

GUI & DATABASE



Arranged By:

Rajendra Rakha Arya Prabaswara

(1941720080/21)

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

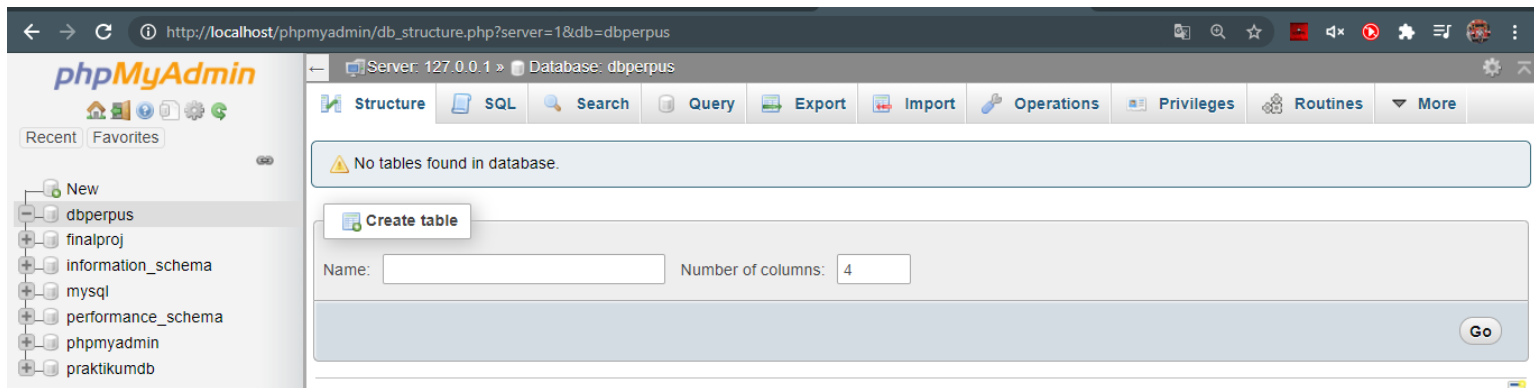
POLITEKNIK NEGERI MALANG



Experiment 1

Creating a database.

1. The first step of this experiment is to create a database. Install XAMPP, open phpMyAdmin, create dbperpus database, and the tables:



dbperpus kategori
idkategori : int(11)
nama : varchar(255)
keterangan : varchar(255)

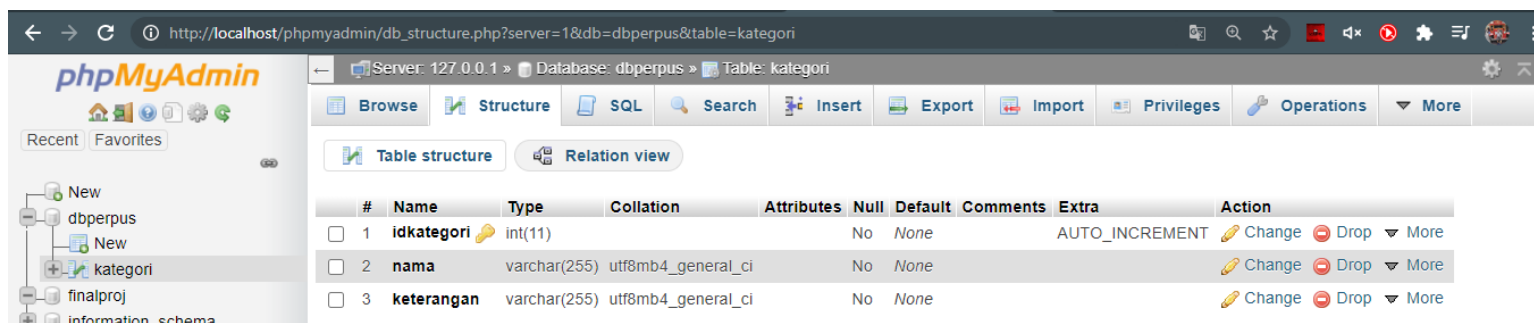
dbperpus buku
idbuku : int(11)
idkategori : int(11)
judul : varchar(255)
penerbit : varchar(255)
penulis : varchar(255)

dbperpus anggota
idanggota : int(11)
nama : varchar(255)
alamat : varchar(255)
telepon : varchar(25)

dbperpus peminjaman
idpeminjaman : int(11)
idanggota : int(11)
idbuku : int(11)
tanggalpinjam : date
tanggalkembali : date

Set all id primary key in each table (idanggota, idkategori, idpeminjaman, idbuku) with Auto Increment.

➤ Kategori





➤ **Buku**

Server: 127.0.0.1 » Database: dbperpus » Table: buku

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	idbuku	int(11)		No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	idkategori	int(11)		No	None			Change Drop More
<input type="checkbox"/>	3	judul	varchar(255)	utf8mb4_general_ci	No	None			Change Drop More
<input type="checkbox"/>	4	penerbit	varchar(255)	utf8mb4_general_ci	No	None			Change Drop More
<input type="checkbox"/>	5	penulis	varchar(255)	utf8mb4_general_ci	No	None			Change Drop More

➤ **Anggota**

Server: 127.0.0.1 » Database: dbperpus » Table: anggota

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	idanggota	int(11)		No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	nama	varchar(255)	utf8mb4_general_ci	No	None			Change Drop More
<input type="checkbox"/>	3	alamat	varchar(255)	utf8mb4_general_ci	No	None			Change Drop More
<input type="checkbox"/>	4	telepon	varchar(25)	utf8mb4_general_ci	No	None			Change Drop More

➤ **Peminjaman**

Server: 127.0.0.1 » Database: dbperpus » Table: peminjaman

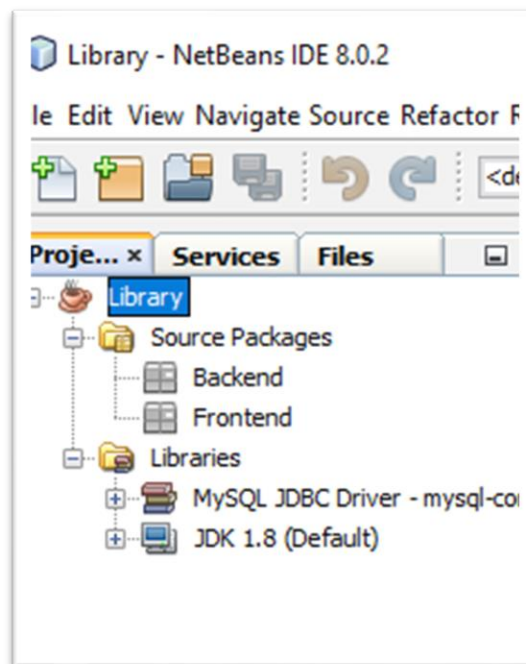
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	idpeminjaman	int(11)		No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	idanggota	int(11)		No	None			Change Drop More
<input type="checkbox"/>	3	idbuku	int(11)		No	None			Change Drop More
<input type="checkbox"/>	4	tanggalpinjam	date		No	None			Change Drop More
<input type="checkbox"/>	5	tanggalkembali	date		No	None			Change Drop More



Experiment 2

Preparing project.

1. Create a new project, named Library.
2. In the project explorer, right click on Libraries→ Add Library, select the MySQL JDBC Driver.
3. Create package **frontend** and **backend**. How to create a package is, in the project explorer, right click on the Source Packages→ New → Java Package, give her the package name (frontend, backend).





Experiment 3

Creating helper classes to execute SQL queries.

1. On the backend package, create DBHelper class.
2. Import java.sql. *
3. In this class there are those methods include:
 - a. **bukaKoneksi ()**, To open a connection to the database
 - b. **insertQueryGetId (String query)**, to insert into the table and returns the ID generated by the database (yield Auto Increment).
 - c. **executeQuery (String query)**, To execute a query that does not return a value (eg: insert, update, delete).
 - d. **selectQuery (String query)**, To execute a select query which returns the query results.

```
6 package Backend;
7
8 import java.sql.*;
9
10 /**
11  * @author Rajendra Rakha
12  */
13
14 public class DBHelper {
15
16     private static Connection connection;
17
18     public static void bukaKoneksi() {
19         if (connection == null) {
20             try {
21                 String url = "jdbc:mysql://localhost:3306/dbperpus";
22                 String user = "root";
23                 String password = "";
24                 DriverManager.registerDriver(new com.mysql.jdbc.Driver());
25                 connection = DriverManager.getConnection(url, user, password);
26             } catch (SQLException e) {
27                 System.out.println("Error connection");
28             }
29         }
30     }
31
32     public static int insertQueryGetId(String query) {
33         bukaKoneksi();
34         int num = 0;
35         int result = -1;
36
37         try {
38             Statement stmt = connection.createStatement();
39             num = stmt.executeUpdate(query, Statement.RETURN_GENERATED_KEYS);
40
41             ResultSet rs = stmt.getGeneratedKeys();
42
43             if (rs.next()) {
44                 result = rs.getInt(1);
45             }
46
47             rs.close();
48             stmt.close();
49         } catch (Exception e) {
50             e.printStackTrace();
51             result = -1;
52         }
53
54         return result;
55     }
56
57     public static boolean executeQuery(String query) {
58         bukaKoneksi();
59         boolean result = false;
60
61         try {
62             Statement stmt = connection.createStatement();
63             stmt.executeUpdate(query);
64
65             result = true;
66
67             stmt.close();
68         } catch (Exception e) {
69             e.printStackTrace();
70         }
71
72         return result;
73     }
74
75     public static ResultSet selectQuery(String query) {
76         bukaKoneksi();
77         ResultSet rs = null;
78
79         try {
80             Statement stmt = connection.createStatement();
81             rs = stmt.executeQuery(query);
82         } catch (Exception e) {
83             e.printStackTrace();
84         }
85
86         return rs;
87     }
88 }
89
```



Experiment 4

Creating a class to handle CRUD class in table **kategori**.

1. On the backend package, create a new class **Kategori**.
2. Add import `java.util.ArrayList` and `java.sql.*`
3. Add attributes corresponding fields in the tables **Kategori**.
4. Add getters setters for each attribute. You can use the NetBeans Code Insert facility. To do this, right-click anywhere in the editor, choose Insert Code, select Setter and Getter, check all of the attributes (`idkategori`, `name`, `description`).
5. Add custom default constructor and a constructor, which is used to set the name and description attributes. `Idkategori` attribute can not be set, because this id will be generated automatically through AutoIncrement features in MySQL.
6. Add method `getById ()` to get a class object in the database by its id.
7. Add method `getAll ()` to get all the data categories in the database, and accommodated to the `ArrayList <Kategori>`.
8. Add method `search ()` in order to perform data searches. This method is similar to the method `getAll ()` but different querynya.
9. Add method `save ()`. This method has two functions, namely the insert and update. If the data is entered yet (`idkategori = 0`) then it will automatically insert. If the input data already exists, it automatically updates.
10. Add method `delete ()` to do the removal operation on the table in the database **Kategori**.



```
6 package Backend;
7
8 /**
9  *
10  * @author Rajendra Rakha
11  */
12 import java.util.*;
13 import java.sql.*;
14
15 public class Kategori {
16     private int idkategori;
17     private String nama, keterangan;
18
19     public Kategori() {
20     }
21
22     public Kategori(String nama, String keterangan) {
23         this.nama = nama;
24         this.keterangan = keterangan;
25     }
26
27     public int getIdKategori() {
28         return idkategori;
29     }
30
31     public void setIdKategori(int idkategori) {
32         this.idkategori = idkategori;
33     }
34
35     public String getNama() {
36         return nama;
37     }
38
39     public void setNama(String nama) {
40         this.nama = nama;
41     }
42
43     public String getKeterangan() {
44         return keterangan;
45     }
46
47     public void setKeterangan(String keterangan) {
48         this.keterangan = keterangan;
49     }
50
51     public String toString() {
52         return nama;
53     }
54
55     public Kategori getById(int id){
56         Kategori kat = new Kategori();
57         ResultSet rs = DBHelper.selectQuery("SELECT * FROM kategori WHERE idkategori = " + id + "");
58
59         try {
60             while(rs.next()){
61                 kat = new Kategori();
62                 kat.setIdKategori(rs.getInt("idkategori"));
63                 kat.setNama(rs.getString("nama"));
64                 kat.setKeterangan(rs.getString("keterangan"));
65             }
66         } catch (Exception e) {
67             e.printStackTrace();
68         }
69         return kat;
70     }
71
72     public ArrayList<Kategori> getAll(){
73         ArrayList<Kategori> ListKategori = new ArrayList();
74         ResultSet rs = DBHelper.selectQuery("SELECT * FROM kategori");
75
76         try {
77             while (rs.next()) {
78                 Kategori kat = new Kategori();
79                 kat.setIdKategori(rs.getInt("idkategori"));
80                 kat.setNama(rs.getString("nama"));
81                 kat.setKeterangan(rs.getString("keterangan"));
82
83                 ListKategori.add(kat);
84             }
85         } catch (Exception e) {
86             e.printStackTrace();
87         }
88         return ListKategori;
89     }
90
91     public ArrayList<Kategori> search(String keyword){
92         ArrayList<Kategori> ListKategori = new ArrayList();
93         String sql = "SELECT * FROM kategori WHERE "
94             + "nama LIKE '%" + keyword + "%'"
95             + "OR keterangan LIKE '%" + keyword + "%'";
96         ResultSet rs = DBHelper.selectQuery(sql);
97
98         try {
99             while (rs.next()) {
100                 Kategori kat = new Kategori();
101                 kat.setIdKategori(rs.getInt("idkategori"));
102                 kat.setNama(rs.getString("nama"));
103                 kat.setKeterangan(rs.getString("keterangan"));
104             }
105         }
```




```
104
105         ListKategori.add(kat);
106     }
107     } catch (Exception e) {
108         e.printStackTrace();
109     }
110     return ListKategori;
111 }
112
113 public void save(){
114     if(getById(idkategori).getIdKategori() == 0){
115         String SQL = "INSERT INTO kategori (nama, keterangan) VALUES("
116             + "'" + this.nama + "', "
117             + "'" + this.keterangan + "'"
118             + ")";
119         this.idkategori = DBHelper.insertQueryGetId(SQL);
120     } else {
121         String SQL = "UPDATE kategori set "
122             + " nama = '" + this.nama + "', "
123             + " keterangan = '" + this.keterangan + "'"
124             + " WHERE idkategori = '" + this.idkategori + "'";
125         DBHelper.executeQuery(SQL);
126     }
127 }
128
129 public void delete(){
130     String SQL = "DELETE FROM kategori WHERE idkategori = '" + this.idkategori + "'";
131     DBHelper.executeQuery(SQL);
132 }
133 }
134
```




Experiment 5

Trying backed that has been made by operating through a text-based frontend (console). This experiment can you skip if you have been convinced that you have made the backend is functioning properly.

```
10  * @author Rajendra Rakha
11  */
12  public class TestBackend {
13
14      public static void main(String[] args) {
15          Kategori kat1 = new Kategori("Novel", "collection of paperback books");
16          Kategori kat2 = new Kategori("Reference", "scholarly reference book");
17          Kategori kat3 = new Kategori("Comic", "Comic children");
18
19          // test insert
20          kat1.save();
21          kat2.save();
22          kat3.save();
23
24          // test update
25          kat2.setKeterangan("Collection of scientific reference books");
26          kat2.save();
27
28          // test delete
29          kat3.delete();
30
31          // test select all
32          for (Kategori k : new Kategori().getAll()) {
33              System.out.println("name:" + k.getNama() + ", Ket:" + k.getKeterangan());
34          }
35
36          // test search
37          for (Kategori k : new Kategori().search("scientific")) {
38              System.out.println("name:" + k.getNama() + ", Ket:" + k.getKeterangan());
39          }
40      }
41  }
```

OUTPUT

```
run:
name:Novel, Ket:collection of paperback books
name:Reference, Ket:Collection of scientific reference books
name:Reference, Ket:Collection of scientific reference books
BUILD SUCCESSFUL (total time: 1 second)
```



Experiment 6

In this experiment we will create a GUI interface for the class Kategori.

1. In the frontend package, create a JFrame with FrmKategori name. To do this, right click on the package frontend → New → JFrame Form.

Title 1	Title 2	Title 3	Title 4

```
1 package frontend;
2
3 import backend.*;
4 import java.util.ArrayList;
5 import javax.swing.table.DefaultTableModel;
6
7 public class FrmKategori extends javax.swing.JFrame {
8
9     public FrmKategori() {
10         initComponents();
11         tampilkanData();
12         kosongkanForm();
13     }
14
15     public void kosongkanForm() {
16         txtIdKategori.setText("");
17         txtNama.setText("");
18         txtKeterangan.setText("");
19     }
20 }
```

```
21 public final void tampilkanData() {
22     String[] kolom = {"ID", "Nama", "Keterangan"};
23     ArrayList<Kategori> list = new Kategori().getAll();
24     Object rowData[] = new Object[3];
25
26     tblKategori.setModel(new DefaultTableModel(new Object[][](), kolom));
27
28     for (Kategori kat : list) {
29         rowData[0] = kat.getIdKategori();
30         rowData[1] = kat.getNama();
31         rowData[2] = kat.getKeterangan();
32
33         ((DefaultTableModel) tblKategori.getModel()).addRow(rowData);
34     }
35 }
36
37 public final void cari(String keyword) {
38     String[] kolom = {"ID", "Nama", "Keterangan"};
39     ArrayList<Kategori> list = new Kategori().search(keyword);
40     Object rowData[] = new Object[3];
41
42     tblKategori.setModel(new DefaultTableModel(new Object[][](), kolom));
43
44     for (Kategori kat : list) {
45         rowData[0] = kat.getIdKategori();
46         rowData[1] = kat.getNama();
47         rowData[2] = kat.getKeterangan();
48
49         ((DefaultTableModel) tblKategori.getModel()).addRow(rowData);
50     }
51 }
```



— □ ×

ID Kategori

Nama Kategori

Keterangan

ID	Nama	Keterangan
1	Novel	Koleksi buku novel
2	Referensi	Koleksi buku referensi i...
4	Referensi	koleksi buku referensi il...
5	Fiksis	buku fiksi aja
6	Fiksis	buku fiksi aja

— □ ×

ID Kategori

Nama Kategori

Keterangan

ID	Nama	Keterangan
1	Novel	Koleksi buku novel

]



Experiment 6 Question

Do the same for Member data!

1. Create Member class on the backend package, complete its attributes and methods.
2. Perform test on TestBackend class on the frontend package.

```
1 package backend;
2
3 import java.sql.ResultSet;
4 import java.util.ArrayList;
5
6 public class Anggota {
7
8     private int idAnggota;
9     private String nama, alamat, telepon;
10
11     public Anggota() {
12     }
13
14     public Anggota(String nama, String alamat, String telepon) {
15         this.nama = nama;
16         this.alamat = alamat;
17         this.telepon = telepon;
18     }
19
20     public int getIdAnggota() {
21         return idAnggota;
22     }
23
24     public void setIdAnggota(int idAnggota) {
25         this.idAnggota = idAnggota;
26     }
27
28     public String getNama() {
29         return nama;
30     }
31
32     public void setNama(String nama) {
33         this.nama = nama;
34     }
35
36     public String getAlamat() {
37         return alamat;
38     }
39
40     public void setAlamat(String alamat) {
41         this.alamat = alamat;
42     }
43
44     public String getTelepon() {
45         return telepon;
46     }
47
48     public void setTelepon(String telepon) {
49         this.telepon = telepon;
50     }
51
52     public Anggota getById(int id) {
53         Anggota ang = new Anggota();
54         ResultSet rs = DBHelper.selectQuery("SELECT * FROM anggota WHERE idanggota = " + id + "");
55
56         try {
57             while (rs.next()) {
58                 ang = new Anggota();
59                 ang.setIdAnggota(rs.getInt("idanggota"));
60                 ang.setNama(rs.getString("nama"));
61                 ang.setAlamat(rs.getString("alamat"));
62                 ang.setTelepon(rs.getString("telepon"));
63             }
64         } catch (Exception e) {
65             e.printStackTrace();
66         }
67         return ang;
68     }
69
70     public ArrayList<Anggota> getAll() {
71         ArrayList<Anggota> ListAnggota = new ArrayList();
72         ResultSet rs = DBHelper.selectQuery("SELECT * FROM anggota");
73
74         try {
75             while (rs.next()) {
76                 Anggota ang = new Anggota();
77                 ang.setIdAnggota(rs.getInt("idanggota"));
78                 ang.setNama(rs.getString("nama"));
79                 ang.setAlamat(rs.getString("alamat"));
80                 ang.setTelepon(rs.getString("telepon"));
81
82                 ListAnggota.add(ang);
83             }
84         }
```



```
107     } catch (Exception e) {  
108         e.printStackTrace();  
109     }  
110     return ListAnggota;  
111 }  
112  
113 public void save() {  
114     if (getById(idAnggota).getIdAnggota() == 0) {  
115         String SQL = "INSERT INTO anggota (nama, alamat, telepon) VALUES(" +  
116             + "" + this.nama + ", " +  
117             + "" + this.alamat + ", " +  
118             + "" + this.telepon + "" +  
119             + ")";  
120         this.idAnggota = DBHelper.insertQueryGetId(SQL);  
121     } else {  
122         String SQL = "UPDATE anggota set " +  
123             + "nama = " + this.nama + ", " +  
124             + "alamat = " + this.alamat + ", " +  
125             + "telepon = " + this.telepon + "" +  
126             + " WHERE idanggota = " + this.idAnggota + "";  
127         DBHelper.executeQuery(SQL);  
128     }  
129 }  
130  
131 public void delete() {  
132     String SQL = "DELETE FROM anggota WHERE idanggota = " + this.idAnggota + "";  
133     DBHelper.executeQuery(SQL);  
134 }  
135 }  
136
```

run:

```
Nama: Rakha, Alamat: Palembang, Telepon: 0823231241  
Nama: Arya, Alamat: Jakarta, Telepon: 081303556400  
Nama: Rakha, Alamat: Palembang, Telepon: 0823231241  
Nama: Arya, Alamat: Jakarta, Telepon: 081303556400  
BUILD SUCCESSFUL (total time: 1 second)
```



Experiment 7

Members create a form for data.

1. Create FrmAnggota on frontend and fill in the component package, the method and its events.

```
1 package frontend;
2 import backend.*;
3 import java.util.ArrayList;
4 import javax.swing.JOptionPane;
5 import javax.swing.table.DefaultTableModel;
6
7 public class FrmAnggota extends javax.swing.JFrame {
8     public FrmAnggota() {
9         initComponents();
10        tampilkanData();
11        kosongkanForm();
12    }
13
14    public void kosongkanForm(){
15        txtIdAnggota.setText("");
16        txtNamaAnggota.setText("");
17        txtAlamat.setText("");
18        txtTelepon.setText("");
19    }
20
21    public final void tampilkanData(){
22        String[] kolom = {"ID", "Nama", "Alamat", "Telepon"};
23        ArrayList<Anggota> list = new Anggota().getAll();
24        Object rowData[] = new Object[4];
25
26        tblAnggota.setModel(new DefaultTableModel(new Object[][] {}, kolom));
27
28        for (Anggota ang : list) {
29            rowData[0] = ang.getIdAnggota();
30            rowData[1] = ang.getNama();
31            rowData[2] = ang.getAlamat();
32            rowData[3] = ang.getTelepon();
33
34            ((DefaultTableModel)tblAnggota.getModel()).addRow(rowData);
35        }
36    }
37
38    public final void cari(String keyword){
39        String[] kolom = {"ID", "Nama", "Alamat", "Telepon"};
40        ArrayList<Anggota> list = new Anggota().search(keyword);
41        Object rowData[] = new Object[4];
42
43        tblAnggota.setModel(new DefaultTableModel(new Object[][] {}, kolom));
44
45        for (Anggota ang : list) {
46            rowData[0] = ang.getIdAnggota();
47            rowData[1] = ang.getNama();
48            rowData[2] = ang.getAlamat();
49            rowData[3] = ang.getTelepon();
50
51            ((DefaultTableModel)tblAnggota.getModel()).addRow(rowData);
52        }
53    }
54 }
```

```
54
55 public boolean checkInput(String nama, String alamat, String telepon){
56     boolean res = true;
57     if(nama.equals("") && alamat.equals("") && telepon.equals("")){
58         res = false;
59     } else if(nama.equals("")) {
60         res = false;
61     } else if (alamat.equals("")) {
62         res = false;
63     } else if (telepon.equals("")){
64         res = false;
65     }
66     return res;
67 }
68
69 @SuppressWarnings("unchecked")
70 Generated Code
71
72 private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
73     // TODO add your handling code here:
74     Anggota ang = new Anggota();
75
76     boolean res = checkInput(txtNamaAnggota.getText(), txtAlamat.getText(), txtTelepon.getText());
77
78     if(res){
79         ang.setIdAnggota(Integer.parseInt(txtIdAnggota.getText()));
80         ang.setNama(txtNamaAnggota.getText());
81         ang.setAlamat(txtAlamat.getText());
82         ang.setTelepon(txtTelepon.getText());
83         ang.save();
84         txtIdAnggota.setText(Integer.toString(ang.getIdAnggota()));
85         tampilkanData();
86     } else {
87         JOptionPane.showMessageDialog(this, "Nama, alamat, dan telepon harus diisi");
88     }
89 }
90
91 private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
92     // TODO add your handling code here:
93     if (tblAnggota.getSelectionModel().isEmpty()){
94         JOptionPane.showMessageDialog(this, "Silahkan pilih data yang akan dihapus");
95         kosongkanForm();
96     } else {
97         DefaultTableModel model = (DefaultTableModel)tblAnggota.getModel();
98         int row = tblAnggota.getSelectedRow();
99
100        Anggota ang = new Anggota().getById(Integer.parseInt(model.getValueAt(row, 0).toString()));
101        ang.delete();
102        kosongkanForm();
103        tampilkanData();
104    }
105 }
```



ID Anggota

Nama Anggota

Alamat

Telepon

Title 1	Title 2	Title 3	Title 4

ID Anggota

Nama Anggota

Alamat

Telepon

ID	Nama	Alamat	Telepon
22	Rakha	Palembang	0823231241
23	Arya	Jakarta	081303556400



For Book data, approximately the same way as the data Kategori and members. Only different is:

- Dialing getKategori (). GetIdKategori () on the query, insert and update to set idkategori in table book
- SELECT queries involving join table on getById method (), getAll () and search () .

The complete code book class you can see in Appendix 1. To test book on the frontend, you can see in Appendix 2.

```
1 package backend;
2 import java.util.ArrayList;
3 import java.sql.*;
4
5 public class Buku {
6     private int idbuku;
7     private Kategori kategori;
8     private String judul, penerbit, penulis;
9
10    public Buku() {
11    }
12
13    public Buku(Kategori kategori, String judul, String penerbit, String penulis) {
14        this.kategori = kategori;
15        this.judul = judul;
16        this.penerbit = penerbit;
17        this.penulis = penulis;
18    }
19
20    public int getIdbuku() {
21        return idbuku;
22    }
23
24    public void setIdbuku(int idbuku) {
25        this.idbuku = idbuku;
26    }
27
28    public Kategori getKategori() {
29        return kategori;
30    }
31
32    public void setKategori(Kategori kategori) {
33        this.kategori = kategori;
34    }
35
36    public String getJudul() {
37        return judul;
38    }
39
40    public void setJudul(String judul) {
41        this.judul = judul;
42    }
43
44    public String getPenerbit() {
45        return penerbit;
46    }
```

```
48    public void setPenerbit(String penerbit) {
49        this.penerbit = penerbit;
50    }
51
52    public String getPenulis() {
53        return penulis;
54    }
55
56    public void setPenulis(String penulis) {
57        this.penulis = penulis;
58    }
59
60    public Buku getById(int id){
61        Buku buku = new Buku();
62
63        String query = "SELECT "
64            + "b.idbuku AS idbuku, "
65            + "b.judul AS judul, "
66            + "b.penerbit AS penerbit, "
67            + "b.penulis AS penulis, "
68            + "k.idkategori AS idkategori, "
69            + "k.nama AS nama, "
70            + "k.keterangan AS keterangan "
71            + "FROM buku b "
72            + "LEFT JOIN kategori k ON b.idkategori = k.idkategori "
73            + "WHERE b.idbuku = " + id + "" ;
74
75        ResultSet rs = DBHelper.selectQuery(query);
76
77        try {
78            while(rs.next()){
79                buku = new Buku();
80                buku.setIdbuku(rs.getInt("idbuku"));
81                buku.getKategori().setIdKategori(rs.getInt("idkategori"));
82                buku.getKategori().setName(rs.getString("nama"));
83                buku.getKategori().setKeterangan(rs.getString("keterangan"));
84                buku.setJudul(rs.getString("judul"));
85                buku.setPenerbit(rs.getString("penerbit"));
86                buku.setPenulis(rs.getString("penulis"));
87            }
88        } catch (Exception e) {
89            e.printStackTrace();
90        }
91
92        return buku;
93    }
```

```
94    public ArrayList<Buku> getAll(){
95        ArrayList<Buku> ListBuku = new ArrayList();
96        String query = "SELECT b.idbuku AS idbuku, b.judul AS judul, b.penerbit AS penerbit, b.penulis AS penulis, k.idkategori AS idkategori, k.nama AS nama, k.keterangan AS keterangan "
97            + "FROM buku AS b LEFT JOIN kategori AS k ON b.idkategori = k.idkategori ";
98        ResultSet rs = DBHelper.selectQuery(query);
99
100
101        try {
102            while(rs.next()){
103                Buku buku = new Buku();
104                buku.setIdbuku(rs.getInt("idbuku"));
105                buku.getKategori().setIdKategori(rs.getInt("idkategori"));
106                buku.getKategori().setName(rs.getString("nama"));
107                buku.getKategori().setKeterangan(rs.getString("keterangan"));
108                buku.setJudul(rs.getString("judul"));
109                buku.setPenerbit(rs.getString("penerbit"));
110                buku.setPenulis(rs.getString("penulis"));
111
112                ListBuku.add(buku);
113            }
114        } catch (Exception e) {
115            e.printStackTrace();
116        }
117
118        return ListBuku;
119    }
120
121    public ArrayList<Buku> search(String keyword){
122        ArrayList<Buku> ListBuku = new ArrayList();
123
124
125        String query = "SELECT b.idbuku AS idbuku, b.judul AS judul, b.penerbit AS penerbit, b.penulis AS penulis, k.idkategori AS idkategori, k.nama AS nama, k.keterangan AS keterangan "
126            + "FROM buku AS b LEFT JOIN kategori AS k ON b.idkategori = k.idkategori "
127            + "WHERE b.judul LIKE '%" + keyword + "%'"
128            + "OR b.penerbit LIKE '%" + keyword + "%'"
129            + "OR b.penulis LIKE '%" + keyword + "%'";
130        ResultSet rs = DBHelper.selectQuery(query);
131
132
133        try {
134            while(rs.next()){
135                Buku buku = new Buku();
136                buku.setIdbuku(rs.getInt("idbuku"));
137                buku.getKategori().setIdKategori(rs.getInt("idkategori"));
138                buku.getKategori().setName(rs.getString("nama"));
139                buku.getKategori().setKeterangan(rs.getString("keterangan"));
140                buku.setJudul(rs.getString("judul"));
141                buku.setPenerbit(rs.getString("penerbit"));
142                buku.setPenulis(rs.getString("penulis"));
143            }
144        }
```



```
143         DBHelper.executeUpdate(SQL);
144     }
145     } catch (Exception e) {
146         e.printStackTrace();
147     }
148 }
149 return ListBuku;
150 }
151
152 public void save(){
153     if(getById(idbuku).getIdbuku() == 0){
154         String SQL = "INSERT INTO buku (idkategori, judul, penerbit, penulis) VALUES("
155             + "" + this.getKategori().getIdKategori() + ","
156             + "" + this.judul + ","
157             + "" + this.penerbit + ","
158             + "" + this.penulis + ""
159             + ")";
160
161         this.idbuku = DBHelper.insertQueryGetId(SQL);
162     } else {
163         String SQL = "UPDATE buku SET "
164             + "idkategori = " + this.getKategori().getIdKategori() + ","
165             + "judul = " + this.judul + ","
166             + "penerbit = " + this.penerbit + ","
167             + "penulis = " + this.penulis + ""
168             + "WHERE idbuku = " + this.idbuku + "";
169         DBHelper.executeUpdate(SQL);
170     }
171 }
172
173 public void delete(){
174     String SQL = "DELETE FROM buku WHERE idbuku = " + this.idbuku + "";
175     DBHelper.executeUpdate(SQL);
176 }
177 }
178 }
```

run:

Kategori: Novel, Judul: Timun Mas
Kategori: Reference, Judul: Aljabar Linier
Kategori: Novel, Judul: Timun Mas
BUILD SUCCESSFUL (total time: 1 second)



Experiment 8

Creating a GUI for Data Book, which is equipped with a combo box to select the Kategori that is connected with the table Kategori.

```
1 package frontend;
2
3 import backend.*;
4 import java.util.ArrayList;
5 import javax.swing.DefaultComboBoxModel;
6 import javax.swing.JOptionPane;
7 import javax.swing.table.DefaultTableModel;
8
9 public class FrmBuku extends javax.swing.JFrame {
10
11     /**
12      * Creates new form FrmBuku
13      */
14     public FrmBuku() {
15         initComponents();
16         tampilkanData();
17         tampilkanCmbKategori();
18         kembalikanForm();
19     }
20
21     public void kembalikanForm() {
22         txtIdBuku.setText("");
23         cmbKategori.setSelectedIndex(0);
24         txtJudul.setText("");
25         txtPenerbit.setText("");
26         txtPenulis.setText("");
27     }
28
29     public void tampilkanCmbKategori() {
30         cmbKategori.setModel(new DefaultComboBoxModel(new Kategori().getAll().toArray()));
31     }
32
33     public final void tampilkanData() {
34         String[] kolom = {"ID", "Kategori", "Judul", "Penerbit", "Penulis"};
35         ArrayList<Buku> list = new Buku().getAll();
36         Object rowData[] = new Object[5];
37
38         tblBuku.setModel(new DefaultTableModel(new Object[][] {}, kolom));
39
40         for (int i = 0; i < list.size(); i++) {
41             rowData[0] = list.get(i).getIdBuku();
42             rowData[1] = list.get(i).getKategori().getNama();
43             rowData[2] = list.get(i).getJudul();
44             rowData[3] = list.get(i).getPenerbit();
45             rowData[4] = list.get(i).getPenulis();
46
47             ((DefaultTableModel) tblBuku.getModel()).addRow(rowData);
48         }
49     }
50 }
```

```
50
51 public final void cari(String keyword) {
52     String[] kolom = {"ID", "Kategori", "Judul", "Penerbit", "Penulis"};
53     ArrayList<Buku> list = new Buku().search(keyword);
54     Object rowData[] = new Object[5];
55
56     tblBuku.setModel(new DefaultTableModel(new Object[][] {}, kolom));
57
58     for (Buku buk : list) {
59         rowData[0] = buk.getIdBuku();
60         rowData[1] = buk.getJudul();
61         rowData[2] = buk.getKategori().getNama();
62         rowData[3] = buk.getPenerbit();
63         rowData[4] = buk.getPenulis();
64
65         ((DefaultTableModel) tblBuku.getModel()).addRow(rowData);
66     }
67 }
68
69 private boolean checkInput(String judul, String penerbit, String penulis) {
70     boolean res = true;
71     if (judul.equals("") && penerbit.equals("") && penulis.equals("")) {
72         res = false;
73     } else if (judul.equals("")) {
74         res = false;
75     } else if (penerbit.equals("")) {
76         res = false;
77     } else if (penulis.equals("")) {
78         res = false;
79     }
80     return res;
81 }
```



Simulasi Aplikasi Manajemen Perpustakaan

ID Buku:

Kategori:

Judul:

Penerbit:

Penulis:

ID	Kategori	Judul	Penerbit	Penulis
10	Novel	Timun Mas	Elex Media	Bang Supit
11	Reference	Aljabar Linier	Springer	Alex Baldwin



Assignment

1. Make Peminjaman class.

```
1 package backend;
2
3 import java.sql.ResultSet;
4 import java.sql.SQLException;
5 import java.text.SimpleDateFormat;
6 import java.util.ArrayList;
7 import java.util.Date;
8 import java.util.Locale;
9
10 public class Peminjaman {
11
12     private int idPeminjaman;
13     private Anggota anggota;
14     private Buku buku;
15     private String tanggalPinjam, tanggalKembali;
16
17     public Peminjaman() {
18     }
19
20     public Peminjaman(Anggota anggota, Buku buku, String tanggalPinjam, String tanggalKembali) {
21         this.anggota = anggota;
22         this.buku = buku;
23         this.tanggalPinjam = tanggalPinjam;
24         this.tanggalKembali = tanggalKembali;
25     }
26
27     public int getIdPeminjaman() {
28         return idPeminjaman;
29     }
30
31     public void setIdPeminjaman(int idPeminjaman) {
32         this.idPeminjaman = idPeminjaman;
33     }
34
35     public Anggota getAnggota() {
36         return anggota;
37     }
38
39     public void setAnggota(Anggota anggota) {
40         this.anggota = anggota;
41     }
42
43     public Buku getBuku() {
44         return buku;
45     }
46
47     public void setBuku(Buku buku) {
48         this.buku = buku;
49     }
50
51     public String getTanggalPinjam() {
52         return tanggalPinjam;
53     }
54 }
```

```
55 public void setTanggalPinjam(String tanggalPinjam) {
56     this.tanggalPinjam = tanggalPinjam;
57 }
58
59 public String getTanggalKembali() {
60     return tanggalKembali;
61 }
62
63 public void setTanggalKembali(String tanggalKembali) {
64     this.tanggalKembali = tanggalKembali;
65 }
66
67 public Peminjaman getById(int id) {
68     Peminjaman pen = new Peminjaman();
69
70     String query = "SELECT * FROM peminjaman p "
71         + "LEFT JOIN anggota a ON p.idanggota = a.idanggota "
72         + "LEFT JOIN buku b ON b.idbuku = p.idbuku "
73         + "WHERE p.idpeminjaman = " + id + " ";
74     ResultSet rs = DBHelper.selectQuery(query);
75
76     try {
77         while (rs.next()) {
78             pen = new Peminjaman();
79             Anggota ang = new Anggota();
80             Buku buk = new Buku();
81             pen.setAnggota(ang);
82             pen.setBuku(buk);
83
84             pen.setIdPeminjaman(rs.getInt("idpeminjaman"));
85             pen.setTanggalPinjam(rs.getString("tanggalpinjam"));
86             pen.setTanggalKembali(rs.getString("tanggalkembali"));
87             pen.getAnggota().setIdAnggota(rs.getInt("idanggota"));
88             pen.getAnggota().setNama(rs.getString("nama"));
89             pen.getAnggota().setAlamat(rs.getString("alamat"));
90             pen.getAnggota().setTelepon(rs.getString("telepon"));
91             pen.getBuku().setIdBuku(rs.getInt("idbuku"));
92             pen.getBuku().setJudul(rs.getString("judul"));
93             pen.getBuku().setPenerbit(rs.getString("penerbit"));
94             pen.getBuku().setPenulis(rs.getString("penulis"));
95         }
96     } catch (Exception e) {
97         e.printStackTrace();
98     }
99     return pen;
100 }
101
102 public ArrayList<Peminjaman> getAll() {
103     ArrayList<Peminjaman> Peminjaman = new ArrayList();
104     String query = "SELECT * FROM peminjaman p "
105         + "LEFT JOIN anggota a ON p.idanggota = a.idanggota "
106         + "LEFT JOIN buku b ON b.idbuku = p.idbuku ";
107     ResultSet rs = DBHelper.selectQuery(query);
108 }
```



```
100
101
102 try {
103     while (rs.next()) {
104         Peminjaman pen = new Peminjaman();
105         Anggota ang = new Anggota();
106         Buku buk = new Buku();
107         pen.setAnggota(ang);
108         pen.setBuku(buk);
109
110         pen.setIdPeminjaman(rs.getInt("idpeminjaman"));
111         pen.setTanggalPinjam(rs.getString("tanggalpinjam"));
112         pen.setTanggalKembali(rs.getString("tanggalkembali"));
113         pen.getAnggota().setIdAnggota(rs.getInt("idanggota"));
114         pen.getAnggota().setNama(rs.getString("nama"));
115         pen.getAnggota().setAlamat(rs.getString("alamat"));
116         pen.getAnggota().setTelepon(rs.getString("telepon"));
117         pen.getBuku().setIdBuku(rs.getInt("idbuku"));
118         pen.getBuku().setJudul(rs.getString("judul"));
119         pen.getBuku().setPenerbit(rs.getString("penerbit"));
120         pen.getBuku().setPenulis(rs.getString("penulis"));
121         Peminjaman.add(pen);
122     }
123 } catch (Exception e) {
124     e.printStackTrace();
125 }
126
127 return Peminjaman;
128
129
130
131
132
133
134
135
136 public ArrayList<Peminjaman> search(String keyword) {
137     ArrayList<Peminjaman> Peminjaman = new ArrayList();
138     String query = "SELECT * FROM peminjaman p "
139         + "LEFT JOIN anggota a ON p.idanggota = a.idanggota "
140         + "LEFT JOIN buku b ON p.idbuku = b.idbuku "
141         + "WHERE a.nama LIKE '%" + keyword + "%' "
142         + "OR a.alamat LIKE '%" + keyword + "%' "
143         + "OR a.telepon LIKE '%" + keyword + "%' "
144         + "OR b.judul LIKE '%" + keyword + "%' "
145         + "OR b.penerbit LIKE '%" + keyword + "%' "
146         + "OR b.penulis LIKE '%" + keyword + "%' ";
147     ResultSet rs = DBHelper.selectQuery(query);
148
149     try {
150         while (rs.next()) {
151             Peminjaman pen = new Peminjaman();
152             pen.setIdPeminjaman(rs.getInt("idpeminjaman"));
153             pen.setTanggalPinjam(rs.getString("tanggalpinjam"));
154             pen.setTanggalKembali(rs.getString("tanggalkembali"));
155             pen.getAnggota().setIdAnggota(rs.getInt("idanggota"));
156             pen.getAnggota().setNama(rs.getString("nama"));
157             pen.getAnggota().setAlamat(rs.getString("alamat"));
158             pen.getAnggota().setTelepon(rs.getString("telepon"));
159             pen.getBuku().setIdBuku(rs.getInt("idbuku"));
160             pen.getBuku().setJudul(rs.getString("judul"));
161             pen.getBuku().setPenerbit(rs.getString("penerbit"));
162             pen.getBuku().setPenulis(rs.getString("penulis"));
```

```
163         Peminjaman.add(pen);
164     }
165 } catch (Exception e) {
166     e.printStackTrace();
167 }
168
169 return Peminjaman;
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```



2. Make FrmPeminjaman

The form contains the following elements:

- ID: [Text Field]
- ID Anggota: [Text Field] Cari Nama Anggota
- ID Buku: [Text Field] Cari Judul Buku
- Tanggal Pinjam: [Text Field] Format: YYYY/MM/DD
- Tanggal Kembali: [Text Field] Format: YYYY/MM/DD
- Buttons: Simpan, Tambah Baru, Hapus
- Table with 4 columns (Title 1, Title 2, Title 3, Title 4) and 5 rows.

```
1 package frontend;
2
3 import backend.*;
4 import java.text.ParseException;
5 import java.text.SimpleDateFormat;
6 import java.util.ArrayList;
7 import javax.swing.JOptionPane;
8 import javax.swing.table.DefaultTableModel;
9
10 public class FrmPeminjaman extends javax.swing.JFrame {
11
12     /**
13      * Creates new form FrmPeminjaman
14      */
15     public FrmPeminjaman() {
16         initComponents();
17         tampilkanData();
18         kosongkanForm();
19     }
20
21     public boolean isValidDate(String inDate) {
22         SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd");
23         dateFormat.setLenient(false);
24         try {
25             dateFormat.parse(inDate.trim());
26         } catch (ParseException pe) {
27             return false;
28         }
29         return true;
30     }
31
32     public void kosongkanForm() {
33         txtIdBuku.setText("");
34         txtIdAnggota.setText("");
35         txtJudulBuku.setText("");
36         txtNamaAnggota.setText("");
37         txtTanggalKembali.setText("");
38         txtTanggalPinjam.setText("");
39     }
40
41     private Anggota cariAnggota(int idAnggota) {
42         Anggota ang = new Anggota().getId(idAnggota);
43         if (ang != null) {
44             return ang;
45         }
46         return null;
47     }
48
49 }
```

```
50 private Buku cariBuku(int idBuku) {
51     Buku buku = new Buku().getId(idBuku);
52     if (buku != null) {
53         return buku;
54     }
55     return null;
56 }
57
58 public final void tampilkanData() {
59     String[] kolom = {"ID", "Nama", "Judul Buku", "Tanggal Pinjam", "Tanggal Kembali"};
60     ArrayList<Peminjaman> pen = new Peminjaman().getAll();
61     Object rowData[] = new Object[5];
62     tblPeminjaman.setModel(new DefaultTableModel(new Object[][] {}, kolom));
63
64     for (int i = 0; i < pen.size(); i++) {
65         rowData[0] = pen.get(i).getIdPeminjaman();
66         rowData[1] = pen.get(i).getAnggota().getNama();
67         rowData[2] = pen.get(i).getBuku().getJudul();
68         rowData[3] = pen.get(i).getTanggalPinjam();
69         rowData[4] = pen.get(i).getTanggalKembali();
70
71         ((DefaultTableModel) tblPeminjaman.getModel()).addRow(rowData);
72     }
73 }
74
75 private boolean checkInput(String tanggalPinjam, String tanggalKembali) {
76     boolean res = true;
77
78     if (tanggalPinjam.equals("") && tanggalKembali.equals("")) {
79         res = false;
80     } else if (tanggalPinjam.equals("")) {
81         res = false;
82     } else if (tanggalKembali.equals("")) {
83         res = false;
84     }
85     return res;
86 }
87
88 /**
89  * This method is called from within the constructor to initialize the form.
90  * WARNING: Do NOT modify this code. The content of this method is always
91  * regenerated by the Form Editor.
92  */
93 @SuppressWarnings("unchecked")
94 // Generated Code
95
96 private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
97     // TODO add your handling code here:
98 }
```




ID

0

ID Anggota

Cari

Nama Anggota

ID Buku

Cari

Judul Buku

Tanggal Pinjam

Format: YYYY/MM/DD

Tanggal Kembali

Format: YYYY/MM/DD

Simpan

Tambah Baru

Hapus

ID	Nama	Judul Buku	Tanggal Pinjam	Tanggal Kembali
25	Rakha	Timun Mas	2020-01-03	2020-11-10
26	Arya	Aljabar Linier	2000-03-13	2021-10-10