LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL 5
LATIHAN EXCEPTION HANDLING

Oleh :

Teuku Bintang Hadi Negara

2311103139

S1SI-07-C

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS REKAYASA INDUSTRI

UNIVERSITAS TELKOM PURWOKERTO

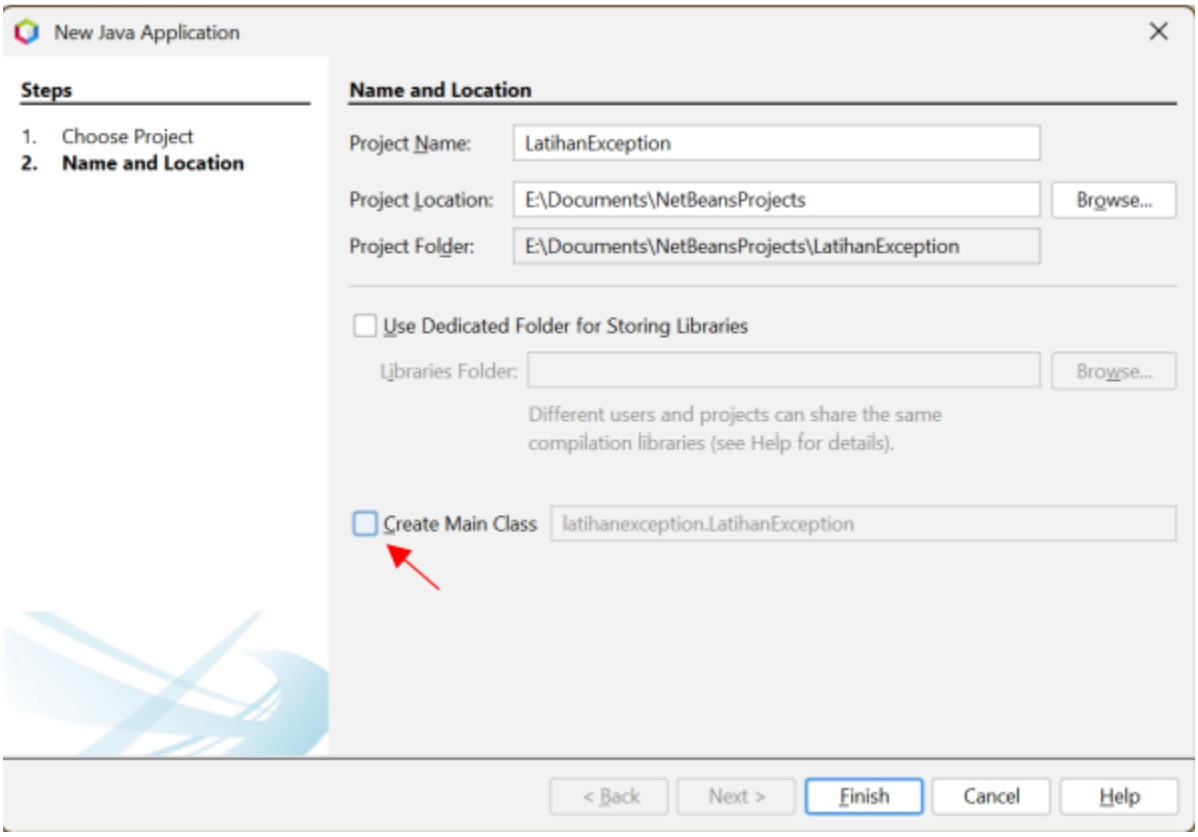
2024

A. Studi Kasus

Sistem Informasi Perpustakaan – Input Buku

B. GUIDED

1. Buat project baru pada Netbeans. Pada saat membuat project, hilangkan centang pada Create Main Class



2. Tambahkan package baru dengan nama perpustakaan dengan cara klik kanan pada Source Packages → New → Java Package...
3. Pada package perpustakaan, buat Class baru dengan nama Buku

Buku.java

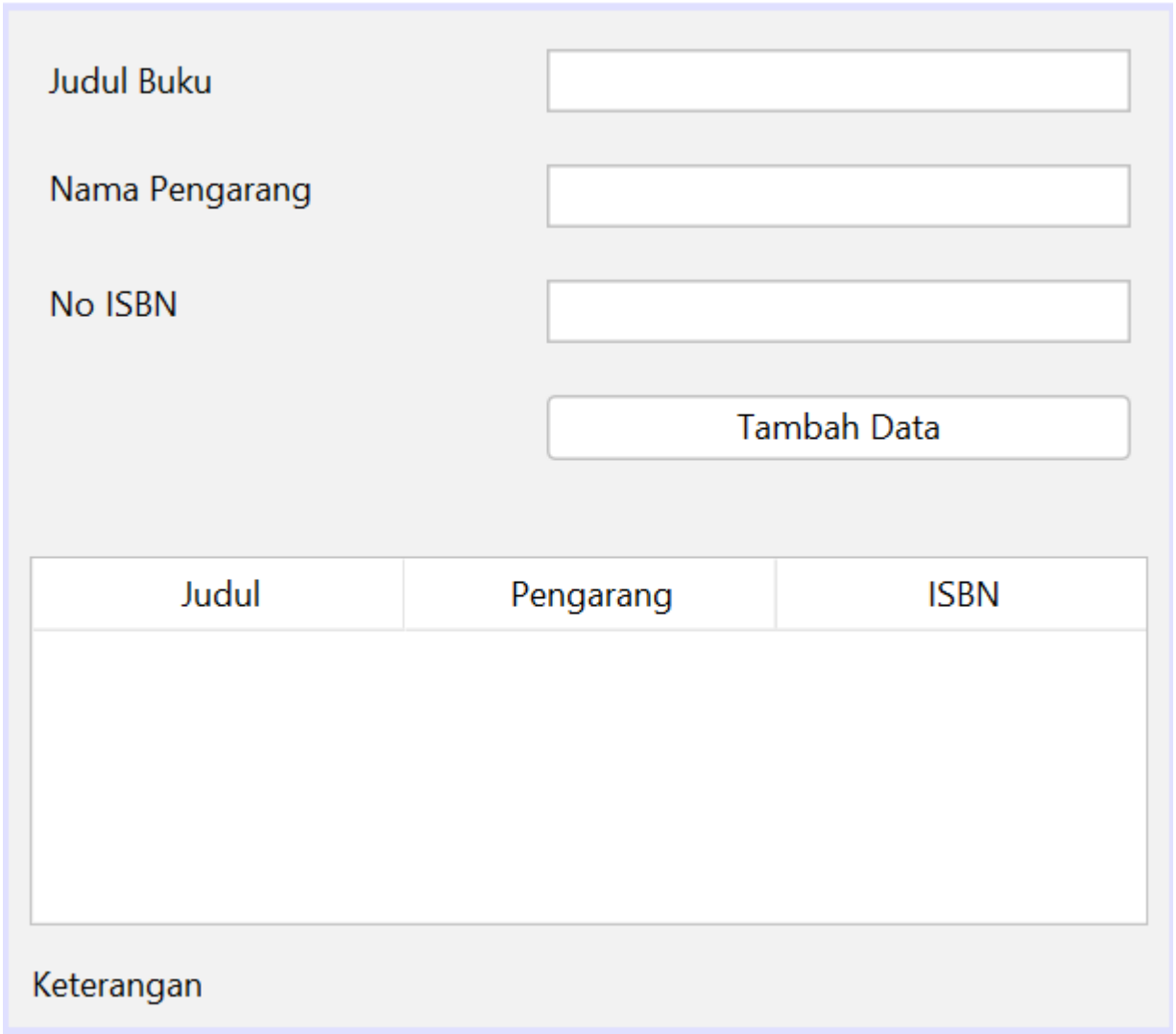
```
package Perpustakaan;


/**
 *
 * @author Teuku Bintang Hadi Negara
```

```
 */
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;

    public Buku(String judul, String pengarang, String isbn){
        this.judul = judul;
        this.pengarang = pengarang;
        this.isbn = isbn;
    }

    public String getJudul() {
        return judul;
    }

    public String getPengarang() {
        return pengarang;
    }

    public String getIsbn() {
        return isbn;
    }
}
```

Penjelasan : Kode di atas mendefinisikan sebuah kelas Java bernama Buku yang berada dalam paket Perpustakaan. Kelas ini memiliki tiga atribut privat: judul, pengarang, dan isbn, yang masing-masing merepresentasikan informasi sebuah buku. Konstruktor pada kelas ini digunakan untuk menginisialisasi nilai-nilai atribut tersebut saat objek dari kelas Buku dibuat. Selain itu, terdapat tiga metode getter (getJudul, getPengarang, dan getIsbn) yang memungkinkan akses ke nilai masing-masing atribut secara aman dari luar kelas. Struktur ini

mengikuti prinsip enkapsulasi untuk menjaga keamanan data dalam pemrograman berorientasi objek.

4. Tambahkan tampilan program untuk input data dengan cara klik kanan pada package perpustakaan → New →JFrame Form... , beri nama InputBuku
5. Buat tampilan seperti di bawah ini:



6. Klik 2x pada tombol Tambah Data, tambahkan kode berikut

```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    String judul = txtJudul.getText();
    String pengarang = txtPengarang.getText();
```

```
    String isbn = txtIsbn.getText();

    // Membuat objek Buku baru
    Buku buku = new Buku(judul, pengarang, isbn);

    // Menambah data buku ke tabel
    DefaultTableModel data = (DefaultTableModel)
    tblBuku.getModel();

    data.addRow(new Object[]{buku.getJudul(), buku.getPengarang(),
    buku.getIsbn()});

    // Reset form input
    txtJudul.setText("");
    txtPengarang.setText("");
    txtIsbn.setText("");
}
```

Penjelasan : Kode di atas adalah metode yang dijalankan ketika tombol submit ditekan dalam aplikasi berbasis GUI. Metode ini mengambil input dari tiga bidang teks (txtJudul, txtPengarang, dan txtIsbn) dan menggunakannya untuk membuat objek Buku baru. Setelah objek buku dibuat, data buku ditambahkan ke tabel menggunakan model tabel (DefaultTableModel). Data yang dimasukkan meliputi judul, pengarang, dan ISBN buku. Setelah data berhasil ditambahkan, metode ini mereset isi bidang teks menjadi kosong agar siap untuk input berikutnya. Hal ini memungkinkan pengguna untuk memasukkan data baru tanpa harus menghapus input sebelumnya secara manual.

7. Jalankan project lalu coba tambahkan data, maka akan muncul data pada tabel. Namun ketika form kosong lalu klik tambah data, maka akan ada data kosong yang masuk pada tabel. Hal ini bisa diantisipasi dengan membuat user defined exception
8. Buat sebuah class baru dengan nama ValidasiInputException

   ValidasiInputException.java

```
package Perpustakaan;
```

```
/**
 *
 * @author Teuku Bintang Hadi Negara
 */
public class ValidasiInputException extends Exception {
    public ValidasiInputException(String message) {
        super(message);
    }
}
```

Penjelasan : Kode di atas mendefinisikan kelas ValidasiInputException yang berada dalam paket Perpustakaan. Kelas ini adalah turunan dari kelas bawaan Java Exception, yang digunakan untuk membuat jenis pengecualian khusus. Konstruktor kelas ini menerima parameter berupa pesan (message) yang akan diteruskan ke konstruktor induk (super). Dengan membuat pengecualian kustom ini, pengembang dapat memberikan pesan kesalahan yang spesifik dan lebih informatif ketika terjadi masalah validasi input dalam aplikasi. Kelas ini dirancang untuk digunakan dalam mekanisme penanganan kesalahan (error handling) yang lebih terstruktur.

9. Ubah kode pada Class buku

   Buku.java

```
package Perpustakaan;


/**
 *
 * @author Teuku Bintang Hadi Negara
 */
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;


    public Buku(String judul, String pengarang, String isbn) throws
ValidasiInputException {
```

```
    if (judul.isEmpty()) {
        throw new ValidasiInputException("Judul buku tidak boleh kosong!");
    }

    if (pengarang.isEmpty()) {
        throw new ValidasiInputException("Pengarang tidak boleh kosong!");
    }

    if (!isbn.matches("\\d{13}")) {
        throw new ValidasiInputException("ISBN harus 13 digit angka!");
    }

    this.judul = judul;
    this.pengarang = pengarang;
    this.isbn = isbn;
    }
    public String getJudul() {
    return judul;
    }
    public String getPengarang() {
    return pengarang;
    }
    public String getIsbn() {
    return isbn;
    }
}
```

Penjelasan : Pada kode yang ditambahkan, konstruktor kelas Buku kini dilengkapi dengan validasi input. Sebelum menyimpan nilai atribut judul, pengarang, dan isbn, dilakukan pengecekan untuk memastikan bahwa judul dan pengarang tidak kosong, serta isbn harus terdiri dari 13 digit angka. Jika salah satu kondisi ini tidak terpenuhi, maka akan dilemparkan pengecualian ValidasiInputException dengan pesan yang sesuai.

Validasi ini memastikan bahwa data yang dimasukkan valid sebelum objek Buku dapat dibuat dan disimpan, meningkatkan keandalan aplikasi.

10. Klik 2x pada tombol Tambah Data, ubah kode berikut

```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            String judul = txtJudul.getText();
            String pengarang = txtPengarang.getText();
            String isbn = txtIsbn.getText();

            // Membuat objek Buku baru
            Buku buku = new Buku(judul, pengarang, isbn);

            // Menambah data buku ke tabel
            DefaultTableModel data = (DefaultTableModel)
            tblBuku.getModel();
            data.addRow(new Object[]{buku.getJudul(),
            buku.getPengarang(), buku.getIsbn()});

            // Reset form input
            txtJudul.setText("");
            txtPengarang.setText("");
            txtIsbn.setText("");
            lblKeterangan.setForeground(Color.GREEN);
            lblKeterangan.setText("Buku berhasil ditambahkan!");
          }catch (ValidasiInputException e) {
            lblKeterangan.setForeground(Color.RED);
            lblKeterangan.setText("Error: " + e.getMessage());
          }
    }
```

Penjelasan : Pada kode yang ditambahkan, blok try-catch digunakan untuk menangani pengecualian yang mungkin terjadi saat pengguna mengisi form dan menekan tombol submit. Di dalam blok try, objek Buku dibuat dengan mengambil data dari input pengguna, lalu data buku ditambahkan ke tabel. Jika proses ini berhasil, form akan di-reset, dan label keterangan (lblKeterangan) akan menampilkan pesan sukses dengan warna hijau. Namun, jika terjadi pengecualian ValidasiInputException (misalnya, input tidak valid), pengecualian tersebut akan ditangani dalam blok catch, dan label keterangan akan menampilkan pesan error dengan warna merah. Ini memberikan feedback yang jelas kepada pengguna terkait keberhasilan atau kegagalan input.

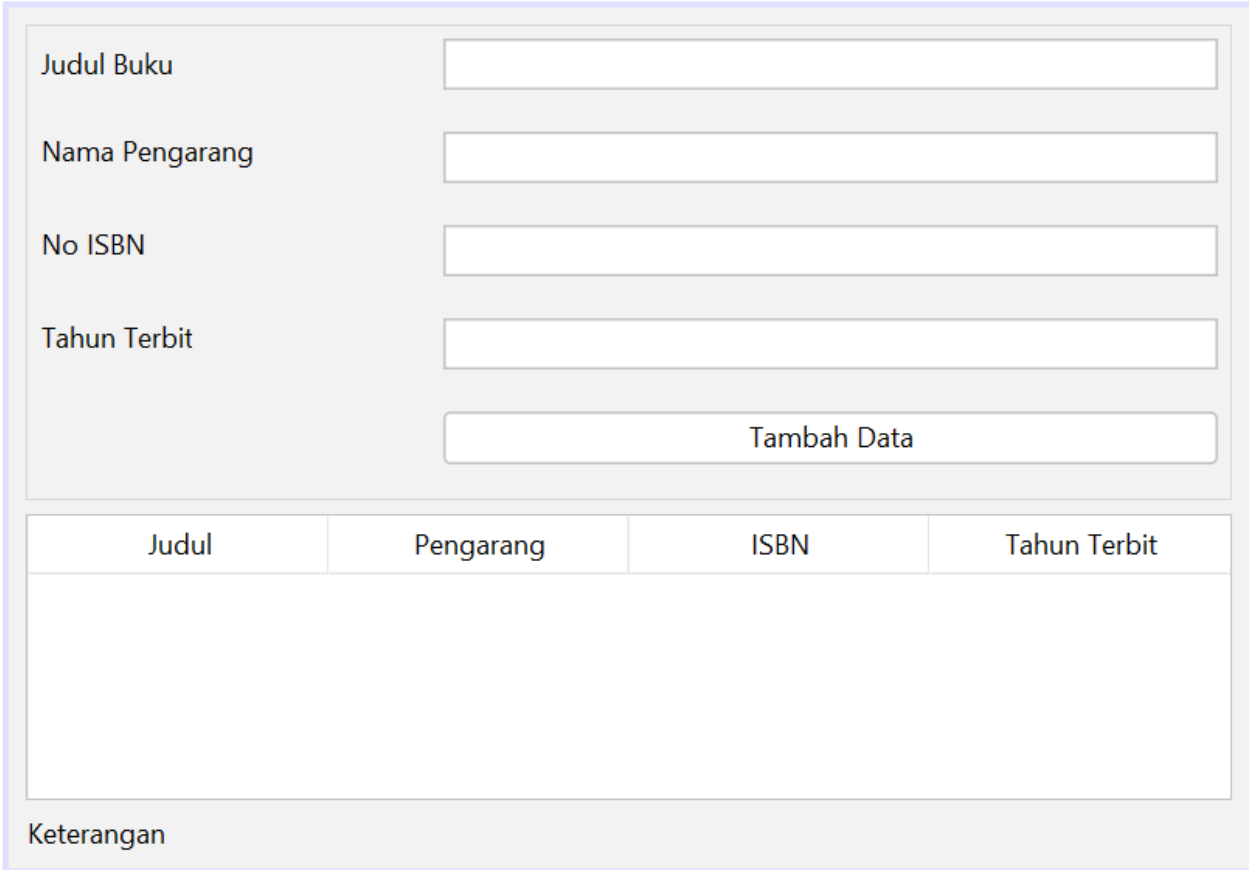
11. Output

C. UNGUIDED

Tambahkan inputan berikut pada aplikasi:

1. Tahun Terbit: Validasi untuk angka empat digit.

InputBuku.java



Penjelasan : Kita memperbarui tampilannya agar label, textfield, dan column table dari Tahun Terbit bisa muncul dan untuk variable dari textfield Tahun Terbit adalah txtThnTerbit

Buku.java

```
package Perpustakaan;


/**
 *
 * @author Teuku Bintang Hadi Negara
```

```
 */
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;
    private String tahunTerbit;

    public Buku(String judul, String pengarang, String isbn, String tahunTerbit) throws ValidasiInputException {
    if (judul.isEmpty()) {
        throw new ValidasiInputException("Judul buku tidak boleh kosong!");
    }

    if (pengarang.isEmpty()) {
        throw new ValidasiInputException("Pengarang tidak boleh kosong!");
    }

    if (!isbn.matches("\\d{13}")) {
        throw new ValidasiInputException("ISBN harus 13 digit angka!");
    }

    if (!tahunTerbit.matches("\\d{4}")) {
        throw new ValidasiInputException("Tahun Terbit harus 4 digit angka!");
    }

    this.judul = judul;
    this.pengarang = pengarang;
```

```
    this.isbn = isbn;
    this.tahunTerbit = tahunTerbit;

    }

    public String getJudul() {
        return judul;
    }

    public String getPengarang() {
        return pengarang;
    }

    public String getIsbn() {
        return isbn;
    }

    public String getTahunTerbit() {
        return tahunTerbit;
    }
}
```

Penjelasan : Pada kode yang ditambahkan, atribut baru tahunTerbit ditambahkan ke kelas Buku, lengkap dengan validasi di konstruktor. Validasi memastikan bahwa tahunTerbit hanya berisi angka sebanyak 4 digit, menggunakan metode matches. Jika validasi gagal, pengecualian ValidasiInputException akan dilemparkan dengan pesan error yang relevan. Selain itu, getter getTahunTerbit juga ditambahkan untuk mengakses nilai atribut tahunTerbit. Dengan penambahan ini, kelas Buku kini mampu menyimpan dan memvalidasi informasi tahun terbit dari sebuah buku, melengkapi atribut lainnya.

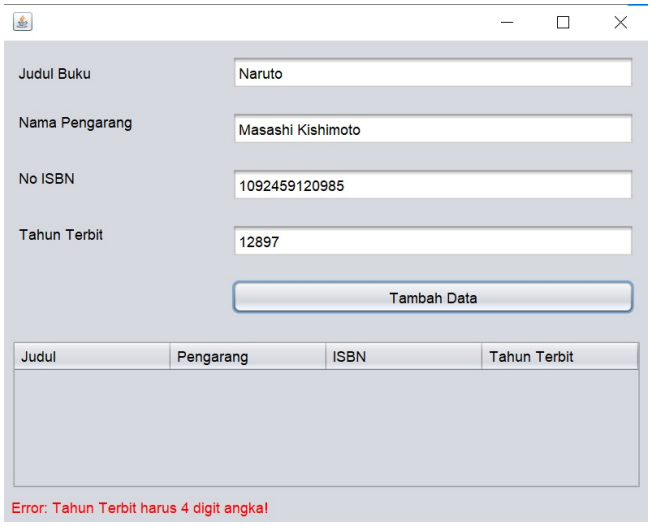
Modifikasi kodingan btnSubmitActionPerformed

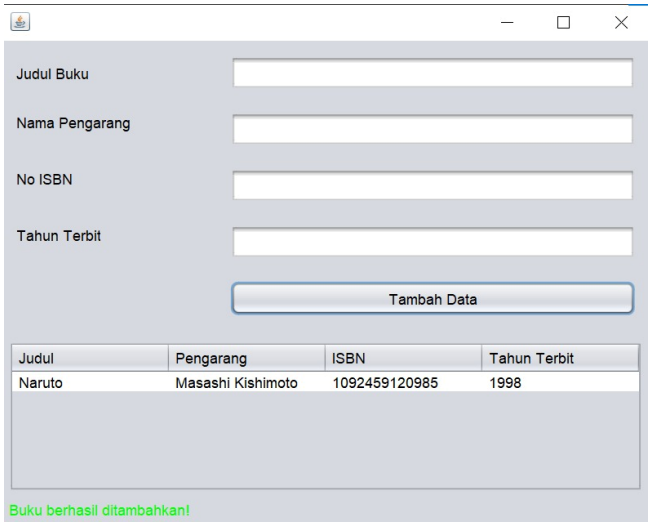
```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String judul = txtBuku.getText();
        String pengarang = txtPengarang.getText();
        String isbn = txtIsbn.getText();
        String tahunTerbit = txtThnTerbit.getText();

        // Membuat objek Buku baru
        Buku buku = new Buku(judul, pengarang, isbn, tahunTerbit);

        // Menambah data buku ke tabel
        DefaultTableModel data = (DefaultTableModel)
        tblBuku.getModel();
        data.addRow(new  Object[]{buku.getJudul(),buku.getPengarang(),  buku.getIsbn(),
buku.getTahunTerbit()});

        // Reset form input
        txtBuku.setText("");
        txtPengarang.setText("");
        txtIsbn.setText("");
        txtThnTerbit.setText("");

        lblKeterangan.setForeground(Color.GREEN);
        lblKeterangan.setText("Buku berhasil ditambahkan!");
      }catch (ValidasiInputException e) {
        lblKeterangan.setForeground(Color.RED);
```

```
            lblKeterangan.setText("Error: " + e.getMessage());
        }
    }
```

Penjelasan : Pada kode yang ditambahkan, variabel tahunTerbit diambil dari input pengguna melalui txtThnTerbit. Konstruktor kelas Buku kemudian diperbarui untuk menerima tahunTerbit sebagai parameter, yang juga ditambahkan ke baris data tabel melalui model tabel (DefaultTableModel). Selain itu, proses reset form input kini mencakup pengosongan txtThnTerbit. Dengan penambahan ini, aplikasi dapat menyimpan dan menampilkan informasi tahun terbit untuk setiap buku yang ditambahkan, memperluas fungsionalitas form input.

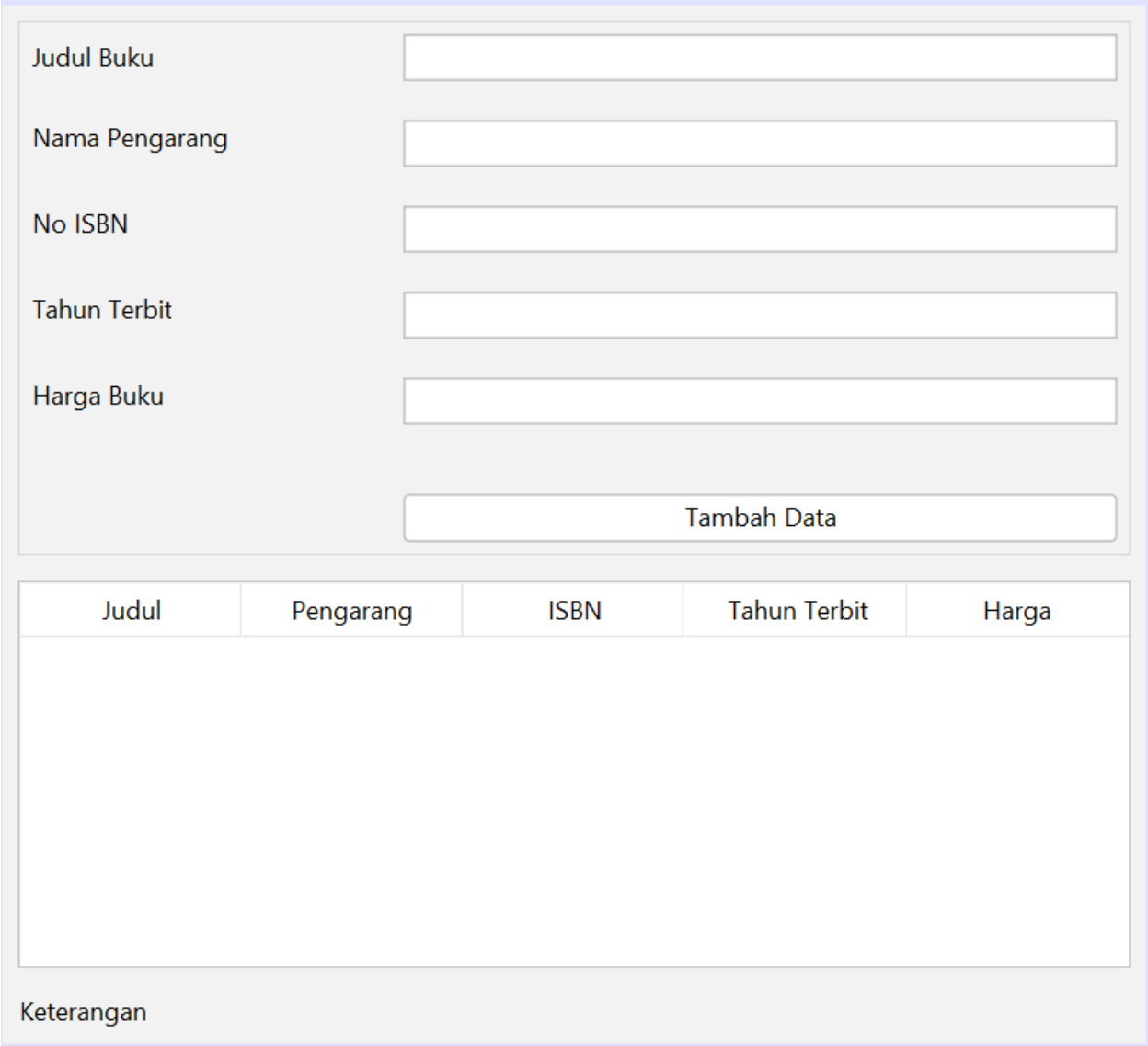
Output





2. Harga Buku: Validasi untuk angka positif.

InputBuku.java



Penjelasan : Kita memperbarui tampilannya agar label, textfield, dan column table dari Harga Buku bisa muncul dan untuk variable dari textfield Harga Buku adalah txtHarga

Buku.java

```
package Perpustakaan;


/**
```

```
 *
 * @author Teuku Bintang Hadi Negara
 */
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;
    private String tahunTerbit;
    private String harga;

    public Buku(String judul, String pengarang, String isbn, String tahunTerbit, String harga) throws ValidasiInputException {
    if (judul.isEmpty()) {
        throw new ValidasiInputException("Judul buku tidak boleh kosong!");
    }

    if (pengarang.isEmpty()) {
        throw new ValidasiInputException("Pengarang tidak boleh kosong!");
    }

    if (!isbn.matches("\\d{13}")) {
        throw new ValidasiInputException("ISBN harus 13 digit angka!");
    }

    if (!tahunTerbit.matches("\\d{4}")) {
        throw new ValidasiInputException("Tahun Terbit harus 4 digit angka!");
    }
```

```
try {
    long hargaNumber = Long.parseLong(harga); // Ubah ISBN menjadi angka
    if (hargaNumber < 0) {
        throw new ValidasiInputException("Harga tidak boleh angka negatif!");
    }
} catch (NumberFormatException e) {
    throw new ValidasiInputException("Input harus berupa angka!");
}

this.judul = judul;
this.pengarang = pengarang;
this.isbn = isbn;
this.tahunTerbit = tahunTerbit;
this.harga = harga;

}

public String getJudul() {
    return judul;
}

public String getPengarang() {
    return pengarang;
}

public String getIsbn() {
```

```
        return isbn;
    }

    public String getTahunTerbit() {
        return tahunTerbit;
    }

    public String getHarga() {
        return harga;
    }
}
```

Penjelasan : Pada kode yang ditambahkan, atribut baru harga ditambahkan ke kelas Buku untuk menyimpan informasi harga buku. Validasi dilakukan dalam konstruktor untuk memastikan bahwa input harga hanya berupa angka dan tidak negatif. Jika harga tidak dapat diubah menjadi angka (karena formatnya salah) atau bernilai negatif, maka pengecualian ValidasiInputException akan dilempar dengan pesan error yang relevan. Selain itu, getter getHarga juga ditambahkan untuk mengakses nilai atribut harga. Penambahan ini memungkinkan kelas Buku memvalidasi dan menyimpan informasi harga buku, menambah detail pada data buku yang dikelola aplikasi.

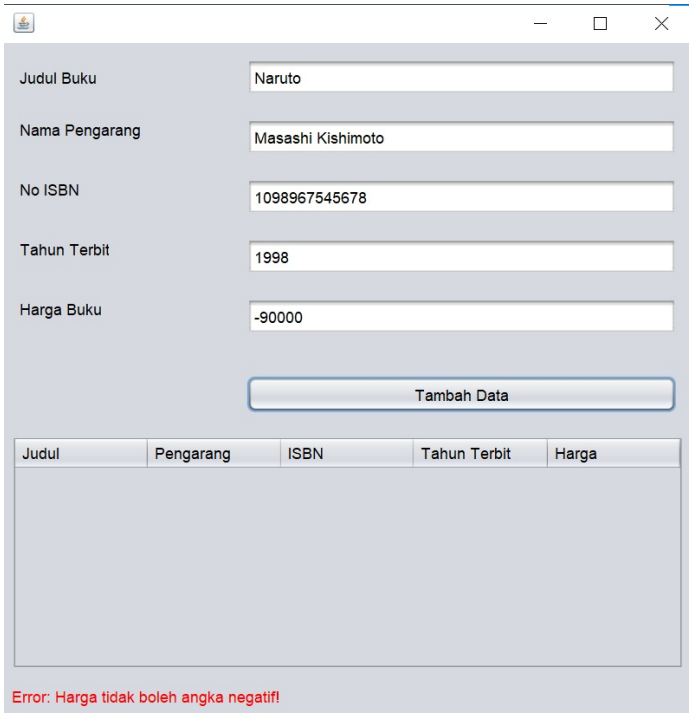
Modifikasi kodingan btnSubmitActionPerformed

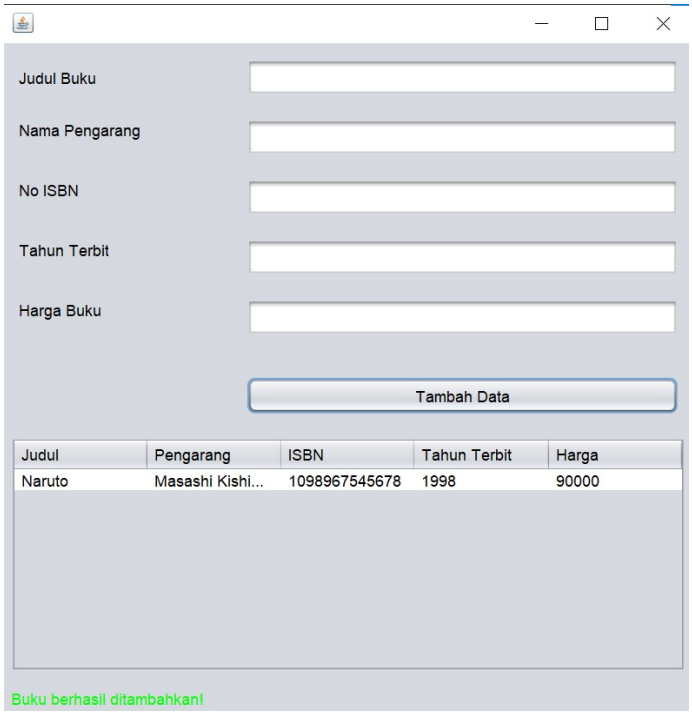
```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
        try {
                String judul = txtBuku.getText();
                String pengarang = txtPengarang.getText();
                String isbn = txtIsbn.getText();
                String tahunTerbit = txtThnTerbit.getText();
                String harga = txtHarga.getText();
```

```
            // Membuat objek Buku baru
            Buku buku = new Buku(judul, pengarang, isbn, tahunTerbit, harga);


            // Menambah data buku ke tabel
            DefaultTableModel data = (DefaultTableModel)
            tblBuku.getModel();
            data.addRow(new  Object[]{buku.getJudul(),buku.getPengarang(),  buku.getIsbn(),
buku.getTahunTerbit(), buku.getHarga()});


            // Reset form input
            txtBuku.setText("");
            txtPengarang.setText("");
            txtIsbn.setText("");
            txtThnTerbit.setText("");
            txtHarga.setText("");


            lblKeterangan.setForeground(Color.GREEN);
            lblKeterangan.setText("Buku berhasil ditambahkan!");
          }catch (ValidasiInputException e) {
            lblKeterangan.setForeground(Color.RED);
            lblKeterangan.setText("Error: " + e.getMessage());
          }
    }
```

Penjelasan : Pada kode yang ditambahkan, variabel harga diambil dari input pengguna melalui txtHarga dan diteruskan ke konstruktor kelas Buku. Saat data buku ditambahkan ke tabel melalui DefaultTableModel, atribut harga juga dimasukkan sebagai kolom baru. Selain itu, proses reset form input kini mencakup pengosongan txtHarga. Dengan penambahan ini, aplikasi mampu mengelola informasi harga buku, sehingga memberikan data yang lebih lengkap dan fungsionalitas form input yang diperluas.

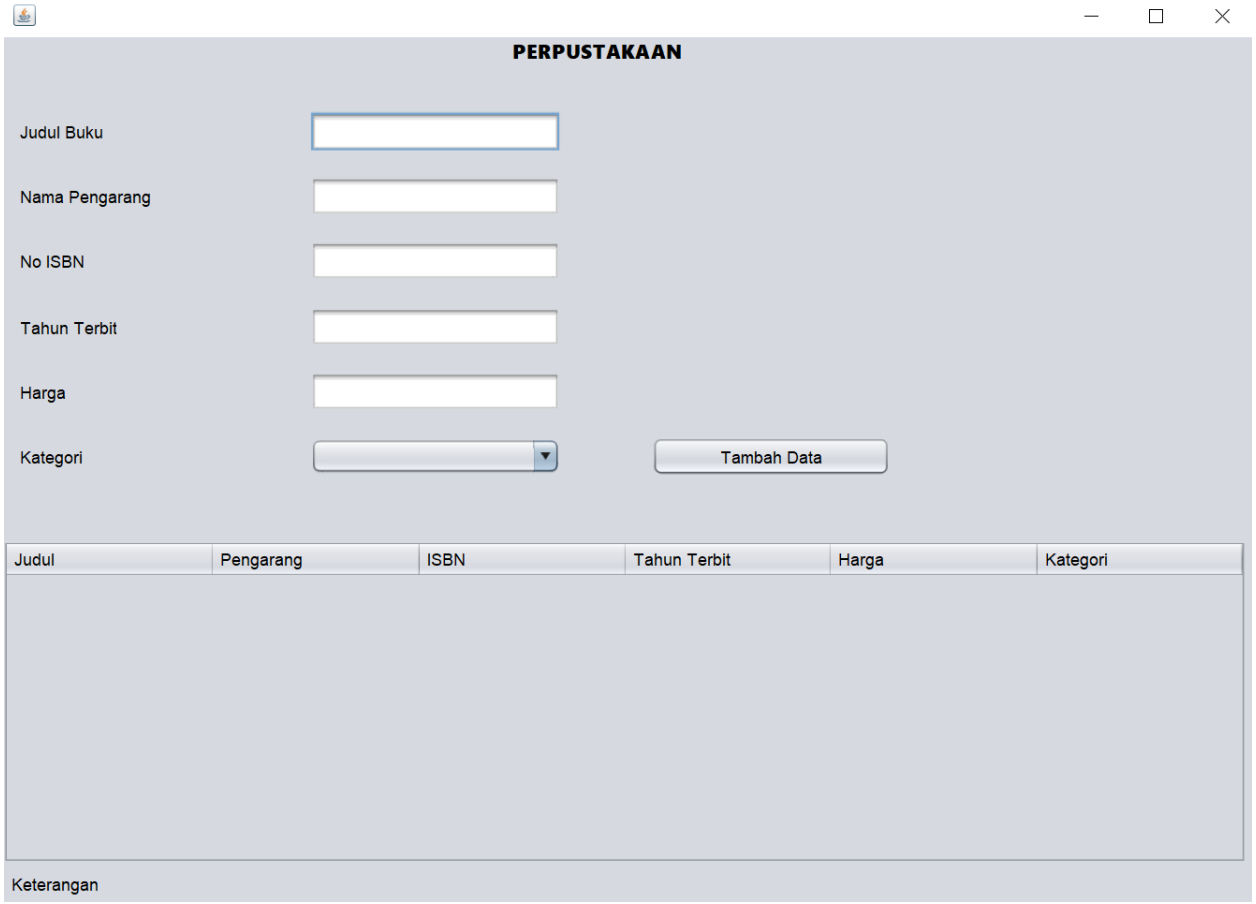
Output


Judul Buku
Naruto
Nama Pengarang
Masashi Kishimoto
No ISBN
1098967545678
Tahun Terbit
1998
Harga Buku
-90000
Tambah Data
Judul
Pengarang
ISBN
Tahun Terbit
Harga
Error: Harga tidak boleh angka negatif!


Judul Buku
Nama Pengarang
No ISBN
Tahun Terbit
Harga Buku
Tambah Data
Judul
Pengarang
ISBN
Tahun Terbit
Harga
Naruto
Masashi Kishi...
1098967545678
1998
90000
Buku berhasil ditambahkan!

3. Kategori Buku: Validasi untuk input tidak kosong. (dalam bentuk Dropdown)

InputBuku.java



Penjelasan : : Kita memperbarui tampilannya agar label, combo box, dan column table dari Kategori Buku bisa muncul dan untuk variable dari combo box Kategori Buku adalah cbKategori

Buku.java

```
package Perpustakaan;


/**
 *
 * @author Teuku Bintang Hadi Negara
 */
```

```
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;
    private String tahunTerbit;
    private String harga;
    private String kategori;

    public Buku(String judul, String pengarang, String isbn, String tahunTerbit, String harga, String kategori) throws ValidasiInputException {
    if (judul.isEmpty()) {
        throw new ValidasiInputException("Judul buku tidak boleh kosong!");
    }

    if (pengarang.isEmpty()) {
        throw new ValidasiInputException("Pengarang tidak boleh kosong!");
    }

    if (!isbn.matches("\\d{13}")) {
        throw new ValidasiInputException("ISBN harus 13 digit angka!");
    }

    if (!tahunTerbit.matches("\\d{4}")) {
        throw new ValidasiInputException("Tahun Terbit harus 4 digit angka!");
    }

    try {
```

```
    long hargaNumber = Long.parseLong(harga); // Ubah ISBN menjadi angka
    if (hargaNumber < 0) {
        throw new ValidasiInputException("Harga tidak boleh angka negatif!");
    }
} catch (NumberFormatException e) {
    throw new ValidasiInputException("Input harus berupa angka!");
}

if (kategori == null || kategori.isEmpty()) {
    throw new ValidasiInputException("Kategori harus dipilih!");
}

this.judul = judul;
this.pengarang = pengarang;
this.isbn = isbn;
this.tahunTerbit = tahunTerbit;
this.harga = harga;
this.kategori = kategori;

}

public String getJudul() {
    return judul;
}

public String getPengarang() {
    return pengarang;
```

```
    }

    public String getIsbn() {
        return isbn;
    }

    public String getTahunTerbit() {
        return tahunTerbit;
    }

    public String getHarga() {
        return harga;
    }

    public String getKategori() {
        return kategori;
    }
}
```

Penjelasan : Pada kode yang ditambahkan, atribut baru kategori ditambahkan ke kelas Buku untuk menyimpan informasi kategori buku. Validasi dilakukan dalam konstruktor untuk memastikan bahwa kategori tidak kosong atau bernilai null. Jika validasi gagal, pengecualian ValidasiInputException akan dilemparkan dengan pesan error yang relevan. Getter getKategori juga ditambahkan untuk mengakses nilai atribut kategori. Dengan penambahan ini, kelas Buku kini dapat mengelola dan memvalidasi informasi kategori buku, melengkapi detail data buku yang dikelola aplikasi.

Modifikasi kodingan btnSubmitActionPerformed

```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
        try {
```

```
            String judul = txtJudul.getText();
            String pengarang = txtPengarang.getText();
            String isbn = txtIsbn.getText();
            String tahunTerbit = txtThnTerbit.getText();
            String harga = txtHarga.getText();
            String kategori = (String) cbKategori.getSelectedItem();

            // Membuat objek Buku baru
            Buku buku = new Buku(judul, pengarang, isbn, tahunTerbit, harga, kategori);

            // Menambah data buku ke tabel
            DefaultTableModel data = (DefaultTableModel)
            tblBuku.getModel();
            data.addRow(new  Object[]{buku.getJudul(),buku.getPengarang(),  buku.getIsbn(),
buku.getTahunTerbit(), buku.getHarga(), buku.getKategori()});

            // Reset form input
            txtJudul.setText("");
            txtPengarang.setText("");
            txtIsbn.setText("");
            txtThnTerbit.setText("");
            txtHarga.setText("");
            cbKategori.setSelectedIndex(0);

            lblKeterangan.setForeground(Color.GREEN);
            lblKeterangan.setText("Buku berhasil ditambahkan!");
          }catch (ValidasiInputException e) {
```

```
            lblKeterangan.setForeground(Color.RED);
            lblKeterangan.setText("Error: " + e.getMessage());
        }
    }
```

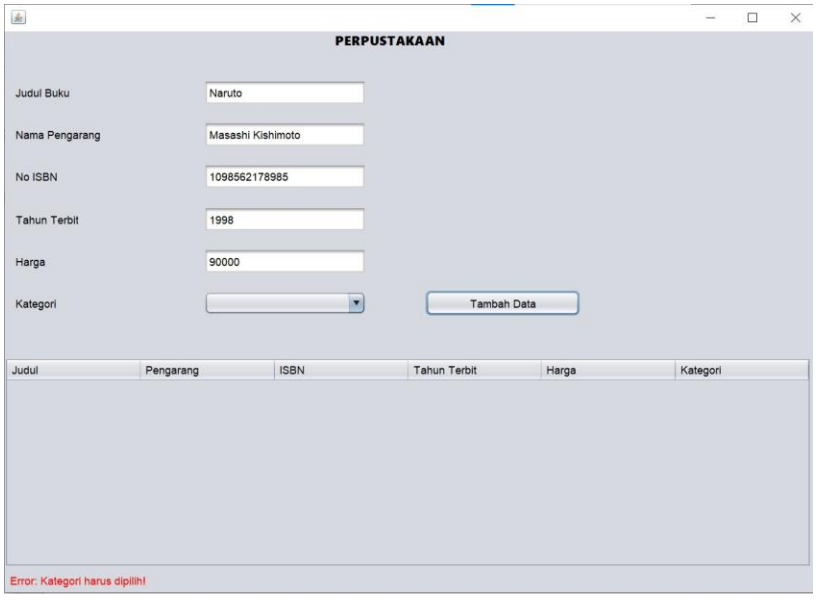
Penjelasan : Pada kode yang ditambahkan, variabel kategori diperoleh dari elemen dropdown cbKategori menggunakan metode getSelectedItem, yang mengembalikan item yang dipilih sebagai String. Variabel ini kemudian diteruskan ke konstruktor kelas Buku saat membuat objek baru. Saat menambah data buku ke tabel melalui DefaultTableModel, atribut kategori juga ditambahkan sebagai kolom baru. Selain itu, proses reset form input kini mencakup pengaturan ulang elemen dropdown cbKategori ke indeks awal (opsi pertama). Dengan penambahan ini, aplikasi mendukung pengelolaan kategori buku secara efektif, memberikan input dan output data yang lebih terstruktur.

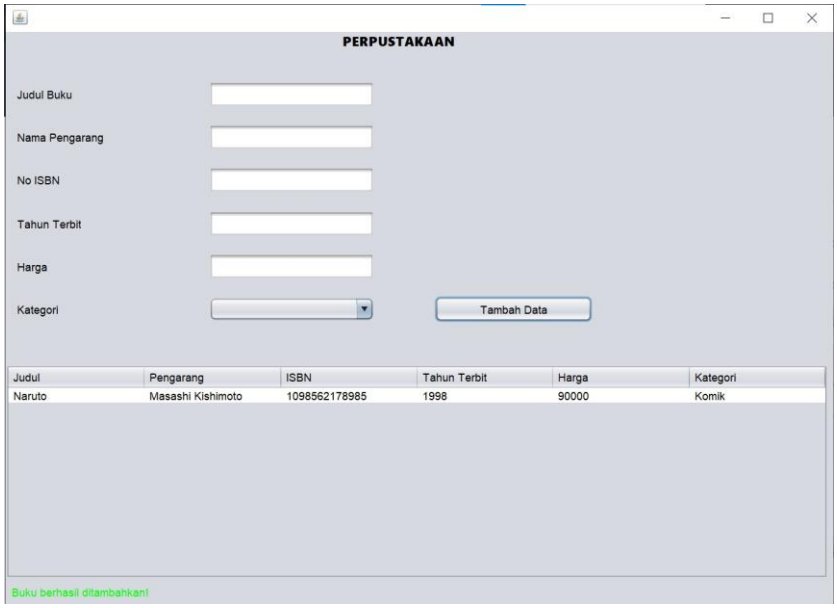
Customize code pada ComboBox

```
cbKategori.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {null, "Komik", "Novel", "Religi"}));
```

Penjelasan : Kode diatas menjelaskan isian comboBox, bisa diliat ada null yang berarti tidak ada data yang berarti saat kita memilih null maka aka nada validasi yang muncul, lalu ada komik,novel dan religi

Output




PERPUSTAKAAN
Judul Buku
Nama Pengarang
No ISBN
Tahun Terbit
Harga
Kategori
Tambah Data
Judul
Pengarang
ISBN
Tahun Terbit
Harga
Kategori
Naruto
Masashi Kishimoto
1098562178985
1998
90000
Komik
Buku berhasil ditambahkan!