

**LAPORAN PRAKTIKUM**  
**PEMROGRAMAN BERORIENTASI OBJEK**  
**MODUL 5**  
**LATIHAN EXCEPTION**



**Oleh :**

Vicky Mahya Mafaza

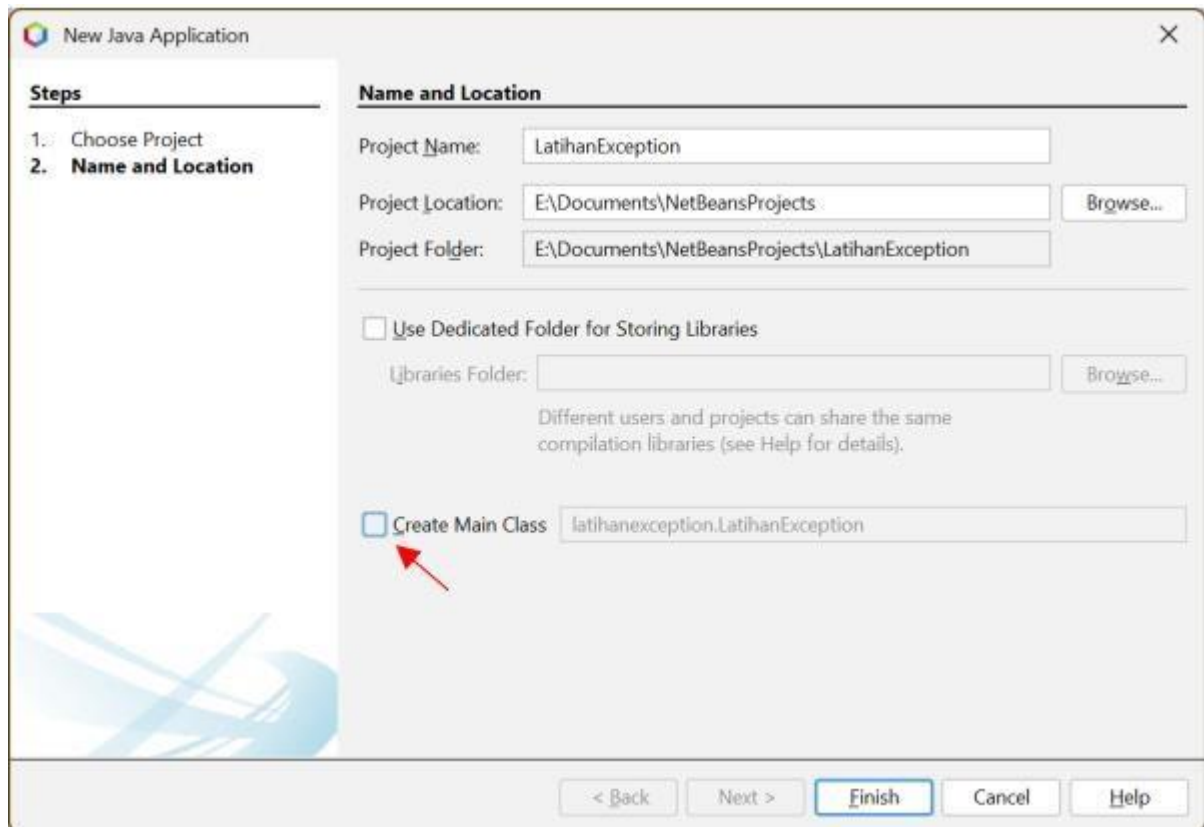
2311103113

S1SI-07-C

**PROGRAM STUDI SISTEM INFORMASI**  
**FAKULTAS REKAYASA INDUSTRI**  
**UNIVERSITAS TELKOM PURWOKERTO**  
**2024**

## A. 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

```
/*
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
template
 */
package Perpustakaan;

/**
```

```

*
* @author Vicky Mahya Mafaza
*/
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 adalah kelas Buku dalam Java yang merepresentasikan data buku dengan atribut privat judul, pengarang, dan isbn. Konstruktor digunakan untuk menginisialisasi atribut tersebut saat objek dibuat. Tiga metode getter (getJudul, getPengarang, getIsbn) disediakan untuk mengakses nilai atribut secara aman, menerapkan prinsip enkapsulasi. Kelas ini cocok untuk digunakan dalam sistem pengelolaan data buku.

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:

Judul Buku

Nama Pengarang

No ISBN

Tambah Data

Judul	Pengarang	ISBN
-------	-----------	------

Keterangan

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 implementasi metode `btnSubmitActionPerformed` yang menangani aksi ketika tombol "Submit" diklik pada antarmuka pengguna. Metode ini mengambil data dari tiga input teks (`txtJudul`, `txtPengarang`, dan `txtIsbn`), lalu membuat objek baru dari kelas `Buku` dengan data tersebut. Selanjutnya, data buku ditambahkan sebagai baris baru ke tabel (`tblBuku`) menggunakan model tabel `DefaultTableModel`. Setelah data berhasil dimasukkan ke tabel, input teks pada form direset agar siap untuk entri data baru. Kode ini digunakan untuk menambahkan data buku secara dinamis ke tabel dalam aplikasi berbasis Java Swing.

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`

`ValidInputException.java`

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
 * change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package Perpustakaan;

/**
 *
 * @author ASUS
 */

```

```
public class ValidInputException extends Exception{  
    public  
    ValidInputException(String  
    message) {  
        super(message);  
    }  
}
```

Penjelasan : Kode di atas mendefinisikan kelas ValidInputException dalam paket Perpustakaan, yang merupakan subclass dari Exception. Kelas ini digunakan untuk membuat jenis pengecualian (exception) khusus dengan pesan tertentu. Konstruktor ValidInputException menerima sebuah string (message) sebagai parameter, yang kemudian diteruskan ke konstruktor Exception induknya menggunakan kata kunci super. Dengan ini, kelas tersebut dapat digunakan untuk menangani kesalahan atau validasi input yang tidak sesuai dalam aplikasi, dengan memberikan pesan kesalahan yang spesifik dan informatif kepada pengguna atau pengembang.

9. Ubah kode pada Class buku

Buku.java

```
/*
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 *      to change this license
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
 *      template
 */
package Perpustakaan;

/**
 *
 *      @author Vicky Mahya Mafaza
 */
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;

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

```

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

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

    if (!isbn.matches("\\d{13}")) {            throw new
ValidInputException("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 : Penambahan kode pada konstruktor Buku adalah validasi input untuk memastikan data yang dimasukkan sesuai aturan. Jika judul atau pengarang kosong, atau jika isbn bukan 13 digit angka, maka akan melemparkan pengecualian `ValidInputException` dengan pesan yang relevan. Validasi ini menjaga agar objek Buku hanya dibuat dengan data yang valid, meningkatkan integritas data dalam 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 (ValidInputException e) {  
  
            lblKeterangan.setForeground(Color.RED);  
            lblKeterangan.setText("Error: " + e.getMessage());  
        }  
    }  
}
```

Penjelasan : Penambahan pada kode ini adalah implementasi mekanisme penanganan kesalahan menggunakan blok try-catch. Dalam blok try, jika validasi input dalam konstruktor Buku gagal, pengecualian `ValidInputException` akan dilempar. Blok catch menangkap pengecualian tersebut dan menampilkan pesan kesalahan melalui label `lblKeterangan` dengan warna merah. Sebaliknya, jika data valid, pesan sukses akan ditampilkan di `lblKeterangan` dengan warna hijau. Penambahan ini memberikan umpan balik visual kepada pengguna terkait status penginputan data.

11. Jalankan project, coba tambahkan data baru. Ketika form kosong maka akan muncul peringatan pada bagian bawah aplikasi

The screenshot shows a Java Swing application window with a light blue background. It contains three text input fields with labels to their left: 'Judul Buku' with the value 'Tentang Kamu', 'Nama Pengarang' with the value 'Tere Liye', and 'No ISBN' with the value '1098'. Below these fields is a button labeled 'Tambah Data'. At the bottom of the window, there is a table with three columns: 'Judul', 'Pengarang', and 'ISBN'. The table is currently empty. A red error message 'Error: ISBN harus 13 digit angka!' is displayed at the bottom left of the window. A watermark 'Activate Windows' is visible in the bottom right corner.

Judul	Pengarang	ISBN
-------	-----------	------

Error: ISBN harus 13 digit angka!

Judul Buku

Nama Pengarang

No ISBN

Tambah Data

Judul	Pengarang	ISBN
Tentang Kamu	Tere Liye	1098216749875

Buku berhasil ditambahkan!

## B. UNGUIDED

Tambahkan inputan berikut pada aplikasi:

1. Tahun Terbit: Validasi untuk angka empat digit.

Buku.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 * license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package Perpustakaan;

/**
 *
 */

```

```
* @author Vicky Mahya Mafaza
*/

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

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

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

        if (!tahunTerbit.matches("\\d{4}")) {
            throw new ValidInputException("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 : Penambahan pada kode ini adalah atribut baru tahunTerbit beserta validasinya dalam konstruktor. Validasi memastikan bahwa tahunTerbit diisi dengan tepat 4 digit angka. Jika tidak sesuai, akan melempar pengecualian ValidInputException dengan pesan "Tahun Terbit harus 4 digit angka!". Selain itu, metode getter getTahunTerbit() ditambahkan untuk mengakses nilai tahunTerbit. Penambahan ini memungkinkan penyimpanan dan validasi informasi tahun terbit buku.

InputBuku.java

```

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

    String judul = txtJudul.getText();

```

```

String pengarang = txtPengarang.getText();

String isbn = txtIsbn.getText();

String tahunTerbit = txttahunTerbit.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          txtJudul.setText("");

txtPengarang.setText("");          txtIsbn.setText("");
txttahunTerbit.setText("");


lblKeterangan.setForeground(Color.GREEN);

lblKeterangan.setText("Buku berhasil ditambahkan!");

} catch (ValidInputException e) {


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

}


}

```

---

Penjelasan : Penambahan pada kode ini mencakup pengambilan input dari txttahunTerbit, yang digunakan untuk menginisialisasi atribut tahunTerbit saat membuat objek Buku. Data tahun terbit juga ditambahkan ke tabel melalui model DefaultTableModel. Setelah data berhasil dimasukkan, input txttahunTerbit direset bersama input lainnya. Dengan penambahan ini, aplikasi kini dapat mengelola informasi tahun terbit buku secara lengkap.

## Output

 — □ ×

Judul Buku	<input type="text" value="RINDU"/>
Nama Pengarang	<input type="text" value="Tere Liye"/>
No ISBN	<input type="text" value="1021405945812"/>
Tahun Terbit Buku	<input type="text" value="18"/>

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit
-------	-----------	------	--------------

Error: Tahun Terbit harus 4 digit angka!

Judul Buku

Nama Pengarang

No ISBN

Tahun Terbit Buku

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit
RINDU	Tere Liye	1021405945812	2012

Buku berhasil ditambahkan!

2. Harga Buku: Validasi untuk angka positif.

Buku.java

```

/*
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
template
 */
package Perpustakaan;

/**
 *
 *      @author Vicky Mahya Mafaza
 */
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 ValidInputException {    if (judul.isEmpty()) {
        throw new ValidInputException("Judul buku tidak boleh kosong!");
    }

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

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

    if (!tahunTerbit.matches("\\d{4}")) {
        throw new ValidInputException("Tahun Terbit harus 4 digit angka!");
    }    try {
        long hargaNumber = Long.parseLong(harga); // Ubah ISBN menjadi angka    if
(hargaNumber < 0) {
            throw new ValidInputException("Harga tidak boleh angka negatif!");
        }
    }
}

```

```

    }

    } catch (NumberFormatException e) {

        throw new ValidInputException("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 : Penambahan pada kode ini meliputi atribut baru harga beserta validasinya. Validasi memastikan bahwa input harga adalah angka positif. Jika harga bernilai negatif, atau bukan angka valid, akan melempar pengecualian ValidInputException dengan pesan yang relevan. Selain itu, metode getter getHarga() ditambahkan untuk mengakses nilai atribut harga. Penambahan ini memungkinkan aplikasi mengelola informasi harga buku dengan validasi yang sesuai.

InputBuku.java

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

```

        txtIsbn.setText("");
    txttahunTerbit.setText("");
    txtHarga.setText("");

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

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

}

```

Penjelasan : Penambahan pada kode ini mencakup pengambilan input dari txtHarga untuk atribut harga saat membuat objek Buku. Data harga juga ditambahkan ke tabel melalui model DefaultTableModel. Setelah data berhasil dimasukkan, input txtHarga direset bersama dengan input lainnya. Penambahan ini memungkinkan aplikasi untuk mengelola informasi harga buku secara lengkap, dengan validasi yang sudah diterapkan pada konstruktor kelas Buku.

## Output

Judul Buku: Negeri Para Bedebah

Nama Pengarang: Tere Liye

No ISBN: 1021879002876

Tahun Terbit Buku: 2015

Harga Buku: -89898989

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit	Harga
-------	-----------	------	--------------	-------

Error: Harga tidak boleh angka negatif!

Judul Buku

Nama Pengarang

No ISBN

Tahun Terbit Buku

Harga Buku

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit	Harga
Negeri Para ...	Tere Liye	1021879002...	2015	120000

Buku berhasil ditambahkan!

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

Buku.java

```

/*
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 *      Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
template
 */
package Perpustakaan;

/**
 *
 *      @author Vicky Mahya Mafaza
 */
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 ValidInputException {    if (judul.isEmpty()) {

    throw new ValidInputException("Judul buku tidak boleh kosong!");

}

    if (pengarang.isEmpty()) {

        throw new ValidInputException("Pengarang tidak boleh kosong!");

    }

    if (!isbn.matches("\\d{13}")) {

        throw new ValidInputException("ISBN harus 13 digit angka!");

    }

    if (!tahunTerbit.matches("\\d{4}")) {

        throw new ValidInputException("Tahun Terbit harus 4 digit angka!");

    }    try {

        long hargaNumber = Long.parseLong(harga); // Ubah ISBN menjadi angka    if
(hargaNumber < 0) {

            throw new ValidInputException("Harga tidak boleh angka negatif!");

        }

```

```

    } catch (NumberFormatException e) {
        throw new ValidInputException("Input harus berupa angka!");
    }

    if (kategori == null || kategori.isEmpty()) {
        throw new ValidInputException("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 : Penambahan pada kode ini adalah atribut baru kategori beserta validasinya. Validasi memastikan bahwa input kategori tidak kosong atau null. Jika kategori tidak

dipilih, maka pengecualian `ValidInputException` akan dilempar dengan pesan "Kategori harus dipilih!". Selain itu, metode getter `getKategori()` ditambahkan untuk mengakses nilai atribut kategori. Penambahan ini memungkinkan aplikasi untuk menyimpan dan memvalidasi kategori buku yang dimasukkan, menambah detail informasi buku yang dikelola.

`InputBuku.java`

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

        cbKategori.setSelectedIndex(0);

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

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

        }

    }
}

```

Penjelasan : Penambahan pada kode ini adalah pengambilan nilai dari combo box cbKategori untuk mendapatkan kategori yang dipilih oleh pengguna, yang kemudian digunakan untuk membuat objek Buku. Setelah objek buku dibuat, data kategori juga ditambahkan ke tabel. Selain itu, setelah data berhasil dimasukkan, nilai input untuk cbKategori direset ke indeks pertama (kosong atau default). Penambahan ini memungkinkan aplikasi untuk menyertakan kategori buku dalam pengelolaan data dan memberikan umpan balik yang sesuai kepada pengguna.

## Output

Judul Buku

Pulang-Pergi

Nama Pengarang

Tere Liye

No ISBN

1209788916745

Tahun Terbit Buku

2018

Harga Buku

2000000

Kategori Buku

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit	Harga	Kategori
-------	-----------	------	--------------	-------	----------

Error: Kategori harus dipilih!

Judul Buku

Nama Pengarang

No ISBN

Tahun Terbit Buku

Harga Buku

Kategori Buku

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit	Harga	Kategori
Pulang-Pergi	Tere Liye	120978891...	2018	2000000	Novel

Buku berhasil ditambahkan!

