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.

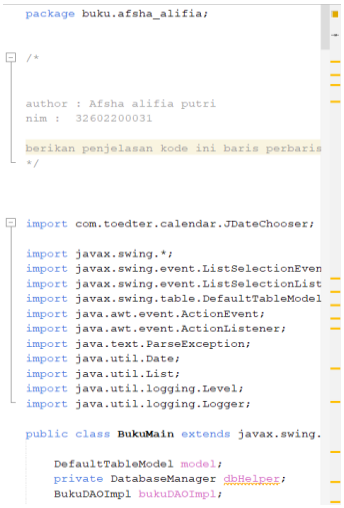
1.2 Tujuan

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 File Utama (BukuMain)

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



```
package buku.afsha_alifia;

/*

author : Afsha alifia putri
nim :  32602200031

berikan penjelasan kode ini baris perbaris
*/

import com.toedter.calendar.JDateChooser;

import javax.swing.*;
import javax.swing.event.ListSelectionEven
import javax.swing.event.ListSelectionList
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.

    DefaultTableModel model;
    private DatabaseManager dbHelper;
    BukuDAOImpl bukuDAOImpl;
```

Gambar 1. 1 File Utama MainBuku

```
    DefaultTableModel model;
    private DatabaseManager dbHelper;
    BukuDAOImpl bukuDAOImpl;

    // Variables declaration - do not modify
    JButton btn_hapus;
    JButton btn_keluar1;
    JButton btn_tambah;
    JButton btn_ubah;
    JTextField id_buku;
    JTextField isbn;
    JLabel jlabelProgram;
    JLabel jLabel1;
    JLabel jLabel10;
    JLabel jLabel2;
    JLabel jLabel3;
    JLabel jLabel4;
    JLabel jLabel5;
    JLabel jLabel6;
    JLabel jLabel7;
    JLabel jLabel8;
    JLabel jLabel9;
    JScrollPane jScrollPane1;
    JTextField judul_buku;
    JTextField kode_buku;
    JTextField penerbit;
    JTextField pengarang;
    JTable tabel;
    JDateChooser create_at;
    JDateChooser update_at;
    // End of variables declaration//GEN-END
```

Gambar 1. 2 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(columnNames:judul, rowCount:0);
    tabel.setModel(dataModel: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(evt:e);
            } catch (ParseException ex) {
                Logger.getLogger(name:BukuMain.class.getName()).log(level:Level.SEVERE, msg:null, thrown:ex);
            }
        }
    });
```

Gambar 1. 3 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(name:BukuMain.class.getName()).log(level: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(name:BukuMain.class.getName()).log(level:Level.SEVERE, msg:null, thrown:ex);
        }
    }
});

// Listener untuk tombol keluar
btn_keluar1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        btn_keluarActionPerformed(evt:e);
    }
```

Gambar 1. 4 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. 5 BukuMain 5

```
                buku.getPengarang(),
                buku.getPenerbit(),
                buku.getIsbn(),
                DateUtils.dateToString(date:buku.getCreateAt()),
                DateUtils.dateToString(date:buku.getUpdateAt())
            };
            model.addRow(rowData:row);
        }
    }

    // Listener untuk tombol tambah
    private void btn_tambahActionPerformed(ActionEvent evt) throws ParseException {
        // Mendapatkan data dari inputan 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 menyertakan ID
        Buku buku = new Buku(idBuku: 0, judulBuku: judul, kodeBuku: kode, pengarang: pengarangBuku, penerbit: penerbitBuku, isbn: isbnBuku, createAt: createDate, updateAt: createDate

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

        // Menampilkan ulang data pada tabel
        tampilkan();

        // Mengosongkan inputan
```

Gambar 1. 6 BukuMain 6

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

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

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

        // Mendapatkan data dari inputan 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 updateDate = new Date();

        // Membuat objek buku
        Buku buku = new Buku(idBuku: id, judulBuku: judul, kodeBuku: kode, pengarang: pengarangBuku, penerbit: penerbitBuku, isbn: isbnBuku, createAt: updateDate, updateAt: updateDat
```

Gambar 1. 7 BukuMain 7

Gambar 1. 8 BukuMain 8

Gambar 1. 9 BukuMain 9

Gambar 1. 10 BukuMain 10

2.2 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
nim :  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;
```

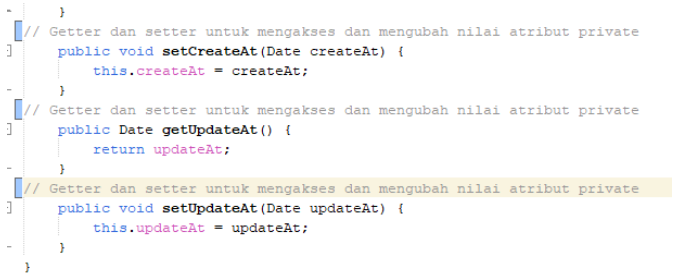
Gambar 1. 11 Kelas Buku 1

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

Gambar 1. 13 Buku 3



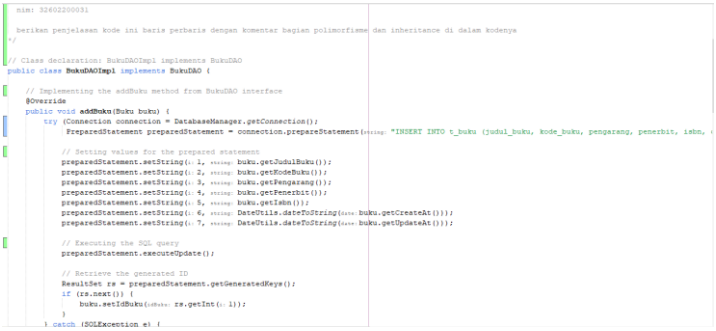
Gambar 1. 14 Buku 4

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.

2.3 Kelas BukuDAOImpl

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

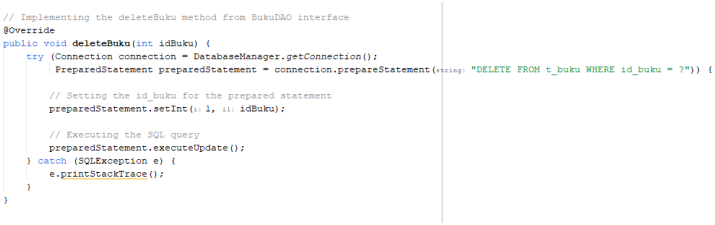


Gambar 1. 15 Kelas BukuDAOImpl 1

Gambar 1. 16 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(
                    idBuku: resultSet.getInt(1),
                    judulBuku: resultSet.getString(2),
                    kodeBuku: resultSet.getString(3),
                    pengarang: resultSet.getString(4),
                    penerbit: resultSet.getString(5),
                    isbn: resultSet.getString(6),
                    createAt: DateUtils.stringToDate(dateString: resultSet.getString(7)),
                    updateAt: DateUtils.stringToDate(dateString: 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. 17 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. 18 BukuDAOImpl 4

2.4 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`.

2.5 Interface BukuDAO

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

Gambar 1. 19 BukuDAO Interface

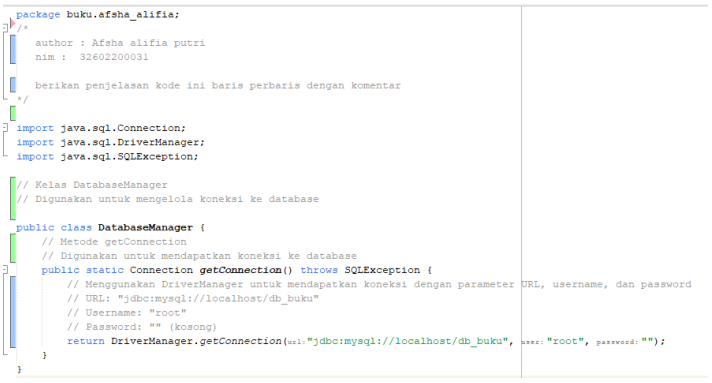
Penjelasan :

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.

2.6 DbManager

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

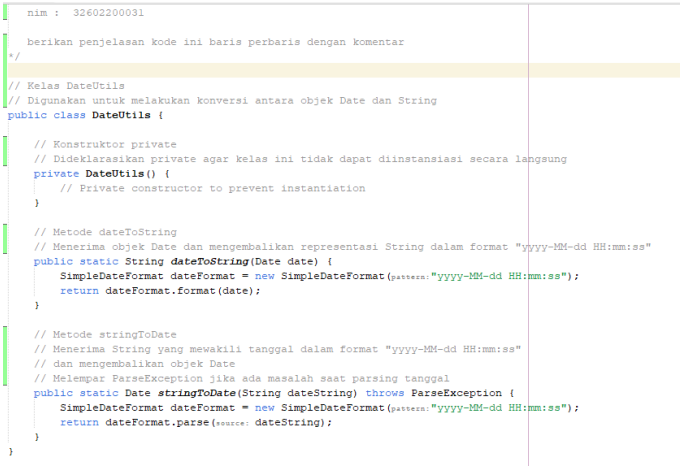


```
package buku.afsha_alifia;
/*
   author : Afsha alifia putri
   nim :  32602200031

   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(url:"jdbc:mysql://localhost/db_buku", user:"root", password:"");
    }
}
```

Gambar 1. 20 DB Manager

2.7 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(pattern:"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(pattern:"yyyy-MM-dd HH:mm:ss");
        return dateFormat.parse(source: dateString);
    }
}
```

Gambar 1. 21 DateUtils

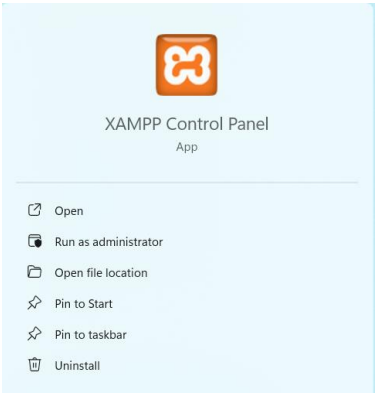
Penjelasan :

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.

BAB III Implementasi Program

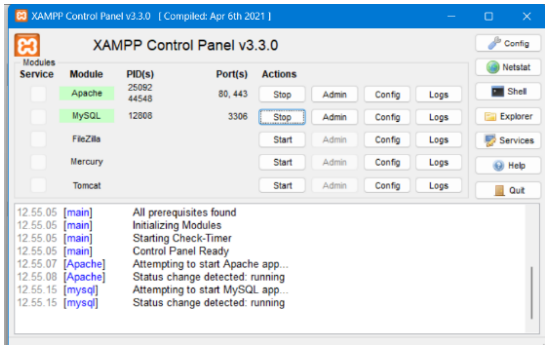
3.1 Menjalankan Program

1. Buka Xampp



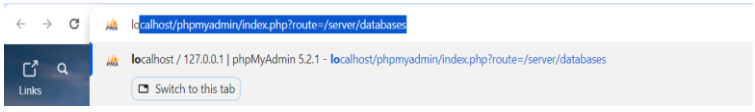
Gambar 1. 22 Open Xampp

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



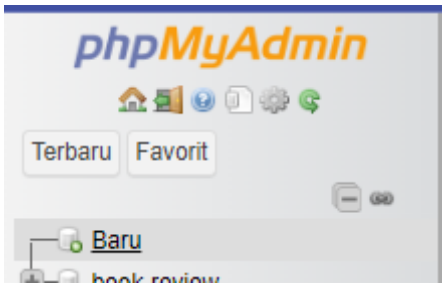
Gambar 1. 23 Sukses start xampp

3. Buka chrome, akses localhost



Gambar 1. 24 Akses localhos

4. Lalu klick baru, untuk buat db_buku

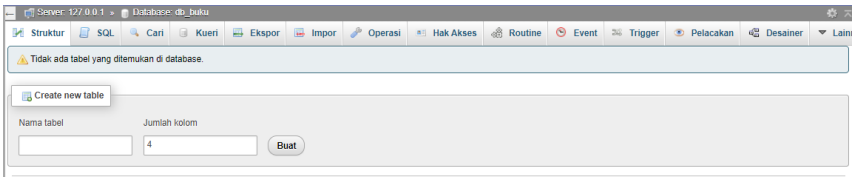


Gambar 1. 25 menuju ke buat db

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

Gambar 1. 26 Buat db_buku

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

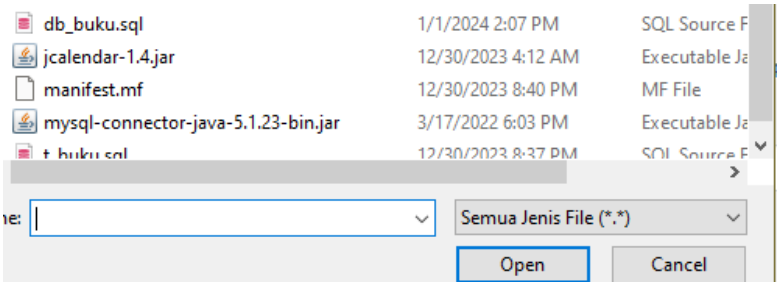


Gambar 1. 27 Menuju impor tabel

7. Lalu klick pilih file, untuk memilih

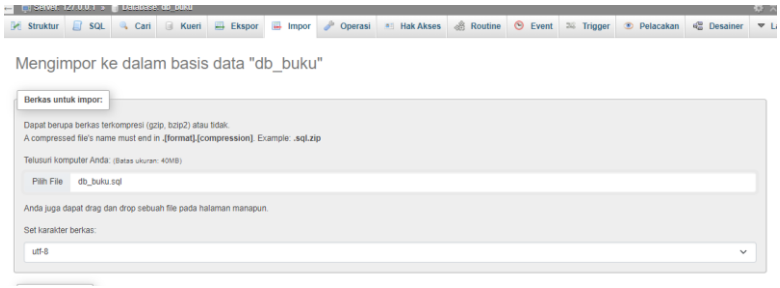
Gambar 1. 28 Klick file menuju direktori

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



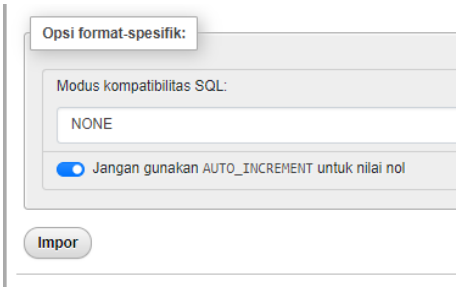
Gambar 1. 29 Pilih file, klick ope

9. Maka akan terpilih file db_buku



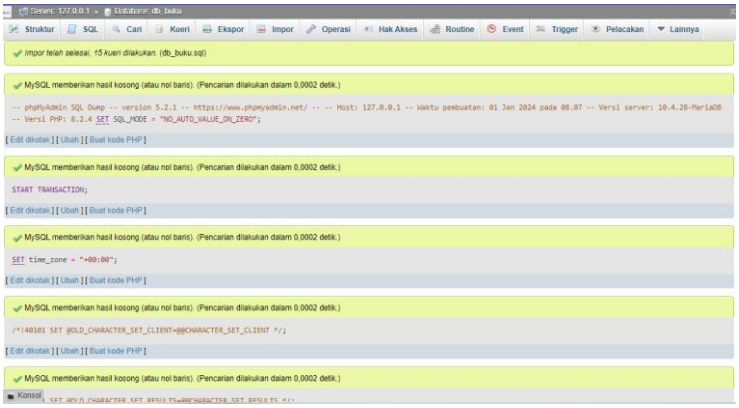
Gambar 1. 30 file terpilih

10. Setelah itu scroll kebawah untuk klick impor



Gambar 1. 31 klick impor

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

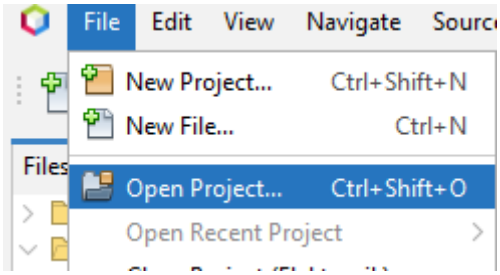


Gambar 1. 32 Sukses impor tabel dan buat db

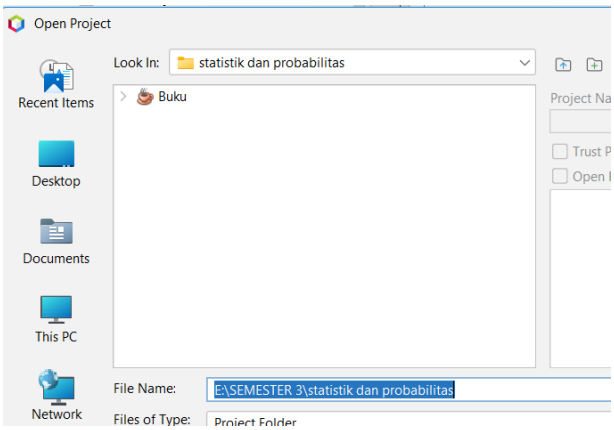
12. Untuk menjalankan program klick neatbeans

Gambar 2. 1 Open Neatbeans

13. Open new project, arahkan ke elektronik

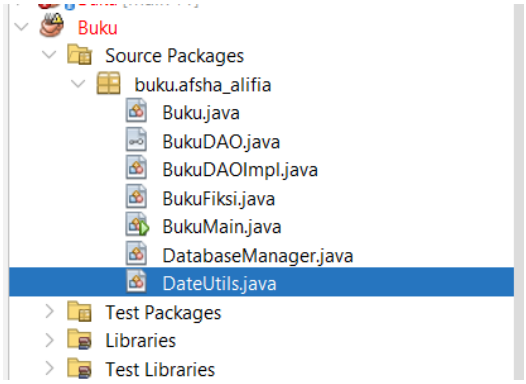


Gambar 2. 2 Open *Projects*



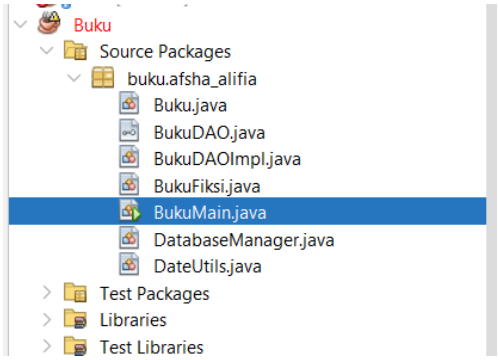
Gambar 2. 3 Arahkan ke *projects*

14. Buka src/buku/afsha_alifia



Gambar 2. 4 Open struktur *project*

15. Klick file BukuMain.java, lalu klick kanan run file



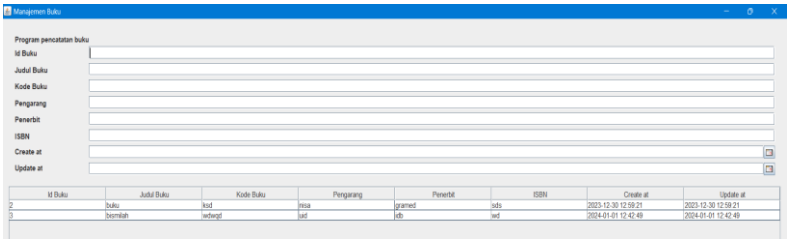
Gambar 2. 5 Run program

16. Tunggu program muncul

Gambar 2. 6 Tunggu program muncul

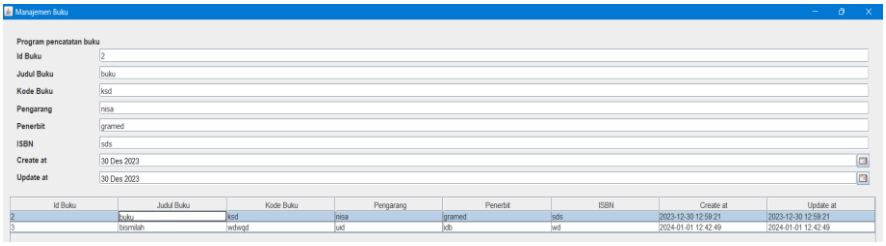
3.2 Menambahkan Buku

1. Isi semua data, lalu klick tambah

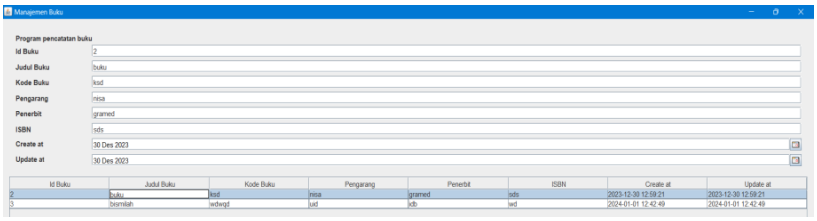


Gambar 2. 7 Tambah Buku

2. Jika sudah, klick salah satu item data dari tabel, coba edit di kolom text, lalu klick ubah

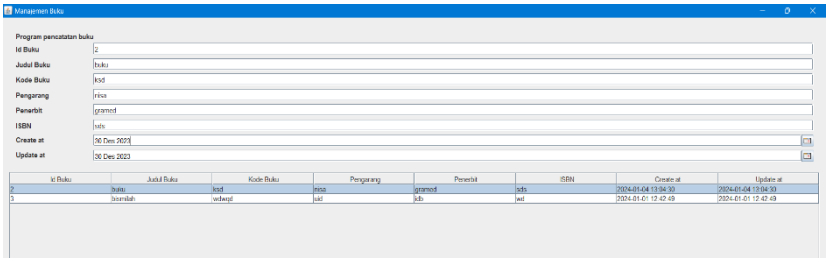


Gambar 2. 8 Buku sebelum dirubah



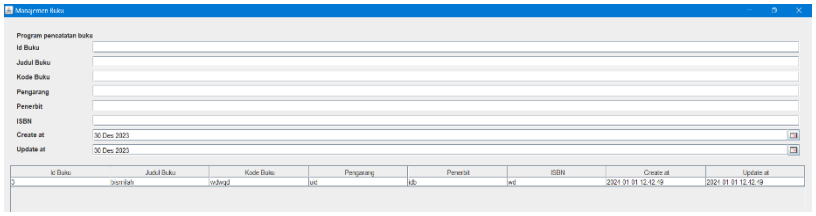
Gambar 1. 33 Buku yang akan dirubah

3. Buku di tabel akan berubah setelah diklick ubah



Gambar 1. 34 Buku berhasil dirubah

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



Gambar 2. 9 Buku berhasil dihapus