

**LAPORAN TUGAS BESAR
PEMROGRAMAN BERORIENTASI OBJEK
SISTEM PENCATATAN BUKU**



Disusun Oleh :

NAMA : Afsha Alifia Putri

NIM : 32602200031

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM SULTAN AGUNG
SEMARANG**

2022

BAB I Pendahuluan

1.1 Latar belakang

Pengelolaan buku merupakan aspek penting dalam kehidupan sehari-hari, baik untuk keperluan pribadi maupun institusional. Pemanfaatan sistematis dalam mencatat dan melacak informasi buku menjadi suatu kebutuhan.

1.2 Tujuan

Tujuan dari pengembangan program manajemen buku ini adalah meningkatkan efisiensi dan akurasi dalam pencatatan buku, memudahkan pengguna dalam mencari dan memanipulasi data buku, serta menyediakan basis data yang terorganisir dengan baik.

BAB II Struktur Program

2.1 Database Buku

DbBuku merupakan kelas yang berisi informasi dan konfigurasi terkait database buku.

```
1  |-- phpMyAdmin SQL Dump
2  -- version 5.2.1
3  -- https://www.phpmyadmin.net/
4  --
5  -- Host: 127.0.0.1
6  -- Waktu pembuatan: 01 Jan 2024 pada 08.07
7  -- Versi server: 10.4.28-MariaDB
8  -- Versi PHP: 8.2.4
9
10 SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
11 START TRANSACTION;
12 SET time_zone = "+00:00";
13
14
15 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
16 /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
17 /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
18 /*!40101 SET NAMES utf8mb4 */;
19
20 --
21 -- Database: `db_buku`
22 --
23
24 --
25 -----
26 --
27 -- Struktur dari tabel `t_buku`
28 --
29
30 CREATE TABLE `t_buku` (
31   `id_buku` int(11) NOT NULL,
32
```

Gambar 1. 1 DB Buku 1

```
--
-- Struktur dari tabel `t_buku`
--
--
CREATE TABLE `t_buku` (
  `id_buku` int(11) NOT NULL,
  `judul_buku` varchar(50) NOT NULL,
  `kode_buku` varchar(255) NOT NULL,
  `penganang` varchar(30) NOT NULL,
  `penerbit` varchar(30) NOT NULL,
  `isbn` varchar(255) NOT NULL,
  `create_at` datetime DEFAULT NULL,
  `update_at` datetime DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
--
-- Dumping data untuk tabel `t_buku`
--
INSERT INTO `t_buku` (`id_buku`, `judul_buku`, `kode_buku`, `penganang`, `penerbit`, `isbn`, `create_at`, `update_at`) VALUES
(2, 'buku', 'ksd', 'nisa', 'gramed', 'sds', '2023-12-30 12:59:21', '2023-12-30 12:59:21'),
(3, 'bismillah', 'wdwd', 'uid', 'idb', 'wd', '2024-01-01 12:42:49', '2024-01-01 12:42:49');
--
-- Indexes for dumped tables
--
--
-- Indeks untuk tabel `t_buku`
--
```

Gambar 1. 2 Buku 2

```

--
-- Indexes for dumped tables
--

--
-- Indeks untuk tabel `t_buku`
--
ALTER TABLE `t_buku`
  ADD PRIMARY KEY (`id_buku`);

--
-- AUTO_INCREMENT untuk tabel yang dibuang
--
--
-- AUTO_INCREMENT untuk tabel `t_buku`
--
ALTER TABLE `t_buku`
  MODIFY `id_buku` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Gambar 1. 3 Buku 3

Penjelasan :

1. Versi Database dan Server:

- a. Database bernama db_buku.
- b. Server menggunakan MariaDB versi 10.4.28.
- c. PHP versi 8.2.4.

2. Struktur Tabel:

a. Tabel bernama t_buku.

b. Kolom-kolom tabel:

- 1) id_buku (Integer, Auto-increment, Primary Key)
- 2) judul_buku (Varchar dengan panjang maksimal 50)
- 3) kode_buku (Varchar dengan panjang maksimal 255)
- 4) pengarang (Varchar dengan panjang maksimal 30)
- 5) penerbit (Varchar dengan panjang maksimal 30)
- 6) isbn (Varchar dengan panjang maksimal 255)
- 7) create_at (Datetime, default NULL)
- 8) update_at (Datetime, default NULL)

c. Tabel menggunakan engine InnoDB dan karakter set utf8mb4.

3. Data Tabel t_buku:

Tabel t_buku memiliki dua baris data:

- a. ID: 2, Judul: 'buku', Kode: 'ksd', Pengarang: 'nisa', Penerbit: 'gramed',

ISBN: 'sds', Create At: '2023-12-30 12:59:21', Update At: '2023-12-30 12:59:21'

b. ID: 3, Judul: 'bismilah', Kode: 'wdwqd', Pengarang: 'uid', Penerbit: 'idb', ISBN: 'wd', Create At: '2024-01-01 12:42:49', Update At: '2024-01-01 12:42:49'

4. Indeks:

a. Tabel t_buku memiliki indeks primary key pada kolom id_buku.

5. Auto-increment:

a. Kolom id_buku diatur sebagai auto-increment dengan nilai awal 4.

6. Charset dan Konfigurasi Lainnya:

a. Database diatur menggunakan charset utf8mb4.

b. Dilakukan beberapa pengaturan awal seperti SQL_MODE dan TIME_ZONE.

7. Pengaturan Karakter Lama:

a. Menyimpan pengaturan karakter lama sebelum dilakukan perubahan.

File ini dapat digunakan untuk mengembalikan struktur database dan datanya di server MySQL/MariaDB.

2.2 File Utama (BukuMain)

BukuMain merupakan file utama program yang berisi fungsi main untuk menjalankan aplikasi manajemen buku.

```
package buku.afsha_alifia;

import java.util.*;

author : Afsha alifia putri
nim : 32602200031
berikan penjelasan kode ini baris perbaris dengan komentar
*/

//import library
import com.toedter.calendar.JDateChooser;
import javax.swing.*;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import javax.swing.table.DefaultTableModel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.text.ParseException;
import java.util.Date;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

public class BukuMain extends javax.swing.JFrame {

    // Deklarasi variabel
    DefaultTableModel model;
```

Gambar 1. 4 File Utama MainBuku

```

// Deklarasi variabel
DefaultTableModel model;
private DatabaseManager dbHelper;
BukuDAOImpl bukuDAOImpl;

// Komponen GUI
JButton btn_hapus;
JButton btn_keluar1;
JButton btn_tambah;
JButton btn_ubah;
JTextField id_buku;
JTextField isbn;
JLabel jLabelProgram;
JLabel jLabel1;
JLabel jLabel10;
JLabel jLabel12;
JLabel jLabel13;
JLabel jLabel14;
JLabel jLabel15;
JLabel jLabel16;
JLabel jLabel17;
JLabel jLabel18;
JLabel jLabel19;
JScrollPane jScrollPane1;
JTextField judul_buku;
JTextField kode_buku;
JTextField penerbit;
JTextField pengarang;
JTable tabel;
JDateChooser create_at;
JDateChooser update_at;

```

Gambar 1. 5 BukuMain 2

```

JDateChooser update_at;

// Variabel indeks baris yang dipilih
private int selectedRowIndex = -1;

// Konstruktor
public BukuMain() throws ParseException {
    initComponents(); // Inisialisasi komponen
    dbHelper = new DatabaseManager(); // Inisialisasi DatabaseHelper
    bukuDAOImpl = new BukuDAOImpl();
    String[] judul = {"Id Buku", "Judul Buku", "Kode Buku", "Pengarang", "Penerbit", "ISBN", "Create at", "Update at"};
    model = new DefaultTableModel(judul, new Object[0]);
    tabel.setModel(model);
    tampilkan(); // Menampilkan data buku pada tabel
}

// Inisialisasi komponen GUI
public void initComponents() {
    // ... (kode yang di-generate secara otomatis oleh GUI Builder)

    // Listener untuk tombol tambah
    btn_tambah.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            try {
                btn_tambahActionPerformed(e);
            } catch (ParseException ex) {
                Logger.getLogger(BukuMain.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    });
}

```

Gambar 1. 6 BukuMain 3

```

// Listener untuk tombol ubah
btn_ubah.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            btn_ubahActionPerformed(evt:e);
        } catch (ParseException ex) {
            Logger.getLogger(BukuMain.class.getName()).log(Level.SEVERE, msg:null, thrown: ex);
        }
    }
});

// Listener untuk tombol hapus
btn_hapus.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            btn_hapusActionPerformed(evt:e);
        } catch (ParseException ex) {
            Logger.getLogger(BukuMain.class.getName()).log(Level.SEVERE, msg:null, thrown: ex);
        }
    }
});

// Listener untuk tombol keluar
btn_keluar.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        btn_keluarActionPerformed(evt:e);
    }
});

```

Gambar 1. 7 BukuMain 4

```

        btn_keluarActionPerformed(evt:e);
    }
}

// Listener untuk pemilihan baris pada tabel
tabel.getSelectionModel().addListSelectionListener(new ListSelectionListener() {
    @Override
    public void valueChanged(ListSelectionEvent e) {
        if (!e.getValueIsAdjusting()) {
            selectedRowIndex = tabel.getSelectedRow();
            if (selectedRowIndex >= 0) {
                fillFormWithSelectedData();
            }
        }
    }
});

// Menampilkan data buku pada tabel
private void tampilkan() throws ParseException {
    // Bersihkan tabel
    model.setRowCount(rowCount: 0);

    // Ambil data dari database dan tampilkan di tabel
    List<Buku> listBuku = bukuDAOImpl.getAllBuku();
    for (Buku buku : listBuku) {
        Object[] row = {
            buku.getIdBuku(),
            buku.getJudulBuku(),
            buku.getKodeBuku(),
            buku.getPengarang(),
            buku.getPenerbit()
        };
    }
}

```

Gambar 1. 8 BukuMain 5

```

        buku.getPengarang(),
        buku.getPenerbit(),
        buku.getIdBuku(),
        buku.getJudulBuku(),
        buku.getKodeBuku(),
        buku.getPengarang(),
        buku.getPenerbit()
    };
    model.addRow(row);
}

// Listener untuk tombol tambah
private void btn_tambahActionPerformed(ActionEvent evt) throws ParseException {
    // Mendapatkan data dari input user
    String judul = judul_buku.getText();
    String kode = kode_buku.getText();
    String pengarangBuku = pengarang.getText();
    String penerbitBuku = penerbit.getText();
    String isbnBuku = isbn.getText();

    // Mendapatkan tanggal saat ini
    Date createDate = new Date();

    // Membuat objek buku tanpa menyatakan ID
    Buku buku = new Buku(isbn: 0, judul: judul, kode: kode, pengarang: pengarangBuku, penerbit: penerbitBuku, isbn: isbnBuku, createDate: createDate, updateDate: createDate);

    // Menambahkan buku ke database
    bukuDAOImpl.addBuku(buku);

    // Menampilkan ulang data pada tabel
    tampilkan();

    // Menampilkan pesan
}

```

Gambar 1. 9 BukuMain 6

```

// Mengosongkan inputan
judul_buku.setText("");
kode_buku.setText("");
pengarang.setText("");
penerbit.setText("");
isbn.setText("");
}

// Listener untuk tombol simpan
private void btn_simpanActionPerformed(ActionEvent evt) throws ParseException {
// Memastikan bahwa baris telah dipilih
if (selectedRowIndex < 0) {
JOptionPane.showMessageDialog(this, message, "Pilih buku yang akan diubah");
return;
}

// Mendapatkan ID buku dari baris yang dipilih
int id = Integer.parseInt(model.getValueAt(selectedRowIndex, column: 0).toString());

// Mengambil data dari baris yang
String judul = judul_buku.getText();
String kode = kode_buku.getText();
String pengarang = pengarang.getText();
String penerbit = penerbit.getText();
String isbn = isbn.getText();

// Mendapatkan tanggal saat ini
Date updateDate = new Date();

// Membuat objek buku
Buku buku = new Buku(idBuku, id, judulBuku, kodeBuku, kode, pengarang, penerbitBuku, penerbit, isbnBuku, updateDate, updateDate);
}

```

Gambar 1. 10 BukuMain 7

```

// Mendapatkan tanggal saat ini
Date updateDate = new Date();

// Membuat objek buku
Buku buku = new Buku(idBuku, id, judulBuku, kodeBuku, kode, pengarang, penerbitBuku, penerbit, isbnBuku, updateDate, updateDate);

// Mengubah buku di database
bukuDAOImpl.updateBuku(buku);

// Menampilkan ulang data pada tabel
tampilkan();

// Mengosongkan inputan
id_buku.setText("");
judul_buku.setText("");
kode_buku.setText("");
pengarang.setText("");
penerbit.setText("");
isbn.setText("");
selectedRowIndex = -1;
}

// Listener untuk tombol hapus
private void btn_hapusActionPerformed(ActionEvent evt) throws ParseException {
// Memastikan bahwa baris telah dipilih
if (selectedRowIndex < 0) {
JOptionPane.showMessageDialog(this, message, "Pilih buku yang akan dihapus");
return;
}

// Mendapatkan ID buku dari baris yang dipilih
int id = Integer.parseInt(id_buku.getText());
}

```

Gambar 1. 11 BukuMain 8

```

// Mendapatkan ID buku dari baris yang dipilih
int id = Integer.parseInt(id_buku.getText());

// Menghapus buku dari database
bukuDAOImpl.deleteBuku(idBuku: id);

// Menampilkan ulang data pada tabel
tampilkan();

// Mengosongkan inputan
id_buku.setText("");
judul_buku.setText("");
kode_buku.setText("");
pengarang.setText("");
penerbit.setText("");
isbn.setText("");
selectedRowIndex = -1;
}

// Listener untuk tombol keluar
private void btn_keluarActionPerformed(ActionEvent evt) {
System.exit(status: 0);
}

// Mengisi formulir dengan data yang dipilih dari tabel
private void fillFormWithSelectedData() {
id_buku.setText(model.getValueAt(row:selectedRowIndex, column: 0).toString());
judul_buku.setText(model.getValueAt(row:selectedRowIndex, column: 1).toString());
kode_buku.setText(model.getValueAt(row:selectedRowIndex, column: 2).toString());
pengarang.setText(model.getValueAt(row:selectedRowIndex, column: 3).toString());
penerbit.setText(model.getValueAt(row:selectedRowIndex, column: 4).toString());
isbn.setText(model.getValueAt(row:selectedRowIndex, column: 5).toString());
}

```

Gambar 1. 12 BukuMain 9


```

penerbit.setText(": " + model.getValueAt(row:selectedRowIndex, column: 4).toString());
isbn.setText(": " + model.getValueAt(row:selectedRowIndex, column: 5).toString());

// Menetapkan tanggal ke JDateChooser
try {
    Date createDate = DateUtils.stringToDate(dateString: model.getValueAt(row:selectedRowIndex, column: 6).toString());
    create_bt.setDate(date: createDate);
} catch (ParseException ex) {
    ex.printStackTrace();
}

try {
    Date updateDate = DateUtils.stringToDate(dateString: model.getValueAt(row:selectedRowIndex, column: 7).toString());
    update_bt.setDate(date: updateDate);
} catch (ParseException ex) {
    ex.printStackTrace();
}
}

// Metode utama untuk menjalankan aplikasi
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                new BukuMain().setVisible(true);
            } catch (ParseException ex) {
                Logger.getLogger(BukuMain.class.getName()).log(Level.SEVERE, msg:null, thrown: ex);
            }
        }
    });
}

```

Gambar 1. 13 BukuMain 10

Penjelasan :

1. Inisialisasi Komponen GUI:

- a. Mendeklarasikan dan menginisialisasi komponen GUI seperti tombol, teksfield, label, dan tabel.
- b. Menyiapkan model untuk tabel dan mengatur header tabel.

2. Konstruktor BukuMain:

- a. Memanggil metode initComponents untuk menginisialisasi komponen.
- b. Menginisialisasi objek DatabaseManager dan BukuDAOImpl.
- c. Membuat model tabel dengan judul kolom.
- d. Menetapkan model tabel pada komponen tabel.
- e. Memanggil metode tampilkan untuk menampilkan data buku pada tabel.

3. Metode initComponents:

- a. Membuat dan mengatur tata letak komponen GUI menggunakan GroupLayout.
- b. Menambahkan action listener untuk tombol-tombol dan tabel.

4. Metode tampilkan:

- a. Menghapus semua baris pada model tabel.
- b. Mengambil data buku dari database melalui BukuDAOImpl dan menambahkannya ke model tabel.

5. Listener untuk Tombol Tambahan:

- a. btn_tambahActionPerformed: Menangani penambahan buku ke database dan menampilkan ulang data pada tabel.
 - b. btn_ubahActionPerformed: Menangani pembaruan data buku di database dan menampilkan ulang data pada tabel.
 - c. btn_hapusActionPerformed: Menangani penghapusan buku dari database dan menampilkan ulang data pada tabel.
 - d. btn_keluarActionPerformed: Menangani penutupan aplikasi.
6. Listener untuk Pemilihan Baris pada Tabel:
- a. Mengatur ListSelectionListener untuk mendeteksi pemilihan baris pada tabel.
 - b. Menyimpan indeks baris yang dipilih.
 - c. Mengisi formulir dengan data dari baris yang dipilih.
7. Metode fillFormWithSelectedData:
- a. Mengisi komponen formulir dengan data dari baris yang dipilih pada tabel.
8. Metode main:
- a. Metode utama untuk menjalankan aplikasi Swing.

2.3 Kelas Buku

Kelas Buku memiliki atribut judulBuku, kodeBuku, pengarang, penerbit, isbn, createAt, dan updateAt. Setiap atribut memiliki metode getter dan setter.

```
package buku.afsha_alifia;

/**
 * author : Afsha alifia putri
 * idm : 32602200031
 * berikan penjelasan kode ini baris perbaris dengan komentar, bagian polimorfisme, getter setter, constructor, inheritance
 */
import java.util.Date;

public class Buku {
    // Atribut private untuk menyimpan informasi buku
    private int idBuku;
    private String judulBuku;
    private String kodeBuku;
    private String pengarang;
    private String penerbit;
    private String isbn;
    private Date createAt;
    private Date updateAt;

    // Constructor untuk membuat objek Buku dengan memberikan nilai awal pada atribut
    public Buku(int idBuku, String judulBuku, String kodeBuku, String pengarang, String penerbit, String isbn, Date createAt, Date updateAt) {
        this.idBuku = idBuku;
        this.judulBuku = judulBuku;
        this.kodeBuku = kodeBuku;
        this.pengarang = pengarang;
        this.penerbit = penerbit;
        this.isbn = isbn;
        this.createAt = createAt;
        this.updateAt = updateAt;
    }
}
```

Gambar 1. 14 Kelas Buku 1

```

        this.createAt = createdAt;
        this.updateAt = updateAt;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public int getIdBuku() {
        return idBuku;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setIdBuku(int idBuku) {
        this.idBuku = idBuku;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public String getJudulBuku() {
        return judulBuku;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setJudulBuku(String judulBuku) {
        this.judulBuku = judulBuku;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public String getKodeBuku() {
        return kodeBuku;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setKodeBuku(String kodeBuku) {
        this.kodeBuku = kodeBuku;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public String getPengarang() {
        return pengarang;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private

```

Gambar 1. 15 Kelas Buku 2

```

    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public String getPengarang() {
        return pengarang;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setPengarang(String pengarang) {
        this.pengarang = pengarang;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public String getPenerbit() {
        return penerbit;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setPenerbit(String penerbit) {
        this.penerbit = penerbit;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public String getIsbn() {
        return isbn;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setIsbn(String isbn) {
        this.isbn = isbn;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public Date getCreateAt() {
        return createdAt;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setCreateAt(Date createdAt) {
        this.createdAt = createdAt;
    }
}

```

Gambar 1. 16 Buku 3

```

    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setCreateAt(Date createdAt) {
        this.createdAt = createdAt;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public Date getUpdateAt() {
        return updateAt;
    }

    // Getter dan setter untuk mengakses dan mengubah nilai atribut private
    public void setUpdateAt(Date updateAt) {
        this.updateAt = updateAt;
    }
}

```

Gambar 1. 17 Buku 4

Penjelasan :

1. Constructor (public Buku(...)):

- a. Membuat objek Buku dengan memberikan nilai awal pada atribut.
 - b. Digunakan saat pembuatan objek baru dari kelas Buku.
2. Getter dan Setter:
- a. Getter digunakan untuk mendapatkan nilai dari atribut private.
 - b. Setter digunakan untuk mengubah nilai dari atribut private.
 - c. Memungkinkan akses terkontrol ke atribut kelas Buku.

2.4 Kelas BukuDAOImpl

BukuImpl berisi implementasi logika bisnis terkait entitas Buku, termasuk penambahan, pengubahan, dan penghapusan data buku..

```

id: 32602200031
Berikan penjelasan kode ini baris perbaris dengan komentar bagian polymorfisme dan inheritance di dalam kodenya
// Class declaration: BukuDAOImpl implements BukuDAO
public class BukuDAOImpl implements BukuDAO {

    // Implementing the addBuku method from BukuDAO interface
    @Override
    public void addBuku(Buku buku) {
        try {
            Connection connection = DatabaseManager.getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement("INSERT INTO t_buku (judul_buku, kode_buku, pengarang, penerbit, tahun, ...)");

            // Setting values for the prepared statement
            preparedStatement.setString(1, buku.getJudulBuku());
            preparedStatement.setString(2, buku.getKodeBuku());
            preparedStatement.setString(3, buku.getPengarang());
            preparedStatement.setString(4, buku.getPenerbit());
            preparedStatement.setString(5, buku.getTahun());
            preparedStatement.setString(6, buku.getIdBuku());
            preparedStatement.setString(7, DateUtil.dateToString(buku.getCreateDate()));

            // Executing the SQL query
            preparedStatement.executeUpdate();

            // Retrieve the generated ID
            ResultSet rs = preparedStatement.getGeneratedKeys();
            if (rs.next()) {
                buku.setIdBuku(rs.getInt(1));
            }
        } catch (SQLException e) {
            // ...
        }
    }
}

```

Gambar 1. 18 Kelas BukuDAOImpl 1

```

}
} catch (SQLException e) {
    e.printStackTrace();
}
}

// Implementing the updateBuku method from BukuDAO interface
@Override
public void updateBuku(Buku buku) {
    try {
        Connection connection = DatabaseManager.getConnection();
        PreparedStatement preparedStatement = connection.prepareStatement("UPDATE t_buku SET judul_buku = ?, kode_buku = ?, pengarang = ?, penerbit = ?, tahun = ?, id_buku = ?");

        // Setting values for the prepared statement
        preparedStatement.setString(1, buku.getJudulBuku());
        preparedStatement.setString(2, buku.getKodeBuku());
        preparedStatement.setString(3, buku.getPengarang());
        preparedStatement.setString(4, buku.getPenerbit());
        preparedStatement.setString(5, buku.getTahun());
        preparedStatement.setString(6, buku.getIdBuku());
        preparedStatement.setString(7, DateUtil.dateToString(buku.getCreateDate()));

        // The rest for ID buku
        preparedStatement.setInt(8, buku.getIdBuku());

        // Executing the SQL query
        preparedStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Implementing the deleteBuku method from BukuDAO interface

```

Gambar 1. 19 Kelas BukuDAOImpl 2

```

// Implementing the getAllBuku method from BukuDAO interface
@Override
public List<Buku> getAllBuku() throws ParseException {
    List<Buku> bukuList = new ArrayList<>();

    try (Connection connection = DatabaseManager.getConnection();
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery("SELECT * FROM t_buku")) {

        while (resultSet.next()) {
            // Creating a new Buku object with data from the result set
            Buku buku = new Buku(
                resultSet.getInt(1),
                resultSet.getString(2),
                resultSet.getString(3),
                resultSet.getString(4),
                resultSet.getString(5),
                resultSet.getString(6),
                DateUtils.stringToDate(resultSet.getString(7)),
                DateUtils.stringToDate(resultSet.getString(8))
            );

            // Adding the Buku object to the list
            bukuList.add(buku);
        }

    } catch (SQLException | ParseException e) {
        e.printStackTrace();
    }

    // Returning the list of Buku objects
}

```

Gambar 1. 20 BukuDAOImpl 3

```

// Implementing the deleteBuku method from BukuDAO interface
@Override
public void deleteBuku(int idBuku) {
    try (Connection connection = DatabaseManager.getConnection();
        PreparedStatement preparedStatement = connection.prepareStatement("DELETE FROM t_buku WHERE id_buku = ?")) {

        // Setting the id_buku for the prepared statement
        preparedStatement.setInt(1, idBuku);

        // Executing the SQL query
        preparedStatement.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

Gambar 1. 21 BukuDAOImpl 4

Penjelasan :

1. Kelas BukuDAOImpl mengimplementasikan interface BukuDAO. Ini adalah contoh polimorfisme karena kelas ini dapat digunakan di mana saja di mana BukuDAO diharapkan.
2. Method addBuku:
Implementasi dari metode addBuku di interface BukuDAO. Tidak ada contoh polimorfisme atau inheritance yang spesifik di sini.
3. Method updateBuku:
Implementasi dari metode updateBuku di interface BukuDAO. Tidak ada contoh polimorfisme atau inheritance yang spesifik di sini.
4. Method getAllBuku:
Implementasi dari metode getAllBuku di interface BukuDAO. Tidak ada contoh polimorfisme atau inheritance yang spesifik di sini.
5. Method deleteBuku:
Implementasi dari metode deleteBuku di interface BukuDAO. Tidak ada contoh polimorfisme atau inheritance yang spesifik di sini.

2.5 Buku Fiksi

Sebuah class yang mendapatkan inheritance / pewarisan dari kelas buku.

```
// Package declaration
package buku.afsha_alifia;

// Import statements
import java.util.Date;

// Inheritance dari Buku
// BukuFiksi extends Buku, menggunakan inheritance dari kelas Buku
public class BukuFiksi extends Buku {
    private String genre;

    // Constructor untuk kelas BukuFiksi yang memanggil constructor kelas Buku menggunakan "super"
    public BukuFiksi(int idBuku, String judulBuku, String kodeBuku, String pengarang, String penerbit, String isbn, Date createAt, Date updateAt, String genre) {
        super(idBuku, judulBuku, kodeBuku, pengarang, penerbit, isbn, createAt, updateAt);
        this.genre = genre;
    }

    // Getter untuk mendapatkan genre
    public String getGenre() {
        return genre;
    }

    // Setter untuk mengatur genre
    public void setGenre(String genre) {
        this.genre = genre;
    }
}
```

Penjelasan :

1. Package Declaration:

- ``package buku.afsha_alifia;``: Mendefinisikan paket (package) dari kelas ``BukuFiksi`` dan menunjukkan lokasi tempat kelas ini berada dalam struktur proyek.

2. Import Statements:

- ``import java.util.Date;``: Mengimpor kelas ``Date`` dari paket ``java.util`` untuk digunakan dalam deklarasi kelas.

3. Inheritance:

- ``BukuFiksi extends Buku;``: Menunjukkan bahwa kelas ``BukuFiksi`` merupakan turunan dari kelas ``Buku``, mengadopsi atribut dan metode dari kelas induknya.

4. Constructor:

- ``public BukuFiksi(int idBuku, String judulBuku, String kodeBuku, String pengarang, String penerbit, String isbn, Date createAt, Date updateAt, String genre) { ... }``: Constructor kelas ``BukuFiksi`` yang memanggil constructor kelas ``Buku`` menggunakan kata kunci ``super``, inisialisasi atribut kelas ini, serta atribut kelas induknya.

5. Getter and Setter:

- ``public String getGenre() { return genre; }``: Getter untuk mendapatkan nilai dari atribut ``genre``.

- ``public void setGenre(String genre) { this.genre = genre; }``: Setter

untuk mengatur nilai atribut `genre`.

2.6 Interface BukuDAO

BukuDAO (Data Access Object) berfungsi sebagai perantara antara aplikasi dan database untuk operasi-operasi terkait buku..

```
author : Afsha alifia putri  
nim : 31402200031  
  
Berikan penjelasan kode ini baris perbaris dengan komentar, bagian interface  
/*  
*/  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.sql.Timestamp;  
import java.text.ParseException;  
import java.util.ArrayList;  
import java.util.List;  
  
// Interface BukuDAO  
// Digunakan sebagai kerangka kerja untuk kelas yang akan mengakses dan memanipulasi data buku dalam database  
  
public interface BukuDAO {  
    // Metode untuk menambahkan buku ke database  
    void addBuku(Buku buku);  
  
    // Metode untuk memperbarui informasi buku di database  
    void updateBuku(Buku buku);  
  
    // Metode untuk mendapatkan daftar semua buku dari database  
    List<Buku> getAllBuku() throws ParseException;  
  
    // Metode untuk menghapus buku dari database berdasarkan ID  
    void deleteBuku(int idBuku);  
}
```

Gambar 1. 22 BukuDAO Interface

Penjelasan :

Interface:

1. Interface BukuDAO digunakan sebagai kontrak yang harus diikuti oleh kelas-kelas yang bertanggung jawab untuk berinteraksi dengan data buku di database.
2. Metode-metode di dalam interface ini mencakup operasi dasar seperti menambahkan, memperbarui, mengambil semua data, dan menghapus buku.
3. Interface memberikan kerangka kerja yang diperlukan untuk implementasi kelas-kelas DAO (Data Access Object) yang akan berkomunikasi dengan database. Implementasi dari interface ini dapat bervariasi tergantung pada jenis database atau penyimpanan data yang digunakan.

2.7 DbManager

DbManager digunakan untuk mengelola koneksi ke database, sehingga aplikasi dapat berinteraksi dengan basis data..

```

package buku.efshe_alifia;

/*
 * author : Afshe alifia putri
 * nim : 52602200031
 * berikan penjelasan kode ini baris perbaris dengan komentar
 */

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

// Kelas DatabaseManager
// Digunakan untuk mengelola koneksi ke database

public class DatabaseManager {
    // Metode getConnection
    // Digunakan untuk mendapatkan koneksi ke database
    public static Connection getConnection() throws SQLException {
        // Menggunakan DriverManager untuk mendapatkan koneksi dengan parameter URL, username, dan password
        // URL: "jdbc:mysql://localhost/db_buku"
        // Username: "root"
        // Password: "" (kosong)
        return DriverManager.getConnection("jdbc:mysql://localhost/db_buku", "root", "");
    }
}

```

Gambar 1. 23 DB Manager

Penjelasan :

1. Database Connection:

- a. Kelas DatabaseManager menyediakan metode statis getConnection untuk mendapatkan koneksi ke database.
- b. Koneksi ke database MySQL dilakukan menggunakan JDBC (Java Database Connectivity).
- c. URL koneksi MySQL: "jdbc:mysql://localhost/db_buku"
- d. Username MySQL: "root"
- e. Password MySQL: "" (kosong)

2. Exception Handling:

- a. Metode getConnection mendeklarasikan bahwa dapat melempar SQLException, dan pemanggilannya di dalam kode yang menggunakan koneksi ini harus menangani pengecualian tersebut.

2.8 DateUtils

Date Utils menyediakan fungsi-fungsi untuk manipulasi tanggal seperti konversi dari Date ke String dan sebaliknya.


```

nim : 32602200031

berikan penjelasan kode ini baris perbaris dengan komentar
*/

// Kelas DateUtils
// Digunakan untuk melakukan konversi antara objek Date dan String
public class DateUtils {

    // Konstruktor private
    // Dideklarasikan private agar kelas ini tidak dapat diinstansiasi secara langsung
    private DateUtils() {
        // Private constructor to prevent instantiation
    }

    // Metode dateToString
    // Menerima objek Date dan mengembalikan representasi String dalam format "yyyy-MM-dd HH:mm:ss"
    public static String dateToString(Date date) {
        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        return dateFormat.format(date);
    }

    // Metode stringToDate
    // Menerima String yang mewakili tanggal dalam format "yyyy-MM-dd HH:mm:ss"
    // dan mengembalikan objek Date
    // Melempar ParseException jika ada masalah saat parsing tanggal
    public static Date stringToDate(String dateString) throws ParseException {
        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        return dateFormat.parse(dateString);
    }
}

```

Gambar 1. 24 DateUtils

Penjelasan :

1. Utility Class:

- a. Kelas DateUtils adalah kelas utilitas yang menyediakan metode-metode untuk konversi antara objek Date dan String dalam format tertentu.
- b. Konstruktor dinyatakan sebagai private untuk mencegah instansiasi langsung dari kelas ini.

2. Date Formatting:

- a. Metode dateToString menerima objek Date dan mengembalikan representasi String dalam format "yyyy-MM-dd HH:mm:ss".
- b. Metode stringToDate menerima String yang mewakili tanggal dalam format yang sama dan mengembalikan objek Date.
- c. Menggunakan kelas SimpleDateFormat untuk melakukan konversi tanggal ke String dan sebaliknya.

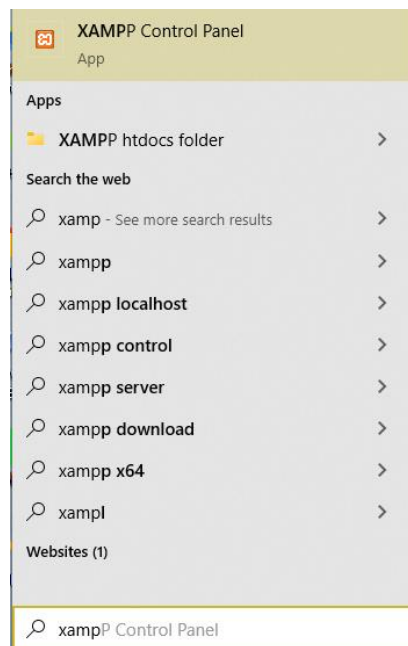
3. Exception Handling:

- a. Metode stringToDate melempar ParseException jika ada masalah saat parsing tanggal. Pemanggilan metode ini di dalam kode yang menggunakannya harus menangani pengecualian ini.

BAB III Implementasi Program

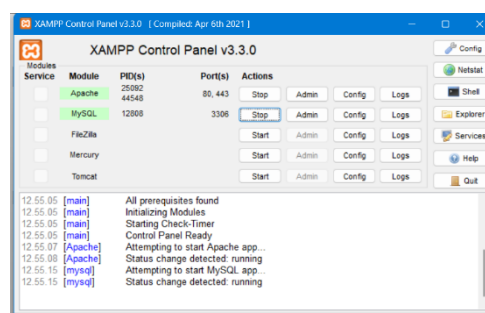
3.1 Menjalankan Program

1. Buka Xampp



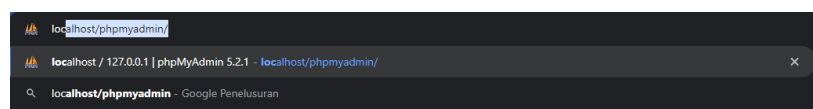
Gambar 1. 25 Open Xampp

2. Start apache & mysql, jika sudah berhasil sukses hijau



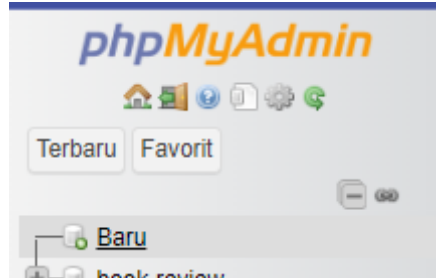
Gambar 1. 26 Sukses start xampp

3. Buka chrome, akses localhost



Gambar 1. 27 Akses localhost

4. Lalu klik baru, untuk buat db_buku



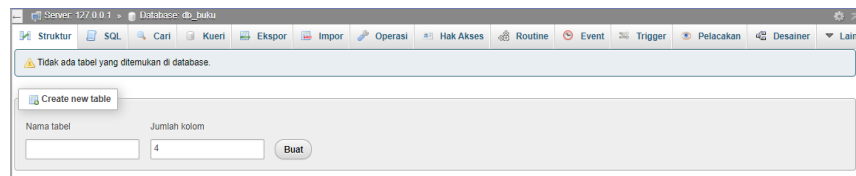
Gambar 1. 28 menuju ke buat db

5. Buat db baru dengan nama db_buku, lalu klik buat



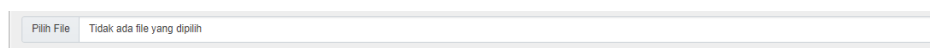
Gambar 1. 29 Buat db_buku

6. klik impor table_buku, jika berhasil masuk ke Langkah selanjutnya



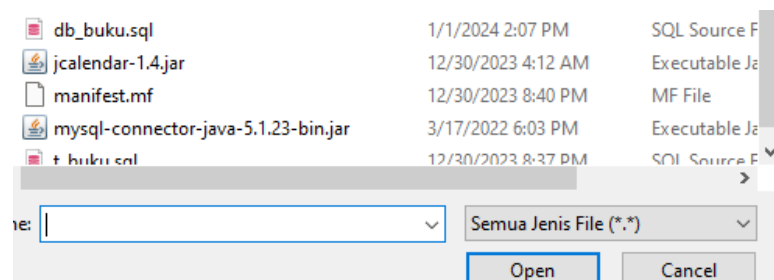
Gambar 1. 30 Menuju impor tabel

7. Lalu klik pilih file, untuk memilih



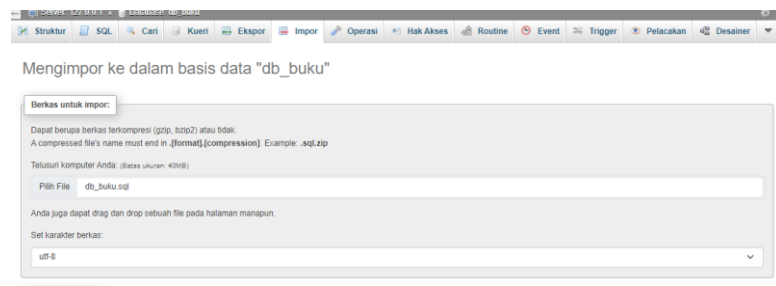
Gambar 1. 31 Klik file menuju direktori

8. Arahkan ke file db_buku.sql, lalu klik Open,



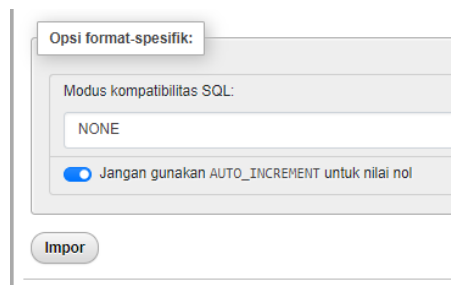
Gambar 1. 32 Pilih file, klik ope

9. Maka akan terpilih file db_buku



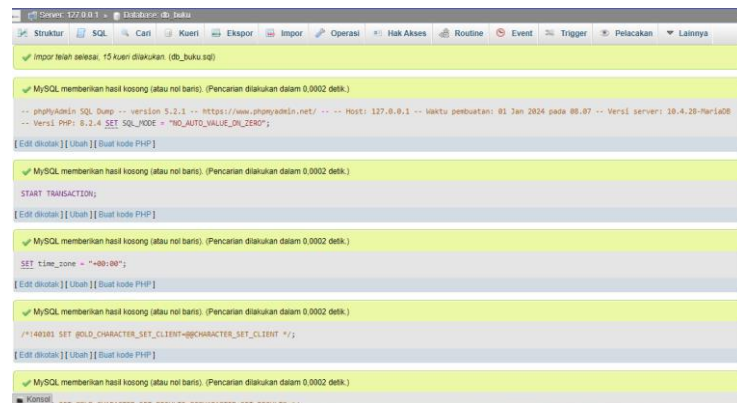
Gambar 1. 33 file terpilih

10. Setelah itu scroll kebawah untuk klik impor



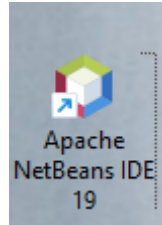
Gambar 1. 34 klik impor

11. Jika telah berhasil, maka import tabel_buku dan pembuatan db buku berhasil sukses



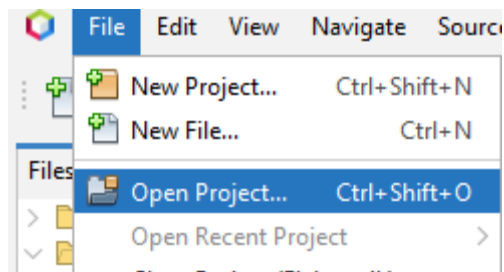
Gambar 1. 35 Sukses impor tabel dan buat db

12. Untuk menjalankan program klik neatbeans

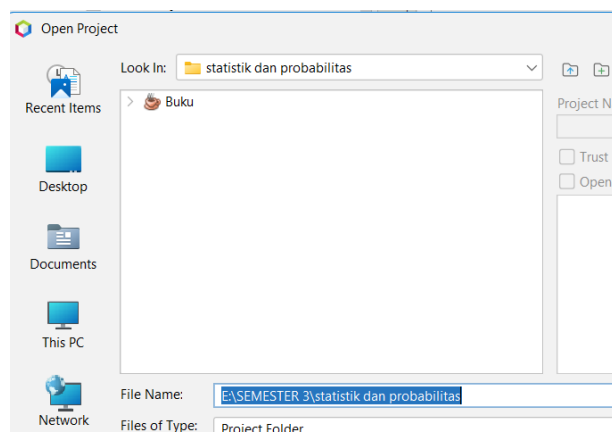


Gambar 2. 1 Open Neatbeans

13. Open new project, arahkan ke elektronik

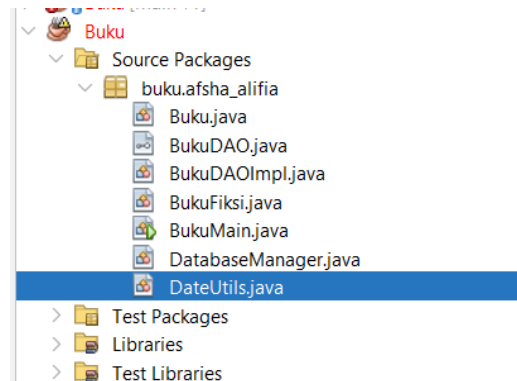


Gambar 2. 2 Open Projects



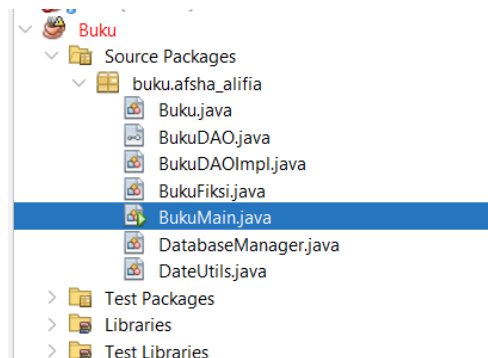
Gambar 2. 3 Arahkan ke projec

14. Buka src/buku/afsha_alifia



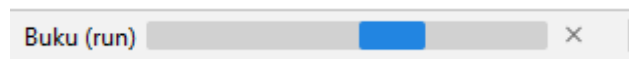
Gambar 2. 4 Open struktur project

15. Klik file BukuMain.java, lalu klik kanan file atau shift + f6



Gambar 2. 5 Run program

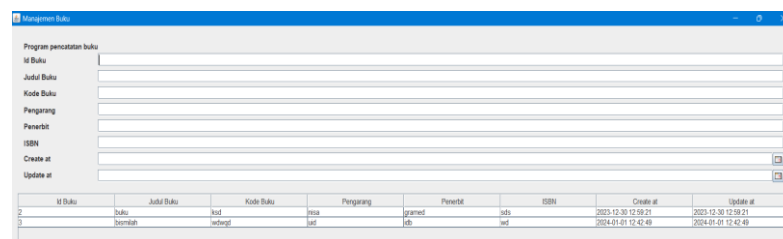
16. Tunggu program muncul



Gambar 2. 6 Tunggu program muncul

3.2 Menambahkan Buku

1. Isi semua data, lalu klik tambah



Gambar 2. 7 Tambah Buku

- Jika sudah, klik salah satu item data dari tabel, coba edit di kolom text, lalu klik ubah

The screenshot shows the 'Managemen Buku' application window. The top section is the 'Program pencatatan buku' form, which includes fields for 'Id Buku', 'Judul Buku', 'Kode Buku', 'Pengarang', 'Penerbit', 'ISBN', 'Create at', and 'Update at'. Below the form is a table with the following columns: 'Id Buku', 'Judul Buku', 'Kode Buku', 'Pengarang', 'Penerbit', 'ISBN', 'Create at', and 'Update at'. The table contains two rows of data.

	Id Buku	Judul Buku	Kode Buku	Pengarang	Penerbit	ISBN	Create at	Update at
2	Buku	led	ntsa	gramed	led	led	2023-12-30 12:49:21	2023-12-30 12:49:21
3	Isimallah	ledbergit	led	led	led	led	2024-01-01 12:42:49	2024-01-01 12:42:49

Gambar 2. 8 Buku sebelum dirubah

The screenshot shows the 'Managemen Buku' application window. The top section is the 'Program pencatatan buku' form, which includes fields for 'Id Buku', 'Judul Buku', 'Kode Buku', 'Pengarang', 'Penerbit', 'ISBN', 'Create at', and 'Update at'. Below the form is a table with the following columns: 'Id Buku', 'Judul Buku', 'Kode Buku', 'Pengarang', 'Penerbit', 'ISBN', 'Create at', and 'Update at'. The table contains two rows of data.

	Id Buku	Judul Buku	Kode Buku	Pengarang	Penerbit	ISBN	Create at	Update at
2	Buku	led	ntsa	gramed	led	led	2023-12-30 12:49:21	2023-12-30 12:49:21
3	Isimallah	ledbergit	led	led	led	led	2024-01-01 12:42:49	2024-01-01 12:42:49

Gambar 1. 36 Buku yang akan dirubah

- Buku di tabel akan berubah setelah diklick ubah

The screenshot shows the 'Managemen Buku' application window. The top section is the 'Program pencatatan buku' form, which includes fields for 'Id Buku', 'Judul Buku', 'Kode Buku', 'Pengarang', 'Penerbit', 'ISBN', 'Create at', and 'Update at'. Below the form is a table with the following columns: 'Id Buku', 'Judul Buku', 'Kode Buku', 'Pengarang', 'Penerbit', 'ISBN', 'Create at', and 'Update at'. The table contains two rows of data.

	Id Buku	Judul Buku	Kode Buku	Pengarang	Penerbit	ISBN	Create at	Update at
2	Buku	led	ntsa	gramed	led	led	2023-12-30 12:49:21	2023-12-30 12:49:21
3	Isimallah	ledbergit	led	led	led	led	2024-01-01 12:42:49	2024-01-01 12:42:49

Gambar 1. 37 Buku berhasil dirubah

4. Jika ingin menghapus item buku, klick hapus maka buku akan terhapus dari tabel

The screenshot shows a web application window titled "Management buku". It contains a form for adding a new book with the following fields: "Id Buku", "Judul Buku", "Kode Buku", "Pengarang", "Penerbit", "ISBN", "Create at", and "Update at". The "Create at" and "Update at" fields are pre-filled with "30 Dec 2023". Below the form is a table with 10 columns: "Id Buku", "Judul Buku", "Kode Buku", "Pengarang", "Penerbit", "ISBN", "Create at", and "Update at". The table contains one row of data with the following values: "1", "Jurnal Baru", "123456", "John Doe", "ABC", "123456789", "2021-01-01 12:34:56", and "2021-01-01 12:34:56".

	Id Buku	Judul Buku	Kode Buku	Pengarang	Penerbit	ISBN	Create at	Update at
1		Jurnal Baru	123456	John Doe	ABC	123456789	2021-01-01 12:34:56	2021-01-01 12:34:56

Gambar 2. 9 Buku berhasil dihapus