

GUI dan Database

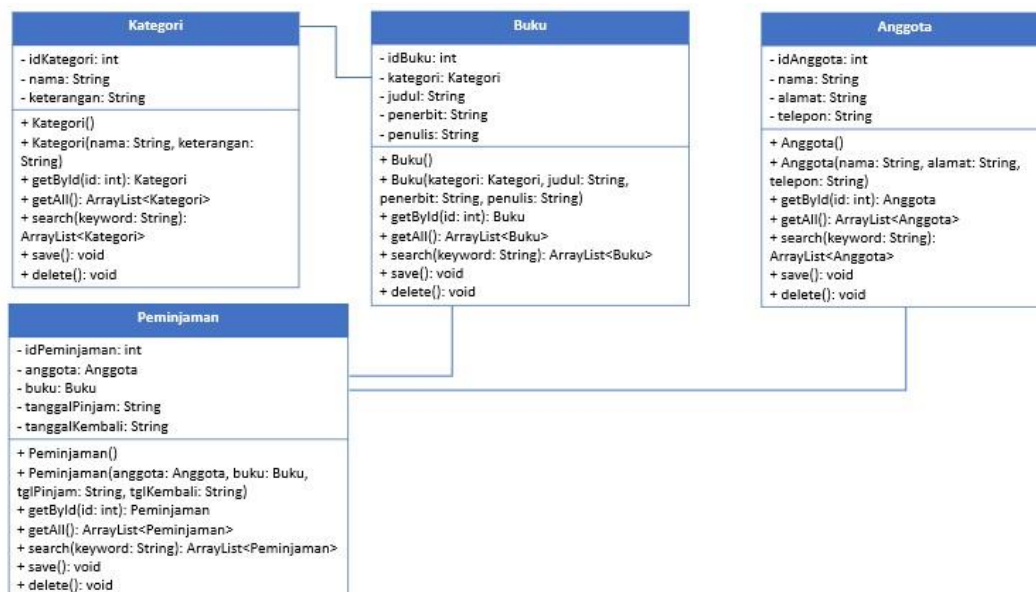
Nama : Rizqi Rohmatul Huda
Kelas : 2G-TI
No.Absen : 26
NIM : 2141720264

1. Kompetensi

Setelah menempuh materi percobaan ini, mahasiswa mampu:

1. Menggunakan paradigma berorientasi objek untuk interaksi dengan database
2. Membuat Graphical User Interface (GUI)

2. Pendahuluan



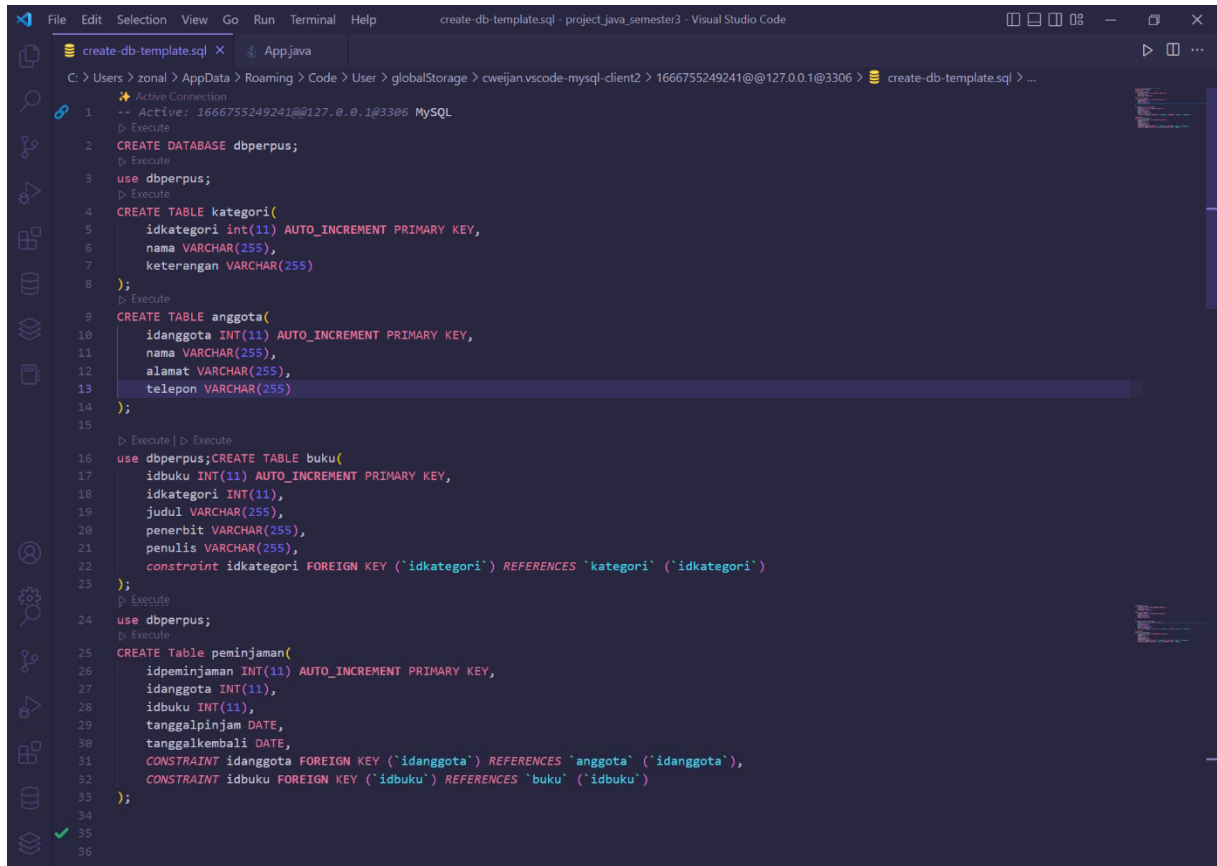
3. Percobaan

3.1 Percobaan 1

Membuat database.

dbperpus kategori	dbperpus buku	dbperpus anggota	dbperpus peminjaman
idkategori : int(11)	idbuku : int(11)	idanggota : int(11)	idpeminjaman : int(11)
nama : varchar(255)	idkategori : int(11)	nama : varchar(255)	idanggota : int(11)
keterangan : varchar(255)	judul : varchar(255)	alamat : varchar(255)	idbuku : int(11)
	penerbit : varchar(255)	telepon : varchar(25)	tanggalpinjam : date
	penulis : varchar(255)		tanggalkembali : date

Sintaks database perpustakaan



```
1 -- Active Connection
2 -- Active: 1666755249241@127.0.0.1@3306 MySQL
3 > Execute
4 CREATE DATABASE dbperpus;
5 > Execute
6 use dbperpus;
7 > Execute
8 CREATE TABLE kategori(
9     idkategori int(11) AUTO_INCREMENT PRIMARY KEY,
10     nama VARCHAR(255),
11     keterangan VARCHAR(255)
12 );
13 > Execute
14 CREATE TABLE anggota(
15     idanggota INT(11) AUTO_INCREMENT PRIMARY KEY,
16     nama VARCHAR(255),
17     alamat VARCHAR(255),
18     telepon VARCHAR(255)
19 );
20 > Execute | > Execute
21 use dbperpus; CREATE TABLE buku(
22     idbuku INT(11) AUTO_INCREMENT PRIMARY KEY,
23     idkategori INT(11),
24     judul VARCHAR(255),
25     penerbit VARCHAR(255),
26     penulis VARCHAR(255),
27     constraint idkategori FOREIGN KEY ('idkategori') REFERENCES 'kategori' ('idkategori')
28 );
29 > Execute
30 use dbperpus;
31 > Execute
32 CREATE Table peminjaman(
33     idpeminjaman INT(11) AUTO_INCREMENT PRIMARY KEY,
34     idanggota INT(11),
35     idbuku INT(11),
36     tanggalpinjam DATE,
37     tanggalkembali DATE,
38     CONSTRAINT idanggota FOREIGN KEY ('idanggota') REFERENCES 'anggota' ('idanggota'),
39     CONSTRAINT idbuku FOREIGN KEY ('idbuku') REFERENCES 'buku' ('idbuku')
40 );
41 > Execute
```

Database perpustakaan

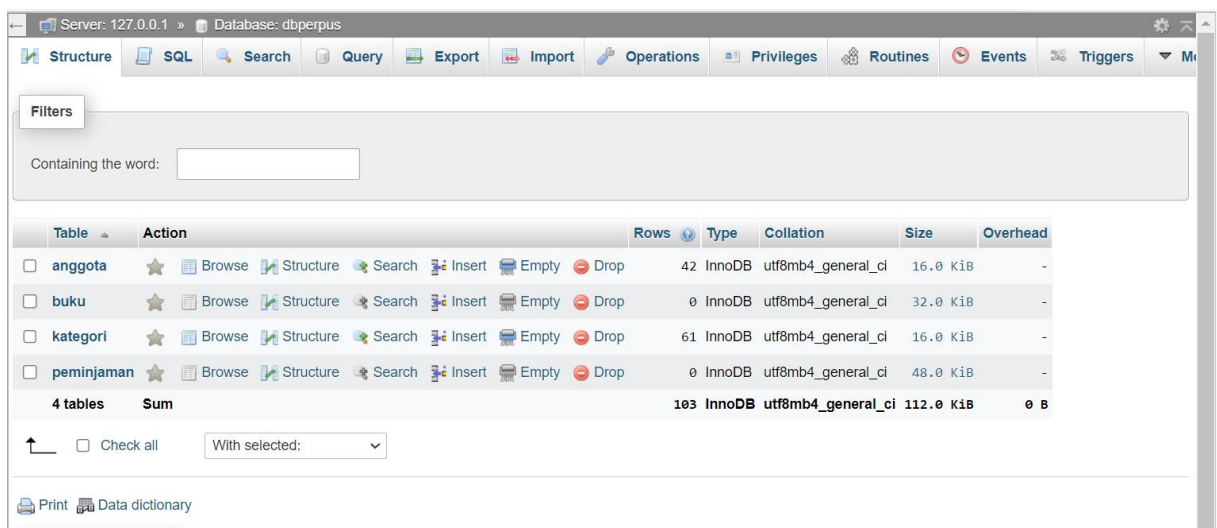


Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> anggota		42	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> buku		0	InnoDB	utf8mb4_general_ci	32.0 KiB	-
<input type="checkbox"/> kategori		61	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> peminjaman		0	InnoDB	utf8mb4_general_ci	48.0 KiB	-
4 tables	Sum	103	InnoDB	utf8mb4_general_ci	112.0 KiB	0 B

☐ Check all With selected:

Print Data dictionary

3.2 Percobaan 2

Mempersiapkan project

1. Buat project baru, beri nama **Perpustakaan**.
2. Pada project explorer, klik kanan pada Libraries → Add Library, pilih MySQL JDBC Driver.
3. Jika tidak ada pilihan MySQL JDBC Driver:
 - a. Buka link <https://dev.mysql.com/downloads/connector/j/>
 - b. Pilih "Platform Independent" pada opsi Operating System
 - c. Download ZIP Archive kemudian extract
 - d. Pada project yang dibuat, klik kanan pada Libraries, kemudian add JAR/Folder... dan pilih file jar yang telah diextract sebelumnya
4. Buat package **frontend** dan **backend**. Cara membuat package adalah, pada project explorer, klik kanan pada Source Packages → New → Java Package, beri nama package nya (frontend, backend).

3.3 Percobaan 3

Membuat class helper untuk mengeksekusi query SQL.

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package backend;
6  import java.sql.*;
7  public class DBHelper {
8      private static Connection koneksi;
9
10     public static void bukaKoneksi() {
11         if(koneksi == null){
12             try{
13                 String url = "jdbc:mysql://localhost:3306/dbperpus";
14                 String user = "root";
15                 String password = "";
16                 DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver());
17                 koneksi = DriverManager.getConnection(url, user, password);
18             }
19             catch (SQLException t){
20                 System.out.println("Error koneksi!");
21             }
22         }
23     }
24
25     public static int insertQueryGetId(String query){
26         bukaKoneksi();
27         int result = -1;
28
29         try{
30             Statement stmt = koneksi.createStatement();
31             stmt.executeUpdate( String:query, 1:Statement.RETURN_GENERATED_KEYS);
32
33             ResultSet rs = stmt.getGeneratedKeys();
34
35             if(rs.next()){
36                 result = rs.getInt( 1:1);
37             }
38             rs.close();
39             stmt.close();
40         }
41         catch(Exception e){
42             e.printStackTrace();
43             result = -1;
44         }
45         return result;
46     }
47
48     public static boolean executeQuery(String query){
49         bukaKoneksi();
50         boolean result = false;
51
52         try{
53             Statement stmt = koneksi.createStatement();
54             stmt.executeUpdate( String:query);
55
56             result = true;
```

```
59     }
60     catch (Exception e){
61         e.printStackTrace();
62     }
63 }
64
65 return result;
66 }
67
68 public static ResultSet selectQuery(String query){
69     bukaKoneksi();
70     ResultSet rs = null;
71
72     try{
73         Statement stmt = koneksi.createStatement();
74         rs = stmt.executeQuery( string,query);
75     }
76     catch (Exception e){
77         e.printStackTrace();
78     }
79     return rs;
80 }
81 }
```

3.4 Percobaan 4

Membuat class **Kategori** untuk menghandle CRUD pada tabel kategori.

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help perustakaan - Apache NetBeans IDE 15 Search (Ctrl+I)
295.5/500.0MB
Start Page X FrmKategori.java X Kategori.java X TestBackend.java X Anggota.java X FrmAnggota.java X DBHelper.java X
Source History
1 /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5 package backend;
6
7 import java.util.ArrayList;
8 import java.sql.*;
9 public class Kategori {
10     private int idkategori;
11     private String nama;
12     private String keterangan;
13
14     public int getIdkategori() {
15         return this.idkategori;
16     }
17
18     public void setIdkategori(int idkategori) {
19         this.idkategori = idkategori;
20     }
21
22     public String getName() {
23         return this.nama;
24     }
25
26     public void setName(String nama) {
27         this.nama = nama;
28     }
29
30     public String getKeterangan() {
31         return this.keterangan;
32     }
33
34     public void setKeterangan(String keterangan) {
35         this.keterangan = keterangan;
36     }
37
38     public Kategori() {
39
40     }
41
42     public Kategori(String nama, String keterangan){
43         this.nama = nama;
44         this.keterangan = keterangan;
45     }
46
47     public static Kategori getById(int id){
48         Kategori kat = null;
49         ResultSet rs = DBHelper.selectQuery("SELECT * FROM " + "WHERE idkategori = " + id + "");
50
51         try {
52             while(rs.next()){
53                 kat = new Kategori();
54                 kat.setIdkategori(rs.getInt( string:"idkategori"));
55                 kat.setName( rs.getString( string:"nama"));
56                 kat.setKeterangan( rs.getString( string:"keterangan"));
57             }
58         }
```

```
58     }
59     catch (Exception e) {
60         e.printStackTrace();
61     }
62 }
63 return kat;
64 }
65
66 public static ArrayList<Kategori> getAll() {
67     ArrayList<Kategori> listKategori = new ArrayList<>();
68     ResultSet rs = DBHelper.selectQuery(query: "SELECT * FROM kategori");
69
70     try {
71         while(rs.next()) {
72             Kategori kat = new Kategori();
73             kat.setIdkategori(rs.getInt( string: "idkategori"));
74             kat.setNama( nama: rs.getString( string: "nama"));
75             kat.setKeterangan( keterangan: rs.getString( string: "keterangan"));
76
77             listKategori.add( e: kat);
78         }
79     }
80     catch (Exception e) {
81         e.printStackTrace();
82     }
83     return listKategori;
84 }
85
86 public static ArrayList<Kategori> search(String keyword) {
87     ArrayList<Kategori> listKategori = new ArrayList<>();
88
89     String query = "SELECT * FROM kategori"
90                 + " WHERE nama LIKE '%" + keyword + "%'"
91                 + " OR keterangan LIKE '%" + keyword + "%'";
92
93     ResultSet rs = DBHelper.selectQuery(query);
94
95     try {
96         while(rs.next()) {
97             Kategori kat = new Kategori();
98             kat.setIdkategori(rs.getInt( string: "idkategori"));
99             kat.setNama( nama: rs.getString( string: "nama"));
100             kat.setKeterangan( keterangan: rs.getString( string: "keterangan"));
101
102             listKategori.add( e: kat);
103         }
104     }
105
106     catch (Exception e) {
107         e.printStackTrace();
108     }
109
110     return listKategori;
111 }
112
113 public void save() {
114     if(this.idkategori == 0) {
115         String query = "INSERT INTO kategori(nama, keterangan) VALUES ("
116                     + " '" + this.nama + "', "
117                     + " '" + this.keterangan + "'"
118                     + " )";
119         this.idkategori = DBHelper.insertQueryGetId(query);
120     }
121     else {
122         String query = "UPDATE kategori SET "
123                     + " nama = '" + this.nama + "'"
124                     + " WHERE idkategori = '" + this.idkategori + "'";
125         DBHelper.executeQuery(query);
126     }
127 }
128
129 public void delete() {
130     String SQL = "DELETE FROM kategori WHERE idkategori = '" + this.idkategori + "'";
131     DBHelper.executeQuery( query: SQL);
132 }
133
134 }
```

3.5 Percobaan 5

Mencoba backed yang sudah dibuat dengan mengoperasikannya lewat frontend berbasis teks (console). Percobaan ini dapat anda skip jika anda telah yakin bahwa backend yang anda buat sudah berfungsi dengan baik.

Code class TestBackend

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help perpustakaan - Apache NetBeans IDE 15
390.5/500.0MB
Start Page x FrmKategori.java x Kategori.java x TestBackend.java x Anggota.java x FrmAnggota.java x DBHelper.java x
Source History
package frontend;

/**
 *
 * @author Hp
 */
import backend.*;

public class TestBackend {
    public static void main(String[] args) {
        Kategori kat1 = new Kategori(1, "Novel", "Koleksi buku novel");
        Kategori kat2 = new Kategori(2, "Referensi", "Buku referensi ilmiah");
        Kategori kat3 = new Kategori(3, "Komik", "Komik anak-anak");

        // test insert
        kat1.save();
        kat2.save();
        kat3.save();

        // test update
        kat2.setKeterangan("Koleksi buku referensi ilmiah");
        kat2.save();

        // test delete
        kat1.delete();

        // test select all
        for (Kategori k : Kategori.getAll()) {
            System.out.println("Nama: " + k.getNama() + ", Ket: " + k.getKeterangan());
        }

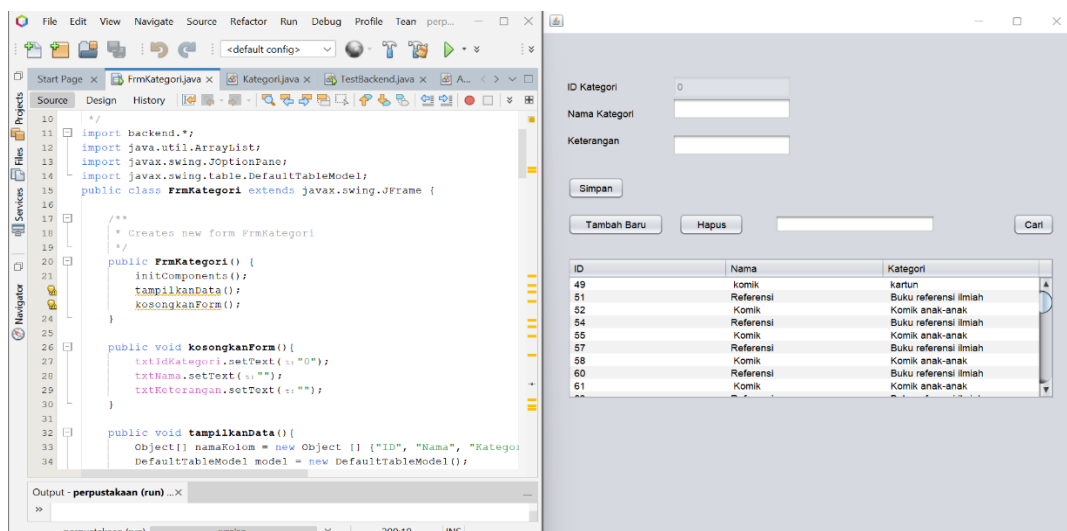
        // test search
        for (Kategori k : Kategori.search("Referensi")) {
            System.out.println("Nama: " + k.getNama() + ", Ket: " + k.getKeterangan());
        }
    }
}
```

3. Jalankan TestBackend dengan klik kanan, Run File. Cocokkan outputnya:

```
run:
Nama: Referensi, Ket: Koleksi buku referensi ilmiah
Nama: Komik, Ket: Komik anak-anak
Nama: Referensi, Ket: Koleksi buku referensi ilmiah
BUILD SUCCESSFUL (total time: 1 second)
```

3.6 Percobaan 6

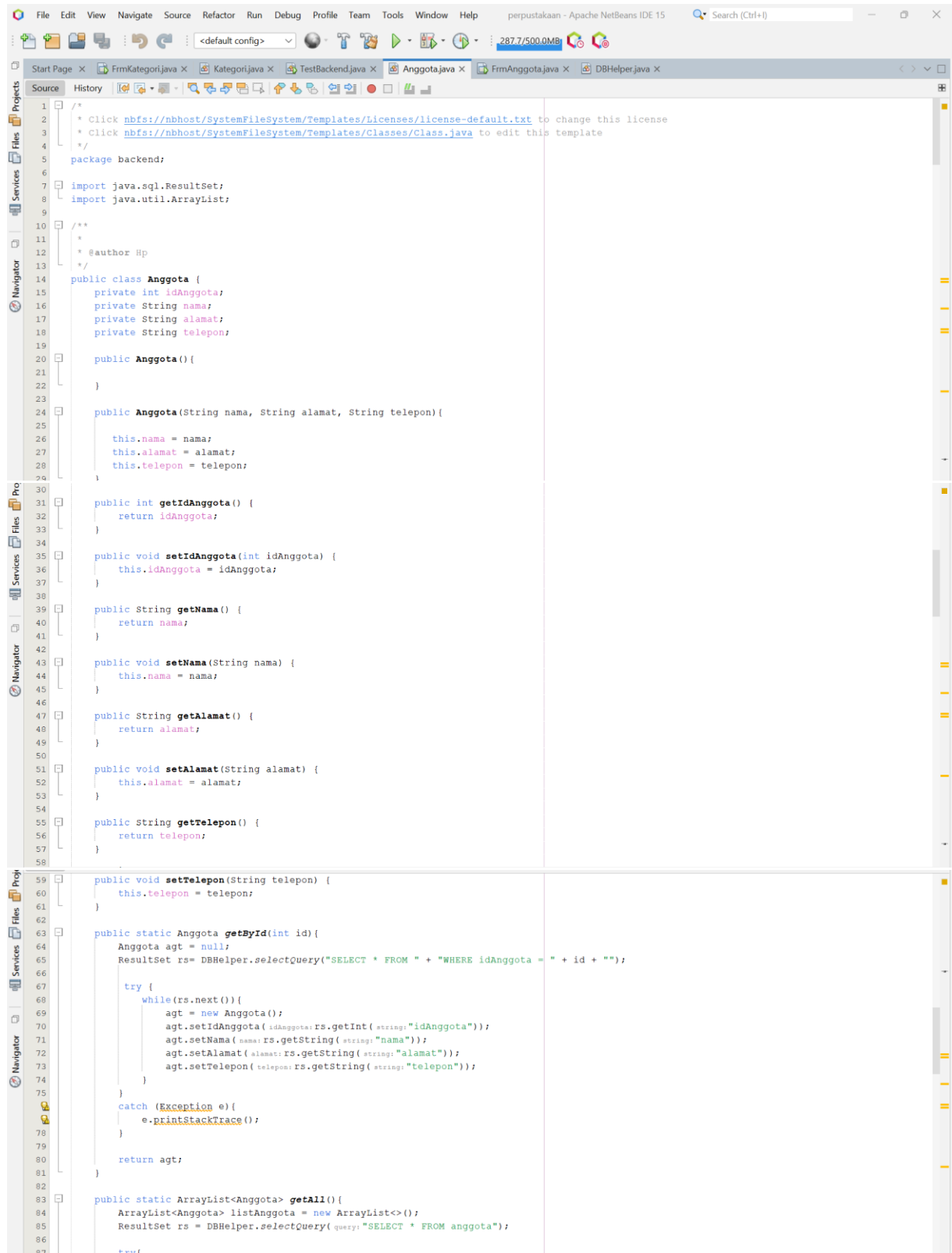
Pada percobaan ini kita akan membuat interface GUI untuk class **Kategori**.



3.7 Percobaan 7

Lakukan hal yang sama untuk data **Anggota**!

1. Buat class **Anggota** pada package **backend**, lengkapi atribut dan method-nya.



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package backend;
6
7  import java.sql.ResultSet;
8  import java.util.ArrayList;
9
10
11  /**
12   * @author Hp
13   */
14  public class Anggota {
15      private int idAnggota;
16      private String nama;
17      private String alamat;
18      private String telepon;
19
20      public Anggota() {
21
22      }
23
24      public Anggota(String nama, String alamat, String telepon) {
25
26          this.nama = nama;
27          this.alamat = alamat;
28          this.telepon = telepon;
29      }
30
31      public int getIdAnggota() {
32          return idAnggota;
33      }
34
35      public void setIdAnggota(int idAnggota) {
36          this.idAnggota = idAnggota;
37      }
38
39      public String getName() {
40          return nama;
41      }
42
43      public void setName(String nama) {
44          this.nama = nama;
45      }
46
47      public String getAlamat() {
48          return alamat;
49      }
50
51      public void setAlamat(String alamat) {
52          this.alamat = alamat;
53      }
54
55      public String getTelepon() {
56          return telepon;
57      }
58
59      public void setTelepon(String telepon) {
60          this.telepon = telepon;
61      }
62
63      public static Anggota getById(int id) {
64          Anggota agt = null;
65          ResultSet rs = DBHelper.selectQuery("SELECT * FROM " + "WHERE idAnggota = " + id + "");
66
67          try {
68              while(rs.next()) {
69                  agt = new Anggota();
70                  agt.setIdAnggota(rs.getInt("idAnggota"));
71                  agt.setName(rs.getString("nama"));
72                  agt.setAlamat(rs.getString("alamat"));
73                  agt.setTelepon(rs.getString("telepon"));
74              }
75          } catch (Exception e) {
76              e.printStackTrace();
77          }
78
79          return agt;
80      }
81
82      public static ArrayList<Anggota> getAll() {
83          ArrayList<Anggota> listAnggota = new ArrayList<>();
84          ResultSet rs = DBHelper.selectQuery("SELECT * FROM anggota");
85
86          try {
87
```

```

88         while(rs.next()){
89             Anggota agt = new Anggota();
90             agt = new Anggota();
91             agt.setIdAnggota(rs.getInt( string:"idAnggota"));
92             agt.setNama( nama: rs.getString( string:"nama"));
93             agt.setAlamat( alamat: rs.getString( string:"alamat"));
94             agt.setTelepon( telepon: rs.getString( string:"telepon"));
95
96             listAnggota.add( agt);
97         }
98     }
99     catch (Exception e){
100         e.printStackTrace();
101     }
102     return listAnggota;
103 }
104
105 public static ArrayList<Anggota> search(String keyword){
106     ArrayList<Anggota> listAnggota = new ArrayList<>();
107
108     String query = "SELECT * FROM anggota"
109                 + " WHERE nama LIKE '%" + keyword + "%'"
110                 + " OR alamat LIKE '%" + keyword + "%'"
111                 + " OR telepon LIKE '%" + keyword + "%'";
112
113     ResultSet rs = DBHelper.selectQuery(query);
114
115     try {
116         while(rs.next()){
117             Anggota agt = new Anggota();
118             agt.setIdAnggota(rs.getInt( string:"idAnggota"));
119             agt.setNama( nama: rs.getString( string:"nama"));
120             agt.setAlamat( alamat: rs.getString( string:"alamat"));
121             agt.setTelepon( telepon: rs.getString( string:"telepon"));
122             listAnggota.add( agt);
123         }
124     }
125     catch (Exception e){
126         e.printStackTrace();
127     }
128     return listAnggota;
129 }
130
131 public void save(){
132     if(this.idAnggota == 0){
133         String query = "INSERT INTO anggota(nama, alamat, telepon) VALUES ("
134                     + " ' " + this.nama + ", "
135                     + " ' " + this.alamat + ", " + this.telepon + "' ) ";
136         this.idAnggota = DBHelper.insertQueryGetId(query);
137     }
138     else{
139         String query = "UPDATE anggota SET"
140                     + " nama = ' " + this.nama + ", "
141                     + " alamat = ' " + this.alamat + ", "
142                     + " telepon = ' " + this.telepon + "' "
143                     + " WHERE idAnggota = ' " + this.idAnggota + "'";
144         DBHelper.executeQuery(query);
145     }
146 }
147
148 public void delete(){
149     String SQL = "DELETE FROM anggota WHERE idAnggota = ' " + this.idAnggota + "' ";
150     DBHelper.executeQuery( query:SQL);
151 }
152
153 }

```

2. Lakukan test pada class TestBackend pada package **frontend**.

```

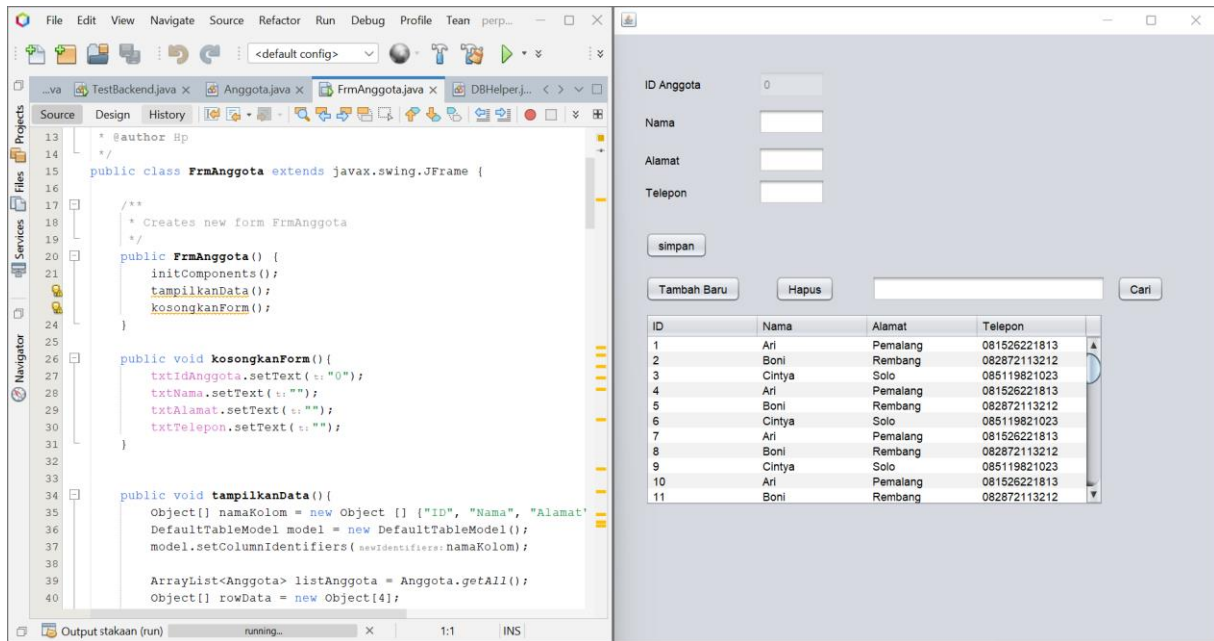
Output - perpustakaan (run) ✖
Nama : Ari, Alamat : Pemalang, Telepon: 081526221813
Nama : Boni, Alamat : Rembang, Telepon: 082872113212
Nama : Cintya, Alamat : Solo, Telepon: 085119821023

```

3.8 Percobaan 8

Buat form untuk data **Anggota**.

1. Buat **FrmAnggota** pada package **frontend** dan lengkapi komponen, method serta event-nya.

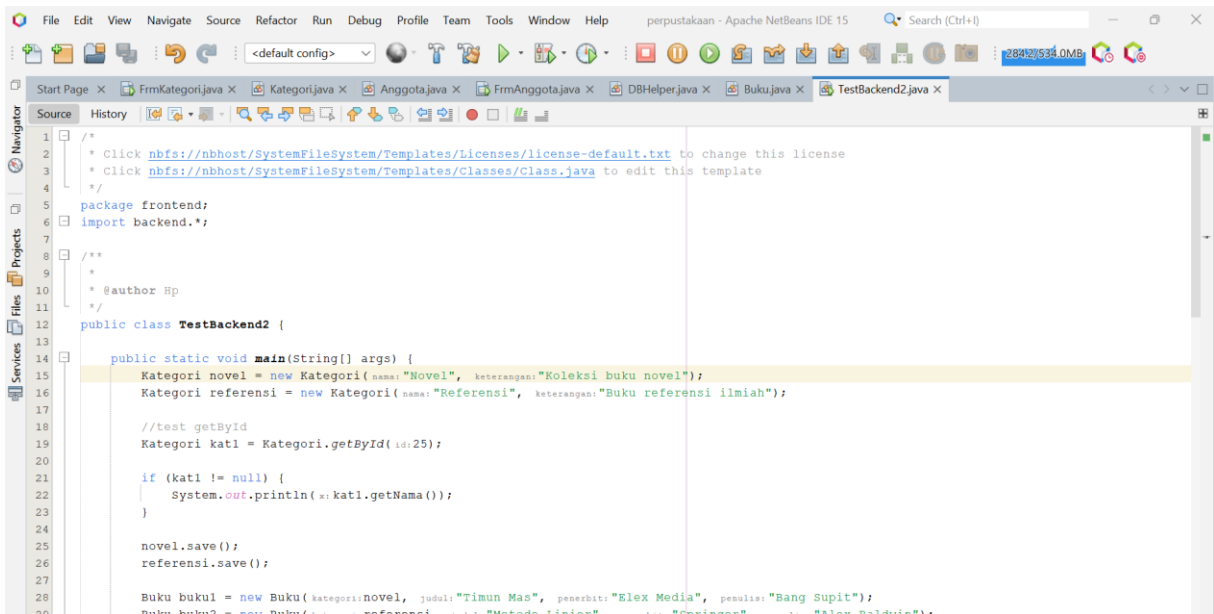


3.9 Percobaan 9

Untuk data **Buku**, caranya kurang lebih sama seperti data Kategori dan Anggota. Hanya saja yang berbeda adalah:

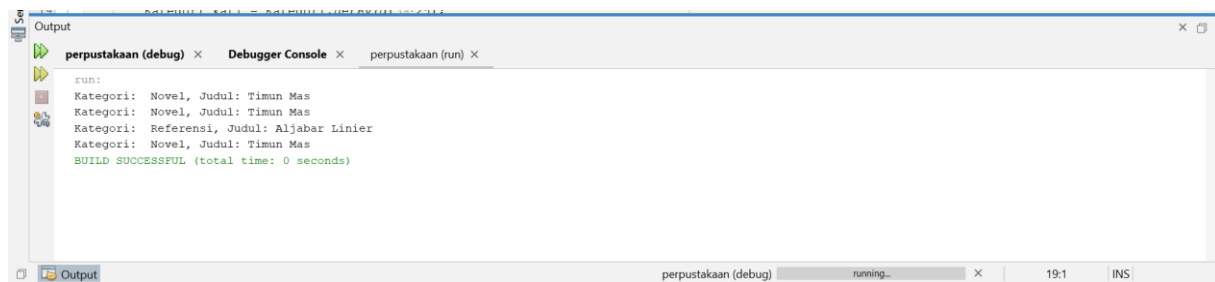
- Pemanggilan `getKategori().getIdKategori()` pada query insert dan update untuk mengeset `idkategori` pada tabel buku
- Query select yang melibatkan join table pada method `getById()`, `getAll()` dan `search()`.

Kode lengkap class Buku dapat anda lihat di **Lampiran 1**. Untuk test buku pada **frontend**, bisa anda lihat di **Lampiran 2**.



```
30     Buku buku3 = new Buku(kategori:novel, judul:"Bintang Terang", penerbit:"Erlangga", penulis:"Mat Sewoot");
31
32     // test insert
33     buku1.save();
34     buku2.save();
35     buku3.save();
36
37     // test update
38     buku2.setJudul(judul:"Aljabar Linier");
39     buku2.save();
40
41     //test getById
42     Buku buku = Buku.getById(id:1);
43
44     if (buku != null) {
45         System.out.println("Kategori: " + buku.getKategori().getNama() + ", Judul: "
46             + buku.getJudul());
47     }
48
49     // test delete
50     buku3.delete();
51
52     // test select all
53     for (Buku b : Buku.getAll()) {
54         System.out.println("Kategori: " + b.getKategori().getNama() + ", Judul: "
55             + b.getJudul());
56     }
57
58     // test search
59     for (Buku b : Buku.search(keyword:"timun")) {
60         System.out.println("Kategori: " + b.getKategori().getNama() + ", Judul: "
61             + b.getJudul());
62     }
63 }
64
65 }
```

Hasil running :



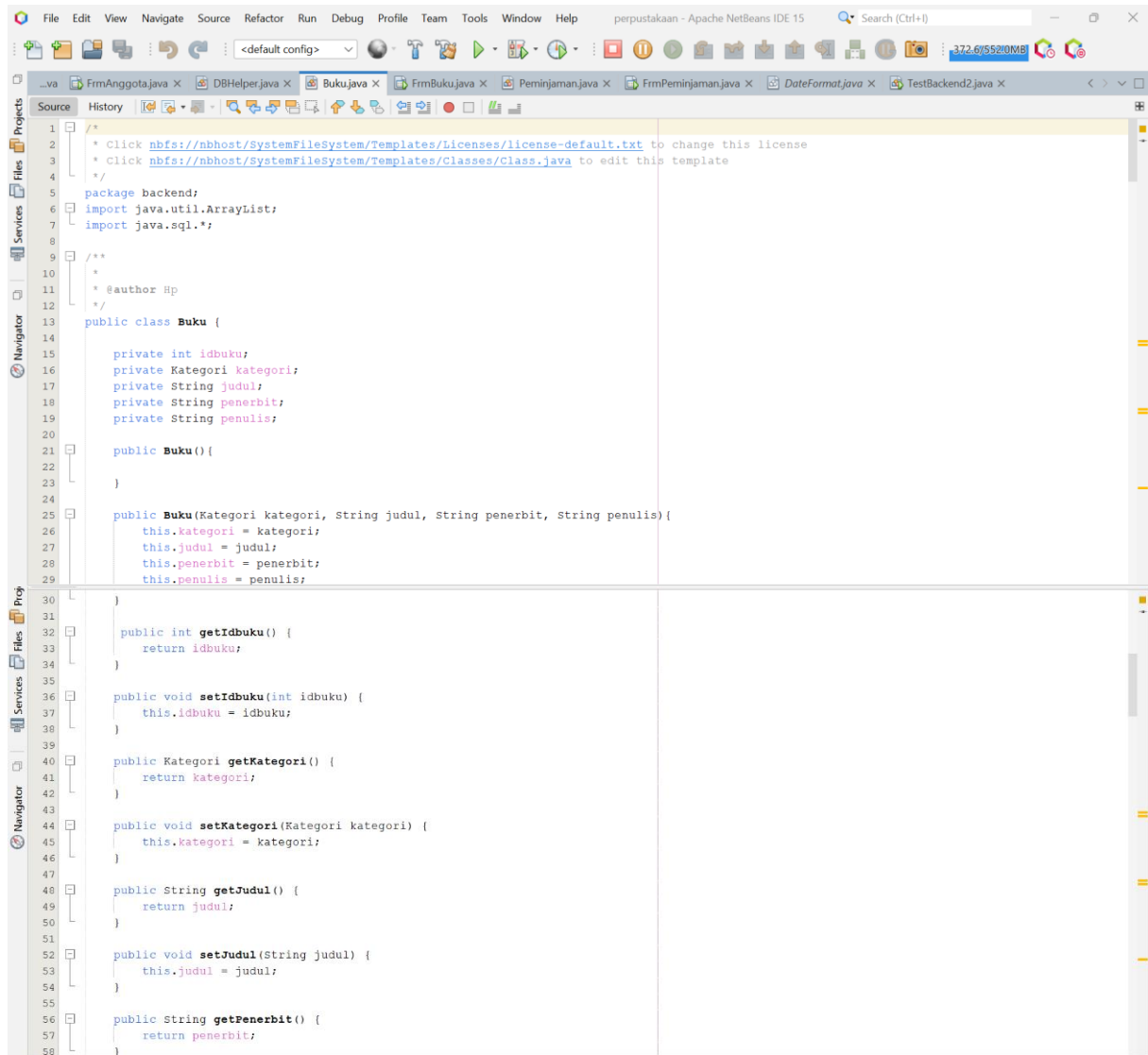
```
Output
perpustakaan (debug) x Debugger Console x perpustakaan (run) x

run:
Kategori: Novel, Judul: Timun Mas
Kategori: Novel, Judul: Timun Mas
Kategori: Referensi, Judul: Aljabar Linier
Kategori: Novel, Judul: Timun Mas
BUILD SUCCESSFUL (total time: 0 seconds)
```

3.10 Percobaan 10

Membuat GUI untuk data Buku, yang dilengkapi dengan combo box untuk memilih kategori yang terhubung dengan tabel kategori.

Code class Buku



```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package backend;
6  import java.util.ArrayList;
7  import java.sql.*;
8
9  /**
10   *
11   * @author Hp
12   */
13  public class Buku {
14
15      private int idbuku;
16      private Kategori kategori;
17      private String judul;
18      private String penerbit;
19      private String penulis;
20
21      public Buku() {
22
23      }
24
25      public Buku(Kategori kategori, String judul, String penerbit, String penulis) {
26          this.kategori = kategori;
27          this.judul = judul;
28          this.penerbit = penerbit;
29          this.penulis = penulis;
30      }
31
32      public int getIdbuku() {
33          return idbuku;
34      }
35
36      public void setIdbuku(int idbuku) {
37          this.idbuku = idbuku;
38      }
39
40      public Kategori getKategori() {
41          return kategori;
42      }
43
44      public void setKategori(Kategori kategori) {
45          this.kategori = kategori;
46      }
47
48      public String getJudul() {
49          return judul;
50      }
51
52      public void setJudul(String judul) {
53          this.judul = judul;
54      }
55
56      public String getPenerbit() {
57          return penerbit;
58      }
59  }
```

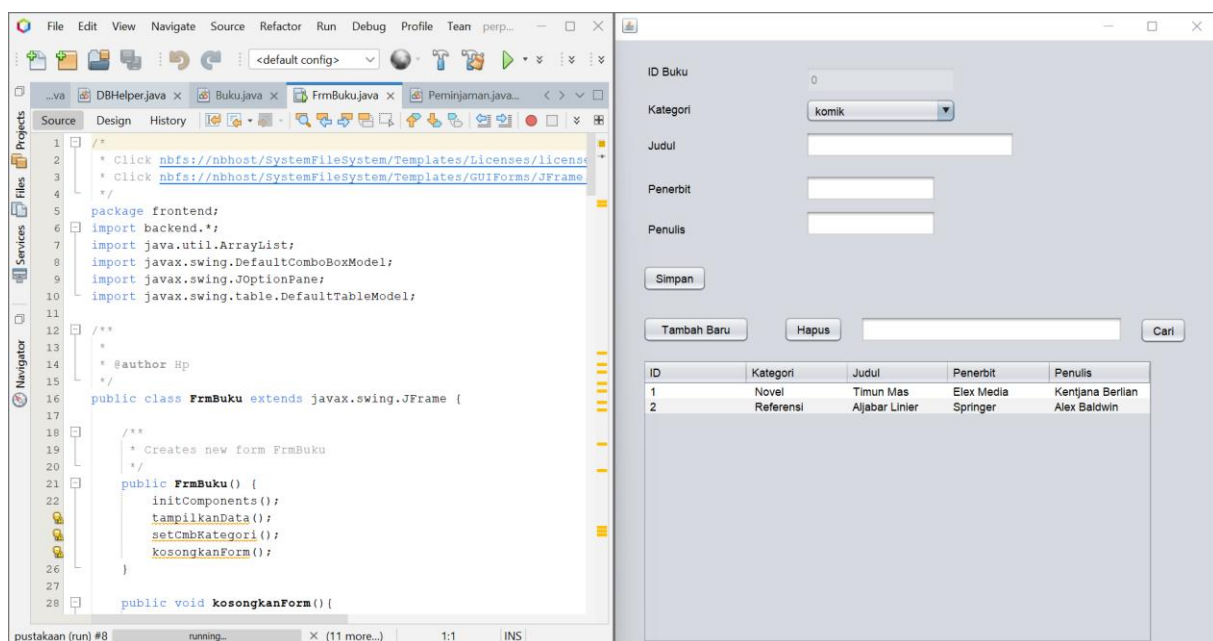
```
59
60 public void setPenerbit(String penerbit) {
61     this.penerbit = penerbit;
62 }
63
64 public String getPenulis() {
65     return penulis;
66 }
67
68 public void setPenulis(String penulis) {
69     this.penulis = penulis;
70 }
71
72 public static ArrayList<Buku> getAll() {
73     ArrayList<Buku> listBuku = new ArrayList();
74
75     String query = "SELECT buku.*, kategori.nama, kategori.keterangan"
76                   + " FROM buku"
77                   + " LEFT JOIN kategori ON buku.idkategori = kategori.idkategori";
78
79     ResultSet rs = DBHelper.selectQuery(query);
80
81     try{
82         while(rs.next()) {
83             Kategori kat = new Kategori();
84             kat.setIdkategori(rs.getInt( string:"idkategori"));
85             kat.setNama( nama:rs.getString( string:"nama"));
86             kat.setKeterangan( keterangan:rs.getString( string:"keterangan"));
87
88             Buku buku = new Buku();
89             buku.setIdbuku(rs.getInt( string:"idbuku"));
90             buku.setKategori( kategori:kat);
91             buku.setJudul( judul:rs.getString( string:"judul"));
92             buku.setPenerbit( penerbit:rs.getString( string:"penerbit"));
93             buku.setPenulis( penulis:rs.getString( string:"penulis"));
94
95             listBuku.add( e:buku);
96         }
97     } catch(Exception e){
98         e.printStackTrace();
99     }
100
101     return listBuku;
102 }
103
104 public static Buku getById(int id){
105     Buku buku = null;
106
107     String query = "SELECT buku.*, kategori.nama, kategori.keterangan"
108                   + " FROM buku"
109                   + " LEFT JOIN kategori ON buku.idkategori = kategori.idkategori"
110                   + " WHERE idbuku = " + id;
111
112     ResultSet rs = DBHelper.selectQuery(query);
113
114     try{
115         while(rs.next()){
116             Kategori kat = new Kategori();
117             kat.setIdkategori(rs.getInt( string:"idkategori"));
118             kat.setNama( nama:rs.getString( string:"nama"));
119             kat.setKeterangan( keterangan:rs.getString( string:"keterangan"));
120
121             buku = new Buku();
122             buku.setIdbuku( idbuku:rs.getInt( string:"idbuku"));
123             buku.setKategori( kategori:kat);
124             buku.setJudul( judul:rs.getString( string:"judul"));
125             buku.setPenerbit( penerbit:rs.getString( string:"penerbit"));
126             buku.setPenulis( penulis:rs.getString( string:"penulis"));
127
128         }
129     } catch(Exception e){
130         e.printStackTrace();
131     }
132
133     return buku;
134 }
135
136 public static ArrayList<Buku> search (String keyword){
137     ArrayList<Buku> listBuku = new ArrayList();
138
139     String query = "SELECT buku.*, kategori.nama, kategori.keterangan"
140                   + " FROM buku"
141                   + " LEFT JOIN kategori ON buku.idkategori = kategori.idkategori"
```

```

144         + " WHERE judul LIKE '%" + keyword + "%'"
145         + " OR penerbit LIKE '%" + keyword + "%'"
146         + " OR penulis LIKE '%" + keyword + "%'";
147
148     ResultSet rs = DBHelper.selectQuery(query);
149
150     try{
151         while(rs.next()){
152             Kategori kat = new Kategori();
153             kat.setIdkategori(rs.getInt( string: "idkategori"));
154             kat.setNama( nama: rs.getString( string: "nama"));
155             kat.setKeterangan( keterangan: rs.getString( string: "keterangan"));
156
157             Buku buku = new Buku();
158             buku.setIdbuku( idbuku: rs.getInt( string: "idbuku"));
159             buku.setKategori( kategori: kat);
160             buku.setJudul( judul: rs.getString( string: "judul"));
161             buku.setPenerbit( penerbit: rs.getString( string: "penerbit"));
162             buku.setPenulis( penulis: rs.getString( string: "penulis"));
163
164             listBuku.add( buku);
165         }
166     }
167
168     catch(Exception e){
169         e.printStackTrace();
170     }
171
172     return listBuku;
173 }
174
175 public void save(){
176     if(this.idbuku == 0){
177         String query = "INSERT INTO buku(idkategori, judul, penerbit, penulis) VALUES("
178             + " " + this.kategori.getIdkategori() + ", "
179             + " " + this.judul + ", "
180             + " " + this.penerbit + ", "
181             + " " + this.idbuku + ") ";
182
183         this.idbuku = DBHelper.insertQueryGetId(query);
184     }
185     else{
186         String query = "UPDATE buku SET"
187             + " idkategori = " + this.kategori.getIdkategori() + ", "
188             + " judul = " + this.judul + ", "
189             + " penerbit = " + this.penerbit + ", "
190             + " penulis = " + this.penulis + ", "
191             + " WHERE idbuku = " + this.idbuku + " ";
192
193         DBHelper.executeQuery(query);
194     }
195 }
196
197 public void delete(){
198     String query = "DELETE FROM buku WHERE idbuku = " + this.idbuku;
199     DBHelper.executeQuery(query);
200 }
201
202 }
203

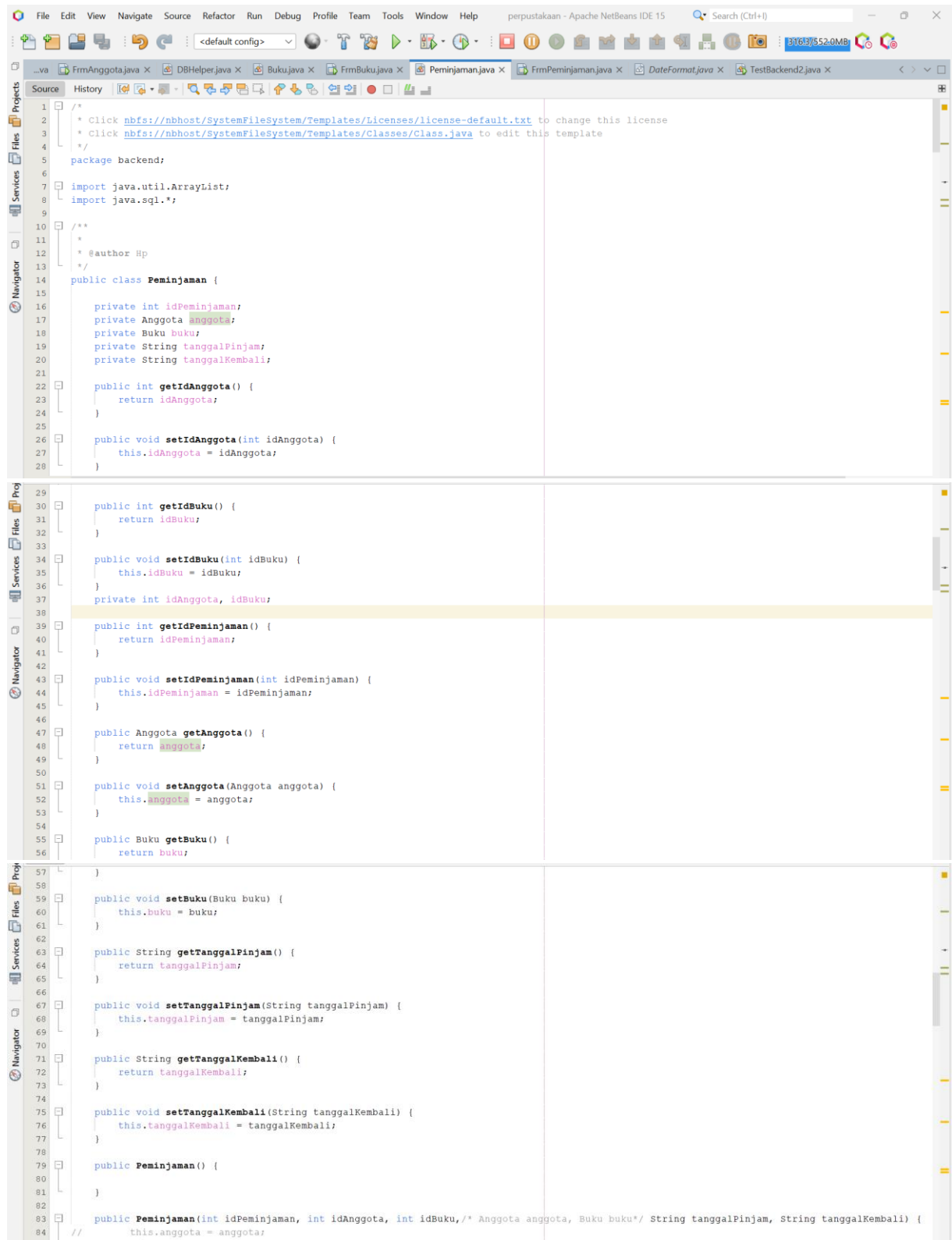
```

1. Pada package **frontend**, buat JFrame FrmBuku. Susun formnya sebagai berikut:



4. Tugas

1. Buatlah class Peminjaman.



```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package backend;
6
7  import java.util.ArrayList;
8  import java.sql.*;
9
10 /**
11  *
12  * @author Hp
13  */
14 public class Peminjaman {
15
16     private int idPeminjaman;
17     private Anggota anggota;
18     private Buku buku;
19     private String tanggalPinjam;
20     private String tanggalKembali;
21
22     public int getIdAnggota() {
23         return idAnggota;
24     }
25
26     public void setIdAnggota(int idAnggota) {
27         this.idAnggota = idAnggota;
28     }
29
30     public int getIdBuku() {
31         return idBuku;
32     }
33
34     public void setIdBuku(int idBuku) {
35         this.idBuku = idBuku;
36     }
37     private int idAnggota, idBuku;
38
39     public int getIdPeminjaman() {
40         return idPeminjaman;
41     }
42
43     public void setIdPeminjaman(int idPeminjaman) {
44         this.idPeminjaman = idPeminjaman;
45     }
46
47     public Anggota getAnggota() {
48         return anggota;
49     }
50
51     public void setAnggota(Anggota anggota) {
52         this.anggota = anggota;
53     }
54
55     public Buku getBuku() {
56         return buku;
57     }
58
59     public void setBuku(Buku buku) {
60         this.buku = buku;
61     }
62
63     public String getTanggalPinjam() {
64         return tanggalPinjam;
65     }
66
67     public void setTanggalPinjam(String tanggalPinjam) {
68         this.tanggalPinjam = tanggalPinjam;
69     }
70
71     public String getTanggalKembali() {
72         return tanggalKembali;
73     }
74
75     public void setTanggalKembali(String tanggalKembali) {
76         this.tanggalKembali = tanggalKembali;
77     }
78
79     public Peminjaman() {
80
81     }
82
83     public Peminjaman(int idPeminjaman, int idAnggota, int idBuku, /* Anggota anggota, Buku buku*/ String tanggalPinjam, String tanggalKembali) {
84         //
85         this.anggota = anggota;
```

```

85 //      this.buku = buku;
86      this.idPeminjaman = idPeminjaman;
87      this.idAnggota = idAnggota;
88      this.idBuku = idBuku;
89      this.tanggalPinjam = tanggalPinjam;
90      this.tanggalKembali = tanggalKembali;
91  }
92
93  public static ArrayList<Peminjaman> getAll() {
94      ArrayList<Peminjaman> listPeminjaman = new ArrayList<>();
95
96      String query = "SELECT * FROM peminjaman";
97
98      ResultSet rs = DBHelper.selectQuery(query);
99
100     try {
101         while (rs.next()) {
102             Peminjaman peminjaman = new Peminjaman();
103             peminjaman.setIdPeminjaman(rs.getInt("idPeminjaman"));
104             peminjaman.setIdAnggota(rs.getInt("idAnggota"));
105             peminjaman.setIdBuku(rs.getInt("idBuku"));
106             peminjaman.setTanggalPinjam(rs.getString("tanggalPinjam"));
107             peminjaman.setTanggalKembali(rs.getString("tanggalKembali"));
108
109             listPeminjaman.add(peminjaman);
110         }
111     } catch (Exception e) {
112         e.printStackTrace();
113     }
114
115     return listPeminjaman;
116 }
117
118 public static Peminjaman getById(int id) {
119     Peminjaman peminjaman = null;
120
121     String query = "SELECT * FROM peminjaman WHERE idpeminjaman = " + id;
122
123     ResultSet rs = DBHelper.selectQuery(query);
124
125     try {
126         while (rs.next()) {
127             peminjaman = new Peminjaman();
128             peminjaman.setIdPeminjaman(rs.getInt("idPeminjaman"));
129             peminjaman.setIdAnggota(rs.getInt("idAnggota"));
130             peminjaman.setIdBuku(rs.getInt("idBuku"));
131             peminjaman.setTanggalPinjam(rs.getString("tanggalPinjam"));
132             peminjaman.setTanggalKembali(rs.getString("tanggalKembali"));
133         }
134     } catch (Exception e) {
135         e.printStackTrace();
136     }
137
138     return peminjaman;
139 }
140

```

```

141 public static ArrayList<Peminjaman> search(String keyword) {
142     ArrayList<Peminjaman> listPeminjaman = new ArrayList<>();
143
144     String query = "SELECT * FROM peminjaman p LEFT JOIN anggota a ON p.idanggota = a.idanggota LEFT JOIN buku b ON b.idbuku = p.idbuku WHERE
145     + keyword + '%' OR b.judul LIKE '%" + keyword + '%' OR tanggalpinjam LIKE '%" + keyword
146     + '%' OR tanggalkembali LIKE '%" + keyword + '%';
147
148     ResultSet rs = DBHelper.selectQuery(query);
149
150     try {
151         while (rs.next()) {
152             Peminjaman peminjaman = new Peminjaman();
153             peminjaman.setIdPeminjaman(rs.getInt("idPeminjaman"));
154             peminjaman.setIdAnggota(rs.getInt("idAnggota"));
155             peminjaman.setIdBuku(rs.getInt("idBuku"));
156             peminjaman.setTanggalPinjam(rs.getString("tanggalPinjam"));
157             peminjaman.setTanggalKembali(rs.getString("tanggalKembali"));
158
159             listPeminjaman.add(peminjaman);
160         }
161     } catch (Exception e) {
162         e.printStackTrace();
163     }
164
165     return listPeminjaman;
166 }
167
168 public void save() {
169
170     if (this.getIdPeminjaman() == 0) {
171         String query = "INSERT INTO peminjaman (idanggota, idbuku, tanggalpinjam, tanggalkembali) VALUES ("
172         + this.getIdAnggota() + ", " + this.getIdBuku() + ", " + this.getTanggalPinjam() + ", "
173         + this.getTanggalKembali() + ")";
174
175         this.getIdPeminjaman() = DBHelper.insertQueryGetId(query);
176     } else {
177         String query = "UPDATE peminjaman SET idanggota = " + this.getIdAnggota() + ", idbuku = "
178         + this.getIdBuku() + ", tanggalpinjam = " + this.getTanggalPinjam() + ", tanggalkembali = "
179         + this.getTanggalKembali() + " WHERE idpeminjaman = " + this.getIdPeminjaman();
180
181         DBHelper.executeQuery(query);
182     }
183
184     public void delete() {
185         String query = "DELETE FROM peminjaman WHERE idpeminjaman = " + this.getIdPeminjaman();
186
187         DBHelper.executeQuery(query);
188     }
189
190 }
191

```

2. Buatlah form **FrmPeminjaman** dan susun sebagai berikut:

ID

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

Title 1	Title 2	Title 3	Title 4

3. Atur kode program agar dapat menangani transaksi peminjaman dan pengembalian.

Note:

Pada textbox ID Anggota, pengguna tinggal memasukkan ID anggota, kemudian menekan tombol Cari. Jika ketemu, maka label **"Nama Anggota"** yang ada di samping tombol Cari tersebut akan menampilkan nama anggota dari ID yang dimasukkan tadi. Begitu juga dengan ID Buku.

Hasil running FrmPeminjaman

