	<input type="button" value="Add Data"/>													
<input style="background-color: #0070C0; color: white; border: none; padding: 5px 10px; border-radius: 5px; font-weight: bold; margin-right: 10px;" type="button" value="Cari"/> <input type="text" value="Cari Data"/>															
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>NIK</th> <th>Nama</th> <th>Kota Tujuan</th> <th>Kota Asal</th> <th>Tanggal Berangkat</th> <th>Harga</th> </tr> </thead> <tbody> <tr> <td colspan="6" style="height: 50px;"></td> </tr> </tbody> </table>				NIK	Nama	Kota Tujuan	Kota Asal	Tanggal Berangkat	Harga						
NIK	Nama	Kota Tujuan	Kota Asal	Tanggal Berangkat	Harga										
				<input style="background-color: #0070C0; color: white; border: none; padding: 5px 10px; border-radius: 5px; margin-bottom: 5px;" type="button" value="Hapus"/> <input style="background-color: #0070C0; color: white; border: none; padding: 5px 10px; border-radius: 5px; margin-bottom: 5px;" type="button" value="Edit"/> <input style="background-color: #0070C0; color: white; border: none; padding: 5px 10px; border-radius: 5px;" type="button" value="Beli"/>											
				<input style="background-color: #0070C0; color: white; border: none; padding: 5px 10px; border-radius: 5px; margin-bottom: 5px;" type="button" value="Show"/> <input style="background-color: #0070C0; color: white; border: none; padding: 5px 10px; border-radius: 5px;" type="button" value="Keluar"/>											

Deskripsi Program:

- **btEditActionPerformed**

Aksi ini digunakan untuk mengedit data pemesanan tiket bus yang sudah ada. Pengguna akan diminta untuk memasukkan informasi tiket yang ingin diedit, seperti nama penumpang, tanggal keberangkatan dan lain-lain. Setelah itu, data yang sudah diedit akan diperbarui.

- **btAddActionPerformed**

Aksi ini digunakan untuk menambahkan data pemesanan tiket bus baru. Pengguna akan diminta untuk memasukkan informasi tiket seperti kota asal, kota tujuan, nama penumpang, tanggal keberangkatan dan lain-lain. Setelah itu, tiket akan terdaftar ketika user menekan tombol add data.

- **btKeluarActionPerformed**

Aksi ini digunakan untuk keluar dari program pemesanan tiket bus. Program kemudian akan kembali menuju menu utama.

- **btBeliActionPerformed**

Aksi ini digunakan untuk menampilkan daftar pemesanan tiket bus yang telah dilakukan. Program akan menampilkan informasi pemesanan tiket bus seperti NIK, nama penumpang, tanggal keberangkatan, harga, dan lain-lain.

- **btHapusActionPerformed**

Aksi ini digunakan untuk menghapus data pemesanan tiket bus yang telah ditambah oleh pengguna. Pengguna akan memilih baris data yang ingin dihapus, kemudian menekan tombol hapus pada samping table.

...va showPesawat.java × FormKereta.java × showKereta.java × showBus.java × FormBus.java × JPasswordField.j...

Source Design History

```

24 public class showBus extends javax.swing.JFrame {
25     private NonEditableTableModel tabelshowbus;
26     public void tambahDataTabelTiket(String nama, String NIK, String kotaAsal, String tujuan, String tanggalBerangkat, String noHP) {
27         DefaultTableModel model = (DefaultTableModel) tabelshowbus.getModel();
28         model.insertRow(row:0, new Object[]{nama, NIK, kotaAsal, tujuan, tanggalBerangkat, harga});
29
30         int rowCount = model.getRowCount();
31         int totalHarga = 0;
32
33         for (int i = 0; i < rowCount; i++) {
34             Object value = model.getValueAt(row:i, column: 5);
35             if (value != null) {
36                 totalHarga += Integer.parseInt(value.toString());
37             }
38         }
39
40         model.setValueAt(totalHarga, row:0, column: 5); // Mengisi kolom harga dengan totalHarga
41     }
42
43     public void setTotalHarga(int totalHarga) {
44         labelTotalHarga.setText(String.valueOf(totalHarga));
45     }
46
47     public void updateData(showBus showPesawatInstance, DefaultTableModel model) {
48         showPesawatInstance.tabelshowbus.setModel(model);
49     }
50 }
```

...va showPesawat.java × FormKereta.java × showKereta.java × showBus.java × FormBus.java × JPasswordField.j...

Source Design History

```

51     public void tambahDataTabelPenumpang(String nama, String NIK, String kotaAsal, String tujuan, String tanggalBerangkat, String noHP) {
52         DefaultTableModel model = (DefaultTableModel) tablePenumpang.getModel();
53         model.insertRow(row:0,new Object[]{nama, NIK, kotaAsal, tujuan, tanggalBerangkat, alamat, noHP});
54
55         int rowCount = model.getRowCount();
56
57     }
58     public void updateDataPenumpang() {
59         DefaultTableModel model = (DefaultTableModel) tablePenumpang.getModel();
60         model.setRowCount(rowCount:0); // Menghapus semua baris dalam tabel
61
62         // Konfigurasi koneksi database
63         String url = "jdbc:mysql://localhost:3306/tiket";
64         String user = "Bunga";
65         String passwordDb = "123";
66
67         // Query SQL untuk mendapatkan data dari tabel penumpang
68         String penumpangQuery = "SELECT nama, NIK, kotaAsal, tujuan, tanggalBerangkat, Alamat, noHP FROM tiket";
69
70         try (Connection conn = DriverManager.getConnection(url, user, password:passwordDb);
71              PreparedStatement stmt = conn.prepareStatement(string:penumpangQuery);
72              var rs = stmt.executeQuery()) {
73
74             while (rs.next()) {
75                 String nama = rs.getString(string: "nama");
76                 String NIK = rs.getString(string: "NIK");
77                 String kotaAsal = rs.getString(string: "kotaAsal");
```

```
...va showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java x JPasswordField.j... < > Source Design History |  105 }  
106     public void showBus() {  
107         initComponents();  
108         // Inisialisasi model tabel penumpang  
109         DefaultTableModel modelPenumpang = new DefaultTableModel(  
110             new Object[][] {},  
111             new String[] {"Nama", "NIK", "Kota Asal", "Kota Tujuan", "Tanggal Berangkat", "Alamat", "No  
112         );  
113  
114         // Atur model tabel penumpang  
115         tablePenumpang.setModel(dataModel: modelPenumpang);  
116  
117         // Inisialisasi model tabel showBus  
118         tabelshowbus = new NonEditableTableModel(  
119             new Object[][] {},  
120             new String[] {"Nama", "NIK", "Kota Asal", "Tujuan", "Tanggal Berangkat", "Harga"}  
121         );  
122  
123         // Atur model tabel showkereta  
124         tablesowbus.setModel(dataModel: tabelshowbus);  
125  
126         updateDataPenumpang();  
127     }
```

Design Preview [showBus]

PEMESANAN TIKET BUS

Nama	NIK	Kota Asal	Kota Tujuan	Tanggal Berangkat	Harga

Total Total

DATA PENUMPANG BUS

Cari

Nama	NIK	Kota Asal	Kota Tujuan	Tanggal Berangkat	Alamat	No Hp

Kembali

Deskripsi Program

- **SetTotalHarga**

Aksi ini digunakan untuk menghitung dan menampilkan total harga dari pemesanan tiket pesawat yang telah diinputkan pengguna.

- **UpdateDataPenumpang**

Aksi ini digunakan untuk mengubah atau memperbarui informasi penumpang dalam pemesanan tiket pesawat. Ketika pengguna menambah data pemesanan tiket, maka program akan mengupdate data penumpang sesuai dengan data yang diinputkan..

- **btCariActionPerformed**

Aksi ini digunakan untuk melakukan pencarian dalam daftar pemesanan tiket pesawat. Pengguna akan diminta untuk memasukkan kriteria pencarian seperti NIK, nama penumpang, atau tanggal keberangkatan. Kemudian program akan memfilter data hingga ditemukan data yang sesuai.

3.1.4 Pesawat

```
...va FormPesawat.java X showPesawat.java X FormKereta.java X showKereta.java X showBus.java X FormBus.java... < >
Source Design History
private void btEditActionPerformed(java.awt.event.ActionEvent evt) {
    // get indeks
    int selectedRow = tabelShow.getSelectedRow();

    // minim ada baris yg di klik
    if (selectedRow == -1) {
        //if none, show pop up
        JOptionPane.showMessageDialog(parentComponent:this, message:"Pilih baris yang akan diedit", title: "Error");
        return;
    }
    int harga = Integer.parseInt(: inputHarga.getText());
    String nama = inputNama.getText();
    String nik = inputNIK.getText();
    String kotaTujuan = inputTujuan.getText();
    String kotaAsal = inputAsal.getText();
    String tanggal = comboTanggal.getSelectedItem().toString();
    String bulan = comboBulan.getSelectedItem().toString();
    String tahun = comboTahun.getSelectedItem().toString();

    //merge jdi string
    String tanggalBerangkat = tanggal + " " + bulan + " " + tahun;

    // Menyimpan data
    Object[] data = {nama, nik, kotaTujuan, kotaAsal, tanggalBerangkat, harga};

    DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
    model.setValueAt(sValue: nik, row:selectedRow, column: 0);
```

...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java.. < > ▾

Source Design History

```

466     model.setValueAt(avalue: kotaTujuan, row:selectedRow, column: 3);
467     model.setValueAt(avalue: kotaAsal, row:selectedRow, column: 2);
468     model.setValueAt(avalue: tanggalBerangkat, row:selectedRow, column: 4);
469     model.setValueAt(avalue: harga, row:selectedRow, column: 5);

470     // Menampilkan pesan bahwa perubahan berhasil disimpan
471     JOptionPane.showMessageDialog(parentComponent:this, message:"Perubahan berhasil disimpan", title: "Informasi");
472     showPesawatInstance.updateData(showPesawatInstance, model);
473 }
474 }

475 private void btAddActionPerformed(java.awt.event.ActionEvent evt) {
476     String hargaText = inputHarga.getText();
477     int harga;
478     try {
479         if (hargaText.isEmpty()) {
480             throw new Exception(message:"Harga harus diisi");
481         }
482         harga = Integer.parseInt(hargaText);
483     } catch (NumberFormatException e) {
484         JOptionPane.showMessageDialog(parentComponent:this, message:"Harga tidak valid", title: "Error", messageType:JOP);
485         return;
486     } catch (Exception e) {
487         JOptionPane.showMessageDialog(parentComponent:this, message:e.getMessage(), title: "Error", messageType:JOP);
488         return;
489     }
490 }
491 }

492 String nama = inputNama.getText();

```

...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java.. < > ▾

Source Design History

```

493     String nik = inputNIK.getText();
494     String kotaTujuan = inputTujuan.getText();
495     String kotaAsal = inputAsal.getText();
496     String tanggal = comboTanggal.getSelectedItem().toString();
497     String bulan = comboBulan.getSelectedItem().toString();
498     String tahun = comboTahun.getSelectedItem().toString();
499     String tanggalBerangkat = tanggal + " " + bulan + " " + tahun;
500     // Menyimpan data
501     Object[] data = {nik,nama,kotaTujuan, kotaAsal, tanggalBerangkat, harga};

502     // add data to tabelShow
503     DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
504     model.addRow(rowData:data);
505 }
506 }

507 private void btKeluarActionPerformed(java.awt.event.ActionEvent evt) {
508     Menu menu = new Menu();
509     menu.setVisible(b: true);
510     this.dispose();
511 }
512 }

513 private void btBeliActionPerformed(java.awt.event.ActionEvent evt) {
514     showPesawat menu = new showPesawat();

515     DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
516     int rowCount = model.getRowCount();
517 }
518 }

```

```
...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus.java... < > Source Design History 
```

519 int totalHarga = 0; // Variabel untuk menyimpan total harga
520
521 // Konfigurasi koneksi database
522 String url = "jdbc:mysql://localhost:3306/tiket";
523 String user = "Bunga";
524 String passwordDb = "123";
525
526 // Query SQL untuk menyimpan data ke tabel pesawat
527 String pesawatQuery = "INSERT INTO pesawat (NIK, nama, kotaAsal, tujuan, tanggalBerangkat, harga) VAI
528
529 // Query SQL untuk menyimpan data ke tabel penumpang
530 String penumpangQuery = "INSERT INTO penumpang (NIK, nama, kotaAsal, tujuan, tanggalBerangkat, Alamat
531
532 try (Connection conn = DriverManager.getConnection(url, user, password:passwordDb);
533 PreparedStatement pesawatStmt = conn.prepareStatement(string: pesawatQuery);
534 PreparedStatement penumpangStmt = conn.prepareStatement(string: penumpangQuery)) {
535
536 for (int i = 0; i < rowCount; i++) {
537 String NIK = model.getValueAt(row:i, column: 0).toString();
538 String nama = model.getValueAt(row:i, column: 1).toString();
539 String kotaAsal = model.getValueAt(row:i, column: 3).toString();
540 String tujuan = model.getValueAt(row:i, column: 2).toString();
541 String tanggalBerangkat = model.getValueAt(row:i, column: 4).toString();
542 int harga = Integer.parseInt(s: model.getValueAt(row:i, column: 5).toString());
543 totalHarga += haga; // Tambahkan harga ke totalHarga
544
545 }


```
...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus...
Source Design History 

```

622 }
623 }
624 // Konfirmasi penghapusan dengan dialog box
625 int confirm = JOptionPane.showConfirmDialog(parentComponent:this, message:"Apakah Anda yakin ingin menghapus");
626 if(confirm == JOptionPane.YES_OPTION) {
627 // Hapus baris dari model tabel
628 DefaultTableModel model = (DefaultTableModel) tabelShow.getModel();
629 model.removeRow(row:selectedRow);

630 // Menampilkan pesan bahwa data berhasil dihapus
631 JOptionPane.showMessageDialog(parentComponent:this, message:"Data berhasil dihapus", title: "Informasi",
632)
633 }
634 }

635  
private void inputCariActionPerformed(java.awt.event.ActionEvent evt) {
 // TODO add your handling code here:
}

636  
private void showActionPerformed(java.awt.event.ActionEvent evt) {
637 showPesawat menu = new showPesawat();
638 DefaultTableModel model = new DefaultTableModel();
639 // kode untuk mengisi model dengan data yang sesuai

640 menu.updateData(showPesawatInstance:menu, model);
641 menu.setVisible(b: true);
642 this.dispose();
643 }

```


```

The screenshot shows a Windows application window titled "Design Preview [FormPesawat]". The main title bar is "Pemesanan Tiket Pesawat". The form contains several input fields: "Kota Asal" and "Nama Penumpang" (text boxes), "Kota Tujuan" and "NIK" (text boxes), "Tanggal Berangkat" (date picker set to Januari 2023) and "Alamat" (text box), "Jam Berangkat" (dropdown menu set to 12.00 WIB) and "No HP" (text box), and "Harga" (text box). Below these are "Add Data" and "Cari" buttons, and a "Cari Data" search bar. A data grid table is displayed with columns: NIK, Name, Kota Tujuan, Kota Asal, Tanggal Berangkat, and Harga. To the right of the grid are buttons for "Hapus", "Edit", and "Beli". At the bottom right are "Show" and "Keluar" buttons.

Deskripsi Program:

- **btEditActionPerformed**

Aksi ini digunakan untuk mengedit data pemesanan tiket pesawat yang sudah ada. Pengguna akan diminta untuk memasukkan informasi tiket yang ingin diedit, seperti nama penumpang, tanggal keberangkatan, dan lain-lain. Setelah itu, data yang sudah diedit akan diperbarui.

- **btAddActionPerformed**

Aksi ini digunakan untuk menambahkan data pemesanan tiket pesawat baru. Pengguna akan diminta untuk memasukkan informasi tiket, seperti kota asal, kota tujuan, nama penumpang, tanggal keberangkatan, dan lain-lain. Setelah itu, tiket akan terdaftar ketika user menekan tombol add data.

- **btKeluarActionPerformed**

Aksi ini digunakan untuk keluar dari program pemesanan tiket pesawat. Program kemudian akan kembali menuju menu utama.

- **btBeliActionPerformed**

Aksi ini digunakan untuk menampilkan daftar pemesanan tiket pesawat yang telah dilakukan. Program akan menampilkan informasi pemesanan tiket pesawat seperti NIK, nama penumpang, tanggal keberangkatan, harga, dan lain-lain.

- **btHapusActionPerformed**

Aksi ini digunakan untuk menghapus data pemesanan tiket pesawat yang telah ditambah oleh pengguna. Pengguna akan memilih baris data yang ingin dihapus, kemudian menekan tombol hapus pada samping table.

```

...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus...
Source Design History | < > v
23 | ...
24 public class showPesawat extends javax.swing.JFrame {
25     private NonEditableTableModel tabelshowpesawat;
26     public void tambahDataTabelTiket(String nama, String NIK, String kotaAsal, String tujuan, String tanggal
27         DefaultTableModel model = (DefaultTableModel) tabelshowpesawat.getModel();
28         model.insertRow(row:0, new Object[]{nama, NIK, kotaAsal, tujuan, tanggalBerangkat, harga});
29
30         int rowCount = model.getRowCount();
31         int totalHarga = 0;
32
33         for (int i = 0; i < rowCount; i++) {
34             Object value = model.getValueAt(row:i, column: 5);
35             if (value != null) {
36                 totalHarga += Integer.parseInt(s: value.toString());
37             }
38         }
39
40         model.setValueAt(avalue: totalHarga, row:0, column: 5); // Mengisi kolom harga dengan totalHarga
41     }
42
43     public void setTotalHarga(int totalHarga) {
44         labelTotalHarga.setText(text: String.valueOf(i: totalHarga));
45     }
46     public void updateData(showPesawat showPesawatInstance, DefaultTableModel model) {
47         showPesawatInstance.tabelshowpesawat.setModel(dataModel: model);
48     }
49 }

```

```

...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus...
Source Design History | < > v
50 public void tambahDataTabelPenumpang(String nama, String NIK, String kotaAsal, String tujuan, String tang
51     DefaultTableModel model = (DefaultTableModel) tablePenumpang.getModel();
52     model.insertRow(row:0,new Object[]{nama, NIK, kotaAsal, tujuan, tanggalBerangkat, alamat, noHp});
53
54     int rowCount = model.getRowCount();
55
56 }
57 public void updateDataPenumpang() {
58     DefaultTableModel model = (DefaultTableModel) tablePenumpang.getModel();
59     model.setRowCount(rowCount:0); // Menghapus semua baris dalam tabel
60
61     // Konfigurasi koneksi database
62     String url = "jdbc:mysql://localhost:3306/tiket";
63     String user = "Bunga";
64     String passwordDb = "123";
65
66     // Query SQL untuk mendapatkan data dari tabel penumpang
67     String penumpangQuery = "SELECT nama, NIK, kotaAsal, tujuan, tanggalBerangkat, Alamat, noHp FROM
68
69     try (Connection conn = DriverManager.getConnection(url, user, password:passwordDb);
70         PreparedStatement stmt = conn.prepareStatement(string: penumpangQuery);
71         var rs = stmt.executeQuery()) {
72
73         while (rs.next()) {
74             String nama = rs.getString(string: "nama");
75             String NIK = rs.getString(string: "NIK");
76             String kotaAsal = rs.getString(string: "kotaAsal");

```

```
...va FormPesawat.java x showPesawat.java x FormKereta.java x showKereta.java x showBus.java x FormBus...
Source Design History < > < >
104 }
105     public void showPesawat() {
106         initComponents();
107         // Inisialisasi model tabel penumpang
108         DefaultTableModel modelPenumpang = new DefaultTableModel(
109             new Object[][] {},
110             new String[] {"Nama", "NIK", "Kota Asal", "Kota Tujuan", "Tanggal Berangkat", "Alamat", "No P");
111     );
112
113         // Atur model tabel penumpang
114         tablePenumpang.setModel(dataModel: modelPenumpang);
115
116         // Inisialisasi model tabel showpesawat
117         tabelshowpesawat = new NonEditableTableModel(
118             new Object[][] {},
119             new String[] {"Nama", "NIK", "Kota Asal", "Tujuan", "Tanggal Berangkat", "Harga"});
120     );
121
122         // Atur model tabel showpesawat
123         tableshowpesawat.setModel(dataModel: tabelshowpesawat);
124
125         updateDataPenumpang();
126     }
```

FormPesawat.java

```

232     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
233         FormPesawat menu = new FormPesawat();
234         menu.setVisible(b: true);
235         this.dispose();           // TODO add your handling code here:
236     }
237
238     private void btCariActionPerformed(java.awt.event.ActionEvent evt) {
239         // TODO add your handling code here:
240         DefaultTableModel model = (DefaultTableModel) tablePenumpang.getModel();
241         TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);
242         tablePenumpang.setRowSorter(sorter);
243         String searchText = inputCari.getText(); // Ganti dengan komponen input pengguna yang sesuai
244         RowFilter<DefaultTableModel, Object> filter = RowFilter.regexFilter(regex: searchText);
245         sorter.setRowFilter(filter);
246         sorter.sort();
247     }
248
249     private void inputCariActionPerformed(java.awt.event.ActionEvent evt) {
250         // TODO add your handling code here:
251     }
252     public static void main(String args[]) {
253         java.awt.EventQueue.invokeLater(new Runnable() {
254             public void run() {
255                 new showPesawat().setVisible(b: true);
256             }
257         });
258     }

```

Design Preview [showPesawat]

PEMESANAN TIKET PESAWAT

Nama	NIK	Kota Asal	Kota Tujuan	Tanggal Berangkat	Harga

Total Total

DATA PENUMPANG PESAWAT

Cari	Cari Data						
Nama	NIK	Kota Asal	Kota Tujuan	Tanggal Berangkat	Alamat	No Hp	

Kembali

Deskripsi Program

- **SetTotalHarga**

Aksi ini digunakan untuk menghitung dan menampilkan total harga dari pemesanan tiket pesawat yang telah diinputkan pengguna.

- **UpdateDataPenumpang**

Aksi ini digunakan untuk mengubah atau memperbarui informasi penumpang dalam pemesanan tiket pesawat. Ketika pengguna menambah data pemesanan tiket, maka program akan mengupdate data penumpang sesuai dengan data yang diinputkan..

- **btCariActionPerformed**

Aksi ini digunakan untuk melakukan pencarian dalam daftar pemesanan tiket pesawat. Pengguna akan diminta untuk memasukkan kriteria pencarian seperti NIK, nama penumpang, atau tanggal keberangkatan. Kemudian program akan memfilter data hingga ditemukan data yang sesuai.

3.1 Diagram Kelas

