

PROJECT UAS PEMROGRAMAN BERORIENTASI OBJEK

SISTEM PERPUSTAKAAN DIGITAL

Dosen Pengampu : Taufik Ridwan S.T., M.Kom



Alya Rahma Khamila 2310631250040

Charina Olivia Tarigan 2310631250009

Dandi Permana 2310631250043

Hana Nabila 2310631250060

Hafiz Rahmansyah 2310631250092

Kelas 4B

PRODI SISTEM INFORMASI

FAKULTAS ILMU KOMPUTER

UNIVERSITAS SINGAPERBANGSA KARAWANG

TAHUN 2025

DAFTAR ISI

BAB I.....	1
PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	3
BAB II.....	4
PEMBAHASAN	4
2.1 Penjelasan Istilah Dalam Oop dan Contoh Penerapan.....	4
2.2 Perancangan Unified Model Language (UML)	9
2.3 Fitur - Fitur.....	16
2.4 Implementasi Kode Program dan Pengujian.....	19
2.5 Pembahasan Output Program.....	108
BAB III	116
KESIMPULAN.....	116
3.1 Kesimpulan	116

KATA PENGANTAR

Puji syukur kita panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga kami dapat menyelesaikan tugas yang berjudul “Laporan Program Terkait Penjualan Sepeda” ini tepat pada waktunya.

Adapun tujuan dari penyusunan laporan ini adalah untuk memenuhi tugas Ujian Akhir Semester (UAS) pada Mata Kuliah Pemrograman Berorientasi Objek. Selain itu, laporan ini juga bertujuan untuk menambah wawasan bagi para pembaca dan penulis mengenai pengimplementasian program berbasis GUI (Bahasa Java) pada sistem penjualan sepeda online.

Terlebih dahulu, kami mengucapkan terima kasih kepada Bapak Taufik Ridwan, S.T., M.T., selaku Dosen pengampu pada Mata Kuliah Pemrograman Berorientasi Objek yang telah memberikan tugas ini sehingga dapat menambah pengetahuan dan wawasan sesuai dengan bidang studi yang kami tekuni ini.

Tidak lupa juga kami mengucapkan terima kasih kepada seluruh pihak yang telah turut memberikan kontribusi dalam penyusunan Laporan Program Terkait Penjualan Sepeda ini. Tentunya, tidak akan bisa maksimal jika tidak mendapat dukungan dari berbagai pihak.

Karena keterbatasan pengetahuan maupun pengalaman maka kami yakin masih banyak kekurangan dalam laporan ini. Oleh karena itu, kami sangat mengharapkan kritik dan saran yang membangun dari pembaca demi kesempurnaan laporan ini. Kami berharap semoga laporan mengenai program berbasis GUI pada sistem penjualan sepeda online yang kami susun ini memberikan manfaat dan juga inspirasi untuk pembaca.

Karawang, Juni 2025

Kelompok 8

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi yang pesat telah mendorong berbagai sektor untuk mengadopsi sistem digital demi meningkatkan efisiensi dan kualitas pelayanan, tak terkecuali pada sektor pendidikan dan perpustakaan. Perpustakaan sebagai pusat informasi dan ilmu pengetahuan memiliki peran penting dalam mendukung kegiatan belajar, riset, dan literasi. Namun, pada kenyataannya, banyak perpustakaan konvensional masih bergantung pada proses manual dalam pengelolaan data buku, pencatatan peminjaman, serta pelayanan terhadap pengguna. Hal ini dapat menimbulkan berbagai permasalahan, seperti hilangnya data, keterlambatan pengembalian, kesulitan dalam pencarian koleksi, serta keterbatasan ruang dan waktu dalam akses terhadap informasi.

Melihat permasalahan tersebut, dibutuhkan suatu sistem informasi perpustakaan yang modern, fleksibel, dan dapat diakses dengan lebih efisien. Oleh karena itu, dikembangkanlah sebuah sistem perpustakaan digital berbasis Java GUI (Graphical User Interface) yang terintegrasi dengan database MySQL. Sistem ini dirancang untuk mendukung proses digitalisasi koleksi eBook, mempermudah pengelolaan data oleh admin, serta memberikan kemudahan akses bagi pengguna dalam menjelajahi, meminjam, dan membaca buku secara daring. Sistem ini menggunakan pendekatan pemrograman berbasis objek (Object-Oriented Programming/OOP) yang mencakup penggunaan class, inheritance, encapsulation, polymorphism, serta exception handling, sehingga struktur kode menjadi modular, terstruktur, dan mudah dikembangkan lebih lanjut.

Dalam sistem ini, terdapat dua jenis pengguna utama yaitu admin dan pengguna umum (user). Admin memiliki akses untuk menambahkan koleksi eBook dengan informasi lengkap, seperti judul buku, penulis, penerbit, sinopsis, harga sewa, serta unggahan gambar buku. Admin juga dapat mengelola data pengguna dan melihat riwayat peminjaman seluruh pengguna. Di sisi lain, pengguna umum dapat melihat koleksi buku yang tersedia, melakukan pencarian berdasarkan judul atau penulis, serta meminjam eBook yang akan otomatis masuk ke dalam riwayat mereka. Masa peminjaman buku dibatasi secara sistematis, di mana setelah waktu sewa berakhir, eBook

tidak dapat diakses lagi oleh pengguna. Fitur ini meningkatkan kedisiplinan dan efisiensi dalam sirkulasi koleksi digital.

Dari sisi antarmuka, sistem ini dirancang dengan tampilan modern dan responsif menggunakan komponen GUI berbasis Java Swing. Setiap tampilan menyesuaikan peran pengguna dan dirancang untuk mudah digunakan, bahkan oleh pengguna awam sekalipun. Proses login, peminjaman, pengelolaan buku, dan pelacakan riwayat dilakukan dengan pendekatan event-driven yang intuitif. Tidak hanya itu, sistem juga mendukung upload dan pratinjau gambar buku, serta pencarian dinamis yang memudahkan pengguna menemukan buku yang mereka inginkan dengan cepat.

Dengan hadirnya sistem perpustakaan digital ini, diharapkan pengelolaan perpustakaan menjadi lebih efisien, transparan, dan fleksibel. Pengguna dapat mengakses koleksi buku tanpa batasan ruang dan waktu, sementara admin dapat mengelola data secara lebih akurat dan terorganisir. Sistem ini juga menjadi contoh penerapan nyata konsep pemrograman berbasis objek dan rekayasa perangkat lunak dalam menyelesaikan permasalahan nyata di dunia pendidikan dan literasi digital.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka rumusan masalah dari pengembangan sistem perpustakaan digital ini dapat dirumuskan sebagai berikut:

1. Bagaimana merancang dan mengimplementasikan sistem perpustakaan digital berbasis Java GUI yang dapat memfasilitasi pengelolaan eBook secara efisien oleh admin dan pengguna?
2. Bagaimana cara mengatur hak akses yang berbeda antara admin dan pengguna umum dalam sistem perpustakaan?
3. Bagaimana sistem dapat mengelola proses peminjaman eBook secara otomatis, termasuk membatasi akses buku setelah masa pinjam berakhir?
4. Bagaimana merancang antarmuka sistem (GUI) yang interaktif, informatif, dan mudah digunakan bagi admin maupun pengguna?
5. Bagaimana mengintegrasikan database MySQL ke dalam sistem untuk menyimpan dan mengelola data buku, pengguna, serta riwayat peminjaman secara terstruktur dan aman?

1.3 Tujuan

Adapun tujuan dari pembangunan sistem perpustakaan digital berbasis Java GUI ini adalah sebagai berikut:

1. Membangun sebuah sistem informasi perpustakaan digital yang mampu mengelola koleksi eBook, peminjaman, dan riwayat pinjam secara terpusat dan efisien.
2. Mengimplementasikan konsep pemrograman berbasis objek (OOP) dalam pengembangan sistem yang modular, terstruktur, dan mudah dikembangkan.
3. Memberikan fitur otentikasi login dengan pembagian hak akses antara admin dan pengguna biasa, agar setiap aktor hanya dapat mengakses fitur yang sesuai.
4. Memfasilitasi admin dalam proses penambahan data buku secara lengkap, termasuk informasi buku dan gambar sampul.
5. Memungkinkan pengguna melakukan pencarian dan peminjaman eBook secara mandiri, serta mengakses riwayat pinjamannya sendiri.
6. Menerapkan mekanisme otomatis yang membatasi akses pengguna terhadap eBook setelah masa pinjam habis.
7. Membangun antarmuka sistem menggunakan Java Swing yang responsif, menarik, dan ramah pengguna (user-friendly).

BAB II

PEMBAHASAN

2.1 Penjelasan Istilah Dalam Oop dan Contoh Penerapan

Object-Oriented Programming (OOP) atau Pemrograman Berbasis Objek adalah paradigma pemrograman yang berfokus pada penggunaan objek dan class untuk memodelkan solusi dari suatu masalah. Dalam sistem perpustakaan digital ini, seluruh struktur program dibangun berdasarkan prinsip-prinsip OOP yang meliputi: Class, Object, Enkapsulasi, Pewarisan (Inheritance), Polimorfisme (Polymorphism), dan Exception Handling. Berikut penjelasan rinci mengenai penerapan konsep tersebut:

Berikut adalah penjelasan setiap materi OOP Java disertai dengan minimal lima contoh implementasi nyata dari sistem perpustakaan digital Anda. Penjelasan ini disusun secara formal dan dapat digunakan untuk keperluan laporan atau dokumentasi teknis.

Implementasi Konsep OOP Java dalam Sistem Perpustakaan Digital

1. Class dan Object

Class adalah struktur dasar yang merepresentasikan entitas dalam program. Object adalah instansiasi dari class tersebut.

Contoh implementasi:

- Book book1 = new Book(1, "Laskar Pelangi", "Andrea Hirata", 3);
- User user = new User(2, "Ali", "ali@gmail.com", "1234");
- Admin admin = new Admin(0, "Administrator", "admin@perpus.com");
- Borrowing pinjam = new Borrowing(1, user, book1, new Date());
- DBConnection conn = new DBConnection();

2. Method

Method merupakan perilaku atau fungsi yang dimiliki oleh suatu class. Method digunakan untuk memproses data internal class.

Contoh implementasi:

- getTitle() dan setTitle() dalam Book.java.
- login(String email, String password) dalam UserController.java.
- updateStock(int delta) dalam Book.java.
- isExpired() dalam Borrowing.java.
- tampilanBuku() dalam AdminDashboard.java.

Setiap method bertanggung jawab atas satu tugas spesifik dan dapat dipanggil oleh objek terkait.

3. Package

Package digunakan untuk mengelompokkan class ke dalam struktur modular agar lebih mudah dalam pengelolaan dan pemeliharaan.

Contoh struktur dan isinya:

- package model → Book.java, User.java, Borrowing.java, Admin.java.
- package controller → BookController.java, UserController.java, BorrowingController.java.
- package view → LoginView.java, AdminDashboard.java, UserDashboard.java, BookListPanel.java.
- package utils → DBConnection.java, DateUtils.java.

Struktur ini mendukung arsitektur MVC (Model-View-Controller).

4. Enkapsulasi

Enkapsulasi adalah teknik membatasi akses langsung ke atribut dengan menggunakan modifier private dan menyediakan method akses publik.

Contoh implementasi:

- Field private String title pada Book.java dengan akses melalui getTitle() dan setTitle().

- Field private String email pada User.java dengan akses melalui getEmail() dan setEmail().
- Field private Date borrowDate pada Borrowing.java dengan akses getBorrowDate().
- Field private int stock pada Book.java dengan update melalui method updateStock(int delta).
- Field private String password pada User.java disembunyikan dan diakses hanya saat login.

Dengan enkapsulasi, data lebih aman dan hanya dapat dimodifikasi melalui mekanisme resmi.

5. Pewarisan (Inheritance)

Inheritance memungkinkan suatu class mewarisi atribut dan method dari class lain.

Contoh implementasi:

- class Admin extends User → Admin merupakan turunan dari User dan dapat memiliki tambahan method tertentu.
- class User memiliki method getEmail(), getName() yang dapat diakses oleh Admin.
- class BookCard extends JPanel dalam view → Pewarisan untuk komponen GUI kustom.
- Semua JFrame seperti AdminDashboard dan UserDashboard secara tidak langsung mewarisi dari class Frame.
- TambahBukuForm extends JDialog → Form tambahan mewarisi dialog Java Swing.

Pewarisan memungkinkan penggunaan kembali kode serta perluasan fungsionalitas.

6. Polymorphism

Polymorphism mengizinkan method yang sama digunakan untuk perilaku berbeda tergantung pada objeknya.

Contoh implementasi:

- Method tampilkanDashboard() berbeda antara AdminDashboard dan UserDashboard.
- Overriding method getRole() di User dan Admin.
- Komponen BookCard dapat dirender berbeda tergantung siapa pengguna yang membuka: admin atau user.
- Panel yang ditampilkan di JFrame berbeda antara admin (BookListPanel + BorrowingListPanel) dan user (BookListPanel + RiwayatPeminjamanPanel).
- Method actionPerformed dari ActionListener diterapkan dengan cara berbeda pada LoginView, TambahBukuForm, dan lainnya.

Polymorphism memperluas fleksibilitas dan mendukung prinsip desain reusable code.

7. Exception Handling

Exception digunakan untuk menangani error secara terstruktur tanpa menghentikan program secara mendadak.

Contoh implementasi:

- try-catch saat membuka koneksi database pada DBConnection.java.
- try-catch saat parsing data peminjaman dari database dalam BorrowingController.
- try-catch saat mengakses file gambar buku di BookCard.java.
- try-catch saat login gagal dalam LoginView.java.
- try-catch saat menambahkan buku baru, jika data tidak valid.

Dengan exception, sistem lebih tangguh terhadap kesalahan runtime.

8. GUI (Graphical User Interface)

Sistem menggunakan Java Swing untuk antarmuka grafis, memudahkan pengguna dalam berinteraksi.

Contoh implementasi:

- LoginView.java menggunakan JFrame, JLabel, JTextField, JPasswordField.
- AdminDashboard.java dan UserDashboard.java menampilkan panel berdasarkan peran pengguna.
- BookListPanel.java menampilkan daftar buku dengan tombol aksi.
- TambahBukuForm.java menampilkan form input untuk admin menambahkan buku.
- BookCard.java menampilkan informasi buku secara visual dan interaktif.

GUI Swing mendukung komponen visual yang kaya untuk pengalaman pengguna yang baik.

9. Event-Driven Programming

Sistem GUI berbasis event-driven artinya segala interaksi pengguna diproses sebagai event.

Contoh implementasi:

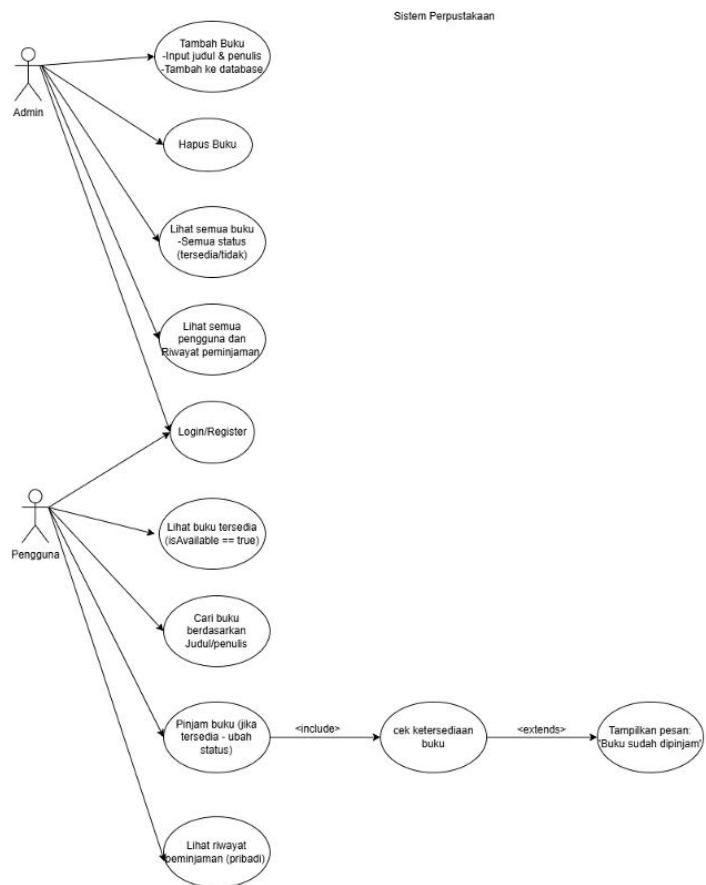
- Klik tombol “Login” ditangani oleh ActionListener di LoginView.java.
- Klik tombol “Tambah Buku” pada AdminDashboard membuka TambahBukuForm.java.
- Klik tombol “Pinjam” pada UserDashboard men-trigger peminjaman.
- Listener pada JComboBox untuk filter daftar buku.
- Event windowClosing pada JFrame untuk menyimpan log keluar pengguna.

Event-driven programming adalah dasar utama interaksi antarmuka pengguna modern.

Kesembilan konsep utama OOP Java telah diterapkan secara menyeluruh dan konsisten pada sistem perpustakaan digital yang dikembangkan. Setiap konsep tidak hanya diimplementasikan secara formal, namun juga terintegrasi ke dalam alur kerja aplikasi secara nyata dan fungsional, mulai dari arsitektur class hingga interaksi antarmuka pengguna.

2.2 Perancangan Unified Model Language (UML)

1. Use Case Diagram



Use Case Diagram untuk Sistem Perpustakaan melibatkan dua aktor utama, yaitu Admin dan Pengguna (User).

1. Aktor

Diagram ini memiliki dua aktor utama:

- Admin

Bertanggung jawab atas manajemen data buku dan pemantauan pengguna serta aktivitas peminjaman.

- Pengguna(User)

Pengguna umum yang dapat melakukan login, melihat dan meminjam buku, serta melihat riwayat peminjamannya sendiri.

2. Admin
 - a. Tambah Buku
 - Deskripsi: Admin dapat menambahkan buku ke sistem dengan menginput judul dan penulis, lalu data tersebut disimpan ke dalam database.
 - Tujuan: Menambahkan koleksi buku ke sistem perpustakaan.
 - b. Hapus Buku
 - Deskripsi: Admin dapat menghapus buku yang tidak lagi tersedia atau relevan dari database.
 - Tujuan: Memastikan data buku di sistem selalu mutakhir.
 - c. Lihat Semua Buku
 - Deskripsi: Admin dapat melihat seluruh data buku, baik yang tersedia maupun tidak.
 - Tujuan: Memantau status koleksi buku di perpustakaan.
 - d. Lihat Semua Pengguna & Riwayat Peminjaman
 - Deskripsi: Admin dapat melihat daftar pengguna yang terdaftar serta seluruh histori peminjaman buku.
 - Tujuan: Mengelola aktivitas pengguna dan memastikan penggunaan sistem yang tepat.
 - e. Login / Register
 - Deskripsi: Proses autentikasi untuk masuk ke sistem atau pendaftaran akun baru.
 - Tujuan: Mengamankan akses dan identifikasi peran pengguna.
3. Use Case untuk Pengguna
 - a. Login / Register
 - Deskripsi: Pengguna melakukan login atau registrasi sebelum mengakses sistem.
 - Tujuan: Untuk memperoleh hak akses ke fitur peminjaman dan riwayat peminjaman.

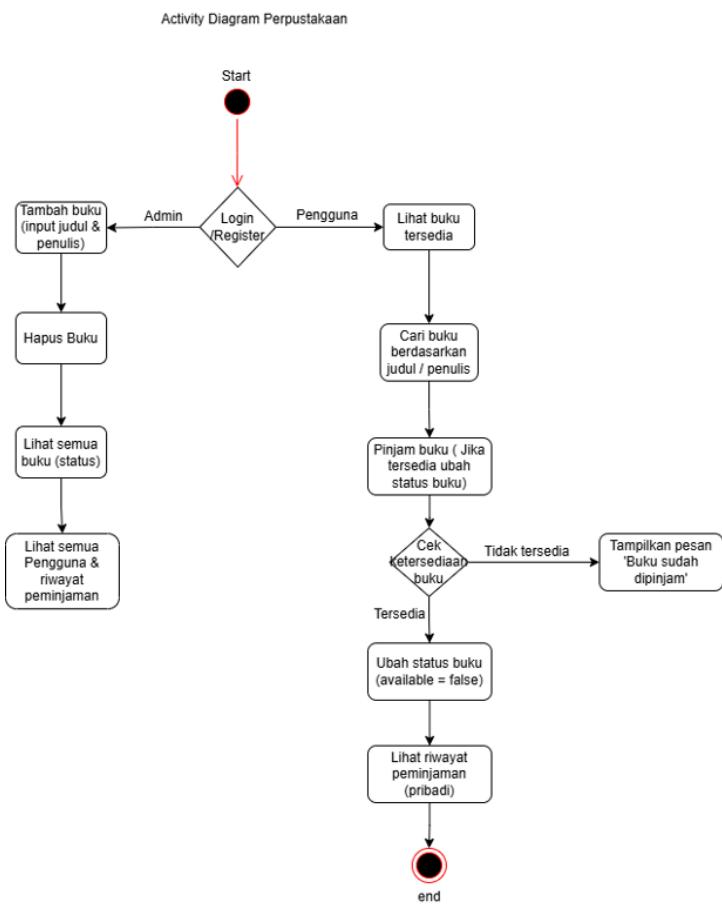
- b. Lihat Buku Tersedia
 - Deskripsi: Pengguna dapat melihat daftar buku yang statusnya "tersedia".
 - Tujuan: Menyaring buku yang dapat dipinjam.
- c. Cari Buku berdasarkan Judul / Penulis
 - Deskripsi: Pengguna dapat mencari buku berdasarkan informasi judul atau nama penulis.
 - Tujuan: Memudahkan pencarian koleksi buku secara efisien.
- d. Pinjam Buku
 - Deskripsi: Pengguna dapat meminjam buku jika tersedia. Jika buku tersedia, maka status buku akan diubah menjadi tidak tersedia.
 - Relasi Include:
 - o Cek Ketersediaan Buku: Sebelum meminjam, sistem akan memeriksa apakah buku masih tersedia.
 - o Tujuan: Memberikan layanan peminjaman buku secara digital.
- e. Lihat Riwayat Peminjaman (Pribadi)
 - Deskripsi: Pengguna dapat melihat riwayat peminjaman mereka sendiri.
 - Tujuan: Memantau aktivitas peminjaman yang pernah dilakukan.

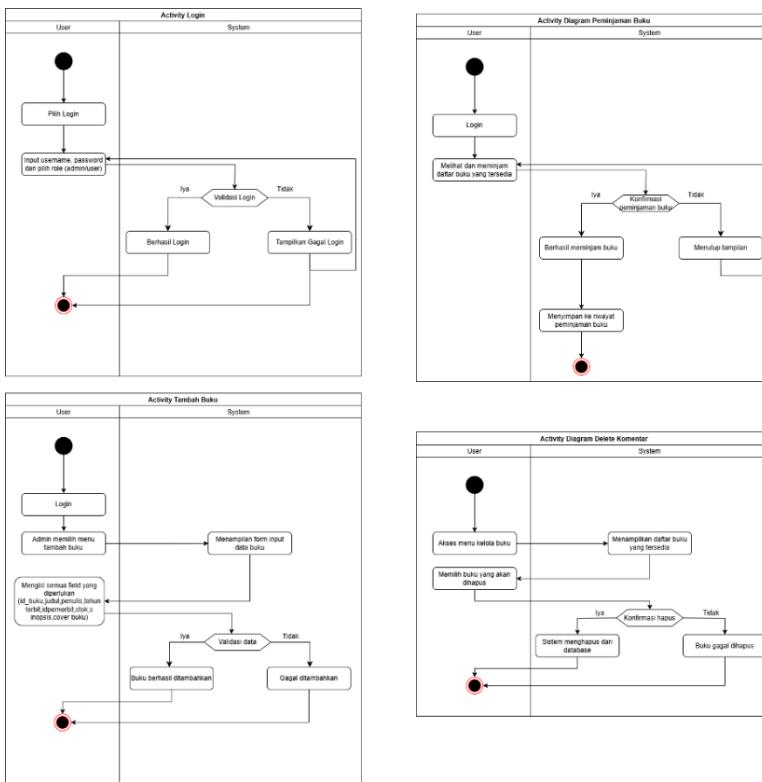
4. Relasi Antar Use Case

- Cek Ketersediaan Buku

Proses peminjaman buku untuk memastikan buku tersedia sebelum statusnya diubah dan proses pinjam dilakukan.

2. Activity Diagram

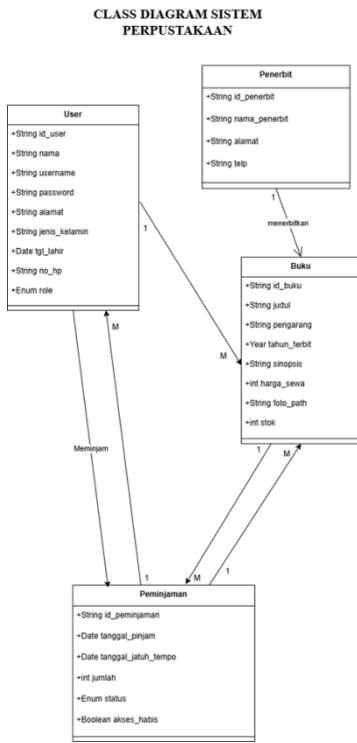




1. Login/Register merupakan awal dari semua aktivitas baik untuk Admin maupun Pengguna.
2. Admin memiliki hak untuk:
 - Tambah/Hapus Buku
 - Melihat semua status buku (tersedia/tidak)
 - Melihat semua pengguna dan riwayat peminjaman.
3. Pengguna hanya dapat:
 - Melihat buku yang tersedia
 - Mencari buku berdasarkan judul/penulis
 - Melakukan peminjaman jika buku tersedia
 - Melihat riwayat peminjaman pribadinya
4. Saat peminjaman dilakukan, dicek dulu ketersediaan buku:
 - Jika tersedia, status buku diubah

- Jika tidak tersedia, muncul pesan “Buku sudah dipinjam”.\\

3. Class Diagram



Class diagram ini menggambarkan struktur statis dari sistem perpustakaan digital, yang terdiri dari beberapa kelas utama: User, Buku, Peminjaman, dan Penerbit. Relasi antar kelas juga digambarkan untuk menunjukkan interaksi dan hubungan antar entitas.

1. Class: User

Mewakili pengguna sistem, baik sebagai admin maupun pengguna biasa.

Atribut:

- **id_user:** ID unik pengguna.
- **nama:** Nama lengkap pengguna.
- **username:** Nama pengguna untuk login.
- **password:** Kata sandi pengguna.
- **alamat:** Alamat tempat tinggal.

- jenis_kelamin: Jenis kelamin.
- tgl_lahir: Tanggal lahir.
- no_hp: Nomor telepon.
- role: Menentukan peran (Admin/Pengguna), disimpan dalam bentuk enum.

Relasi: 1 user dapat melakukan banyak peminjaman (1:M) ke class Peminjaman.

2. Class: Buku

Mewakili data buku yang tersedia di perpustakaan.

Atribut:

- id_buku: ID unik buku.
- judul: Judul buku.
- pengarang: Nama pengarang buku.
- tahun_terbit: Tahun terbit buku.
- sinopsis: Ringkasan isi buku.
- harga_sewa: Biaya sewa buku.
- foto_path: Path gambar sampul buku.
- stok: Jumlah stok buku yang tersedia.

Relasi: 1 buku dapat dipinjam oleh banyak peminjaman (1:M) ke class Peminjaman dan 1 penerbit dapat menerbitkan banyak buku (1:M) ke class Penerbit.

3. Class: Peminjaman

Mewakili transaksi peminjaman buku oleh pengguna.

Atribut:

- id_peminjaman: ID unik untuk transaksi peminjaman.
- tanggal_pinjam: Tanggal saat buku dipinjam.

- tanggal_jatuh_tempo: Batas akhir pengembalian buku.
- jumlah: Jumlah buku yang dipinjam.
- status: Status peminjaman (dipinjam, dikembalikan, terlambat, dll) dalam bentuk enum.
- akses_habis: Menandakan apakah akses buku (misalnya e-book) masih aktif atau sudah kedaluwarsa.

Relasi: M dimiliki oleh 1 user (M:1) dan M melibatkan 1 buku (M:1)

4. Class: Penerbit

Mewakili informasi tentang penerbit buku.

Atribut:

- id_penerbit: ID unik penerbit.
- nama_penerbit: Nama resmi penerbit.
- alamat: Alamat penerbit.
- telp: Nomor telepon penerbit.

Relasi: 1 penerbit dapat menerbitkan banyak buku (1:M)

Relasi Antar Kelas (Asosiasi):

- User – Peminjaman: Satu pengguna dapat memiliki banyak transaksi peminjaman.
- Buku – Peminjaman: Satu buku dapat dipinjam dalam banyak transaksi berbeda.
- Penerbit – Buku: Satu penerbit dapat menerbitkan banyak buku.

2.3 Fitur - Fitur

Fitur-fitur yang tersedia dalam sistem perpustakaan digital berbasis Java GUI + MySQL ini dirancang untuk mempermudah proses pengelolaan dan akses terhadap koleksi eBook secara digital. Sistem menyediakan dua jenis akses pengguna, yaitu Admin dan

Pengguna Umum, dengan hak dan tampilan fitur yang berbeda. Berikut penjelasan lengkap setiap fitur:

1. Login Multi-Role (Admin & Pengguna)

Sistem menyediakan halaman login yang dapat digunakan oleh dua peran utama, yaitu admin dan pengguna umum. Pengguna dapat memilih peran saat login, lalu memasukkan username dan password. Jika autentikasi berhasil, sistem akan mengarahkan pengguna ke dashboard sesuai dengan rolenya. Proses login dilengkapi dengan validasi dan pesan kesalahan apabila data tidak sesuai.

2. Manajemen Pengguna (khusus Admin)

Admin dapat melihat daftar seluruh pengguna terdaftar dalam sistem beserta riwayat peminjaman mereka. Data yang ditampilkan mencakup nama pengguna, ID, dan data peminjaman. Hal ini mempermudah admin dalam memantau aktivitas pengguna serta memastikan bahwa peminjaman berjalan sesuai aturan.

3. Manajemen Buku

Admin dapat mengelola data koleksi eBook dalam sistem. Pengelolaan ini meliputi:

- Menambahkan buku baru dengan informasi lengkap: judul, penulis, penerbit, sinopsis, harga sewa, dan gambar buku (upload).
- Melihat semua buku yang tersedia maupun tidak tersedia (dipinjam).
- Menghapus buku yang belum pernah dipinjam oleh pengguna.

4. Upload dan Tampilkan Gambar Buku

Setiap buku yang ditambahkan dapat disertai gambar sampul. Admin dapat memilih gambar dari penyimpanan lokal, lalu sistem akan menampilkan gambar tersebut dalam bentuk pratinjau (preview) pada tampilan daftar buku. Gambar juga akan tampil pada kartu buku (BookCard) yang dilihat pengguna.

5. Pencarian Buku

Pengguna dapat mencari buku berdasarkan judul atau penulis. Sistem akan menampilkan hasil pencarian secara dinamis dengan memfilter daftar buku yang

sesuai. Hal ini sangat membantu pengguna untuk menemukan koleksi tertentu tanpa harus mencari daftar secara manual.

6. Peminjaman eBook

Pengguna dapat meminjam buku yang tersedia. Proses peminjaman mencakup:

- Validasi ketersediaan buku (status isAvailable)
- Pencatatan tanggal pinjam dan tanggal kembali (otomatis 3 hari dari tanggal pinjam)
- Menonaktifkan tombol “Pinjam” jika buku sudah dipinjam pengguna lain
- Menandai buku sebagai tidak tersedia (isAvailable = false)

7. Sistem Kadaluarsa Peminjaman Otomatis

Sistem secara otomatis menghitung masa aktif peminjaman. Jika sudah melewati tanggal kembali, maka status peminjaman akan berubah menjadi “expired”, dan pengguna tidak dapat lagi mengakses eBook tersebut. Validasi ini dilakukan di tampilan riwayat pinjam dan saat ingin membaca eBook.

8. Riwayat Peminjaman

Pengguna dapat melihat riwayat peminjaman mereka sendiri, sementara admin dapat melihat riwayat seluruh pengguna. Informasi yang ditampilkan meliputi judul buku, tanggal pinjam, tanggal kembali, dan status pinjaman (aktif/kadaluarsa). Riwayat ini membantu pengguna melacak aktivitas peminjamannya.

9. Antarmuka Pengguna (GUI) Interaktif dan Modern

Seluruh sistem dirancang menggunakan Java Swing dengan layout GridLayout, BorderLayout, dan komponen GUI lain yang disusun dengan rapi. Warna latar belakang, font, dan elemen visual disesuaikan agar ramah pengguna. Tampilan dibedakan berdasarkan peran: dashboard admin dan user memiliki menu dan akses fitur yang berbeda.

10. Validasi dan Penanganan Kesalahan

Setiap form dalam sistem dilengkapi dengan validasi, seperti input wajib diisi, pengecekan kesesuaian login, serta pengamanan terhadap kesalahan input data.

Selain itu, sistem juga menerapkan try-catch (exception handling) untuk menghindari crash akibat error koneksi database atau input invalid.

11. Koneksi Database MySQL

Sistem terhubung dengan database MySQL melalui koneksi JDBC. Seluruh data (user, buku, peminjaman) tersimpan dalam database dan dikelola melalui query SQL. Kelas utilitas DBConnection digunakan untuk mengatur koneksi secara modular dan terpisah dari logika utama aplikasi.

2.4 Implementasi Kode Program dan Pengujian

- Struktur folder

The screenshot shows a file explorer window with the following structure:

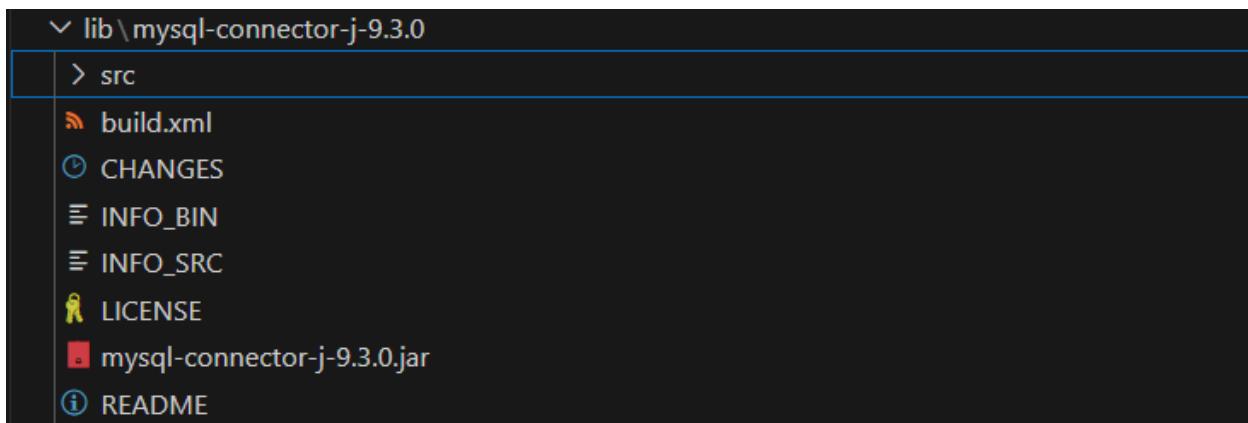
- > images
- > lib\mysql-connector-j-9.3.0
- ✓ src
 - > controller
 - > model
 - > utils
 - > view

Folder utama proyek diberi nama "**UAS PBO**" dan di dalamnya terdapat beberapa subfolder penting. Folder images digunakan untuk menyimpan asset-asset gambar yang mungkin digunakan dalam tampilan antarmuka aplikasi. Folder lib berisi pustaka eksternal, salah satunya adalah mysql-connector-j-9.3.0 yang merupakan library JDBC untuk menghubungkan aplikasi Java dengan database MySQL.

Selanjutnya, folder src berfungsi sebagai tempat menyimpan seluruh kode sumber program, yang dibagi lagi ke dalam beberapa package. Package controller berisi logika kendali aplikasi, menangani alur data dari dan ke tampilan serta model. Package model digunakan untuk merepresentasikan data dalam bentuk class-class objek seperti entitas dalam database, lengkap

dengan atribut dan method getter/setter. Package utils menyimpan class-class pembantu seperti koneksi ke database dan fungsi utilitas lainnya. Sementara itu, package view menangani bagian antarmuka pengguna, baik itu tampilan berbasis GUI maupun teks.

- Struktur file



Didalam projek ini ada beberapa struktur file yang antara lain:

- mysql-connector-j-9.3.0.jar: Merupakan file utama dalam bentuk arsip Java (JAR) yang digunakan untuk mengimpor driver MySQL ke dalam proyek Java. File ini wajib ditambahkan ke classpath agar aplikasi dapat terhubung ke database MySQL.
- LICENSE: Berisi informasi lisensi penggunaan dari library tersebut, biasanya berupa lisensi open-source seperti GPL atau lainnya.
- README: File teks yang berisi petunjuk umum atau dokumentasi singkat mengenai library, seperti cara penggunaan, informasi pengembang, atau catatan penting lainnya.
- CHANGES: Merupakan log perubahan (change log) yang mencatat pembaruan, perbaikan bug, atau penambahan fitur di setiap versi library.
- INFO_BIN dan INFO_SRC: Berisi informasi tentang file biner dan source code library ini. Ini biasanya menjelaskan isi dari file distribusi untuk keperluan dokumentasi atau pengembang.
- build.xml: File konfigurasi build berbasis Apache Ant, digunakan jika ingin membangun (compile) ulang library dari source code-nya.

- src: Merupakan folder source code dari library itu sendiri, tempat di mana kode Java asli dari MySQL Connector/J disimpan. Ini berguna bagi pengembang yang ingin mempelajari atau memodifikasi library secara langsung.

```

src
└── controller
    ├── BookController.java
    ├── BorrowingController.java
    └── UserController.java
└── model
    ├── Admin.java
    ├── Book.java
    ├── Borrowing.java
    └── User.java
└── utils
    ├── DateUtils.java
    └── DBConnection.java
└── view
    ├── AdminDashboard.java
    ├── BookCard.java
    ├── BookListPanel.java
    ├── BorrowingListPanel.java
    ├── EditBookHandler.java
    ├── LoginView.java
    ├── RegisterView.java
    ├── RiwayatPeminjamanPanel.java
    ├── TambahBukuForm.java
    └── UserDashboard.java

```

1. Model

Berisi representasi data atau entitas dalam sistem, masing-masing disimpan dalam class tersendiri:

- Admin.java, User.java ; Mewakili data admin dan pengguna aplikasi.
- Book.java ; Mewakili informasi tentang buku.
- Borrowing.java ; Mewakili data transaksi peminjaman buku

2. Utils

Folder ini berisi class pembantu (utility) yang mendukung fungsionalitas umum dalam aplikasi:

- DateUtils.java ; Kemungkinan digunakan untuk mengatur format dan manipulasi tanggal.
- DBConnection.java ; Menyediakan koneksi ke database, berperan penting dalam komunikasi antara aplikasi dan MySQL.

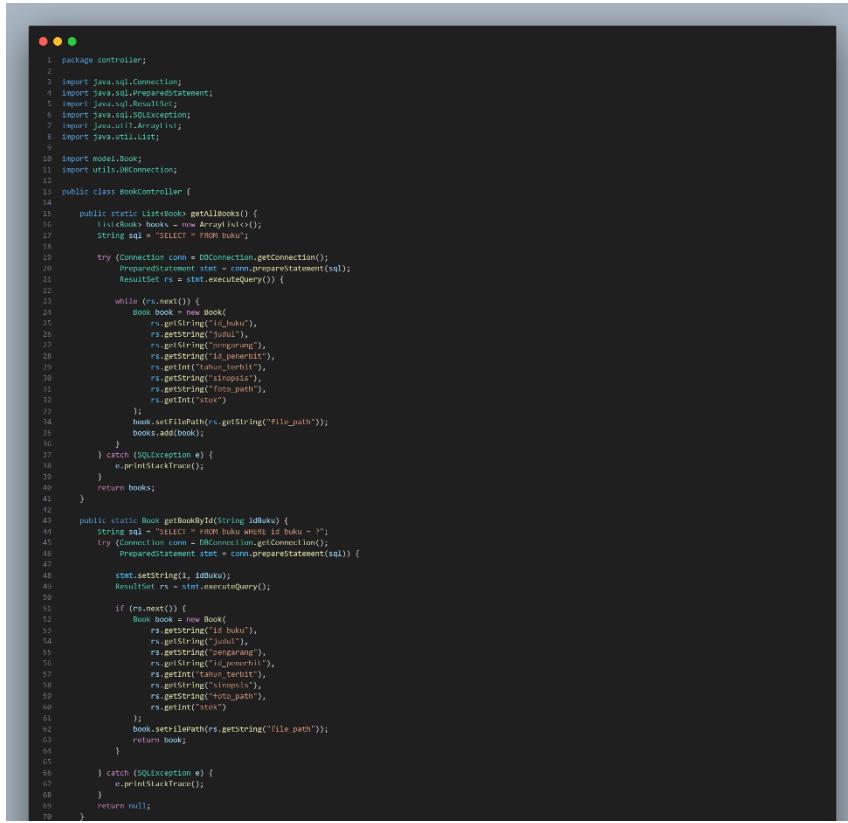
3. View

Berisi class-class yang menampilkan antarmuka pengguna (GUI), kemungkinan menggunakan Swing atau JavaFX. Di antaranya:

- LoginView.java, RegisterView.java ; Tampilan untuk login dan pendaftaran pengguna.
- UserDashboard.java, AdminDashboard.java ; Panel utama setelah login, tergantung peran pengguna.
- BookCard.java, BookListPanel.java, TambahBukuForm.java ; Menampilkan dan mengatur daftar buku serta form penambahan buku.
- BorrowingListPanel.java, RiwayatPeminjamanPanel.java ; Menampilkan data peminjaman dan riwayatnya.
- EditBookHandler.java ; menangani event saat pengguna ingin mengedit buku.

FOLDER CONTROLLER

Book Controller (1)



```
1 package controller;
2
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.util.ArrayList;
8 import java.util.List;
9
10 import model.Book;
11 import utils.DBConnection;
12
13 public class BookController {
14
15     public static List<Book> getAllBooks() {
16         List<Book> books = new ArrayList<>();
17         String sql = "SELECT * FROM book";
18
19         try (Connection conn = DBConnection.getConnection();
20              PreparedStatement stat = conn.prepareStatement(sql);
21              ResultSet rs = stat.executeQuery()) {
22
23             while (rs.next()) {
24                 Book book = new Book();
25                 rs.getString("id_buku"),
26                 rs.getString("judul"),
27                 rs.getString("pengarang"),
28                 rs.getInt("tahun_terbit"),
29                 rs.getString("sinopsis"),
30                 rs.getString("foto_path"),
31                 rs.getInt("stock");
32             }
33             book.setFilePath(rs.getString("file_path"));
34             books.add(book);
35         } catch (SQLException e) {
36             e.printStackTrace();
37         }
38         return books;
39     }
40
41     public static Book getBookByIdCasing (String idbuku) {
42         String sql = "SELECT * FROM book WHERE id_buku = ? ";
43         try (Connection conn = DBConnection.getConnection();
44              PreparedStatement stat = conn.prepareStatement(sql)) {
45
46             stat.setString(1, idbuku);
47             ResultSet rs = stat.executeQuery();
48
49             if (rs.next()) {
50                 Book book = new Book();
51                 rs.getString("id_buku"),
52                 rs.getString("judul"),
53                 rs.getString("pengarang"),
54                 rs.getInt("tahun_terbit"),
55                 rs.getString("sinopsis"),
56                 rs.getString("foto_path"),
57                 rs.getInt("stock");
58             }
59             book.setFilePath(rs.getString("file_path"));
60             return book;
61         } catch (SQLException e) {
62             e.printStackTrace();
63         }
64         return null;
65     }
66 }
```

Kelas ini berisi dua metode utama:

1. public static List<Book> getAllBooks()

Metode ini digunakan untuk mengambil semua data buku dari tabel book di database dan mengembalikannya dalam bentuk list (List<Book>).

Langkah-langkah yang dilakukan:

- Membuat list kosong untuk menampung objek Book.
- Menyusun perintah SQL: SELECT * FROM book.
- Menggunakan koneksi dari DBConnection.getConnection() untuk menjalankan query tersebut.
- Menelusuri hasil (ResultSet) dan untuk setiap baris, membuat objek Book baru menggunakan data dari kolom seperti id_buku, judul, pengarang, dll.

- Menambahkan setiap objek Book ke dalam list.
- Jika terjadi exception (misalnya kesalahan koneksi atau query), error akan dicetak dengan e.printStackTrace().

2. public static Book getBookById(String idBuku)

Metode ini digunakan untuk mengambil data satu buku berdasarkan ID tertentu.

Langkah-langkahnya:

- Menyusun SQL: SELECT * FROM book WHERE id_buku = ? untuk memilih buku berdasarkan ID.
- Menggunakan prepared statement untuk mencegah SQL injection.
- Menyisipkan parameter ID ke dalam query.
- Menjalankan query dan jika hasil ditemukan (rs.next()), maka dibuatlah objek Book dengan data yang sesuai.
- Jika tidak ada data yang ditemukan, fungsi akan mengembalikan null.

Class BookController ini berfungsi sebagai penghubung antara tampilan (view) dan data (model).

Book Controller (2)

```
72     public static boolean insertBook(Book book) {
73         String sql = "INSERT INTO buku (id_buku, judul, pengarang, tahun_terbit, id_penerbit, sinopsis, foto_path, file_path, stok) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)";
74
75         try (Connection conn = DBConnection.getConnection();
76              PreparedStatement stat = conn.prepareStatement(sql)) {
77
78             stat.setString(1, book.getIdBuku());
79             stat.setString(2, book.getTitle());
80             stat.setString(3, book.getAuthor());
81             stat.setInt(4, book.getYear());
82             stat.setString(5, book.getPublisher());
83             stat.setString(6, book.getSynopsis());
84             stat.setString(7, book.getFotoPath());
85             stat.setString(8, book.getFilePath());
86             stat.setInt(9, book.getStok());
87
88             return stat.executeUpdate() > 0;
89         } catch (SQLException e) {
90             e.printStackTrace();
91             return false;
92         }
93     }
94
95     public static boolean updateBook(Book book) {
96         String sql = "UPDATE buku SET judul = ?, pengarang = ?, tahun_terbit = ?, id_penerbit = ?, sinopsis = ?, foto_path = ?, file_path = ? WHERE id_buku = ?";
97
98         try (Connection conn = DBConnection.getConnection();
99              PreparedStatement stat = conn.prepareStatement(sql)) {
100
101            stat.setString(1, book.getTitle());
102            stat.setString(2, book.getAuthor());
103            stat.setInt(3, book.getYear());
104            stat.setString(4, book.getPublisher());
105            stat.setString(5, book.getSynopsis());
106            stat.setString(6, book.getFotoPath());
107            stat.setString(7, book.getFilePath());
108            stat.setInt(8, book.getIdBuku());
109            stat.setString(9, book.getStok());
110
111            return stat.executeUpdate() > 0;
112        } catch (SQLException e) {
113            e.printStackTrace();
114            return false;
115        }
116    }
117
118    public static boolean deleteBook(String idBuku) {
119        String sql = "DELETE FROM buku WHERE id_buku = ?";
120
121        try (Connection conn = DBConnection.getConnection();
122              PreparedStatement stat = conn.prepareStatement(sql)) {
123
124            stat.setString(1, idBuku);
125            return stat.executeUpdate() > 0;
126        } catch (SQLException e) {
127            e.printStackTrace();
128            return false;
129        }
130    }
131
132    public static String getLastIdBuku() {
133        String sql = "SELECT id_buku FROM buku ORDER BY id_buku DESC LIMIT 1";
134
135        try (Connection conn = DBConnection.getConnection();
136              PreparedStatement stat = conn.prepareStatement(sql);
137              ResultSet rs = stat.executeQuery()) {
138
139
140            if (rs.next()) {
141                return rs.getString("id_buku");
142            }
143        } catch (SQLException e) {
144            e.printStackTrace();
145        }
146        return null;
147    }
148
149 }
```

1. public static boolean insertBook(Book book)

Metode ini berfungsi untuk menambahkan (insert) data buku baru ke dalam database.

- SQL yang digunakan: INSERT INTO book VALUES
- Menggunakan PreparedStatement untuk menyisipkan nilai dari objek Book ke dalam query.
- Field yang dimasukkan meliputi: id_buku, judul, pengarang, tahun_terbit, id_penerbit, sinopsis, foto_path, file_path, stok.
- Jika berhasil, akan mengembalikan true; jika gagal atau terjadi exception, akan mengembalikan false.

2. public static boolean updateBook(Book book)

Metode ini berfungsi untuk memperbarui data buku yang sudah ada berdasarkan id_buku.

- SQL yang digunakan: UPDATE book SET ... WHERE id_buku = ?
- Semua field buku diperbarui dengan nilai baru dari objek Book, kecuali id_buku sebagai kunci identifikasi.
- Mengembalikan true jika proses update berhasil; false jika terjadi kesalahan.

3. public static boolean deleteBook(String idBuku)

Metode ini digunakan untuk menghapus data buku dari database berdasarkan id_buku.

- SQL: DELETE FROM book WHERE id_buku
- Menggunakan PreparedStatement untuk mencegah SQL Injection.
- Mengembalikan true jika proses penghapusan berhasil; false jika gagal.

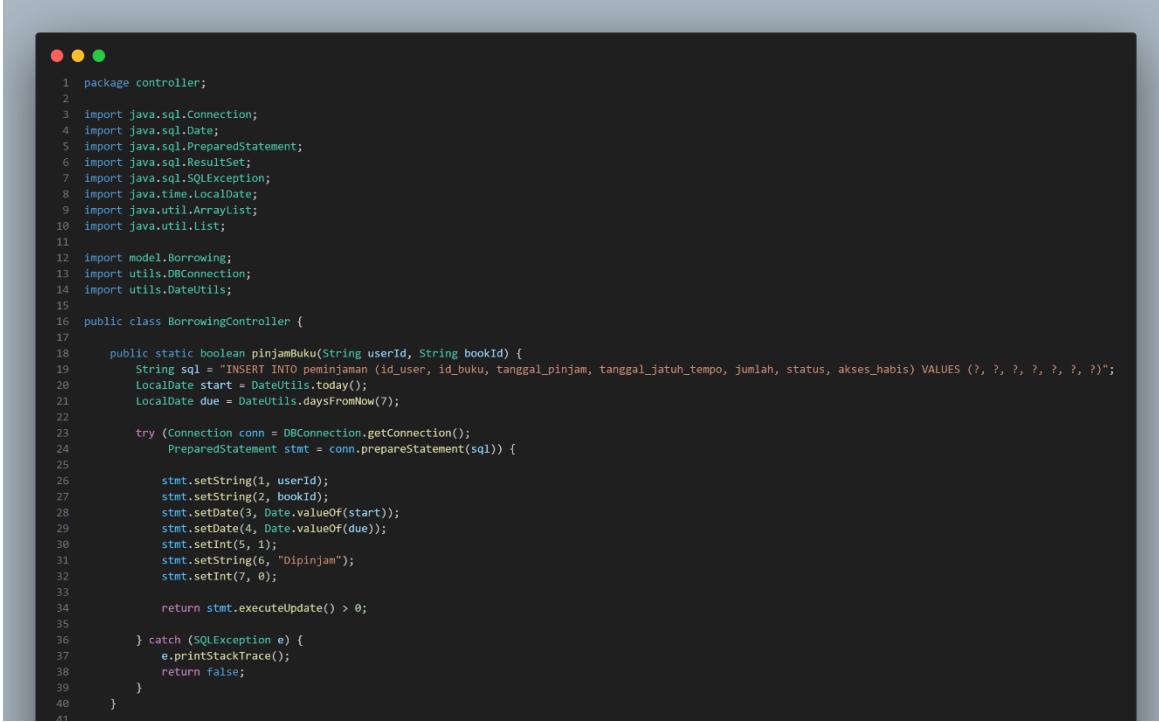
4. public static String getLastIdBuku()

Metode ini digunakan untuk mengambil ID buku terakhir yang dimasukkan ke dalam database.

- SQL: SELECT id_buku FROM book ORDER BY id_buku DESC LIMIT 1
- Ini berguna untuk membuat ID buku baru secara otomatis berdasarkan urutan terakhir.
- Jika ada data, maka akan dikembalikan id_buku terakhir; jika tidak, null.

Kode ini menunjukkan implementasi CRUD (Create, Read, Update, Delete) untuk objek Book di dalam database.

Borrowing Controller (1)



```
1 package controller;
2
3 import java.sql.Connection;
4 import java.sql.Date;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.time.LocalDate;
9 import java.util.ArrayList;
10 import java.util.List;
11
12 import model.Borrowing;
13 import utils.DBConnection;
14 import utils.DateUtils;
15
16 public class BorrowingController {
17
18     public static boolean pinjamBuku(String userId, String bookId) {
19         String sql = "INSERT INTO peminjaman (id_user, id_buku, tanggal_pinjam, tanggal_jatuh_tempo, jumlah, status, akses_habis) VALUES (?, ?, ?, ?, ?, ?, ?)";
20         LocalDate start = DateUtils.today();
21         LocalDate due = DateUtils.daysFromNow(7);
22
23         try (Connection conn = DBConnection.getConnection();
24              PreparedStatement stmt = conn.prepareStatement(sql)) {
25
26             stmt.setString(1, userId);
27             stmt.setString(2, bookId);
28             stmt.setDate(3, Date.valueOf(start));
29             stmt.setDate(4, Date.valueOf(due));
30             stmt.setInt(5, 1);
31             stmt.setString(6, "Dipinjam");
32             stmt.setInt(7, 0);
33
34             return stmt.executeUpdate() > 0;
35
36         } catch (SQLException e) {
37             e.printStackTrace();
38             return false;
39         }
40     }
41 }
```

Fungsi: public static boolean pinjamBuku(String userId, String bookId)

Fungsi ini digunakan untuk menambahkan data peminjaman buku ke dalam tabel peminjaman pada database.

Proses yang Dilakukan:

1. Menyusun SQL Insert Statement:
2. INSERT INTO peminjaman (id_user, id_buku, tanggal_pinjam, tanggal_jatuh_tempo, jumlah, status, akses_habis)
3. VALUES
4. Mengatur Nilai Tanggal:
 - start: diambil dari tanggal hari ini menggunakan DateUtils.today().
 - due: dihitung 7 hari dari sekarang dengan DateUtils.daysFromNow(7) (tenggat pengembalian).

5. Membuka Koneksi ke Database:

Menggunakan DBConnection.getConnection() untuk mendapatkan objek Connection.

6. Menyiapkan dan Mengisi Statement:

Nilai-nilai yang di-set dalam PreparedStatement:

- id_user: ID peminjam
- id_buku: ID buku
- tanggal_pinjam: tanggal mulai meminjam
- tanggal_jatuh_tempo: tanggal jatuh tempo
- jumlah: 1 (diasumsikan selalu meminjam 1 buku)
- status: "Dipinjam"
- akses_habis: 0 (akses belum habis)

7. Eksekusi Query:

- stmt.executeUpdate() > 0: mengembalikan true jika berhasil menambahkan baris baru.
- Jika terjadi SQLException, error dicetak dan fungsi mengembalikan false.

Borrowing Controller (2)

```
42     public static List<Borrowing> getRiwayat(String userId) {
43         List<Borrowing> list = new ArrayList<>();
44         String sql;
45
46         if (userId == null || userId.isEmpty()) {
47             sql = "SELECT * FROM peminjaman";
48         } else {
49             sql = "SELECT * FROM peminjaman WHERE id_user = ?";
50         }
51
52         try (Connection conn = DBConnection.getConnection();
53              PreparedStatement stmt = conn.prepareStatement(sql)) {
54
55             if (userId != null && !userId.isEmpty()) {
56                 stmt.setString(1, userId);
57             }
58
59             ResultSet rs = stmt.executeQuery();
60
61             while (rs.next()) {
62                 Borrowing b = new Borrowing(
63                     rs.getInt("id_peminjaman"),
64                     rs.getString("id_user"),
65                     rs.getString("id_buku"),
66                     rs.getDate("tanggal_pinjam").toLocalDate(),
67                     rs.getDate("tanggal_jatuh_tempo").toLocalDate(),
68                     rs.getInt("jumlah"),
69                     rs.getString("status"),
70                     rs.getBoolean("akses_habis")
71                 );
72                 list.add(b);
73             }
74
75         } catch (SQLException e) {
76             e.printStackTrace();
77         }
78
79         return list;
80     }
81
82     public static List<Borrowing> getAllBorrowings() {
83         return getRiwayat("");
84     }
85
86     public static boolean deleteBorrowing(int idPeminjaman) {
87         String sql = "DELETE FROM peminjaman WHERE id_peminjaman = ?";
88
89         try (Connection conn = DBConnection.getConnection();
90              PreparedStatement stmt = conn.prepareStatement(sql)) {
91
92             stmt.setInt(1, idPeminjaman);
93             return stmt.executeUpdate() > 0;
94
95         } catch (SQLException e) {
96             e.printStackTrace();
97             return false;
98         }
99     }
100 }
```

1. public static List<Borrowing> getRiwayat(String userId)

Metode ini digunakan untuk mengambil daftar riwayat peminjaman berdasarkan userId.

- Jika userId kosong atau null, maka mengambil semua data (SELECT * FROM peminjaman).
- Jika userId valid, maka mengambil data berdasarkan id_user = ?.
- Hasil query dieksekusi dan setiap baris data disimpan ke dalam objek Borrowing, kemudian ditambahkan ke List<Borrowing>.
- Mengembalikan list berisi seluruh peminjaman milik user tertentu atau seluruh user.

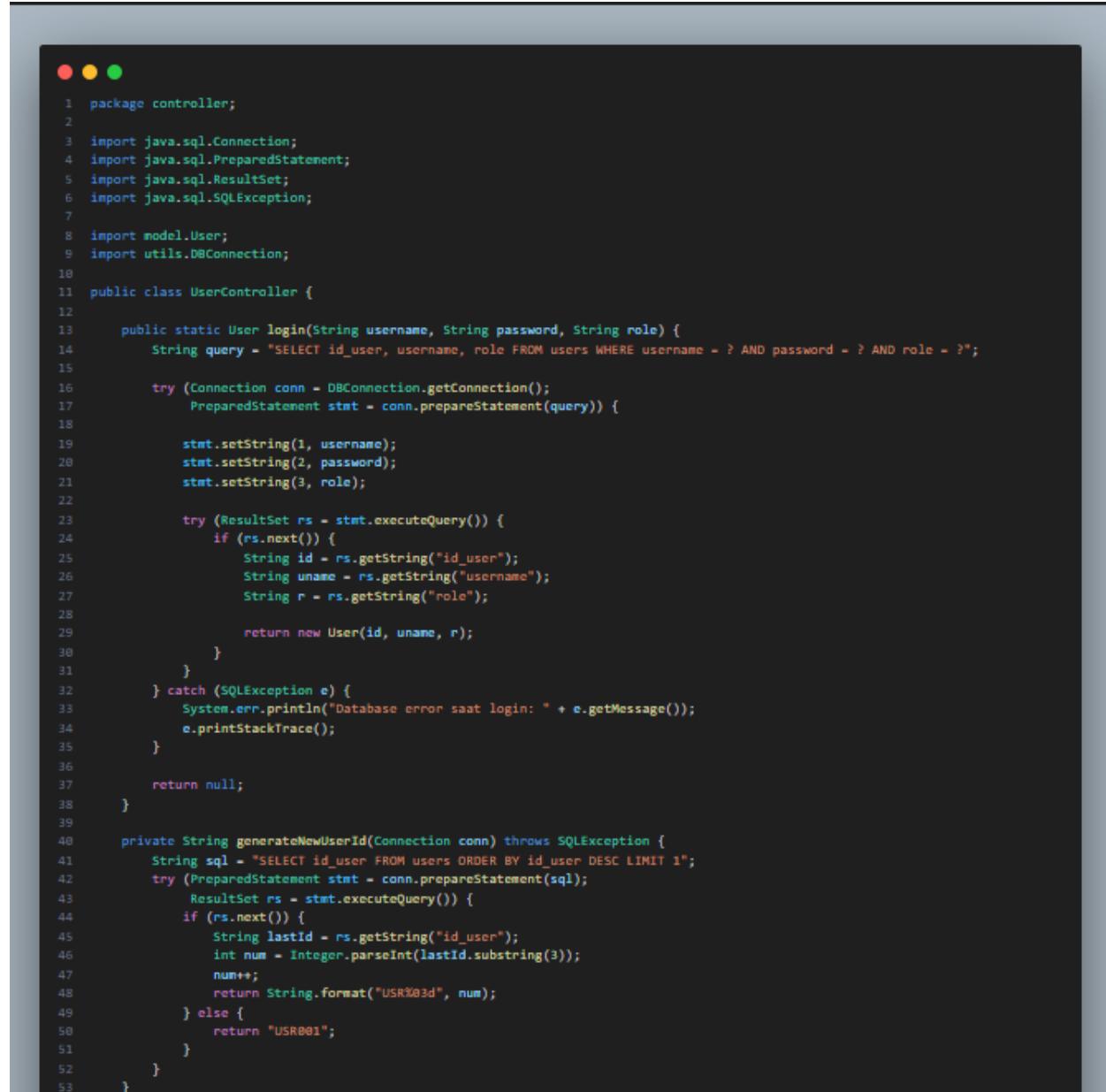
2. public static List<Borrowing> getAllBorrowings()

Metode ini digunakan untuk mengambil seluruh data peminjaman buku.

- Memanggil fungsi getRiwayat(""), yang berarti mengambil semua peminjaman (tanpa filter user).
- Mengembalikan List<Borrowing> yang berisi seluruh riwayat peminjaman di sistem.

2. public static boolean deleteBorrowing(int idPeminjaman) :Metode ini digunakan untuk menghapus data peminjaman berdasarkan id_peminjaman.

Use Controller (1)



```
1 package controller;
2
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7
8 import model.User;
9 import utils.DBConnection;
10
11 public class UserController {
12
13     public static User login(String username, String password, String role) {
14         String query = "SELECT id_user, username, role FROM users WHERE username = ? AND password = ? AND role = ?";
15
16         try (Connection conn = DBConnection.getConnection();
17              PreparedStatement stmt = conn.prepareStatement(query)) {
18
19             stmt.setString(1, username);
20             stmt.setString(2, password);
21             stmt.setString(3, role);
22
23             try (ResultSet rs = stmt.executeQuery()) {
24                 if (rs.next()) {
25                     String id = rs.getString("id_user");
26                     String uname = rs.getString("username");
27                     String r = rs.getString("role");
28
29                     return new User(id, uname, r);
30                 }
31             }
32         } catch (SQLException e) {
33             System.err.println("Database error saat login: " + e.getMessage());
34             e.printStackTrace();
35         }
36
37         return null;
38     }
39
40     private String generateNewUserId(Connection conn) throws SQLException {
41         String sql = "SELECT id_user FROM users ORDER BY id_user DESC LIMIT 1";
42         try (PreparedStatement stmt = conn.prepareStatement(sql);
43              ResultSet rs = stmt.executeQuery()) {
44             if (rs.next()) {
45                 String lastId = rs.getString("id_user");
46                 int num = Integer.parseInt(lastId.substring(3));
47                 num++;
48                 return String.format("USR%03d", num);
49             } else {
50                 return "USR001";
51             }
52         }
53     }
54 }
```

1. public static User login(String username, String password, String role)

Fungsi ini digunakan untuk **melakukan autentikasi user** berdasarkan tiga parameter: username, password, dan role.

- Query yang digunakan:

- `SELECT id_user, username, role FROM users WHERE username = ? AND password = ? AND role = ?`
 - Menggunakan PreparedStatement untuk menghindari SQL Injection.
 - Jika data ditemukan (`rs.next()`), maka akan dikembalikan objek User baru berisi:
 - `id_user`
 - `username`
 - `role`
 - Jika tidak ada data yang cocok, fungsi akan mengembalikan null.
 - Jika ada error SQL, error akan dicetak dan tetap mengembalikan null.
2. `private String generateNewUserId(Connection conn) throws SQLException`

Fungsi ini digunakan untuk menghasilkan ID user baru secara otomatis, dengan format `USR##`.

- Query yang digunakan:
 - `SELECT id_user FROM users ORDER BY id_user DESC LIMIT 1`
- Tujuannya adalah mengambil ID user terakhir.
- Kemudian:
 - Mengambil bagian angka dari ID (`substring(3)`).
 - Mengubahnya menjadi integer (`parseInt`), lalu menambahkan 1.
 - Menggabungkannya kembali ke format ID seperti `USR001`, `USR002`, dst.
 - Jika tabel user masih kosong, akan menghasilkan ID awal `USR001`.

User Controller (2)

```
54     public boolean register(User user) {
55         String sql = "INSERT INTO users (id_user, nama, username, password, alamat, jenis_kelamin, tgl_lahir, no_hp, role) " +
56             "VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)";
57
58         try (Connection conn = DBConnection.getConnection()) {
59             String newId = generateNewUserId(conn);
60             user.setIdUser(newId);
61
62             try (PreparedStatement stmt = conn.prepareStatement(sql)) {
63                 stmt.setString(1, user.getIdUser());
64                 stmt.setString(2, user.getNama());
65                 stmt.setString(3, user.getUsername());
66                 stmt.setString(4, user.getPassword());
67                 stmt.setString(5, user.getAlamat());
68                 stmt.setString(6, user.getJenisKelamin());
69                 stmt.setDate(7, user.getTglLahir());
70                 stmt.setString(8, user.getNoHp());
71                 stmt.setString(9, user.getRole());
72
73                 int result = stmt.executeUpdate();
74                 return result > 0;
75             }
76         } catch (SQLException e) {
77             System.err.println("Database error saat register: " + e.getMessage());
78             e.printStackTrace();
79             return false;
80         }
81     }
82 }
```

1. Deklarasi SQL dan Method

Method ini menerima objek User sebagai parameter. Query SQL INSERT digunakan untuk menyimpan data user ke tabel users.

2. Koneksi dan ID Otomatis

- Membuka koneksi ke database menggunakan DBConnection.
- Menghasilkan ID unik baru untuk user dengan method generateNewUserId(conn) dan menyetelnya ke objek user.

3. Pengisian dan Eksekusi Statement

- Menggunakan PreparedStatement agar lebih aman dari SQL Injection.
- Mengisi semua parameter ? dengan data dari objek user seperti nama, username, password, dll.

- `stmt.executeUpdate()` menjalankan perintah INSERT dan mengembalikan jumlah baris yang terpengaruh. Jika lebih dari 0, artinya insert berhasil.

4. Penanganan Error

- Menangkap dan mencetak kesalahan jika terjadi SQLException selama proses insert.
- Jika ada error, method mengembalikan false.

FOLDER MODEL

Book



```
1 package model;
2
3 public class Book {
4     private String idBuku;
5     private String title;
6     private String author;
7     private String publisher;
8     private int tahunTerbit;
9     private String synopsis;
10    private String imagePath;
11    private int stok;
12    private int dipinjam;
13    private String filePath;
14
15    public Book() {
16    }
17
18    public Book(String idBuku, String title, String author, String publisher, int tahunTerbit,
19                String synopsis, String imagePath, int stok) {
20        this.idBuku = idBuku;
21        this.title = title;
22        this.author = author;
23        this.publisher = publisher;
24        this.tahunTerbit = tahunTerbit;
25        this.synopsis = synopsis;
26        this.imagePath = imagePath;
27        this.stok = stok;
28    }
29
30    public String getIdBuku() {
31        return idBuku;
32    }
33
34    public void setIdBuku(String idBuku) {
35        this.idBuku = idBuku;
36    }
37
38    public String getTitle() {
39        return title;
40    }
41
42    public void setTitle(String title) {
43        this.title = title;
44    }
45
46    public String getAuthor() {
47        return author;
48    }
49
50    public void setAuthor(String author) {
51        this.author = author;
52    }
53
54    public String getPublisher() {
55        return publisher;
56    }
57
58    public void setPublisher(String publisher) {
59        this.publisher = publisher;
60    }
61
62    public int getTahunTerbit() {
63        return tahunTerbit;
64    }
65
66    public void setTahunTerbit(int tahunTerbit) {
67        this.tahunTerbit = tahunTerbit;
68    }
69
70    public String getSynopsis() {
71        return synopsis;
72    }
73
74    public void setSynopsis(String synopsis) {
75        this.synopsis = synopsis;
76    }
77
78    public String getImagePath() {
79        return imagePath;
80    }
81
82    public void setImagePath(String imagePath) {
83        this.imagePath = imagePath;
84    }
85
86    public int getStok() {
87        return stok;
88    }
89
90    public void setStok(int stok) {
91        this.stok = stok;
92    }
93
94    public int getDipinjam() {
95        return dipinjam;
96    }
97
98    public void setDipinjam(int dipinjam) {
99        this.dipinjam = dipinjam;
100    }
101
102    public String getFilePath() {
103        return filePath;
104    }
105
106    public void setFilePath(String filePath) {
107        this.filePath = filePath;
108    }
109}
110
```

Penjelasan Class Book:

Class ini berfungsi sebagai representasi data buku dalam sistem perpustakaan digital, dengan atribut dan method untuk menyimpan dan mengelola informasi terkait buku.

Penjelasan:

- idBuku: ID unik untuk buku.
- title: Judul buku.
- author: Penulis buku.
- publisher: Penerbit buku.
- tahunTerbit: Tahun terbit buku.
- synopsis: Sinopsis buku.
- imagePath: Lokasi gambar buku.
- stok: Jumlah total stok buku.
- dipinjam: Jumlah buku yang sedang dipinjam.
- filePath: Lokasi file digital buku (jika eBook).

2. Constructor

```
public Book() {}
```

Constructor default (tanpa parameter).

```
public Book(String idBuku, String title, String author, String publisher, int tahunTerbit, String synopsis, String imagePath, int stok)
```

Constructor dengan parameter untuk menginisialisasi objek Book dengan data awal.

3. Getter dan Setter

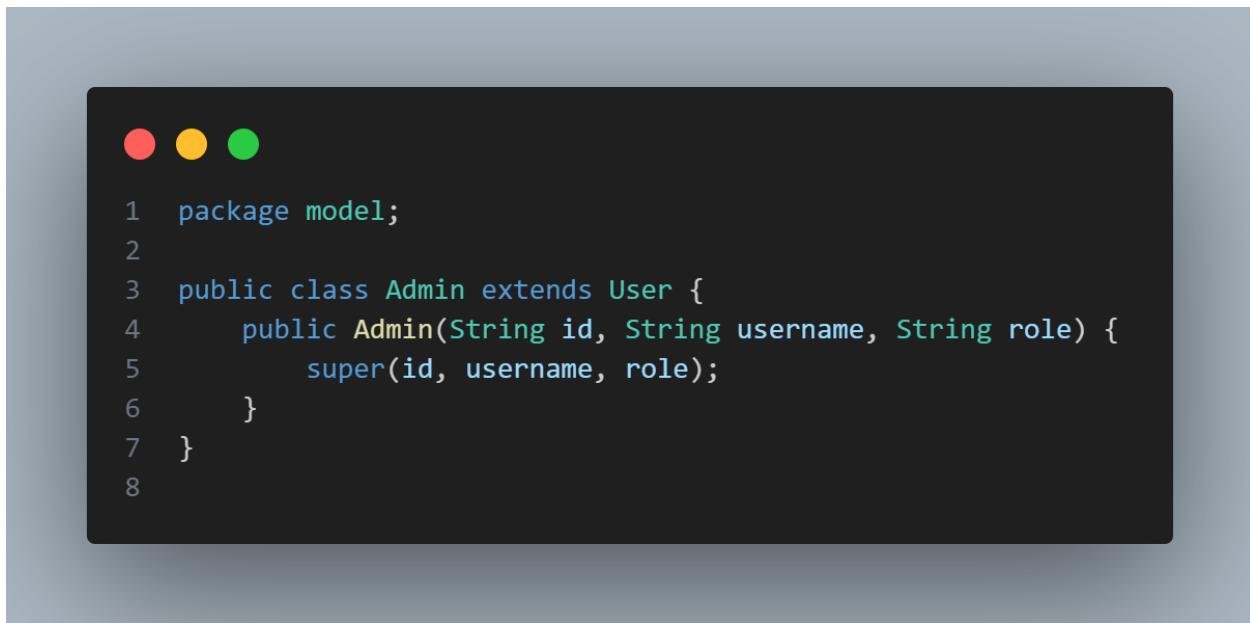
Method getter dan setter disediakan untuk setiap atribut, mengikuti prinsip enkapsulasi OOP, yaitu:

- Getter: Mengambil nilai atribut (getIdBuku(), getTitle(), dll).
- Setter: Mengubah nilai atribut (setTitle(String title), setStok(int stok), dll).

Class Book digunakan untuk:

- Menyimpan data tiap buku dalam sistem.
- Memudahkan pengolahan data melalui objek.
- Memastikan data bersifat privat namun tetap dapat diakses dan dimodifikasi melalui method yang aman.

Admin



The screenshot shows a dark-themed code editor window. At the top, there are three circular icons: red, yellow, and green. Below them, the Java code for the Admin class is displayed:

```
1 package model;
2
3 public class Admin extends User {
4     public Admin(String id, String username, String role) {
5         super(id, username, role);
6     }
7 }
8
```

Penjelasan Class Admin :

```
public class Admin extends User {
```

- Class Admin merupakan subclass (kelas turunan) dari class User.
- Artinya, Admin mewarisi semua atribut dan method dari class User, sehingga Admin adalah bentuk spesifik dari User.

```
public Admin(String id, String username, String role) {
    super(id, username, role);
```

}

- Ini adalah constructor untuk class Admin dengan parameter:
 - id: ID unik admin.
 - username: Nama pengguna admin.
 - role: Peran pengguna (dalam hal ini kemungkinan "admin").
- super(...) dipanggil untuk menjalankan constructor milik superclass (User), agar nilai id, username, dan role diinisialisasi langsung melalui constructor User.

Class Admin digunakan untuk merepresentasikan pengguna dengan role khusus sebagai admin dalam sistem. Karena mewarisi dari class User, class ini tidak perlu mendeklarasikan ulang atribut id, username, dan role, melainkan hanya menyesuaikan fungsi atau perilaku yang dibutuhkan admin (jika ditambahkan nanti).

User



```
1 package model;
2
3 import java.sql.Date;
4
5 public class User {
6     private String idUser;
7     private String username;
8     private String password;
9     private String role;
10
11    private String nama;
12    private String alamat;
13    private String jenisKelamin;
14    private Date tglLahir;
15    private String noHp;
16
17    public User(String idUser, String username, String role) {
18        this.idUser = idUser;
19        this.username = username;
20        this.role = role;
21    }
22
23    public User() {
24    }
25
26    public String getIdUser() {
27        return idUser;
28    }
29
30    public void setIdUser(String idUser) {
31        this.idUser = idUser;
32    }
33
34    public String getUsername() {
35        return username;
36    }
37
38    public void setUsername(String username) {
39        this.username = username;
40    }
41
42    public String getPassword() {
43        return password;
44    }
45
46    public void setPassword(String password) {
47        this.password = password;
48    }
49
50    public String getRole() {
51        return role;
52    }
53
54    public void setRole(String role) {
55        this.role = role;
56    }
57
58    public String getNama() {
59        return nama;
60    }
61
62    public void setNama(String nama) {
63        this.nama = nama;
64    }
65
66    public String getAlamat() {
67        return alamat;
68    }
69
70    public void setAlamat(String alamat) {
71        this.alamat = alamat;
72    }
73
74    public String getJenisKelamin() {
75        return jenisKelamin;
76    }
77
78    public void setJenisKelamin(String jenisKelamin) {
79        this.jenisKelamin = jenisKelamin;
80    }
81
82    public Date getTglLahir() {
83        return tglLahir;
84    }
85
86    public void setTglLahir(Date tglLahir) {
87        this.tglLahir = tglLahir;
88    }
89
90    public String getNoHp() {
91        return noHp;
92    }
93
94    public void setNoHp(String noHp) {
95        this.noHp = noHp;
96    }
97 }
```

Penjelasan Class User:

1. Package dan Import (Baris 1–2)

Class ini berada dalam package model. Import java.sql.Date digunakan untuk merepresentasikan atribut tanggal lahir pengguna dengan format tanggal dari database.

2. Deklarasi Class dan Atribut (Baris 4–13)

public class User merupakan class model yang merepresentasikan data pengguna (user) dalam sistem perpustakaan digital.

Atribut-atribut yang dimiliki class ini adalah:

- idUser : ID unik untuk setiap pengguna
- username : Nama pengguna untuk login
- password : Kata sandi pengguna
- role : Peran pengguna (admin, user)
- nama : Nama lengkap pengguna
- alamat : Alamat tempat tinggal pengguna
- jenisKelamin : Jenis kelamin pengguna
- tglLahir : Tanggal lahir pengguna (menggunakan java.sql.Date)
- noHp : Nomor telepon pengguna

Atribut bersifat private agar hanya bisa diakses melalui method getter dan setter, sesuai prinsip encapsulation.

3. Konstruktor (Baris 15–19 dan 21–22)

Class ini memiliki dua konstruktor:

- User(String idUser, String username, String role) : Konstruktor dengan parameter ID, username, dan role. Digunakan ketika hanya data dasar pengguna yang dibutuhkan, seperti saat proses login atau otorisasi.
- User() : Konstruktor default tanpa parameter. Digunakan saat membuat objek kosong sebelum datanya diisi secara bertahap.

4. Method Getter dan Setter (Baris 24–89)

Setiap atribut memiliki method getter dan setter:

- Method get berfungsi untuk mengambil nilai dari atribut.
- Method set berfungsi untuk mengubah atau mengisi nilai atribut.

Contoh:

- getUsername() dan setUsername(String username) digunakan untuk mengakses dan mengatur data username.
- getTglLahir() dan setTglLahir(Date tglLahir) digunakan untuk mengakses dan mengatur tanggal lahir.

Penggunaan getter dan setter ini mendukung prinsip encapsulation dalam OOP, yaitu menyembunyikan data internal objek dan hanya mengaksesnya melalui method publik.

5. Penerapan OOP

Class User menerapkan prinsip Object-Oriented Programming sebagai berikut:

- Encapsulation: Semua atribut bersifat private dan hanya bisa diakses melalui getter dan setter.
- Abstraction: Detail implementasi disembunyikan dari pengguna class, hanya method publik yang diekspos.
- Object: Class ini merupakan representasi nyata dari entitas pengguna di dalam sistem.
- Reusability: Class ini dapat digunakan di berbagai komponen sistem (login, peminjaman, registrasi, dsb) sebagai representasi data pengguna.

Fungsi utama class ini adalah sebagai model data untuk menyimpan dan mengelola informasi pengguna dalam sistem perpustakaan digital berbasis Java GUI. Class ini mendukung pengelolaan akun, peminjaman, dan interaksi pengguna lainnya dengan sistem.

FOLDER UTILS

DateUtils



```
1 package utils;
2
3 import java.time.LocalDate;
4
5 public class DateUtils {
6     public static LocalDate today() {
7         return LocalDate.now();
8     }
9
10    public static LocalDate daysFromNow(int days) {
11        return LocalDate.now().plusDays(days);
12    }
13}
14
```

Penjelasan Class DateUtils:

Berikut adalah penjelasan codingan untuk class DateUtils yang diselaraskan dengan format penjelasan class lain dalam sistem:

1. Package dan Import (Baris 1)

```
package utils;
```

Menandakan bahwa class ini berada dalam package utils, yang umumnya digunakan untuk menyimpan class-class utilitas atau bantuan dalam sistem.

```
import java.time.LocalDate;
```

Melakukan import terhadap class LocalDate dari package java.time, yang digunakan untuk merepresentasikan tanggal (tanpa waktu) dalam format tahun-bulan-hari (YYYY-MM-DD).

2. Deklarasi Class (Baris 3)

```
public class DateUtils
```

Merupakan class utilitas yang menyediakan method-method statis untuk operasi terkait tanggal.

Class ini tidak memiliki atribut karena tidak menyimpan data apa pun. Semua fungsionalitas disediakan dalam bentuk method static yang bisa dipanggil langsung tanpa harus membuat objek dari class DateUtils.

3. Method today() (Baris 4–6)

```
public static LocalDate today()
```

Method ini bertugas mengembalikan tanggal hari ini sesuai waktu lokal sistem. Menggunakan LocalDate.now() untuk mendapatkan tanggal saat ini. Karena method bersifat static, dapat langsung digunakan tanpa perlu membuat objek class.

Contoh penggunaan:

```
LocalDate tanggalHariIni = DateUtils.today();
```

4. Method daysFromNow(int days) (Baris 8–10)

```
public static LocalDate daysFromNow(int days)
```

Method ini mengembalikan tanggal yang berjarak sejumlah hari dari tanggal sekarang. Misalnya, jika days = 3, maka akan mengembalikan tanggal tiga hari ke depan dari hari ini. Menggunakan method LocalDate.now().plusDays(days) dari API java.time.

Contoh penggunaan:

```
LocalDate tigaHariLagi = DateUtils.daysFromNow(3);
```

5. Materi OOP yang Diterapkan

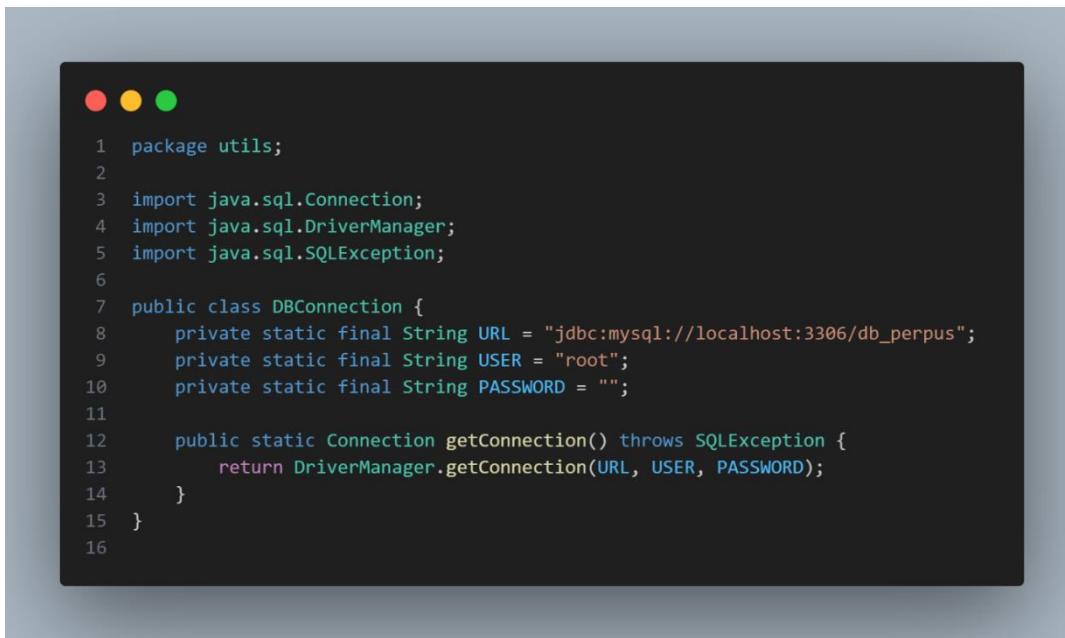
- Static Method: Method tidak terikat pada objek, melainkan class itu sendiri.
- Encapsulation: Menyediakan fungsionalitas terbatas (khusus tanggal) tanpa mengekspos implementasi ke pengguna class.
- Utility Class Design: Class ini berfungsi sebagai alat bantu untuk menyediakan method umum yang dapat digunakan di berbagai bagian sistem.

6. Fitur-fitur Utamanya

- Mendapatkan tanggal saat ini.
- Mendapatkan tanggal sejumlah hari ke depan dari hari ini.
- Memudahkan pengolahan data tanggal secara konsisten di seluruh sistem.

Class ini mendukung pengembangan sistem perpustakaan dengan menyediakan kemudahan dalam perhitungan tanggal, seperti menentukan batas waktu peminjaman e-book atau menghitung tanggal kadaluarsa otomatis.

DBConnection



```
1 package utils;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class DBConnection {
8     private static final String URL = "jdbc:mysql://localhost:3306/db_perpus";
9     private static final String USER = "root";
10    private static final String PASSWORD = "";
11
12    public static Connection getConnection() throws SQLException {
13        return DriverManager.getConnection(URL, USER, PASSWORD);
14    }
15 }
16
```

Penjelasan Class DBConnection :

Berikut penjelasan codingan class DBConnection yang diselaraskan dengan format penjelasan class lain dalam sistem:

1. Package dan Import (Baris 1–3)

```
package utils;
```

Menandakan bahwa class ini berada dalam package utils, yaitu package yang biasanya digunakan untuk menyimpan class-class utilitas atau pembantu.

```
import java.sql.Connection;
```

Mengimpor class Connection dari package java.sql, yang digunakan untuk merepresentasikan koneksi ke database.

```
import java.sql.DriverManager;
```

Mengimpor class DriverManager yang digunakan untuk mengatur koneksi ke database melalui JDBC.

```
import java.sql.SQLException;
```

Mengimpor class SQLException yang digunakan untuk menangani error yang mungkin terjadi selama proses koneksi ke database.

2. Deklarasi Class dan Atribut (Baris 5–9)

```
public class DBConnection
```

Merupakan class utilitas yang menyediakan method untuk mendapatkan koneksi ke database.

Atribut:

- private static final String URL: Menyimpan URL koneksi JDBC ke database MySQL lokal dengan nama database db_perpus.
- private static final String USER: Menyimpan username untuk akses database.
- private static final String PASSWORD: Menyimpan password untuk akses database (dikosongkan karena default MySQL lokal sering tidak memakai password).

Ketiga atribut di atas dideklarasikan static final karena bersifat konstan dan digunakan bersama di seluruh class.

3. Method getConnection() (Baris 11–13)

```
public static Connection getConnection() throws SQLException
```

Merupakan method static yang dapat langsung dipanggil tanpa membuat objek dari class ini. Method ini digunakan untuk membuat dan mengembalikan objek Connection ke database menggunakan DriverManager.getConnection(URL, USER, PASSWORD).

Jika terjadi kesalahan dalam proses koneksi, method ini akan melempar SQLException yang harus ditangani oleh pemanggil method.

Contoh penggunaan:

```
Connection conn = DBConnection.getConnection();
```

4. Materi OOP yang Diterapkan

- Static Method: Method tidak terikat pada objek, melainkan class itu sendiri.
- Exception Handling: Method dideklarasikan untuk melempar SQLException jika terjadi error saat koneksi.
- Encapsulation: Detail URL, username, dan password disimpan dalam atribut privat dan hanya digunakan di dalam class.

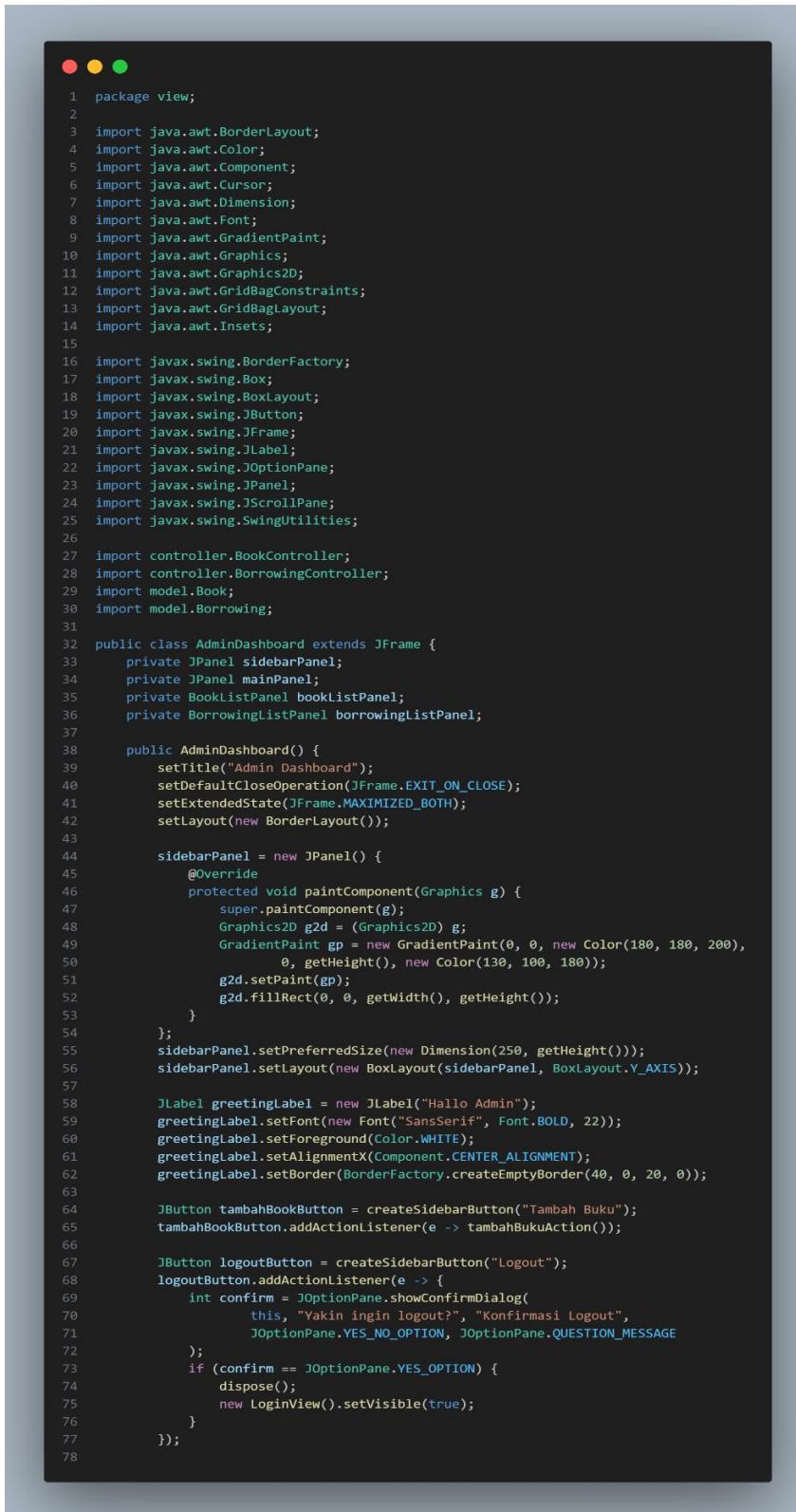
5. Fitur-fitur Utamanya

- Menyediakan koneksi ke database db_perpus secara terpusat.
- Mempermudah akses database dari class lain tanpa perlu menulis ulang konfigurasi koneksi.
- Menyederhanakan manajemen koneksi database dalam sistem.

Class ini berperan penting dalam sistem karena menjadi gerbang utama untuk seluruh operasi database, seperti query peminjaman, pengambilan data buku, dan validasi login. Desain class yang terpisah juga mendukung prinsip pemisahan tanggung jawab (separation of concerns) dalam pengembangan aplikasi.

FOLDER VIEW

Admin Dashboard (1)



```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Component;
6 import java.awt.Cursor;
7 import java.awt.Dimension;
8 import java.awt.Font;
9 import java.awt.GradientPaint;
10 import java.awt.Graphics;
11 import java.awt.Graphics2D;
12 import java.awt.GridBagConstraints;
13 import java.awt.GridBagLayout;
14 import java.awt.Insets;
15
16 import javax.swing.BorderFactory;
17 import javax.swing.Box;
18 import javax.swing.BoxLayout;
19 import javax.swing.JButton;
20 import javax.swing.JFrame;
21 import javax.swing.JLabel;
22 import javax.swing.JOptionPane;
23 import javax.swing.JPanel;
24 import javax.swing.JScrollPane;
25 import javax.swing.SwingUtilities;
26
27 import controller.BookController;
28 import controller.BorrowingController;
29 import model.Book;
30 import model.Borrowing;
31
32 public class AdminDashboard extends JFrame {
33     private JPanel sidebarPanel;
34     private JPanel mainPanel;
35     private BookListPanel booklistPanel;
36     private BorrowingListPanel borrowingListPanel;
37
38     public AdminDashboard() {
39         setTitle("Admin Dashboard");
40         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
41         setExtendedState(JFrame.MAXIMIZED_BOTH);
42         setLayout(new BorderLayout());
43
44         sidebarPanel = new JPanel() {
45             @Override
46             protected void paintComponent(Graphics g) {
47                 super.paintComponent(g);
48                 Graphics2D g2d = (Graphics2D) g;
49                 GradientPaint gp = new GradientPaint(0, 0, new Color(180, 180, 200),
50                     0, getHeight(), new Color(130, 100, 180));
51                 g2d.setPaint(gp);
52                 g2d.fillRect(0, 0, getWidth(), getHeight());
53             }
54         };
55         sidebarPanel.setPreferredSize(new Dimension(250, getHeight()));
56         sidebarPanel.setLayout(new BoxLayout(sidebarPanel, BoxLayout.Y_AXIS));
57
58         JLabel greetingLabel = new JLabel("Hallo Admin");
59         greetingLabel.setFont(new Font("SansSerif", Font.BOLD, 22));
60         greetingLabel.setForeground(Color.WHITE);
61         greetingLabel.setAlignmentX(Component.CENTER_ALIGNMENT);
62         greetingLabel.setBorder(BorderFactory.createEmptyBorder(40, 0, 20, 0));
63
64         JButton tambahBookButton = createSidebarButton("Tambah Buku");
65         tambahBookButton.addActionListener(e -> tambahBukuAction());
66
67         JButton logoutButton = createSidebarButton("Logout");
68         logoutButton.addActionListener(e -> {
69             int confirm = JOptionPane.showConfirmDialog(
70                 this, "Yakin ingin logout?", "Konfirmasi Logout",
71                 JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE
72             );
73             if (confirm == JOptionPane.YES_OPTION) {
74                 dispose();
75                 new LoginView().setVisible(true);
76             }
77         });
78     }
}
```

Admin Dashboard (2)



```
1 JButton editBookButton = createSidebarButton("Edit Buku");
2 editBookButton.addActionListener(e -> editBookAction());
3
4 JButton deleteBookButton = createSidebarButton("Hapus Buku");
5 deleteBookButton.addActionListener(e -> deleteBookAction());
6
7 JButton deleteBorrowingButton = createSidebarButton("Hapus Peminjaman");
8 deleteBorrowingButton.addActionListener(e -> deleteBorrowingAction());
9
10 sidebarPanel.add(Box.createVerticalStrut(20));
11 sidebarPanel.add(tambahBookButton);
12 sidebarPanel.add(Box.createVerticalStrut(10));
13 sidebarPanel.add(editBookButton);
14 sidebarPanel.add(Box.createVerticalStrut(10));
15 sidebarPanel.add(deleteBookButton);
16 sidebarPanel.add(Box.createVerticalStrut(10));
17 sidebarPanel.add(deleteBorrowingButton);
18 sidebarPanel.add(Box.createVerticalStrut(10));
19 sidebarPanel.add(Box.createVerticalStrut(10));
20 sidebarPanel.add(logoutButton);
21
22 mainPanel = new JPanel(new GridBagLayout());
23 mainPanel.setBackground(Color.WHITE);
24 add(sidebarPanel, BorderLayout.WEST);
25 add(mainPanel, BorderLayout.CENTER);
26
27 refreshMainPanel();
28 setVisible(true);
29 }
30
31 private JButton createSidebarButton(String text) {
32     JButton button = new JButton(text);
33     button.setFont(new Font("SansSerif", Font.BOLD, 14));
34     button.setAlignmentX(Component.CENTER_ALIGNMENT);
35     button.setMaximumSize(new Dimension(180, 40));
36     button.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
37     button.setBackground(new Color(50, 50, 100));
38     button.setForeground(Color.WHITE);
39     button.setFocusPainted(false);
40     button.setOpaque(true);
41     return button;
42 }
43
44 private void tambahBukuAction() {
45     TambahBukuForm formPanel = new TambahBukuForm(this);
46     showFormInMainPanel(formPanel);
47 }
48
49 private void editBookAction() {
50     EditBookHandler.editBook(this, bookListPanel);
51 }
52
53 private void deleteBookAction() {
54     Book selected = bookListPanel.getSelectedBook();
55     if (selected == null) {
56         JOptionPane.showMessageDialog(this, "Pilih buku yang ingin dihapus.");
57         return;
58     }
59     int confirm = JOptionPane.showConfirmDialog(this, "Yakin ingin menghapus buku ini?", "Konfirmasi Hapus", JOptionPane.YES_NO_OPTION);
60     if (confirm == JOptionPane.YES_OPTION) {
61         if (BookController.deleteBook(selected.getIdBuku())) {
62             JOptionPane.showMessageDialog(this, "Buku berhasil dihapus.");
63             bookListPanel.reloadData();
64         } else {
65             JOptionPane.showMessageDialog(this, "Gagal menghapus buku.");
66         }
67     }
68 }
```

Admin Dashboard (3)



```
1  private void deleteBorrowingAction() {
2      Borrowing selected = borrowingListPanel.getSelectedBorrowing();
3      if (selected == null) {
4          JOptionPane.showMessageDialog(this, "Pilih data peminjaman yang ingin dihapus.");
5          return;
6      }
7      int confirm = JOptionPane.showConfirmDialog(this, "Yakin ingin menghapus peminjaman ini?", "Konfirmasi Hapus", JOptionPane.YES_NO_OPTION);
8      if (confirm == JOptionPane.YES_OPTION) {
9          if (BorrowingController.deleteBorrowing(selected.getIdPeminjaman())) {
10             JOptionPane.showMessageDialog(this, "Data peminjaman berhasil dihapus.");
11             borrowingListPanel.reloadData();
12         } else {
13             JOptionPane.showMessageDialog(this, "Gagal menghapus peminjaman.");
14         }
15     }
16 }
17
18 public void refreshMainPanel() {
19     mainPanel.removeAll();
20     mainPanel.setLayout(new GridBagLayout());
21
22     GridBagConstraints gbc = new GridBagConstraints();
23     gbc.insets = new Insets(10, 10, 10, 10);
24     gbc.fill = GridBagConstraints.BOTH;
25     gbc.weightx = 1.0;
26
27     bookListPanel = new BookListPanel();
28     borrowingListPanel = new BorrowingListPanel();
29
30     gbc.gridx = 0;
31     gbc.gridy = 0;
32     gbc.weighty = 0.6;
33     mainPanel.add(new JScrollPane(bookListPanel), gbc);
34
35     gbc.gridy = 1;
36     gbc.weighty = 0.4;
37     mainPanel.add(new JScrollPane(borrowingListPanel), gbc);
38
39     mainPanel.revalidate();
40     mainPanel.repaint();
41 }
42
43 private void showFormInMainPanel(JPanel formPanel) {
44     mainPanel.removeAll();
45     mainPanel.setLayout(new BorderLayout());
46     mainPanel.add(formPanel, BorderLayout.CENTER);
47     mainPanel.revalidate();
48     mainPanel.repaint();
49 }
50
51 public static void main(String[] args) {
52     SwingUtilities.invokeLater(AdminDashboard::new);
53 }
54 }
```

Penjelasan Class Admin Dashboard :

1. Package dan Import (Baris 1–18)

Class ini berada dalam package view, yang berarti merupakan bagian dari lapisan tampilan (GUI) dalam arsitektur sistem.

Import yang digunakan antara lain:

- java.awt dan javax.swing: digunakan untuk membangun elemen GUI seperti JFrame, JPanel, JButton, JLabel, dan komponen lainnya.
- controller.BookController dan controller.BorrowingController: untuk menghubungkan GUI dengan logika pengolahan data buku dan peminjaman.

- model.Book dan model.Borrowing: representasi objek data buku dan peminjaman.

2. Deklarasi Class dan Atribut (Baris 20–29)

Class AdminDashboard merupakan turunan dari JFrame yang berfungsi sebagai jendela utama untuk admin.

Atribut yang digunakan:

- sidebarPanel: panel di sisi kiri untuk navigasi menu admin.
- mainPanel: panel utama untuk menampilkan daftar buku dan peminjaman.
- bookListPanel dan borrowingListPanel: panel khusus yang menampilkan daftar buku dan daftar peminjaman.

3. Konstruktor AdminDashboard() (Baris 31–97)

Konstruktor ini menginisialisasi dan membangun seluruh tampilan dashboard admin.

Langkah-langkah yang dilakukan:

- Menyetel judul, operasi close, dan ukuran tampilan.
- Membuat sidebarPanel dengan gradien warna menggunakan paintComponent.
- Menambahkan label ucapan salam "Hallo Admin".
- Membuat tombol menu sidebar: Tambah Buku, Edit Buku, Hapus Buku, Hapus Peminjaman, dan Logout.
- Masing-masing tombol memiliki ActionListener yang mengatur aksi yang dilakukan saat tombol diklik:
 - Tambah Buku: membuka form tambah buku.
 - Edit Buku: memanggil class EditBookHandler untuk mengedit buku yang dipilih.
 - Hapus Buku: menghapus buku yang dipilih setelah konfirmasi.
 - Hapus Peminjaman: menghapus data peminjaman yang dipilih setelah konfirmasi.
 - Logout: keluar dari dashboard dan kembali ke halaman login.
- mainPanel diatur menggunakan GridBagLayout agar tampilan dapat fleksibel dan rapi.
- Panel utama ditambahkan ke frame.

4. Method createSidebarButton(String text) (Baris 99–108)

Method ini membuat tombol dengan gaya yang konsisten untuk sidebar.

Properti tombol diatur seperti warna, font, ukuran, dan kursor agar seragam.

5. Method tambahBukuAction() (Baris 110–112)

Membuka form tambah buku ke dalam mainPanel dengan memanggil class TambahBukuForm.

6. Method editBookAction() (Baris 114–116)

Memanggil method editBook dari class EditBookHandler dan mengoper referensi ke bookListPanel.

7. Method deleteBookAction() (Baris 118–129)

Menghapus data buku dari tabel:

- Mengecek apakah ada buku yang dipilih.
- Jika ya, meminta konfirmasi pengguna.
- Jika konfirmasi diterima, data buku akan dihapus menggunakan BookController.deleteBook.
- Jika berhasil, data di panel buku akan diperbarui (reload).

8. Method deleteBorrowingAction() (Baris 131–142)

Menghapus data peminjaman dari tabel:

- Mengecek apakah ada data peminjaman yang dipilih.
- Jika ada, meminta konfirmasi pengguna.
- Jika konfirmasi diterima, data peminjaman akan dihapus menggunakan BorrowingController.deleteBorrowing.
- Panel daftar peminjaman diperbarui jika penghapusan berhasil.

9. Method refreshMainPanel() (Baris 144–162)

Digunakan untuk menampilkan kembali panel daftar buku dan daftar peminjaman ke dalam mainPanel.

- Menghapus seluruh komponen sebelumnya.
- Mengatur ulang layout.
- Menambahkan panel buku di bagian atas dan panel peminjaman di bagian bawah menggunakan GridBagConstraints.

10. Method showFormInMainPanel(JPanel formPanel) (Baris 164–170)

Mengganti isi mainPanel dengan form lain (misalnya form tambah buku).

- Layout diubah menjadi BorderLayout.
- Form baru ditambahkan ke posisi tengah.

11. Method main(String[] args) (Baris 172–174)

Merupakan method utama untuk menjalankan aplikasi.

- Menjalankan AdminDashboard menggunakan SwingUtilities.invokeLater() untuk memastikan GUI dibangun di thread yang sesuai.

Fitur-fitur utama class ini:

- Menyediakan navigasi sidebar bagi admin untuk mengelola data buku dan peminjaman.
- Mendukung operasi tambah, edit, dan hapus buku.
- Menyediakan fitur hapus data peminjaman.
- Menyediakan logout untuk kembali ke halaman login.
- Menampilkan data buku dan peminjaman dalam dua panel berbeda yang dapat diperbarui.

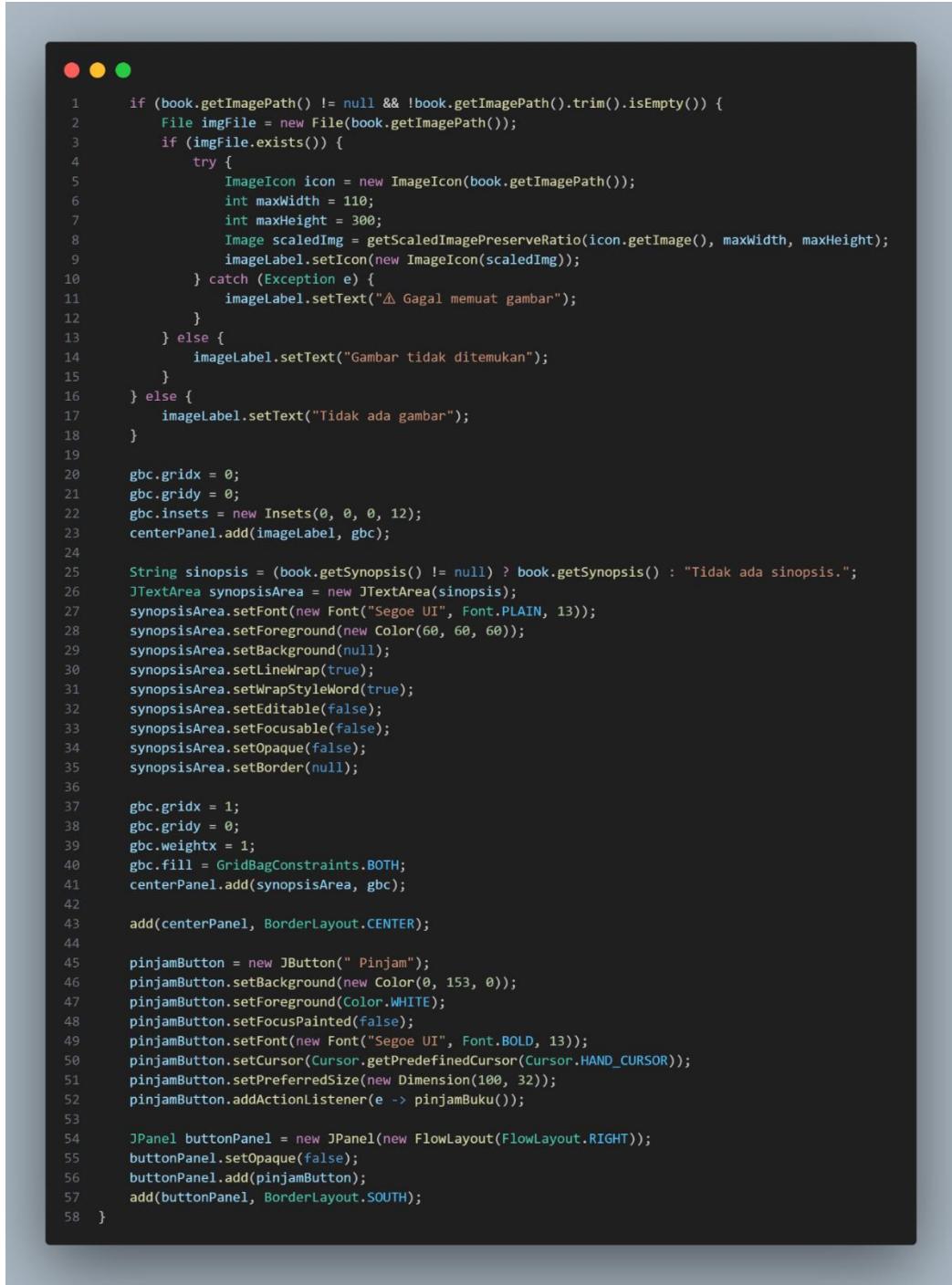
Penerapan prinsip OOP:

- Inheritance: class ini merupakan turunan dari JFrame.
- Encapsulation: logika dikelola dalam method-method privat seperti deleteBookAction, tambahBukuAction, dan sebagainya.
- Event-Driven Programming: penggunaan ActionListener pada tombol-tombol.
- Polymorphism: ditunjukkan dalam penggunaan method yang menerima parameter bertipe JPanel secara dinamis pada method showFormInMainPanel.

Book Card (1)

```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Cursor;
6 import java.awt.Dimension;
7 import java.awt.FlowLayout;
8 import java.awt.Font;
9 import java.awt.GridBagConstraints;
10 import java.awt.GridBagLayout;
11 import java.awt.GridLayout;
12 import java.awt.Image;
13 import java.awt.Insets;
14 import java.io.File;
15
16 import javax.swing.ImageIcon;
17 import javax.swing.JButton;
18 import javax.swing.JLabel;
19 import javax.swing.JOptionPane;
20 import javax.swing.JPanel;
21 import javax.swing.JTextArea;
22 import javax.swing.SwingConstants;
23 import javax.swing.border.CompoundBorder;
24 import javax.swing.border.EmptyBorder;
25 import javax.swing.border.LineBorder;
26
27 import controller.BorrowingController;
28 import model.Book;
29
30 public class BookCard extends JPanel {
31     private Book book;
32     private String userId;
33     private UserDashboard dashboard;
34     private JButton pinjamButton;
35
36     public BookCard(Book book, String userId, UserDashboard dashboard) {
37         this.book = book;
38         this.userId = userId;
39         this.dashboard = dashboard;
40
41         setLayout(new BorderLayout(10, 10));
42         setPreferredSize(new Dimension(400, 260));
43         setBackground(Color.WHITE);
44         setBorder(new CompoundBorder(
45             new LineBorder(new Color(220, 220, 220), 1, true),
46             new EmptyBorder(12, 12, 12, 12)
47         ));
48
49         JPanel header = new JPanel(new GridLayout(2, 1));
50         header.setOpaque(false);
51         JLabel titleLabel = new JLabel(book.getTitle());
52         titleLabel.setFont(new Font("Segoe UI", Font.BOLD, 16));
53         JLabel authorLabel = new JLabel(" " + book.getAuthor());
54         authorLabel.setFont(new Font("Segoe UI", Font.PLAIN, 13));
55         authorLabel.setForeground(new Color(60, 60, 60));
56         header.add(titleLabel);
57         header.add(authorLabel);
58         add(header, BorderLayout.NORTH);
59
60         JPanel centerPanel = new JPanel(new GridBagLayout());
61         centerPanel.setOpaque(false);
62         GridBagConstraints gbc = new GridBagConstraints();
63
64         JLabel imageLabel = new JLabel();
65         imageLabel.setHorizontalAlignment(SwingConstants.CENTER);
66         imageLabel.setVerticalAlignment(SwingConstants.TOP);
```

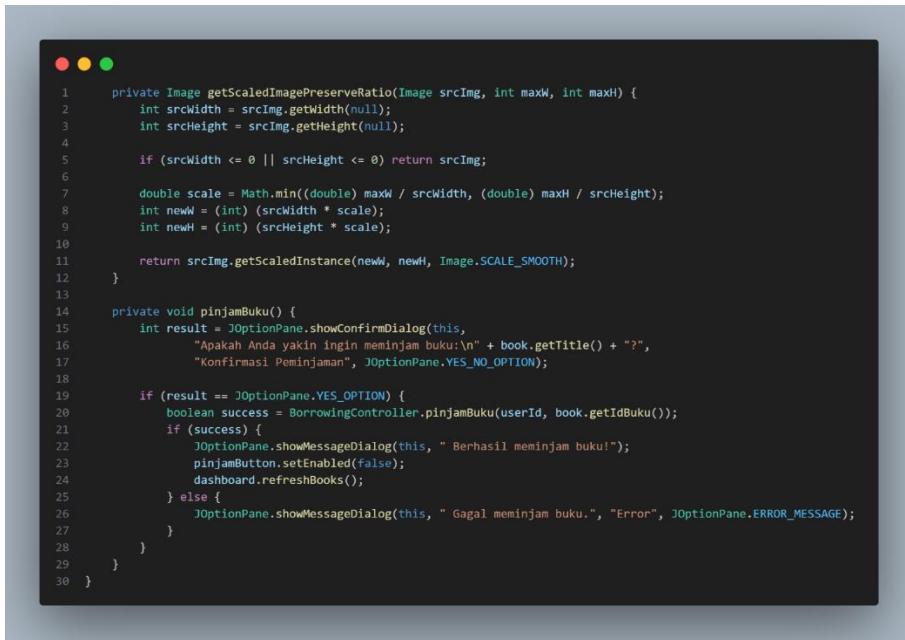
Book Card (2)



The screenshot shows a Java code editor with a dark theme. The code is a Java Swing application for displaying book details. It includes logic to load an image from a file path, set it to a JLabel, and handle exceptions. It also creates a JTextArea for the synopsis, sets its properties like font and foreground color, and adds it to a JPanel. Finally, it creates a JButton for borrowing and adds a JPanel with the button to the main window.

```
1  if (book.getImagePath() != null && !book.getImagePath().trim().isEmpty()) {
2      File imgfile = new File(book.getImagePath());
3      if (imgfile.exists()) {
4          try {
5              ImageIcon icon = new ImageIcon(book.getImagePath());
6              int maxWidth = 110;
7              int maxHeight = 300;
8              Image scaledImg = getScaledImagePreserveRatio(icon.getImage(), maxWidth, maxHeight);
9              imageLabel.setIcon(new ImageIcon(scaledImg));
10         } catch (Exception e) {
11             imageLabel.setText("⚠ Gagal memuat gambar");
12         }
13     } else {
14         imageLabel.setText("Gambar tidak ditemukan");
15     }
16 } else {
17     imageLabel.setText("Tidak ada gambar");
18 }
19
20 gbc.gridx = 0;
21 gbc.gridy = 0;
22 gbc.insets = new Insets(0, 0, 0, 12);
23 centerPanel.add(imageLabel, gbc);
24
25 String sinopsis = (book.getSynopsis() != null) ? book.getSynopsis() : "Tidak ada sinopsis.";
26 JTextArea synopsisArea = new JTextArea(sinopsis);
27 synopsisArea.setFont(new Font("Segoe UI", Font.PLAIN, 13));
28 synopsisArea.setForeground(new Color(60, 60, 60));
29 synopsisArea.setBackground(null);
30 synopsisArea.setLineWrap(true);
31 synopsisArea.setWrapStyleWord(true);
32 synopsisArea.setEditable(false);
33 synopsisArea.setFocusable(false);
34 synopsisArea.setOpaque(false);
35 synopsisArea.setBorder(null);
36
37 gbc.gridx = 1;
38 gbc.gridy = 0;
39 gbc.weightx = 1;
40 gbc.fill = GridBagConstraints.BOTH;
41 centerPanel.add(synopsisArea, gbc);
42
43 add(centerPanel, BorderLayout.CENTER);
44
45 pinjamButton = new JButton(" Pinjam");
46 pinjamButton.setBackground(new Color(0, 153, 0));
47 pinjamButton.setForeground(Color.WHITE);
48 pinjamButton.setFocusPainted(false);
49 pinjamButton.setFont(new Font("Segoe UI", Font.BOLD, 13));
50 pinjamButton.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
51 pinjamButton.setPreferredSize(new Dimension(100, 32));
52 pinjamButton.addActionListener(e -> pinjamBuku());
53
54 JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
55 buttonPanel.setOpaque(false);
56 buttonPanel.add(pinjamButton);
57 add(buttonPanel, BorderLayout.SOUTH);
58 }
```

Book Card (3)

A screenshot of a terminal window showing Java code. The code is a class named BookCard with two methods: getScaledImagePreserveRatio and pinjamBuku. The getScaledImagePreserveRatio method takes an Image source and returns a scaled version. The pinjamBuku method shows a confirmation dialog asking if the user wants to borrow a book, calls a borrowing controller, and then refreshes the dashboard if successful.

Penjelasan Class Book Card :

Berikut adalah penjelasan codingan untuk class BookCard dalam sistem yang sama, disusun dengan format dan gaya penulisan yang selaras dengan penjelasan sebelumnya:

1. Package dan Import (Baris 1–18)

Class ini berada dalam package view, yang menandakan bahwa fungsinya terkait tampilan (GUI). Import yang digunakan antara lain:

- java.awt dan javax.swing digunakan untuk membangun antarmuka pengguna (komponen seperti JPanel, JButton, JLabel, dll).
- java.io.File digunakan untuk mengecek keberadaan gambar buku dari path yang diberikan.
- javax.swing.border digunakan untuk memberikan border pada panel.
- controller.BorrowingController digunakan untuk menangani logika peminjaman buku.
- model.Book digunakan untuk merepresentasikan entitas buku.

2. Deklarasi Class dan Atribut (Baris 20–29)

Class BookCard adalah turunan dari JPanel yang digunakan untuk menampilkan informasi singkat mengenai sebuah buku, lengkap dengan tombol peminjaman.

Atribut yang digunakan:

- Book book: menyimpan informasi buku yang akan ditampilkan.
- String userId: menyimpan ID user yang sedang login.
- UserDashboard dashboard: referensi ke dashboard pengguna untuk melakukan refresh saat buku dipinjam.
- JButton pinjamButton: tombol yang digunakan untuk melakukan peminjaman buku.

3. Konstruktor BookCard(Book book, String userId, UserDashboard dashboard) (Baris 31–101)

Konstruktor ini bertugas menyusun tampilan kartu buku dan menginisialisasi semua komponen GUI yang diperlukan. Langkah-langkahnya:

- Menyimpan parameter ke atribut kelas.
- Menyetel layout, ukuran preferensi, background, dan border panel.
- Membuat panel header berisi judul dan penulis buku.
- Menyusun bagian tengah (centerPanel) menggunakan GridBagLayout. Panel ini berisi gambar buku dan sinopsis.
- Menampilkan gambar buku dalam ukuran yang disesuaikan secara proporsional. Jika gambar tidak ditemukan, akan ditampilkan pesan teks.
- Menampilkan sinopsis buku dalam JTextArea yang tidak dapat diedit.
- Menambahkan tombol "Pinjam" pada bagian bawah panel. Tombol diberi listener untuk menjalankan proses peminjaman buku.

4. Method getScaledImagePreserveRatio(Image srcImg, int maxW, int maxH) (Baris 103–111)

Method ini digunakan untuk mengubah ukuran gambar buku agar tetap proporsional.

Langkah-langkahnya:

- Mengambil ukuran asli gambar.
- Menghitung skala terkecil berdasarkan batas maksimum lebar dan tinggi.
- Mengembalikan gambar dengan ukuran baru menggunakan Image.SCALE_SMOOTH.

5. Method pinjamBuku() (Baris 113–126)

Method ini menangani proses peminjaman buku. Langkah-langkah:

- Menampilkan dialog konfirmasi peminjaman kepada pengguna.
- Jika pengguna menyetujui, method BorrowingController.pinjamBuku dipanggil dengan userId dan ID buku.
- Jika peminjaman berhasil, tombol dinonaktifkan dan dashboard dipanggil untuk memuat ulang daftar buku.
- Jika gagal, ditampilkan pesan error.

Materi OOP yang diterapkan:

- Encapsulation: penggunaan atribut dan method untuk membungkus data dan logika.
- Inheritance: turunan dari JPanel.
- Event-Driven Programming: penggunaan listener pada tombol peminjaman.
- Exception Handling: pengecekan gambar dan penanganan error jika gambar tidak dapat dimuat.
- Polymorphism: penggunaan method yang override seperti setPreferredSize, add, dan lainnya dari superclass JPanel.

Fitur-fitur utamanya:

- Menampilkan informasi singkat buku, termasuk judul, penulis, sinopsis, dan gambar.

- Menyediakan tombol “Pinjam” yang akan menambahkan data ke daftar peminjaman pengguna.
- Mengatur tampilan kartu dengan layout yang rapi dan responsif.
- Menyegarkan tampilan dashboard pengguna setelah peminjaman berhasil.

Class ini berfungsi sebagai komponen individual dari daftar buku dalam antarmuka pengguna dan menjadi penghubung antara user interface dan logika bisnis peminjaman.

Book List Panel (1)



```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Component;
6 import java.awt.Font;
7 import java.util.List;
8
9 import javax.swing.BorderFactory;
10 import javax.swing.JLabel;
11 import javax.swing.JPanel;
12 import javax.swing.JScrollPane;
13 import javax.swing.JTable;
14 import javax.swing.table.DefaultTableCellRenderer;
15 import javax.swing.table.DefaultTableModel;
16
17 import controller.BookController;
18 import model.Book;
19
20 public class BookListPanel extends JPanel {
21     private JTable bookTable;
22     private DefaultTableModel model;
23     private List<Book> daftarBuku;
24
25     public BookListPanel() {
26         setLayout(new BorderLayout());
27         setBorder(BorderFactory.createTitledBorder("Daftar Buku"));
28
29         loadBooks();
30     }
31
32     public void loadBooks() {
33         daftarBuku = BookController.getAllBooks();
34
35         String[] kolom = {"ID Buku", "Judul", "Pengarang", "Penerbit", "Tahun Terbit", "Stok"};
36         Object[][] data = new Object[daftarBuku.size()][kolom.length];
37
38         for (int i = 0; i < daftarBuku.size(); i++) {
39             Book b = daftarBuku.get(i);
40             data[i][0] = b.getIdBuku();
41             data[i][1] = b.getTitle();
42             data[i][2] = b.getAuthor();
43             data[i][3] = b.getPublisher();
44             data[i][4] = b.getTahunTerbit();
45             data[i][5] = b.getStok();
46         }
47
48         model = new DefaultTableModel(data, kolom) {
49             @Override
50             public boolean isCellEditable(int row, int column) {
51                 return false;
52             }
53         };
54
55         if (bookTable == null) {
56             bookTable = new JTable(model);
57             bookTable.setFont(new Font("Arial", Font.PLAIN, 14));
58             bookTable.setRowHeight(40);
59         }
60     }
61 }
```

Penjelasan Class BookListPanel

1. Package dan Import (Baris 1–18)

```
import java.awt.*;
import java.util.List;
import javax.swing.*;
import javax.swing.table.*;
```

```
import controller.BookController;
```

```
import model.Book;
```

- `java.awt.*` dan `javax.swing.*` : Untuk komponen GUI seperti JPanel, JLabel, JTable, JScrollPane, BorderFactory, dll.
- `javax.swing.table.*` : Untuk memanipulasi model tabel (DefaultTableModel, DefaultTableCellRenderer).
- `controller.BookController` : Mengambil data dari database melalui controller.
- `model.Book` : Representasi objek buku.

2. Deklarasi Class dan Atribut (Baris 20–24)

BookListPanel merupakan class turunan dari JPanel, berfungsi sebagai panel GUI untuk menampilkan daftar buku dalam bentuk tabel.

Atribut:

- `bookTable` → Komponen tabel GUI untuk menampilkan data.
- `model` → Model yang mengatur isi data di `bookTable`.
- `daftarBuku` → List objek Book yang akan diisi dari database.

3. Konstruktor BookListPanel() (Baris 26–30)

- `setLayout(new BorderLayout())` → Mengatur layout panel ke bentuk border.
- `setBorder(...)` → Menambahkan border dengan judul “Daftar Buku”.
- `loadBooks()` → Memanggil method untuk mengisi data buku dari database.

4. Method loadBooks() (Baris 32–58)

Bagian 1: Mengambil dan Menyiapkan Data

- Memanggil `getAllBooks()` dari BookController untuk mengambil semua data buku.
- Menentukan header kolom untuk tabel.
- Menyiapkan array data untuk menampung isi tabel.

Bagian 2: Memasukkan Data ke Tabel

Melakukan iterasi pada setiap objek Book.

- Mengambil setiap field menggunakan getter (`getTitle()`, `getAuthor()`, dll) dan mengisinya ke array data.

Bagian 3: Membuat Model Tabel

- Membuat objek DefaultTableModel dengan data dan kolom.
- Override method isCellEditable agar tabel bersifat read-only (tidak bisa diedit langsung).

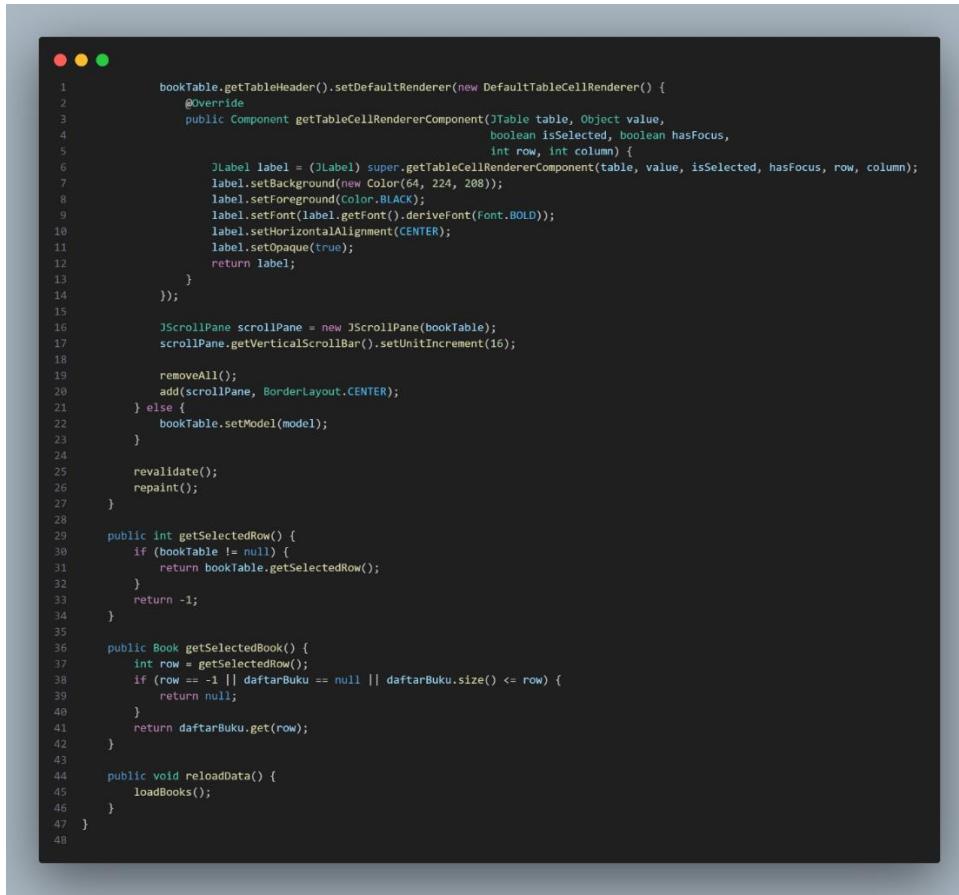
Bagian 4: Inisialisasi Tabel (Jika Belum Dibuat)

- Jika bookTable belum dibuat, maka buat instance baru dari JTable menggunakan model tadi.

Prinsip OOP yang Diterapkan

- Encapsulation: Atribut disimpan sebagai private, manipulasi lewat method (loadBooks()).
- Inheritance: Class BookListPanel mewarisi JPanel.
- Modularitas: Pemisahan antara controller dan view (BookController dan BookListPanel).
- Reusability: Class ini bisa digunakan kembali untuk menampilkan daftar buku di berbagai bagian aplikasi.

Book List Panel (2)



```
1     bookTable.getTableHeader().setDefaultRenderer(new DefaultTableCellRenderer() {
2         @Override
3             public Component getTableCellRendererComponent(JTable table, Object value,
4                 boolean isSelected, boolean hasFocus,
5                 int row, int column) {
6                 JLabel label = (JLabel) super.getTableCellRendererComponent(table, value, isSelected, hasFocus, row, column);
7                 label.setBackground(new Color(64, 224, 208));
8                 label.setForeground(Color.BLACK);
9                 label.setFont(label.getFont().deriveFont(Font.BOLD));
10                label.setHorizontalAlignment(CENTER);
11                label.setOpaque(true);
12                return label;
13            }
14        });
15
16        JScrollPane scrollPane = new JScrollPane(bookTable);
17        scrollPane.getVerticalScrollBar().setUnitIncrement(16);
18
19        removeAll();
20        add(scrollPane, BorderLayout.CENTER);
21    } else {
22        bookTable.setModel(model);
23    }
24
25    revalidate();
26    repaint();
27 }
28
29 public int getSelectedRow() {
30     if (bookTable != null) {
31         return bookTable.getSelectedRow();
32     }
33     return -1;
34 }
35
36 public Book getSelectedBook() {
37     int row = getSelectedRow();
38     if (row == -1 || daftarBuku == null || daftarBuku.size() <= row) {
39         return null;
40     }
41     return daftarBuku.get(row);
42 }
43
44 public void reloadData() {
45     loadBooks();
46 }
47 }
48 }
```

Penjelasan Class BookListPanel (Lanjutan)

1. Styling Header Tabel (Baris 1–14)

Baris ini mengatur tampilan header kolom pada JTable.

2. Menambahkan Tabel ke Panel dengan Scroll (Baris 15–21)

- Membungkus tabel dalam scroll pane agar bisa digulir.
- removeAll() → Menghapus seluruh komponen sebelum menambahkan ulang.

3. Method getSelectedRow() (Baris 26–30)

- Mengembalikan indeks baris yang dipilih oleh user di tabel.
- Jika tidak ada baris dipilih atau tabel null, akan mengembalikan -1.

4. Method getSelectedBook() (Baris 32–38)

- Digunakan untuk mengambil objek Book yang dipilih dari tabel.
- Menggunakan getSelectedRow(), lalu mengakses objek dari daftarBuku.
- Cek keamanan: menghindari akses out-of-bounds atau null.

5. Method reloadData() (Baris 40–46)

- Method publik untuk memuat ulang data dari database.
- Dapat dipanggil dari luar (misalnya dari AdminDashboard) jika buku diperbarui atau ditambah.

Fitur Utama Class BookListPanel

- Menampilkan daftar buku dengan tabel GUI yang rapi dan dapat digulir.
- Menampilkan header kolom dengan warna dan font khusus.
- Mendukung pemilihan baris untuk mengambil objek Book terkait.
- Dapat diperbarui (reload) saat data berubah.

Borrowing List Panel (1)

```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Component;
6 import java.awt.Font;
7 import java.util.List;
8
9 import javax.swing.BorderFactory;
10 import javax.swing.JLabel;
11 import javax.swing.JPanel;
12 import javax.swing.JScrollPane;
13 import javax.swing.JTable;
14 import javax.swing.table.DefaultTableCellRenderer;
15 import javax.swing.table.DefaultTableModel;
16
17 import controller.BorrowingController;
18 import model.Borrowing;
19
20 public class BorrowingListPanel extends JPanel {
21     private JTable peminjamanTable;
22     private DefaultTableModel model;
23     private List<Borrowing> daftarPinjam;
24
25     public BorrowingListPanel() {
26         this("");
27     }
28
29     public BorrowingListPanel(String userId) {
30         setLayout(new BorderLayout());
31         setBorder(BorderFactory.createTitledBorder("Daftar Peminjaman"));
32
33         loadBorrowings(userId);
34     }
35
36     public void loadBorrowings(String userId) {
37         daftarPinjam = BorrowingController.getRiwayat(userId);
38
39         String[] kolom = {"ID Peminjaman", "User ID", "Book ID", "Tanggal Pinjam", "Jatuh Tempo", "Status"};
40         Object[][] data = new Object[daftarPinjam.size()][kolom.length];
41
42         for (int i = 0; i < daftarPinjam.size(); i++) {
43             Borrowing b = daftarPinjam.get(i);
44             data[i][0] = b.getIdPeminjaman();
45             data[i][1] = b.getUserId();
46             data[i][2] = b.getBookId();
47             data[i][3] = b.getTanggalPinjam();
48             data[i][4] = b.getTanggalJatuhTempo();
49             data[i][5] = b.getStatus();
50         }
51
52         model = new DefaultTableModel(data, kolom) {
53             @Override
54             public boolean isCellEditable(int row, int column) {
55                 return false;
56             }
57         };
58     }
59
60     protected void paintComponent(Graphics g) {
61         super.paintComponent(g);
62         // Add your custom painting logic here
63     }
64 }
```

Penjelasan Class BorrowingListPanel

1. Package dan Import (Baris 1–18)

- Class ini berada dalam package view, yang berisi bagian tampilan (GUI) dari aplikasi.
- java.awt.* dan javax.swing.*: Digunakan untuk GUI dan layout panel, tabel, scrollbar, warna, font, dsb.
- javax.swing.table.*: Digunakan untuk manipulasi tabel (JTable, DefaultTableModel, TableCellRenderer).
- controller.BorrowingController: Untuk mengambil data peminjaman dari database.
- model.Borrowing: Representasi objek data peminjaman.

2. Deklarasi Class dan Atribut (Baris 20–23)

- BorrowingListPanel adalah subclass dari JPanel untuk menampilkan daftar peminjaman.
- Atribut penting:
 - peminjamanTable: Komponen tabel untuk menampilkan data.
 - model: Model data tabel.
 - daftarPinjam: List dari objek Borrowing.

3. Konstruktor Default (Baris 25): Konstruktor default akan memanggil konstruktor lain (this("")) dengan parameter userId kosong.

4. Konstruktor dengan userId (Baris 27–31)

- setLayout(new BorderLayout()): Menetapkan layout panel agar dapat menyusun komponen atas-tengah-bawah.
- setBorder(...): Menambahkan border dengan judul “Daftar Peminjaman”.
- loadBorrowings(userId): Memuat data peminjaman untuk user tertentu dari database.

5. Method loadBorrowings(String userId) (Baris 33–57)

Bagian 1: Ambil Data dan Siapkan Array

- Mengambil data riwayat peminjaman dari controller berdasarkan userId.
- Menyiapkan header kolom dan array data untuk menampung isi tabel.

Bagian 2: Isi Data ke Array

- Membuat model tabel dari array data dan kolom.
- Override method isCellEditable agar sel tabel tidak bisa diedit langsung oleh user.

Fitur Utama Class Ini

- Menampilkan daftar peminjaman dalam tabel GUI.
- Menampilkan kolom: ID Peminjaman, User ID, Book ID, Tanggal Pinjam, Jatuh Tempo, Status.
- Mengambil data dari controller berdasarkan user tertentu.
- Data bersifat read-only.

Borrowing List Panel (2)



```
1     if (peminjamanTable == null) {
2         peminjamanTable = new JTable(model);
3         peminjamanTable.setFont(new Font("Arial", Font.PLAIN, 14));
4         peminjamanTable.setRowHeight(40);
5
6         peminjamanTable.getTableHeader().setDefaultRenderer(new DefaultTableCellRenderer() {
7             @Override
8             public Component getTableCellRendererComponent(JTable table, Object value,
9                 boolean isSelected, boolean hasFocus,
10                int row, int column) {
11                 JLabel label = (JLabel) super.getTableCellRendererComponent(table, value, isSelected, hasFocus, row, column);
12                 label.setBackground(new Color(64, 224, 208));
13                 label.setForeground(Color.BLACK);
14                 label.setFont(label.getFont().deriveFont(Font.BOLD));
15                 label.setHorizontalAlignment(CENTER);
16                 label.setOpaque(true);
17                 return label;
18             }
19         });
20
21         JScrollPane scrollPane = new JScrollPane(peminjamanTable);
22         scrollPane.getVerticalScrollBar().setUnitIncrement(16);
23
24         removeAll();
25         add(scrollPane, BorderLayout.CENTER);
26     } else {
27         peminjamanTable.setModel(model);
28     }
29
30     revalidate();
31     repaint();
32 }
33
34     public int getSelectedRow() {
35         if (peminjamanTable != null) {
36             return peminjamanTable.getSelectedRow();
37         }
38         return -1;
39     }
40
41     public Borrowing getSelectedBorrowing() {
42         int row = getSelectedRow();
43         if (row == -1 || daftarPinjam == null || daftarPinjam.size() <= row) {
44             return null;
45         }
46         return daftarPinjam.get(row);
47     }
48
49     public void reloadData() {
50         loadBorrowings("");
51     }
52 }
53 }
```

Penjelasan Class BorrowingListPanel (Lanjutan)

1. Inisialisasi dan Styling Tabel (Baris 1–6)
2. Styling Header Tabel (Baris 7–16)
 - Meng-custom tampilan header kolom tabel (warna & font).
 - Header akan terlihat profesional dan mudah dibaca.
3. Scroll Pane & Penempatan (Baris 18–23)

Membungkus tabel dalam JScrollPane agar bisa digulir secara vertikal.

- setUnitIncrement(16) → mengatur kecepatan scroll agar lebih halus.
- removeAll() → menghapus semua komponen dari panel sebelum menambahkan ulang.

4. Jika Tabel Sudah Dibuat (Baris 24)

Jika peminjamanTable sudah ada, cukup update model-nya dengan data terbaru.

5. Penyegaran Tampilan GUI (Baris 26–27)

- revalidate() → Memperbarui layout dan posisi komponen.
- repaint() → Menggambar ulang tampilan panel agar perubahan terlihat.

6. Method getSelectedRow() (Baris 29–33)

- Mengembalikan nomor baris yang sedang dipilih di tabel.
- Jika belum ada baris dipilih, atau tabel null, maka return -1.

7. Method getSelectedBorrowing() (Baris 35–41)

- Mengembalikan objek Borrowing yang dipilih berdasarkan baris yang dipilih.
- Cek batasan (row valid, daftarPinjam tidak null, dll).

8. Method reloadData() (Baris 43–45)

- Method publik untuk memuat ulang data peminjaman dari awal.
- Secara default memuat seluruh data (userId kosong).
- Bisa dipanggil dari luar class ini (misal dari dashboard admin/user).

Fitur Utama BorrowingListPanel

- Menampilkan daftar peminjaman dalam tabel yang rapi dan terformat.
- Header tabel dengan warna dan font kustom.
- Mendukung scroll vertikal halus.
- Bisa mengambil baris yang dipilih dan objek Borrowing yang sesuai.
- Tabel bisa diperbarui ulang tanpa reload aplikasi.

Edit Book Handler (1)



The screenshot shows a Java code editor window with a dark theme. The title bar has three colored circles (red, yellow, green) at the top. The main area contains the following Java code:

```
1 package view;
2
3 import javax.swing.JOptionPane;
4 import javax.swing.JPanel;
5 import javax.swing.JTextField;
6 import javax.swing.JComboBox;
7
8 import controller.BookController;
9 import model.Book;
10
11 public class EditBookHandler {
12
13     public static void editBook(AdminDashboard dashboard, BookListPanel bookListPanel) {
14         Book selected = bookListPanel.getSelectedBook();
15         if (selected == null) {
16             JOptionPane.showMessageDialog(dashboard, "Pilih buku yang ingin diedit terlebih dahulu.");
17             return;
18         }
19
20         // Komponen input
21         JTextField tfTitle = new JTextField(selected.getTitle());
22         JTextField tfAuthor = new JTextField(selected.getAuthor());
23         JTextField tfYear = new JTextField(String.valueOf(selected.getTahunTerbit()));
24         JComboBox<String> cbPublisher = new JComboBox<>(new String[] {
25             "PEN001 - Gramedia",
26             "PEN002 - Erlangga",
27             "PEN003 - Tiga Serangkai",
28             "PEN004 - Mizan",
29             "PEN005 - Andi Publisher"
30         });
31         tfPublisherSelect(cbPublisher, selected.getPublisher());
32
33         JTextField tfSynopsis = new JTextField(selected.getSynopsis());
34         JTextField tfImagePath = new JTextField(selected.getImagePath());
35         JTextField tfStock = new JTextField(String.valueOf(selected.getStok()));
36
37         JPanel panel = new JPanel();
38         panel.setLayout(new java.awt.GridLayout(0, 1));
39         panel.add(new javax.swing.JLabel("Judul:"));
40         panel.add(tfTitle);
41         panel.add(new javax.swing.JLabel("Pengarang:"));
42         panel.add(tfAuthor);
43         panel.add(new javax.swing.JLabel("Tahun Terbit:"));
44         panel.add(tfYear);
45         panel.add(new javax.swing.JLabel("Penerbit:"));
46         panel.add(cbPublisher);
47         panel.add(new javax.swing.JLabel("Sinopsis:"));
48         panel.add(tfSynopsis);
49         panel.add(new javax.swing.JLabel("Path Foto:"));
50         panel.add(tfImagePath);
51         panel.add(new javax.swing.JLabel("Stok:"));
52         panel.add(tfStock);
53
54         int result = JOptionPane.showConfirmDialog(
55             dashboard,
56             panel,
57             "Edit Buku",
58             JOptionPane.OK_CANCEL_OPTION,
59             JOptionPane.PLAIN_MESSAGE
60         );
61     }
62 }
```

Penjelasan Class EditBookHandler

1. Package dan Import (Baris 1–10)

- Menunjukkan bahwa class ini berada dalam package view.

```
import javax.swing.*; // Untuk komponen GUI seperti JPanel, JTextField,  
JComboBox, JOptionPane
```

```
import controller.BookController; // Untuk menyimpan data perubahan ke database  
(jika ada)
```

```
import model.Book; // Representasi objek buku
```

- Class ini hanya berfungsi sebagai utility/helper untuk menampilkan dan menangani dialog edit data buku.

2. Deklarasi Class (Baris 12)

```
public class EditBookHandler {
```

- Class EditBookHandler bersifat helper class, tidak memperluas JPanel atau JFrame, melainkan digunakan untuk memunculkan dialog edit dari AdminDashboard.

3. Method editBook(...) (Baris 14–60)

```
public static void editBook(AdminDashboard dashboard, BookListPanel bookListPanel)
```

- Static: Bisa dipanggil langsung tanpa perlu membuat objek EditBookHandler.
- Digunakan oleh admin untuk memunculkan form dialog edit buku yang dipilih dari BookListPanel.

4. Baris 15–18: Validasi Pemilihan Buku

- Mengecek apakah ada buku yang sedang dipilih oleh admin.
- Jika tidak ada, tampilkan pesan peringatan dan hentikan proses.

5. Komponen Input untuk Dialog (Baris 21–34)

- Membuat field input dari data buku yang terpilih.

6. Layout Panel Form (Baris 36–51)

- Panel ini akan ditampilkan dalam JOptionPane.
- Layout 1 kolom (label dan input bertumpuk).
- Semua komponen input ditambahkan ke panel.

7. Menampilkan Dialog (Baris 53–59)

- Menampilkan dialog konfirmasi edit.
- Jika user klik OK, maka nilai dari field bisa diambil dan disimpan kembali ke database. Proses menyimpan ke database kemungkinan dilakukan setelah baris ke-60 (di luar gambar), biasanya dengan memanggil BookController.updateBook(...).

Fitur Utama

- Menyediakan form dinamis untuk mengedit data buku.
- Mengisi data awal berdasarkan buku yang dipilih.
- Mendukung perubahan judul, penulis, tahun, penerbit, sinopsis, gambar, dan stok.
- Menggunakan JOptionPane untuk menampilkan panel edit.

Edit Book Handler (2)



```
1     if (result == JOptionPane.OK_OPTION) {
2         try {
3             String title = tfTitle.getText().trim();
4             String author = tfAuthor.getText().trim();
5             int year = Integer.parseInt(tfYear.getText().trim());
6             String publisherId = ((String) cbPublisher.getSelectedItem()).split(" - ")[0];
7             String synopsis = tfSynopsis.getText().trim();
8             String imagePath = tfImagePath.getText().trim();
9             int stock = Integer.parseInt(tfStock.getText().trim());
10
11            if (title.isEmpty() || author.isEmpty() || synopsis.isEmpty() || imagePath.isEmpty()) {
12                throw new IllegalArgumentException("Semua field wajib diisi.");
13            }
14
15            selected.setTitle(title);
16            selected.setAuthor(author);
17            selected.setTahunTerbit(year);
18            selected.setPublisher(publisherId);
19            selected.setSynopsis(synopsis);
20            selected.setImagePath(imagePath);
21            selected.setStok(stock);
22
23            if (BookController.updateBook(selected)) {
24                JOptionPane.showMessageDialog(dashboard, "Buku berhasil diperbarui.");
25                bookListPanel.reloadData();
26            } else {
27                JOptionPane.showMessageDialog(dashboard, "Gagal memperbarui buku.");
28            }
29        } catch (NumberFormatException e) {
30            JOptionPane.showMessageDialog(dashboard, "Tahun dan stok harus berupa angka.");
31        } catch (IllegalArgumentException e) {
32            JOptionPane.showMessageDialog(dashboard, e.getMessage());
33        }
34    }
35}
36
37 private static void tfPublisherSelect(JComboBox<String> combo, String publisherId) {
38     for (int i = 0; i < combo.getItemCount(); i++) {
39         String item = combo.getItemAt(i);
40         if (item.startsWith(publisherId)) {
41             combo.setSelectedIndex(i);
42             break;
43         }
44     }
45 }
46 }
47 }
```

Penjelasan Class EditBookHandler (Lanjutan)

1. Penyimpanan Data Setelah Klik OK (Baris 1–35)

- Mengecek apakah user menekan tombol “OK” pada dialog edit buku.
- Jika ya, masuk ke blok try untuk proses parsing dan validasi input.

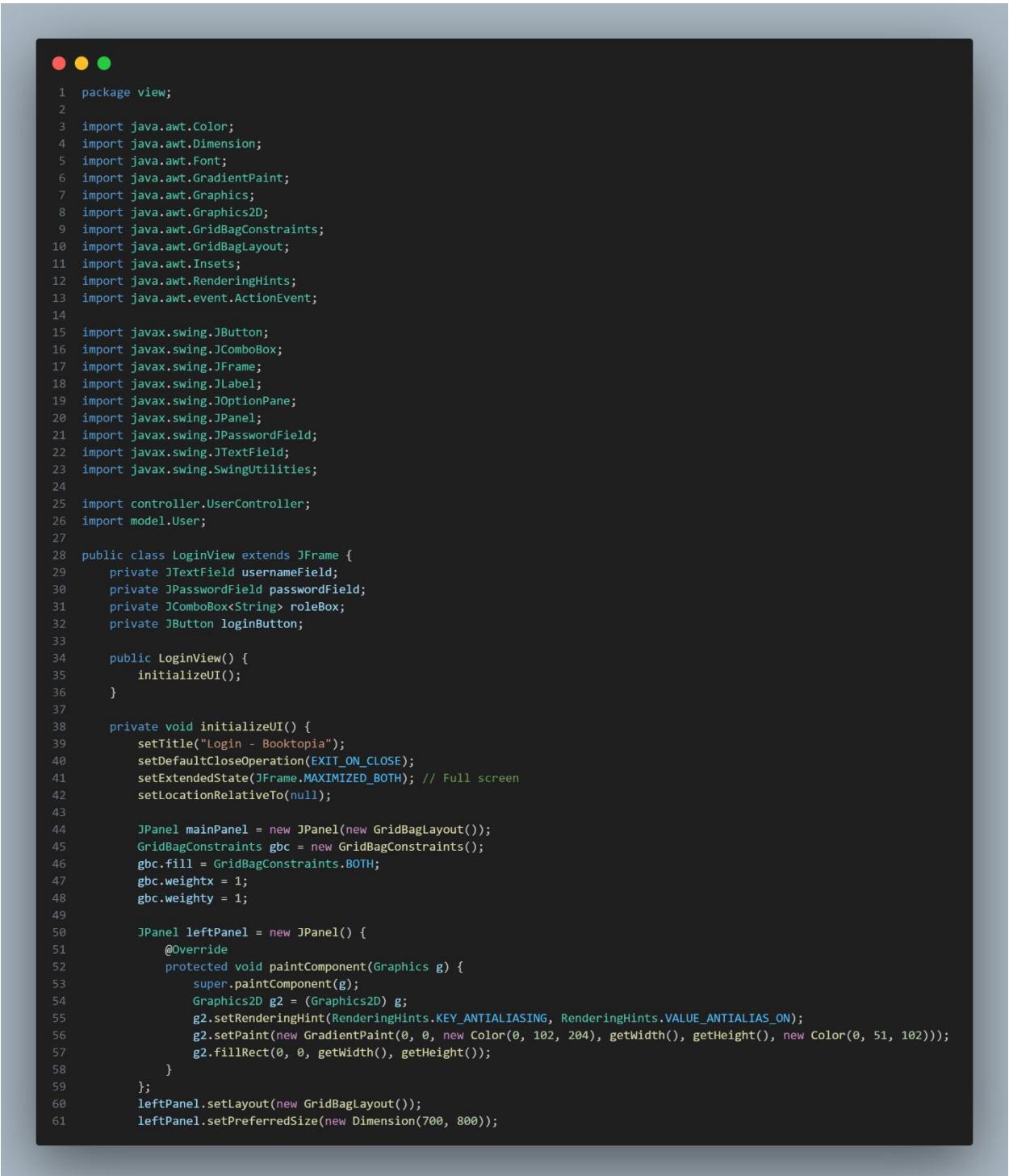
2. Ambil dan Parsing Data Input (Baris 3–10)

- Mengambil nilai dari field input dan membersihkan spasi.
 - publisherId diambil dari awal string comboBox (misalnya: "PEN001 - Gramedia" → "PEN001").
 - Tahun dan stok diparsing ke tipe int.
3. Validasi Data Kosong (Baris 11–13)
- Mengecek apakah ada field yang kosong (kecuali tahun dan stok yang sudah diparsing).
 - Jika ada yang kosong, lempar exception untuk ditangani.
4. Set Nilai Baru ke Objek Book (Baris 15–21)
- Mengatur ulang nilai-nilai baru pada objek Book yang dipilih dengan setter method.
5. Simpan ke Database (Baris 23–29)
- Memanggil method BookController.updateBook() untuk menyimpan perubahan ke database.
 - Jika berhasil, tampilkan pesan sukses dan refresh data buku.
 - Jika gagal, tampilkan pesan error.
6. Penanganan Error (Baris 30–34)
- Menangkap error parsing angka (jika tahun atau stok bukan angka).
 - Menangkap error field kosong dari IllegalArgumentException.
7. Method tfPublisherSelect(...) (Baris 37–46)
- Method pembantu untuk memilih item comboBox sesuai dengan ID penerbit.
 - Loop setiap item, cocokkan dengan publisherId, lalu set ke index yang cocok.

Fitur Utama Tambahan

- Validasi lengkap untuk field input.
- Penanganan error angka dan data kosong.
- Menyimpan data perubahan ke database secara langsung.
- Kombinasi input GUI dan backend controller secara terintegrasi.

Login View (1)



The screenshot shows a Java code editor with a dark theme. The title bar says "Login View (1)". The code is a Java class named LoginView, which extends JFrame. It imports various Java.awt and javax.swing packages, along with UserController and User classes from other packages. The class contains a constructor that initializes the UI, sets the window title, and defines a main panel with a GridBagLayout. A left panel is also defined with a custom paintComponent method that uses Graphics2D to draw a gradient background.

```
1 package view;
2
3 import java.awt.Color;
4 import java.awt.Dimension;
5 import java.awt.Font;
6 import java.awt.GradientPaint;
7 import java.awt.Graphics;
8 import java.awt.Graphics2D;
9 import java.awt.GridBagConstraints;
10 import java.awt.GridBagLayout;
11 import java.awt.Insets;
12 import java.awt.RenderingHints;
13 import java.awt.event.ActionEvent;
14
15 import javax.swing.JButton;
16 import javax.swing.JComboBox;
17 import javax.swing.JFrame;
18 import javax.swing.JLabel;
19 import javax.swing.JOptionPane;
20 import javax.swing.JPanel;
21 import javax.swing.JPasswordField;
22 import javax.swing.JTextField;
23 import javax.swing.SwingUtilities;
24
25 import controller.UserController;
26 import model.User;
27
28 public class LoginView extends JFrame {
29     private JTextField usernameField;
30     private JPasswordField passwordField;
31     private JComboBox<String> roleBox;
32     private JButton loginButton;
33
34     public LoginView() {
35         initializeUI();
36     }
37
38     private void initializeUI() {
39         setTitle("Login - Booktopia");
40         setDefaultCloseOperation(EXIT_ON_CLOSE);
41         setExtendedState(JFrame.MAXIMIZED_BOTH); // Full screen
42         setLocationRelativeTo(null);
43
44         JPanel mainPanel = new JPanel(new GridBagLayout());
45         GridBagConstraints gbc = new GridBagConstraints();
46         gbc.fill = GridBagConstraints.BOTH;
47         gbc.weightx = 1;
48         gbc.weighty = 1;
49
50         JPanel leftPanel = new JPanel() {
51             @Override
52             protected void paintComponent(Graphics g) {
53                 super.paintComponent(g);
54                 Graphics2D g2 = (Graphics2D) g;
55                 g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
56                 g2.setPaint(new GradientPaint(0, 0, new Color(0, 102, 204), getWidth(), getHeight(), new Color(0, 51, 102)));
57                 g2.fillRect(0, 0, getWidth(), getHeight());
58             }
59         };
60         leftPanel.setLayout(new GridBagLayout());
61         leftPanel.setPreferredSize(new Dimension(700, 800));
```

Login View (2)



```
1  JLabel iconLabel = new JLabel("■");
2  iconLabel.setFont(new Font("SansSerif", Font.PLAIN, 80));
3  iconLabel.setForeground(Color.WHITE);
4
5  JLabel textLabel = new JLabel("Booktopia");
6  textLabel.setFont(new Font("SansSerif", Font.BOLD, 28));
7  textLabel.setForeground(Color.WHITE);
8
9  JPanel branding = new JPanel(new GridBagLayout());
10 branding.setOpaque(false);
11 GridBagConstraints bgbc = new GridBagConstraints();
12 bgbc.gridx = 0;
13 branding.add(iconLabel, bgbc);
14 bgbc.gridx = 2;
15 branding.add(textLabel, bgbc);
16
17 leftPanel.add(branding);
18
19 JPanel rightPanel = new JPanel(new GridBagLayout());
20 rightPanel.setBackground(new Color(245, 245, 245));
21 rightPanel.setPreferredSize(new Dimension(700, 800));
22 GridBagConstraints rbc = new GridBagConstraints();
23 rbc.insets = new Insets(15, 15, 15, 15);
24 rbc.fill = GridBagConstraints.HORIZONTAL;
25
26 JLabel welcomeLabel = new JLabel("Selamat Datang Kembali!");
27 welcomeLabel.setFont(new Font("SansSerif", Font.BOLD, 24));
28 welcomeLabel.setForeground(new Color(33, 37, 41));
29 rbc.gridx = 0;
30 rbc.gridy = 0;
31 rbc.gridwidth = 2;
32 rbc.anchor = GridBagConstraints.CENTER;
33 rightPanel.add(welcomeLabel, rbc);
34
35 rbc.gridwidth = 1;
36 rbc.anchor = GridBagConstraints.LINE_END;
37 rbc.gridy++;
38 JLabel usernameLabel = new JLabel("Username:");
39 rightPanel.add(usernameLabel, rbc);
40 rbc.gridx = 1;
41 rbc.anchor = GridBagConstraints.LINE_START;
42 JTextField usernameField = new JTextField(20);
43 rightPanel.add(usernameField, rbc);
44
45 rbc.gridx = 0;
46 rbc.gridy++;
47 rbc.anchor = GridBagConstraints.LINE_END;
48 JLabel passwordLabel = new JLabel("Password:");
49 rightPanel.add(passwordLabel, rbc);
50 rbc.gridx = 1;
51 rbc.anchor = GridBagConstraints.LINE_START;
52 JPasswordField passwordField = new JPasswordField(20);
53 rightPanel.add(passwordField, rbc);
54
55 rbc.gridx = 0;
56 rbc.gridy++;
57 rbc.anchor = GridBagConstraints.LINE_END;
58 JLabel roleLabel = new JLabel("Role:");
59 rightPanel.add(roleLabel, rbc);
60 rbc.gridx = 1;
61 rbc.anchor = GridBagConstraints.LINE_START;
62 JComboBox roleBox = new JComboBox(new String[]{"Admin", "User"});
63 roleBox.setBackground(Color.WHITE);
64 rightPanel.add(roleBox, rbc);
65
66 rbc.gridx = 0;
67 rbc.gridy++;
68 rbc.gridwidth = 2;
69 rbc.anchor = GridBagConstraints.CENTER;
70 JButton loginButton = new JButton("Masuk");
71 loginButton.setPreferredSize(new Dimension(120, 40));
72 loginButton.setBackground(new Color(0, 153, 76));
73 loginButton.setForeground(Color.WHITE);
74 loginButton.setFont(new Font("SansSerif", Font.BOLD, 16));
75 loginButton.setFocusPainted(false);
76 rightPanel.add(loginButton, rbc);
77
78 rbc.gridy++;
79 JButton registerButton = new JButton("Belum punya akun? Daftar di sini");
80 registerButton.setFocusPainted(false);
81 registerButton.setFont(new Font("SansSerif", Font.PLAIN, 12));
82 registerButton.setForeground(new Color(0, 102, 204));
83 registerButton.setBackground(new Color(245, 245, 245));
84 registerButton.setBorderPainted(false);
85 registerButton.setContentAreaFilled(false);
86 registerButton.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
87 rightPanel.add(registerButton, rbc);
88
89 gbc.gridx = 0;
90 mainPanel.add(leftPanel, gbc);
91 gbc.gridx = 1;
92 mainPanel.add(rightPanel, gbc);
93
```

Login View (3)



The screenshot shows a Java code editor with a dark theme. The window title bar has three colored circles (red, yellow, green) in the top-left corner. The main area contains the following Java code:

```
1  loginButton.addActionListener(this::performLogin);
2
3  registerButton.addActionListener(e -> {
4      dispose();
5      new RegisterView().setVisible(true);
6  });
7
8  setVisible(true);
9 }
10
11 private void performLogin(ActionEvent e) {
12     String username = usernameField.getText().trim();
13     String password = new String(passwordField.getPassword()).trim();
14     String role = ((String) roleBox.getSelectedItem()).trim();
15
16     if (username.isEmpty() || password.isEmpty()) {
17         JOptionPane.showMessageDialog(this,
18             "Username dan password tidak boleh kosong",
19             "Error",
20             JOptionPane.ERROR_MESSAGE);
21         return;
22     }
23
24     UserController userController = new UserController();
25     User user = userController.login(username, password, role);
26
27     if (user != null) {
28         dispose();
29         if (role.equalsIgnoreCase("admin")) {
30             System.out.println("Login sebagai Admin berhasil.");
31             new AdminDashboard().setVisible(true);
32         } else {
33             System.out.println("Login sebagai User berhasil: " + user.getUsername());
34             new UserDashboard(user).setVisible(true);
35         }
36     } else {
37         JOptionPane.showMessageDialog(this,
38             "Username, password, atau role salah",
39             "Login Gagal",
40             JOptionPane.ERROR_MESSAGE);
41     }
42 }
43
44 public static void main(String[] args) {
45     SwingUtilities.invokeLater(LoginView::new);
46 }
47 }
48 }
```

Penjelasan Class Login View:

Class LoginView merupakan tampilan antarmuka (GUI) utama yang digunakan untuk proses login pengguna dalam aplikasi Booktopia. Tampilan ini dibangun menggunakan Java Swing dan menerapkan prinsip pemisahan antara tampilan (view) dan logika kontrol (controller). User dapat login sebagai Admin atau User dengan memasukkan data yang sesuai.

Komponen Utama:

1. Input Komponen:

- usernameField: Komponen JTextField untuk menginput username.
- passwordField: Komponen JPasswordField untuk menginput password secara tersembunyi.
- roleBox: Komponen JComboBox yang digunakan untuk memilih role pengguna, yaitu Admin atau User.

2. Tombol Aksi:

- loginButton: Tombol untuk memproses login berdasarkan data yang diinputkan.
- registerButton: Tombol untuk mengarahkan pengguna ke halaman registrasi (RegisterView).

3. Panel dan Layout:

- mainPanel: Panel utama dengan layout GridBagLayout yang membagi tampilan menjadi dua bagian, yaitu panel kiri (leftPanel) dan panel kanan (rightPanel).
- leftPanel: Panel dengan desain gradasi warna biru dan menampilkan branding aplikasi berupa ikon dan nama aplikasi "Booktopia".
- rightPanel: Panel yang berisi form login seperti input username, password, pilihan role, serta tombol aksi.

Fungsi dan Proses Kerja:

1. Inisialisasi Tampilan (initializeUI()):

- Menentukan properti awal dari frame seperti judul, ukuran layar penuh, dan posisi tengah.
- Menyusun layout antar panel dan komponen menggunakan GridBagConstraints.

- Menambahkan desain visual seperti font, warna, dan layout yang bersih dan responsif.
 - Menambahkan event listener pada tombol login dan register.
2. Proses Login (performLogin(ActionEvent e)):
- Mengambil nilai dari field username, password, dan role.
 - Melakukan validasi bahwa username dan password tidak kosong.
 - Memanggil method login() dari UserController untuk memverifikasi login.
 - Jika login berhasil, frame login akan ditutup dan membuka dashboard sesuai role pengguna (AdminDashboard atau UserDashboard).
 - Jika login gagal, akan ditampilkan pesan kesalahan menggunakan JOptionPane.
3. Main Method:
- Method main() digunakan untuk menjalankan aplikasi dan menampilkan tampilan login menggunakan SwingUtilities.invokeLater() agar dijalankan pada Event Dispatch Thread (EDT).

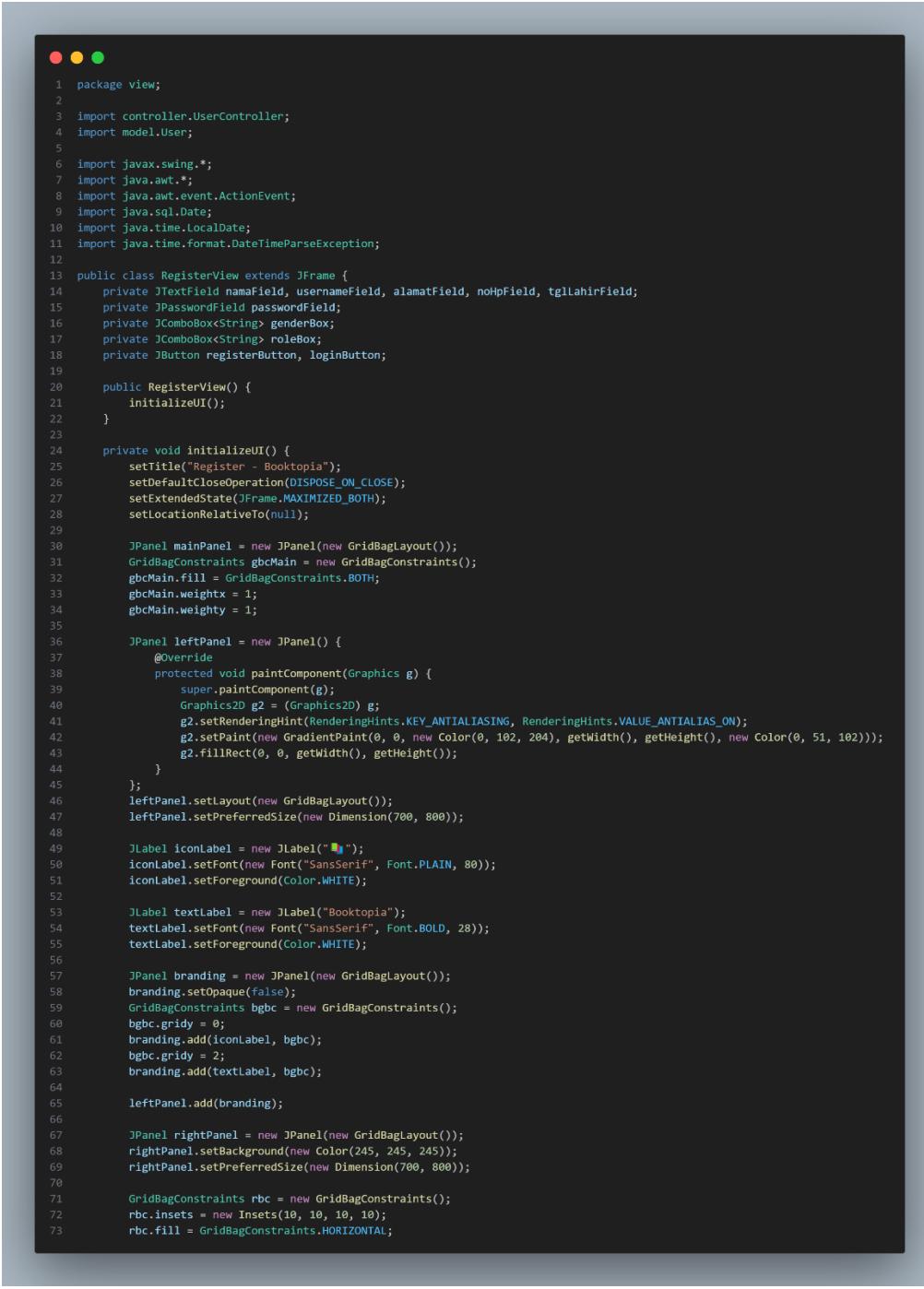
Relasi dengan Class Lain:

- UserController: Digunakan untuk melakukan proses login berdasarkan input user.
- User: Model yang merepresentasikan data pengguna.
- RegisterView: Tampilan pendaftaran akun baru, dibuka dari tombol register.
- AdminDashboard dan UserDashboard: Tampilan utama yang dibuka setelah login berhasil, tergantung role pengguna.

Kesimpulan:

Class LoginView merupakan komponen tampilan awal dalam sistem aplikasi Booktopia yang menangani proses autentikasi pengguna. Tampilan ini dirancang dengan antarmuka modern, responsif, dan mendukung autentikasi berdasarkan role.

Register View (1)



```
1 package view;
2
3 import controller.UserController;
4 import model.User;
5
6 import javax.swing.*;
7 import java.awt.*;
8 import java.awt.event.ActionEvent;
9 import java.sql.Date;
10 import java.time.LocalDate;
11 import java.time.format.DateTimeParseException;
12
13 public class RegisterView extends JFrame {
14     private JTextField namaField, usernameField, alamatField, noHpField, tglLahirField;
15     private JPasswordField passwordField;
16     private JComboBox<String> genderBox;
17     private JComboBox<String> roleBox;
18     private JButton registerButton, loginButton;
19
20     public RegisterView() {
21         initializeUI();
22     }
23
24     private void initializeUI() {
25         setTitle("Register - Booktopia");
26         setDefaultCloseOperation(DISPOSE_ON_CLOSE);
27         setExtendedState(JFrame.MAXIMIZED_BOTH);
28         setLocationRelativeTo(null);
29
30         JPanel mainPanel = new JPanel(new GridBagLayout());
31         GridBagConstraints gbcMain = new GridBagConstraints();
32         gbcMain.fill = GridBagConstraints.BOTH;
33         gbcMain.weightx = 1;
34         gbcMain.weighty = 1;
35
36         JPanel leftPanel = new JPanel() {
37             @Override
38             protected void paintComponent(Graphics g) {
39                 super.paintComponent(g);
40                 Graphics2D g2 = (Graphics2D) g;
41                 g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
42                 g2.setPaint(new GradientPaint(0, 0, new Color(0, 102, 204), getWidth(), getHeight(), new Color(0, 51, 102)));
43                 g2.fillRect(0, 0, getWidth(), getHeight());
44             }
45         };
46         leftPanel.setLayout(new GridBagLayout());
47         leftPanel.setPreferredSize(new Dimension(700, 800));
48
49         JLabel iconLabel = new JLabel("User");
50         iconLabel.setFont(new Font("SansSerif", Font.PLAIN, 80));
51         iconLabel.setForeground(Color.WHITE);
52
53         JLabel textLabel = new JLabel("Booktopia");
54         textLabel.setFont(new Font("SansSerif", Font.BOLD, 28));
55         textLabel.setForeground(Color.WHITE);
56
57         JPanel branding = new JPanel(new GridBagLayout());
58         branding.setOpaque(false);
59         GridBagConstraints bgbc = new GridBagConstraints();
60         bgbc.gridx = 0;
61         branding.add(iconLabel, bgbc);
62         bgbc.gridx = 2;
63         branding.add(textLabel, bgbc);
64
65         leftPanel.add(branding);
66
67         JPanel rightPanel = new JPanel(new GridBagLayout());
68         rightPanel.setBackground(new Color(245, 245, 245));
69         rightPanel.setPreferredSize(new Dimension(700, 800));
70
71         GridBagConstraints rbc = new GridBagConstraints();
72         rbc.insets = new Insets(10, 10, 10, 10);
73         rbc.fill = GridBagConstraints.HORIZONTAL;
```

Register View (2)



```
1 JLabel welcomeLabel = new JLabel("Buat Akun Baru");
2 welcomeLabel.setFont(new Font("SansSerif", Font.BOLD, 24));
3 welcomeLabel.setForeground(new Color(33, 37, 41));
4 rbc.gridx = 0;
5 rbc.gridy = 0;
6 rbc.gridwidth = 2;
7 rbc.anchor = GridBagConstraints.CENTER;
8 rightPanel.add(welcomeLabel, rbc);
9
10 rbc.gridwidth = 1;
11 rbc.anchor = GridBagConstraints.LINE_END;
12
13 rbc.gridy++;
14 rbc.gridx = 0;
15 rightPanel.add(new JLabel("Nama lengkap:"), rbc);
16 rbc.gridx = 1;
17 rbc.anchor = GridBagConstraints.LINE_START;
18 namaField = new JTextField(20);
19 rightPanel.add(namaField, rbc);
20
21 rbc.gridy++;
22 rbc.gridx = 0;
23 rbc.anchor = GridBagConstraints.LINE_END;
24 rightPanel.add(new JLabel("Username:"), rbc);
25 rbc.gridx = 1;
26 rbc.anchor = GridBagConstraints.LINE_START;
27 usernameField = new JTextField(20);
28 rightPanel.add(usernameField, rbc);
29
30 rbc.gridy++;
31 rbc.gridx = 0;
32 rbc.anchor = GridBagConstraints.LINE_END;
33 rightPanel.add(new JLabel("Password:"), rbc);
34 rbc.gridx = 1;
35 rbc.anchor = GridBagConstraints.LINE_START;
36 passwordField = new JPasswordField(20);
37 rightPanel.add(passwordField, rbc);
38
39 rbc.gridy++;
40 rbc.gridx = 0;
41 rbc.anchor = GridBagConstraints.LINE_END;
42 rightPanel.add(new JLabel("Alamat:"), rbc);
43 rbc.gridx = 1;
44 rbc.anchor = GridBagConstraints.LINE_START;
45 alamatField = new JTextField(20);
46 rightPanel.add(alamatField, rbc);
47
48 rbc.gridy++;
49 rbc.gridx = 0;
50 rbc.anchor = GridBagConstraints.LINE_END;
51 rightPanel.add(new JLabel("Jenis Kelamin:"), rbc);
52 rbc.gridx = 1;
53 rbc.anchor = GridBagConstraints.LINE_START;
54 genderBox = new JComboBox<>(new String[]{"Laki-laki", "Perempuan"});
55 genderBox.setBackground(Color.WHITE);
56 rightPanel.add(genderBox, rbc);
57
58 rbc.gridy++;
59 rbc.gridx = 0;
60 rbc.anchor = GridBagConstraints.LINE_END;
61 rightPanel.add(new JLabel("Tanggal Lahir (yyyy-MM-dd):"), rbc);
62 rbc.gridx = 1;
63 rbc.anchor = GridBagConstraints.LINE_START;
64 tglLahirField = new JTextField(20);
65 rightPanel.add(tglLahirField, rbc);
66
67 rbc.gridy++;
68 rbc.gridx = 0;
69 rbc.anchor = GridBagConstraints.LINE_END;
70 rightPanel.add(new JLabel("Nomor HP:"), rbc);
71 rbc.gridx = 1;
72 rbc.anchor = GridBagConstraints.LINE_START;
73 noHpField = new JTextField(20);
74 rightPanel.add(noHpField, rbc);
75
76 rbc.gridy++;
77 rbc.gridx = 0;
78 rbc.anchor = GridBagConstraints.LINE_END;
79 rightPanel.add(new JLabel("Role:"), rbc);
80 rbc.gridx = 1;
81 rbc.anchor = GridBagConstraints.LINE_START;
82 roleBox = new JComboBox<>(new String[]{"user"});
83 roleBox.setEnabled(false);
84 roleBox.setBackground(Color.LIGHT_GRAY);
85 rightPanel.add(roleBox, rbc);
86
87 rbc.gridy++;
88 rbc.gridx = 0;
89 rbc.gridwidth = 2;
90 rbc.anchor = GridBagConstraints.CENTER;
91 registerButton = new JButton("Daftar");
92 registerButton.setPreferredSize(new Dimension(120, 40));
93 registerButton.setBackground(new Color(0, 153, 76));
94 registerButton.setForeground(Color.WHITE);
95 registerButton.setFont(new Font("SansSerif", Font.BOLD, 16));
96 registerButton.setFocusPainted(false);
97 rightPanel.add(registerButton, rbc);
```

Register View (3)



The screenshot shows a Java Swing application window titled "Register View (3)". The window has a dark theme with red, yellow, and green window control buttons at the top. Inside, there is a code editor displaying Java code. The code is related to a registration view, handling button actions, form validation, and user registration logic.

```
1 rbc.gridx++;
2 JButton loginButton = new JButton("Sudah punya akun? Masuk di sini");
3 loginButton.setFocusPainted(false);
4 loginButton.setFont(new Font("SansSerif", Font.PLAIN, 12));
5 loginButton.setForeground(new Color(0, 102, 204));
6 loginButton.setBackground(new Color(245, 245, 245));
7 loginButton.setBorderPainted(false);
8 loginButton.setContentAreaFilled(false);
9 loginButton.setCursor(new Cursor(Cursor.HAND_CURSOR));
10 rightPanel.add(loginButton, rbc);
11
12 gbcMain.gridx = 0;
13 mainPanel.add(leftPanel, gbcMain);
14 gbcMain.gridx = 1;
15 mainPanel.add(rightPanel, gbcMain);
16
17 add(mainPanel);
18
19 registerButton.addActionListener(this::performRegister);
20
21 loginButton.addActionListener(e -> {
22     dispose();
23     new LoginView().setVisible(true);
24 });
25
26 setVisible(true);
27 }
28
29 private void performRegister(ActionEvent e) {
30     try {
31         String nama = namaField.getText().trim();
32         String username = usernameField.getText().trim();
33         String password = new String(passwordField.getPassword()).trim();
34         String alamat = alamatField.getText().trim();
35         String gender = (String) genderBox.getSelectedItem();
36         String tglLahirStr = tglLahirField.getText().trim();
37         String noHp = noHpField.getText().trim();
38         String role = (String) roleBox.getSelectedItem();
39
40         if (nama.isEmpty() || username.isEmpty() || password.isEmpty()) {
41             JOptionPane.showMessageDialog(this, "Nama, Username, dan Password wajib diisi.");
42             return;
43         }
44
45         LocalDate tglLahir;
46         try {
47             tglLahir = LocalDate.parse(tglLahirStr);
48         } catch (DateTimeParseException ex) {
49             JOptionPane.showMessageDialog(this, "Format tanggal lahir salah. Gunakan yyyy-MM-dd.");
50             return;
51         }
52
53         User user = new User();
54         user.setNama(nama);
55         user.setUsername(username);
56         user.setPassword(password);
57         user.setAlamat(alamat);
58         user.setJenisKelamin(gender);
59         user.setTglLahir(Date.valueOf(tglLahir));
60         user.setNoHp(noHp);
61         user.setRole(role);
62
63         boolean success = new UserController().register(user);
64         if (success) {
65             JOptionPane.showMessageDialog(this, "Registrasi berhasil! Silakan login.");
66             new LoginView().setVisible(true);
67             dispose();
68         } else {
69             JOptionPane.showMessageDialog(this, "Registrasi gagal. Username mungkin sudah terdaftar.");
70         }
71
72     } catch (Exception ex) {
73         JOptionPane.showMessageDialog(this, "Terjadi error: " + ex.getMessage());
74         ex.printStackTrace();
75     }
76 }
77 }
78 }
79 }
```

Penjelasan Class Register View:

1. Package dan Import (Baris 1–8)

package view; menandakan bahwa class ini berada di dalam package view.

Import yang digunakan:

- controller.UserController dan model.User: untuk menghubungkan tampilan register dengan proses logika dan data pengguna.
- javax.swing.*: untuk komponen GUI seperti JFrame, JPanel, JTextField, JLabel, JComboBox, dan JButton.
- java.awt.*: digunakan untuk mengatur tampilan GUI seperti warna, layout, font, dan grafik.
- java.awt.event.ActionEvent: untuk menangani event klik tombol.
- java.sql.Date: digunakan untuk konversi tanggal.
- java.time.LocalDate dan java.time.format.DateTimeParseException: untuk parsing input tanggal dari pengguna.

2. Deklarasi Class dan Atribut (Baris 10–18)

Class RegisterView merupakan turunan dari JFrame dan digunakan untuk menampilkan form registrasi pengguna baru.

Atribut-atribut yang digunakan di antaranya:

- JTextField untuk nama lengkap, username, alamat, nomor HP, dan tanggal lahir.
- JPasswordField untuk input kata sandi.
- JComboBox untuk memilih jenis kelamin dan role pengguna.
- JButton untuk tombol daftar dan login.

3. Konstruktor RegisterView() (Baris 20–23)

Memanggil method initializeUI() untuk mengatur dan menampilkan komponen tampilan registrasi ke layar.

4. Method initializeUI() (Baris 25–155)

Method ini bertugas mengatur seluruh tampilan user interface registrasi.

Langkah-langkahnya:

- Menyetel properti dasar JFrame seperti judul, mode fullscreen, posisi tengah layar, dan operasi keluar.
- Membuat dua panel utama: kiri dan kanan, yang diatur dengan GridBagLayout.
- Panel kiri (leftPanel) menampilkan branding aplikasi dengan efek gradasi warna biru. Di dalamnya terdapat icon dan label nama aplikasi.
- Panel kanan (rightPanel) menampilkan form input dengan berbagai field: nama lengkap, username, password, alamat, jenis kelamin, tanggal lahir, nomor HP, dan role.
- Field jenis kelamin diatur dalam combo box berisi pilihan "Laki-laki" dan "Perempuan".
- Field role diisi dengan pilihan default "user" dan dibuat tidak bisa diubah (disabled).
- Dua tombol utama ditambahkan: tombol "Daftar" untuk proses registrasi, dan tombol navigasi ke login "Sudah punya akun? Masuk di sini".
- Masing-masing tombol diberi style visual yang sesuai agar tampilan lebih menarik.
- Action listener ditambahkan untuk:
 - Tombol daftar: memanggil method performRegister().
 - Tombol login: membuka form login (LoginView) dan menutup jendela saat ini.

5. Method performRegister(ActionEvent e) (Baris 157–197)

Method ini menangani logika saat pengguna mengklik tombol daftar. Langkah-langkah yang dilakukan:

- Mengambil nilai dari seluruh input field.
- Validasi untuk memastikan nama, username, dan password tidak kosong.
- Validasi tanggal lahir menggunakan LocalDate.parse. Jika format salah, akan muncul pesan peringatan.

- Membuat objek User dan mengisi semua datanya berdasarkan input dari form.
- Memanggil method register() dari UserController untuk menyimpan data ke database.
- Jika berhasil, ditampilkan pesan sukses dan form login akan dibuka.
- Jika gagal, akan muncul pesan error (misalnya username sudah digunakan).
- Keseluruhan proses dibungkus dalam blok try-catch untuk menangani error yang tidak terduga dan menampilkan pesan kesalahan jika ada.

Fitur utama:

- Form pendaftaran pengguna baru dengan input validasi sederhana.
- Tampilan antarmuka yang modern dengan layout dua panel dan gradient.
- Perpindahan ke halaman login setelah proses registrasi selesai.

Penerapan prinsip OOP:

- Encapsulation: data pengguna dikelola melalui objek User dan dikirim ke controller.
- Event-Driven Programming: penggunaan action listener pada tombol.
- Exception Handling: validasi input dan penggunaan try-catch saat proses registrasi.
- Modular Design: pemisahan antara logika (controller), data (model), dan tampilan (view).

Riwayat Peminjaman (1)

```
1 package view;
2
3 import java.awt.*;
4 import java.awt.event.*;
5 import java.io.File;
6 import java.net.URI;
7 import java.util.List;
8
9 import javax.swing.*;
10 import javax.swing.border.EmptyBorder;
11 import javax.swing.table.*;
12
13 import controller.BorrowingController;
14 import controller.BookController;
15 import model.Borrowing;
16 import model.Book;
17 import model.User;
18
19 public class RiwayatPeminjamanPanel extends JPanel {
20     private JTable table;
21     private DefaultTableModel model;
22     private User user;
23
24     private final Color PRIMARY_COLOR = new Color(36, 52, 92); // #24345C
25     private final Color BG_COLOR = new Color(245, 245, 245);
26
27     private JLabel emptyLabel;
28
29     public RiwayatPeminjamanPanel(User user) {
30         this.user = user;
31         setLayout(new BorderLayout());
32         setBackground(BG_COLOR);
33         setBorder(new EmptyBorder(15, 15, 15, 15));
34
35         JLabel titleLabel = new JLabel("Riwayat Peminjaman Anda");
36         titleLabel.setFont(new Font("Segoe UI", Font.BOLD, 18));
37         titleLabel.setForeground(PRIMARY_COLOR);
38         titleLabel.setHorizontalAlignment(SwingConstants.CENTER);
39
40         JButton refreshButton = new JButton("Refresh");
41         refreshButton.setFont(new Font("Segoe UI", Font.PLAIN, 13));
42         refreshButton.setBackground(new Color(0, 153, 0));
43         refreshButton.setForeground(Color.WHITE);
44         refreshButton.setFocusPainted(false);
45         refreshButton.setCursor(new Cursor(Cursor.HAND_CURSOR));
46         refreshButton.addActionListener(new ActionListener() {
47             @Override
48             public void actionPerformed(ActionEvent e) {
49                 loadData();
50             }
51         });
52     }
53 }
```

Riwayat Peminjaman (2)

```
● ● ●
1 JPanel topPanel = new JPanel(new BorderLayout());
2 topPanel.setBackground(BG_COLOR);
3 topPanel.add(titleLabel, BorderLayout.CENTER);
4 topPanel.add(refreshButton, BorderLayout.EAST);
5 add(topPanel, BorderLayout.NORTH);
6
7 model = new DefaultTableModel(new String[]{"ID", "ID Buku", "Nama Buku", "Tanggal Pinjam", "Tanggal Kembali", "Status", "Detail"}, 0) {
8     @Override
9     public boolean isCellEditable(int row, int column) {
10         return column == 6;
11     }
12 };
13
14 table = new JTable(model);
15 table.setFont(new Font("Segoe UI", Font.PLAIN, 13));
16 table.setRowHeight(30);
17 table.getTableHeader().setFont(new Font("Segoe UI", Font.BOLD, 13));
18 table.getTableHeader().setBackground(PRIMARY_COLOR);
19 table.getTableHeader().setForeground(Color.WHITE);
20
21 table.getColumnModel().getColumn("Detail").setCellRenderer(new TableCellRenderer() {
22     @Override
23     public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column) {
24         if (value instanceof JButton) {
25             return (JButton) value;
26         }
27         return new JButton("Lihat");
28     }
29 });
30
31 table.getColumnModel().getColumn("Detail").setCellEditor(new DefaultCellEditor(new JCheckBox()));
32 private JButton button;
33
34     @Override
35     public Component getTableCellEditorComponent(JTable table, Object value, boolean isSelected, int row, int column) {
36         if (value instanceof JButton) {
37             button = (JButton) value;
38             return button;
39         }
40         return new JButton("Lihat");
41     }
42
43     @Override
44     public Object getCellEditorValue() {
45         return button;
46     }
47 });
48
49 JScrollPane scrollPane = new JScrollPane(table);
50 scrollPane.setBorder(BorderFactory.createLineBorder(new Color(200, 200, 200)));
51 add(scrollPane, BorderLayout.CENTER);
52
53 emptyLabel = new JLabel("Belum ada riwayat peminjaman.");
54 emptyLabel.setFont(new Font("Segoe UI", Font.ITALIC, 16));
55 emptyLabel.setForeground(new Color(120, 120, 120));
56 emptyLabel.setHorizontalAlignment(SwingConstants.CENTER);
57 emptyLabel.setVisible(false);
58 add(emptyLabel, BorderLayout.SOUTH);
59
60 loadData();
61 }
```

Riwayat Peminjaman (3)

```
1  private void loadData() {
2      model.setRowCount(0);
3      List<Borrowing> list = BorrowingController.getRiwayat(user.getIdUser());
4
5      if (list.isEmpty()) {
6          emptyLabel.setVisible(true);
7          table.setVisible(false);
8          table.getTableHeader().setVisible(false);
9      } else {
10         emptyLabel.setVisible(false);
11         table.setVisible(true);
12         table.getTableHeader().setVisible(true);
13
14         for (Borrowing b : list) {
15             Book book = BookController.getBookById(b.getBookId());
16             String namaBuku = (book != null && book.getTitle() != null) ? book.getTitle() : "(Tidak ditemukan)";
17             String fileUrl = (book != null && book.getFilePath() != null) ? book.getFilePath() : null;
18
19             boolean isExpired = b.isExpired();
20
21             JButton detailButton = new JButton("Lihat");
22             detailButton.setBackground(PRIMARY_COLOR);
23             detailButton.setForeground(Color.WHITE);
24             detailButton.setFocusPainted(false);
25             detailButton.setFont(new Font("Segoe UI", Font.PLAIN, 12));
26
27             if (isExpired) {
28                 detailButton.setEnabled(false);
29                 detailButton.setToolTipText("E-book tidak tersedia karena peminjaman sudah kadaluarsa.");
30             } else {
31                 detailButton.addActionListener(new ActionListener() {
32                     @Override
33                     public void actionPerformed(ActionEvent e) {
34                         if (fileUrl != null && !fileUrl.isEmpty()) {
35                             try {
36                                 Desktop.getDesktop().browse(new URI(fileUrl));
37                             } catch (Exception ex) {
38                                 JOptionPane.showMessageDialog(null, "Gagal membuka link e-book: " + ex.getMessage());
39                             }
40                         } else {
41                             JOptionPane.showMessageDialog(null, "URL e-book belum tersedia.");
42                         }
43                     }
44                 });
45             }
46
47             model.addRow(new Object[]{
48                 b.getIdPeminjaman(),
49                 b.getBookId(),
50                 namaBuku,
51                 b.getTanggalPinjam(),
52                 b.getTanggalJatuhTempo(),
53                 isExpired ? "Kadaluarsa" : "Aktif",
54                 detailButton
55             });
56         }
57     }
58
59     revalidate();
60     repaint();
61 }
62
63 public void refresh() {
64     loadData();
65 }
66 }
67 }
```

Penjelasan Class Riwayat Peminjaman :

1. Package dan Import (Baris 1–18)

package view; :Menandakan bahwa class ini berada dalam package view. Import:

- java.awt dan java.awt.event (baris 3–6): Untuk GUI dan event seperti Color, BorderLayout, Font, Cursor, Component, ActionListener.
- java.io.File, java.net.URI (baris 7–8): Untuk akses file dan membuka link e-book.
- java.util.List (baris 9): Untuk menyimpan daftar data riwayat peminjaman.
- javax.swing dan javax.swing.table (baris 11–13): Untuk komponen GUI seperti JPanel, JTable, JScrollPane, JButton, JLabel, DefaultTableModel, dll.
- controller.BorrowingController dan BookController (baris 15–16): Untuk mengakses data peminjaman dan buku dari database.
- model.Borrowing, Book, User (baris 17–18): Representasi entitas data dari sistem.

2. Deklarasi Class dan Atribut (Baris 20–29)

public class RiwayatPeminjamanPanel extends JPanel :Class ini merupakan turunan dari JPanel dan berfungsi sebagai panel GUI untuk menampilkan riwayat peminjaman pengguna.

Atribut:

- JTable table (baris 21): Tabel utama yang menampilkan daftar riwayat.
- DefaultTableModel model (baris 22): Model data tabel.
- User user (baris 23): Data user yang sedang login.
- Color PRIMARY_COLOR & BG_COLOR (baris 25–26): Untuk styling warna utama dan latar belakang.
- JLabel emptyLabel (baris 28): Label yang ditampilkan jika riwayat kosong.

3. Konstruktor RiwayatPeminjamanPanel(User user) (Baris 30–100)

```
public RiwayatPeminjamanPanel(User user)
```

Fungsi utama konstruktor ini:

- Menyimpan user ke atribut user (baris 31).
- Menyetel layout, background, dan border panel (baris 32–34).
- Membuat dan mengatur label judul "Riwayat Peminjaman Anda" (baris 36–39).
- Membuat tombol Refresh (baris 41–48) dan menambahkan ActionListener untuk memuat ulang data (baris 49–53).
- Menyusun topPanel untuk menyatukan titleLabel dan tombol refresh (baris 55–58).
- Inisialisasi model tabel dengan 7 kolom termasuk tombol "Detail" (baris 60–63). Kolom hanya dapat diedit pada kolom terakhir (Detail).
- Inisialisasi JTable dan styling tampilannya (baris 65–69).
- Menambahkan renderer khusus untuk kolom "Detail" (baris 71–76): menampilkan tombol "Lihat".
- Menambahkan editor tombol "Lihat" agar dapat diklik (baris 78–87).
- Membungkus table dalam JScrollPane dan menambahkannya ke panel utama (baris 89–90).
- Menambahkan emptyLabel di bagian bawah jika data kosong (baris 92–95).
- Memanggil loadData() untuk pertama kali saat panel dibuka (baris 97).

4. Method loadData() (Baris 102–167)

```
private void loadData()
```

Method ini bertugas mengambil dan menampilkan riwayat peminjaman dari database.

Langkah-langkah:

- Menghapus semua data sebelumnya di tabel (baris 103).
- Mengambil data list Borrowing dari BorrowingController (baris 104).

- Jika list kosong:
 - Menampilkan emptyLabel, menyembunyikan table dan header (baris 106–109).
- Jika data ada:
 - Menyembunyikan emptyLabel, menampilkan tabel (baris 111–113).
 - Melakukan perulangan pada tiap objek Borrowing (baris 115).
 - Mengambil data buku dari BookController.getBookById (baris 116).
 - Mengecek judul buku dan file URL (baris 117–118).
 - Menentukan apakah peminjaman kadaluarsa (baris 120).
 - Membuat tombol "Lihat" dan menyesuaikan tampilannya (baris 122–125).
 - Jika kadaluarsa: tombol disabled dan menampilkan tooltip (baris 127–129).
 - Jika aktif: tombol akan membuka URL e-book dengan Desktop.getDesktop().browse() (baris 131–137).
 - Menambahkan data ke baris tabel model (baris 139–146).
 - Memanggil revalidate() dan repaint() untuk menyegarkan tampilan (baris 149–150).

Materi OOP yang diterapkan:

- Event-Driven Programming (penggunaan ActionListener).
- Exception Handling (try-catch saat membuka URI e-book).
- Encapsulation (mengelola data lewat method dan objek).
- Polymorphism (custom TableCellRenderer dan TableCellEditor).

5. Method refresh() (Baris 152–154)

```
public void refresh()
```

Method ini bersifat publik dan bisa dipanggil dari class luar (misalnya UserDashboard) untuk me-refresh isi tabel. Fungsi ini hanya memanggil kembali loadData() (baris 153).

Materi OOP: Method, Encapsulation

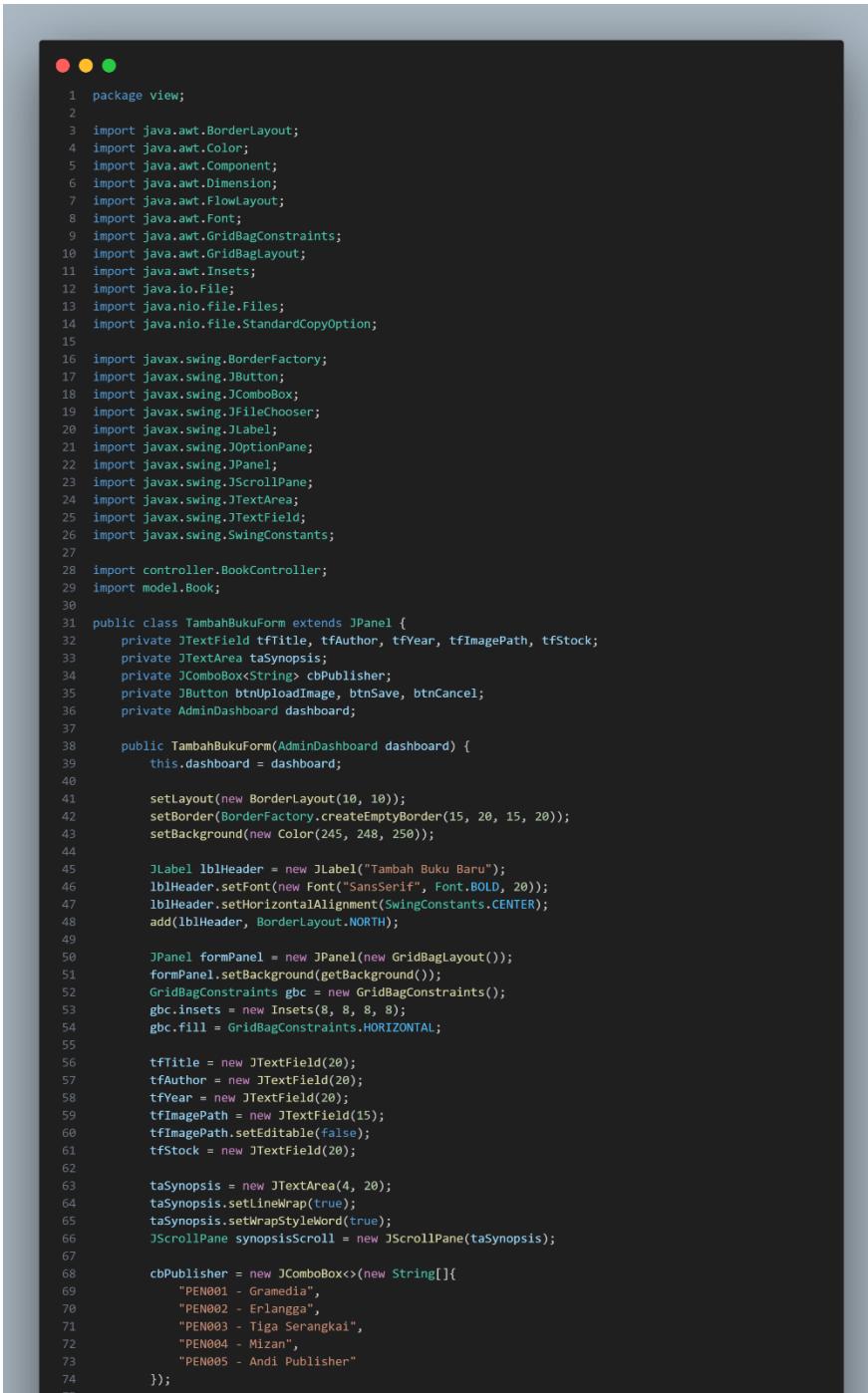
Fitur-fitur utamanya:

- Menampilkan riwayat peminjaman dalam bentuk tabel.
- Memberikan tombol “Lihat” untuk membuka link e-book jika status peminjaman masih aktif.
- Menampilkan pesan “Belum ada riwayat peminjaman” jika daftar kosong.
- Tombol Refresh untuk memperbarui data peminjaman.

Prinsip OOP yang digunakan:

- Inheritance: turunan dari JPanel.
- Encapsulation: atribut disimpan privat dan dimanipulasi melalui method.
- Event-Driven: penggunaan tombol dan listener.
- Polymorphism: custom renderer dan editor tabel.
- Exception Handling: penggunaan try-catch saat membuka URL.

Tambah Buku Form (1)



```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Component;
6 import java.awt.Dimension;
7 import java.awt.FlowLayout;
8 import java.awt.Font;
9 import java.awt.GridBagConstraints;
10 import java.awt.GridBagLayout;
11 import java.awt.Insets;
12 import java.io.File;
13 import java.nio.file.Files;
14 import java.nio.file.StandardCopyOption;
15
16 import javax.swing.BorderFactory;
17 import javax.swing.JButton;
18 import javax.swing.JComboBox;
19 import javax.swing.JFileChooser;
20 import javax.swing.JLabel;
21 import javax.swing.JOptionPane;
22 import javax.swing.JPanel;
23 import javax.swing.JScrollPane;
24 import javax.swing.JTextArea;
25 import javax.swing.JTextField;
26 import javax.swing.SwingConstants;
27
28 import controller.BookController;
29 import model.Book;
30
31 public class TambahBukuForm extends JPanel {
32     private JTextField tfTitle, tfAuthor, tfYear, tfImagePath, tfStock;
33     private JTextArea taSynopsis;
34     private JComboBox<String> cbPublisher;
35     private JButton btnUploadImage, btnSave, btnCancel;
36     private AdminDashboard dashboard;
37
38     public TambahBukuForm(AdminDashboard dashboard) {
39         this.dashboard = dashboard;
40
41         setLayout(new BorderLayout(10, 10));
42         setBorder(BorderFactory.createEmptyBorder(15, 20, 15, 20));
43         setBackground(new Color(245, 248, 250));
44
45         JLabel lblHeader = new JLabel("Tambah Buku Baru");
46         lblHeader.setFont(new Font("SansSerif", Font.BOLD, 20));
47         lblHeader.setHorizontalTextPosition(SwingConstants.CENTER);
48         add(lblHeader, BorderLayout.NORTH);
49
50         JPanel formPanel = new JPanel(new GridBagLayout());
51         formPanel.setBackground(getBackground());
52         GridBagConstraints gbc = new GridBagConstraints();
53         gbc.insets = new Insets(8, 8, 8, 8);
54         gbc.fill = GridBagConstraints.HORIZONTAL;
55
56         tfTitle = new JTextField(20);
57         tfAuthor = new JTextField(20);
58         tfYear = new JTextField(20);
59         tfImagePath = new JTextField(15);
60         tfImagePath.setEditable(false);
61         tfStock = new JTextField(20);
62
63         taSynopsis = new JTextArea(4, 20);
64         taSynopsis.setLineWrap(true);
65         taSynopsis.setWrapStyleWord(true);
66         JScrollPane synopsisScroll = new JScrollPane(taSynopsis);
67
68         cbPublisher = new JComboBox<>(new String[]{
69             "PEN001 - Gramedia",
70             "PEN002 - Erlangga",
71             "PEN003 - Tiga Serangkai",
72             "PEN004 - Mizan",
73             "PEN005 - Andi Publisher"
74         });
75     }
76 }
```

Tambah Buku Form (2)

```
75     btnUploadImage = new JButton("Pilih Gambar");
76     btnUploadImage.addActionListener(e -> {
77         JFileChooser fileChooser = new JFileChooser();
78         int result = fileChooser.showOpenDialog(this);
79         if (result == JFileChooser.APPROVE_OPTION) {
80             File selectedFile = fileChooser.getSelectedFile();
81             String fileName = selectedFile.getName();
82             File destDir = new File("images");
83
84             if (!destDir.exists()) {
85                 destDir.mkdirs();
86             }
87
88             File destFile = new File(destDir, fileName);
89             try {
90                 Files.copy(
91                     selectedFile.toPath(),
92                     destFile.toPath(),
93                     StandardCopyOption.REPLACE_EXISTING
94                 );
95                 tfImagePath.setText("images/" + fileName); // relative path
96             } catch (Exception ex) {
97                 JOptionPane.showMessageDialog(this, "Gagal menyimpan gambar: " + ex.getMessage());
98             }
99         }
100    });
101});
```

Tambah Buku Form (3)

```
1 addField(formPanel, gbc, 0, "Judul:", tfTitle);
2 addField(formPanel, gbc, 1, "Penulis:", tfAuthor);
3 addField(formPanel, gbc, 2, "Tahun Terbit:", tfYear);
4 addField(formPanel, gbc, 3, "Penerbit:", cbPublisher);
5 addField(formPanel, gbc, 4, "Sinopsis:", synopsisScroll);
6 addFieldWithButton(formPanel, gbc, 5, "Path Foto:", tfImagePath, btnUploadImage);
7 addField(formPanel, gbc, 6, "Stok:", tfStock);
8
9 JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 15, 10));
10 buttonPanel.setBackground(getBackground());
11 btnSave = new JButton("Simpan");
12 btnCancel = new JButton("Batal");
13
14 btnSave.setPreferredSize(new Dimension(120, 30));
15 btnCancel.setPreferredSize(new Dimension(120, 30));
16
17 btnSave.addActionListener(e -> saveBook());
18 btnCancel.addActionListener(e -> {
19     clearForm();
20     dashboard.refreshMainPanel();
21 });
22
23 buttonPanel.add(btnSave);
24 buttonPanel.add(btnCancel);
25
26 add(formPanel, BorderLayout.CENTER);
27 add(buttonPanel, BorderLayout.SOUTH);
28 }
29
30 private void saveBook() {
31     String title = tfTitle.getText().trim();
32     String author = tfAuthor.getText().trim();
33     String tahunTerbitStr = tfYear.getText().trim();
34     String synopsis = synopsisScroll.getText().trim();
35     String imagePath = tfImagePath.getText().trim();
36     String stokStr = tfStock.getText().trim();
37
38     String publisherItem = (String) cbPublisher.getSelectedItem();
39     String publisher = publisherItem.split("-")[0];
40
41     if (title.isEmpty() || author.isEmpty() || tahunTerbitStr.isEmpty() || publisher.isEmpty()
42         || synopsis.isEmpty() || imagePath.isEmpty() || stokStr.isEmpty()) {
43         JOptionPane.showMessageDialog(this, "Semua field harus diisi!", "Peringatan", JOptionPane.WARNING_MESSAGE);
44         return;
45     }
46
47     int tahunTerbit;
48     try {
49         tahunTerbit = Integer.parseInt(tahunTerbitStr);
50         stok = Integer.parseInt(stokStr);
51     } catch (NumberFormatException e) {
52         JOptionPane.showMessageDialog(this, "Tahun Terbit dan Stok harus angka!", "Peringatan", JOptionPane.WARNING_MESSAGE);
53         return;
54     }
55
56     Book newBook = new Book();
57     newBook.setIdBook(generateNewId());
58     newBook.setTitle(title);
59     newBook.setAuthor(author);
60     newBook.setPublisher(publisher);
61     newBook.setTahunTerbit(tahunTerbit);
62     newBook.setSynopsis(synopsis);
63     newBook.setImagePath(imagePath);
64     newBook.setStok(stok);
65
66     if (BookController.insertBook(newBook)) {
67         JOptionPane.showMessageDialog(this, "Buku berhasil ditambahkan!");
68         clearForm();
69         dashboard.refreshMainPanel();
70     } else {
71         JOptionPane.showMessageDialog(this, "Gagal menambahkan buku.", "Error", JOptionPane.ERROR_MESSAGE);
72     }
73 }
```

Tambah Buku Form (4)

```
1  private void clearForm() {
2      tfTitle.setText("");
3      tfAuthor.setText("");
4      tfYear.setText("");
5      cbPublisher.setSelectedIndex(0);
6      taSynopsis.setText("");
7      tfImagePath.setText("");
8      tfStock.setText("");
9  }
10
11 private String generateNewId() {
12     String lastId = BookController.getLastIdBuku();
13     if (lastId == null) return "BK001";
14     try {
15         int num = Integer.parseInt(lastId.substring(2)) + 1;
16         return String.format("BK%03d", num);
17     } catch (NumberFormatException e) {
18         return "BK001";
19     }
20 }
21
22 private void addField(JPanel panel, GridBagConstraints gbc, int row, String label, Component field) {
23     gbc.gridx = 0;
24     gbc.gridy = row;
25     panel.add(new JLabel(label), gbc);
26     gbc.gridx = 1;
27     panel.add(field, gbc);
28 }
29
30 private void addFieldWithButton(JPanel panel, GridBagConstraints gbc, int row, String label, Component field, JButton button) {
31     gbc.gridx = 0;
32     gbc.gridy = row;
33     panel.add(new JLabel(label), gbc);
34     gbc.gridx = 1;
35     panel.add(field, gbc);
36     gbc.gridx = 2;
37     panel.add(button, gbc);
38 }
39 }
```

Penjelasan Class Tambah Buku Form :

1. Package dan Import (Baris 1–26)

- package view; :Menandakan bahwa class ini berada dalam package view.
- Import:
- java.awt dan java.awt.event (baris 3–9): Untuk komponen dan tata letak GUI seperti BorderLayout, Color, Font, Insets, Component, Dimension, FlowLayout, dan GridBagLayout.
 - java.io.File dan java.nio.file (baris 10–11): Untuk mengelola file gambar yang dipilih dan menyalinnya ke folder tujuan.

- javax.swing (baris 13–22): Untuk membangun antarmuka pengguna seperti JPanel, JLabel, JButton, JTextField, JTextArea, JComboBox, JScrollPane, JFileChooser, JOptionPane, dan lain-lain.
 - controller.BookController (baris 24): Untuk mengelola penyimpanan data buku ke database.
 - model.Book (baris 25): Sebagai representasi entitas data buku dalam sistem.
2. Deklarasi Class dan Atribut (Baris 28–30)
- ```
public class TambahBukuForm extends JPanel :Menunjukkan bahwa class ini merupakan turunan dari JPanel dan digunakan sebagai panel form untuk menambah data buku baru.
```
- Atribut:
- JTextField tfTitle, tfAuthor, tfYear, tfImagePath, tfStock: Field input untuk judul, pengarang, tahun terbit, path gambar, dan stok buku (baris 29).
  - JTextArea taSynopsis: Input untuk sinopsis buku.
  - JComboBox cbPublisher: Dropdown untuk memilih penerbit.
  - JButton btnUploadImage, btnSave, btnCancel: Tombol untuk memilih gambar, menyimpan buku, dan membatalkan aksi (baris 30).
  - AdminDashboard dashboard: Referensi ke dashboard admin tempat form ini akan ditampilkan.

3. Konstruktor TambahBukuForm(AdminDashboard dashboard) (Baris 32–98)

```
public TambahBukuForm(AdminDashboard dashboard)
```

Fungsi utama konstruktor:

- Menyimpan dashboard ke atribut lokal (baris 33).
- Menyetel layout BorderLayout, border padding, dan latar belakang panel (baris 35–37).
- Membuat dan menambahkan JLabel sebagai header “Tambah Buku Baru” ke bagian atas panel (baris 39–41).

- Membuat formPanel menggunakan GridBagLayout untuk mengatur input secara fleksibel (baris 43–45).
  - Inisialisasi semua field input: tfTitle, tfAuthor, tfYear, tfImagePath (tidak dapat diedit), tfStock, serta taSynopsis yang dibungkus dalam JScrollPane (baris 47–54).
  - Menambahkan pilihan penerbit ke dalam cbPublisher (baris 56–61).
  - Membuat tombol btnUploadImage dengan ActionListener yang membuka JFileChooser untuk memilih gambar, lalu menyalin gambar ke folder images, dan menyimpan path-nya ke tfImagePath (baris 63–74).
  - Menambahkan field input dan label ke formPanel menggunakan method addField dan addFieldWithButton (baris 76–82).
  - Menyusun panel tombol Simpan dan Batal (btnSave dan btnCancel) ke dalam buttonPanel dengan FlowLayout (baris 84–91).
  - Memberi ukuran tetap dan menambahkan ActionListener: btnSave memanggil method saveBook() dan btnCancel akan mengosongkan form dan kembali ke tampilan awal dashboard (baris 93–97).
  - Menambahkan formPanel ke bagian tengah dan buttonPanel ke bagian bawah panel utama (baris 98).
4. Method saveBook() (Baris 100–139)

```
private void saveBook()
```

Method ini bertugas untuk menyimpan data buku baru ke dalam database.

Langkah-langkah:

- Mengambil input dari seluruh field form dan menyimpannya ke variabel lokal (baris 101–106).
- Mengekstrak kode penerbit dari item terpilih di cbPublisher (baris 108–109).
- Melakukan validasi: jika ada field kosong, maka ditampilkan pesan peringatan menggunakan JOptionPane (baris 111–114).

- Melakukan parsing nilai tahun terbit dan stok, jika terjadi NumberFormatException maka ditampilkan pesan bahwa input harus berupa angka (baris 116–120).
- Membuat objek Book baru dan menyetel semua field-nya menggunakan setter (baris 122–130).
- Memanggil BookController.insertBook(newBook):
- Jika berhasil :tampilkan pesan sukses, kosongkan form, dan perbarui dashboard (baris 132–134).
- Jika gagal :tampilkan pesan error (baris 136–137).

Materi OOP: Method, Object, Encapsulation, Exception Handling, Event-Driven.

#### 5. Method clearForm() (Baris 141–148)

```
private void clearForm()
```

Digunakan untuk mengosongkan seluruh isi field input, biasanya setelah data berhasil disimpan atau saat tombol batal diklik. Semua field dikosongkan dan dropdown penerbit dikembalikan ke indeks pertama.

#### 6. Method generateNewId() (Baris 150–174)

```
private String generateNewId()
```

Digunakan untuk membuat ID buku baru secara otomatis.

Langkah-langkah:

- Mengambil ID terakhir dari database melalui BookController.getLastIdBuku() (baris 151).
- Jika ID tidak ditemukan :return BK001 (baris 152).
- Jika ID ditemukan :ambil angka di belakang prefix BK, tambah 1, dan format kembali menjadi string ID seperti BK002, BK003, dst. (baris 154–157).
- Jika parsing gagal :fallback ke BK001 (baris 158–159).

## 7. Method addField(...) dan addFieldWithButton(...) (Baris 176–186)

Method ini digunakan untuk menambahkan pasangan label dan komponen input ke formPanel.

- `addField()`: Menambahkan label dan field biasa (baris 176–180).
- `addFieldWithButton()`: Menambahkan label, field, dan tombol (seperti tombol pilih gambar) dalam satu baris (baris 181–186).

Fitur-fitur utama:

- Menyediakan form antarmuka admin untuk menambah buku ke sistem.
- Mendukung pemilihan gambar melalui file explorer dan menyimpannya ke folder lokal. Otomatisasi ID buku.
- Validasi form input sebelum penyimpanan.
- Interaksi melalui tombol Simpan dan Batal.

Prinsip OOP yang digunakan:

- Inheritance: class TambahBukuForm merupakan turunan dari JPanel.
- Encapsulation: semua atribut bersifat private dan tidak diakses langsung dari luar.
- Event-Driven: tombol-tombol seperti Simpan, Batal, dan Pilih Gambar merespons event klik.
- Exception Handling: diterapkan saat parsing input dan saat menyimpan file gambar.
- Object: membuat dan memanipulasi instance Book sebelum disimpan ke database.

## User Dashboard (1)



```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Component;
6 import java.awt.Cursor;
7 import java.awt.Dimension;
8 import java.awt.FlowLayout;
9 import java.awt.Font;
10 import java.awt.GradientPaint;
11 import java.awt.Graphics;
12 import java.awt.Graphics2D;
13 import java.awt.GridLayout;
14
15 import javax.swing.BorderFactory;
16 import javax.swing.Box;
17 import javax.swingBoxLayout;
18 import javax.swing.JButton;
19 import javax.swing.JLabel;
20 import javax.swing.JOptionPane;
21 import javax.swing.JPanel;
22 import javax.swing.JScrollPane;
23 import javax.swing.JTextField;
24 import javax.swing.SwingConstants;
25 import javax.swing.SwingUtilities;
26 import javax.swing.border.EmptyBorder;
27
28 import controller.BookController;
29 import model.Book;
30 import model.User;
31
32 public class UserDashboard extends javax.swing.JFrame {
33 private User user;
34 private JPanel contentPanel;
35 private JPanel booksGridPanel;
36 private JScrollPane bookScrollPane;
37 private RiwayatPeminjamanPanel riwayatPanel;
38 private JTextField searchField;
39 private JButton activeButton = null;
40
41 public UserDashboard(User user) {
42 this.user = user;
43 setTitle("Booktopia | Selamat datang, " + user.getUsername());
44 setExtendedState(MAXIMIZED_BOTH);
45 setDefaultCloseOperation(EXIT_ON_CLOSE);
46 setLocationRelativeTo(null);
47 getContentPane().setBackground(new Color(245, 245, 245));
48 setLayout(new BorderLayout());
49
50 JPanel sidebar = new JPanel() {
51 @Override
52 protected void paintComponent(Graphics g) {
53 super.paintComponent(g);
54 Graphics2D g2d = (Graphics2D) g;
55 GradientPaint gp = new GradientPaint(
56 0, 0, new Color(0, 102, 204),
57 0, getHeight(), new Color(60, 110, 170));
58 g2d.setPaint(gp);
59 g2d.fillRect(0, 0, getWidth(), getHeight());
60 }
61 };
62 sidebar.setLayout(new BorderLayout());
63 sidebar.setPreferredSize(new Dimension(220, getHeight()));
64 sidebar.setOpaque(false);
65
66 JLabel logoLabel = new JLabel("Booktopia", SwingConstants.CENTER);
67 logoLabel.setFont(new Font("Rockwell", Font.BOLD, 26));
68 logoLabel.setForeground(Color.WHITE);
69 logoLabel.setBorder(new EmptyBorder(30, 10, 30, 10));
70 sidebar.add(logoLabel, BorderLayout.NORTH);
71
72 JPanel buttonPanel = new JPanel();
73 buttonPanel.setLayout(new BoxLayout(buttonPanel, BoxLayout.Y_AXIS));
74 buttonPanel.setOpaque(false);
75 buttonPanel.setBorder(new EmptyBorder(10, 20, 10, 20));
76
77 JButton homeBtn = new JButton("Beranda");
78 JButton pinjamanBtn = new JButton("Pinjaman");
79 JButton logoutBtn = new JButton("Logout");

```

## User Dashboard (2)



The screenshot shows a Java Swing application window titled "User Dashboard". The window has a dark theme with red, yellow, and green window control buttons. Inside the window, there is a code editor displaying Java code. The code is related to a user dashboard application, specifically handling button listeners and UI components like JPanel, JButton, JTextField, and JScrollPane.

```
1 homeBtn.addActionListener(e -> {
2 searchField.setText("");
3 showBooksGrid();
4 setActiveButton(homeBtn);
5 });
6
7 pinjamanBtn.addActionListener(e -> {
8 showRiwayatPanel();
9 setActiveButton(pinjamanBtn);
10 });
11
12 logoutBtn.addActionListener(e -> {
13 int confirm = JOptionPane.showConfirmDialog(this,
14 "Apakah Anda yakin ingin logout?",
15 "Konfirmasi Logout",
16 JOptionPane.YES_NO_OPTION);
17 if (confirm == JOptionPane.YES_OPTION) {
18 dispose();
19 new LoginView();
20 }
21 });
22
23 for (JButton btn : new JButton[]{homeBtn, pinjamanBtn, logoutBtn}) {
24 styleSidebarButton(btn);
25 buttonPanel.add(btn);
26 buttonPanel.add(Box.createVerticalStrut(15));
27 }
28
29 sidebar.add(buttonPanel, BorderLayout.CENTER);
30 add(sidebar, BorderLayout.WEST);
31
32 JPanel topPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT, 10, 5));
33 topPanel.setBackground(new Color(0, 102, 204));
34 topPanel.setBorder(new EmptyBorder(5, 10, 5, 10));
35
36 searchField = new JTextField(25);
37 searchField.setFont(new Font("Segoe UI", Font.PLAIN, 13));
38 searchField.setPreferredSize(new Dimension(200, 28));
39
40 JButton searchBtn = new JButton("Cari");
41 searchBtn.setBackground(new Color(36, 52, 92));
42 searchBtn.setForeground(Color.WHITE);
43 searchBtn.setFocusPainted(false);
44 searchBtn.setFont(new Font("Segoe UI", Font.BOLD, 12));
45 searchBtn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
46 searchBtn.setBorder(BorderFactory.createEmptyBorder(6, 14, 6, 14));
47 searchBtn.setOpaque(true);
48 searchBtn.addActionListener(e -> showBooksGrid());
49
50 topPanel.add(searchField);
51 topPanel.add(searchBtn);
52 add(topPanel, BorderLayout.NORTH);
53
54 contentPanel = new JPanel(new BorderLayout());
55 contentPanel.setBorder(new EmptyBorder(20, 20, 20, 20));
56 contentPanel.setBackground(new Color(245, 245, 245));
57
58 booksGridPanel = new JPanel(new GridLayout(0, 2, 20, 20));
59 booksGridPanel.setBackground(new Color(245, 245, 245));
60
61 bookScrollPane = new JScrollPane(booksGridPanel);
62 bookScrollPane.getVerticalScrollBar().setUnitIncrement(16);
63 bookScrollPane.setViewport().setBackground(new Color(245, 245, 245));
64
65 contentPanel.add(bookScrollPane, BorderLayout.CENTER);
66 add(contentPanel, BorderLayout.CENTER);
67
68 riwayatPanel = new RiwayatPeminjamanPanel(user);
69
70 setActiveButton(homeBtn);
71 showBooksGrid();
72 setVisible(true);
73 }
74 }
```

## User Dashboard (3)

```
1 private void showBooksGrid() {
2 contentPanel.removeAll();
3 booksGridPanel.removeAll();
4
5 String keyword = searchField.getText().trim().toLowerCase();
6 boolean adaBuku = false;
7
8 for (Book b : BookController.getAllBooks()) {
9 if (b.getStok() > 0 && (keyword.isEmpty()
10 || b.getTitle().toLowerCase().contains(keyword)
11 || b.getAuthor().toLowerCase().contains(keyword))) {
12 booksGridPanel.add(new BookCard(b, user.getIdUser(), this));
13 adaBuku = true;
14 }
15 }
16
17 if (!adaBuku) {
18 booksGridPanel.setLayout(new BorderLayout());
19 JLabel kosongLabel = new JLabel("⚠️ Ups! Kami tidak menemukan buku yang cocok.");
20 kosongLabel.setHorizontalAlignment(SwingConstants.CENTER);
21 kosongLabel.setFont(new Font("Segoe UI", Font.ITALIC, 16));
22 booksGridPanel.add(kosongLabel, BorderLayout.CENTER);
23 } else {
24 booksGridPanel.setLayout(new GridLayout(0, 2, 20, 20));
25 }
26
27 contentPanel.add(bookScrollPane, BorderLayout.CENTER);
28 contentPanel.revalidate();
29 contentPanel.repaint();
30
31 SwingUtilities.invokeLater(() -> bookScrollPane.getVerticalScrollBar().setValue(0));
32 }
33
34 private void showRiwayatPanel() {
35 contentPanel.removeAll();
36
37 JScrollPane scrollPane = new JScrollPane(riwayatPanel);
38 scrollPane.getVerticalScrollBar().setUnitIncrement(16);
39 scrollPane.setViewport().setBackground(new Color(245, 245, 245));
40
41 contentPanel.add(scrollPane, BorderLayout.CENTER);
42 contentPanel.revalidate();
43 contentPanel.repaint();
44 }
45
46 public void refreshBooks() {
47 if (activeButton != null && activeButton.getText().contains("Beranda")) {
48 showBooksGrid();
49 }
50 }
```

## User Dashboard (4)

```
 1 private void styleSidebarButton(JButton button) {
 2 Color normalColor = new Color(36, 52, 92);
 3 Color hoverColor = new Color(0, 153, 0);
 4 Color textColor = Color.WHITE;
 5
 6 button.setFocusPainted(false);
 7 button.setContentAreaFilled(true);
 8 button.setOpaque(true);
 9 button.setBackground(normalColor);
10 button.setForeground(textColor);
11 button.setFont(new Font("Segoe UI", Font.BOLD, 14));
12 button.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 10));
13 button.setAlignmentX(Component.CENTER_ALIGNMENT);
14 button.setMaximumSize(new Dimension(180, 40));
15 button.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
16
17 button.addMouseListener(new java.awt.event.MouseAdapter() {
18 public void mouseEntered(java.awt.event.MouseEvent evt) {
19 if (button != activeButton) {
20 button.setBackground(hoverColor);
21 }
22 }
23
24 public void mouseExited(java.awt.event.MouseEvent evt) {
25 if (button != activeButton) {
26 button.setBackground(normalColor);
27 }
28 }
29 });
30 }
31
32 private void setActiveButton(JButton button) {
33 Color normalColor = new Color(36, 52, 92);
34 Color activeColor = new Color(0, 153, 0);
35
36 if (activeButton != null) {
37 activeButton.setBackground(normalColor);
38 }
39 activeButton = button;
40 activeButton.setBackground(activeColor);
41 }
42 }
```

Penjelasan Class Dashboard User :

## 1. Package dan Import (Baris 1–27)

Class UserDashboard berada di dalam package view.

Pada bagian import, UserDashboard mengimpor berbagai class dari:

- `java.awt`: Digunakan untuk pengaturan tampilan GUI seperti `BorderLayout`, `Color`, `Font`, `Component`, `Graphics2D`, `GradientPaint`, dan lainnya.
- `javax.swing`: Digunakan untuk membangun antarmuka grafis (GUI) seperti `JFrame`, `JPanel`, `JButton`, `JLabel`, `JTextField`, `JScrollPane`, `JOptionPane`, `Box`, `BoxLayout`, `BorderFactory`, `EmptyBorder`, dan lain-lain.
- `controller.BookController`: Digunakan untuk mengakses dan mengelola data buku dari backend.
- `model.Book` dan `model.User`: Digunakan sebagai representasi objek buku dan pengguna dalam sistem.

## 2. Deklarasi Class dan Atribut (Baris 29–38)

Deklarasi class: `public class UserDashboard extends JFrame`

Class ini merupakan turunan (subclass) dari `JFrame`, yang berarti merupakan jendela utama untuk tampilan pengguna biasa.

Atribut-atribut utama yang digunakan antara lain:

- `User user`: Data pengguna yang sedang login.
- `JPanel contentPanel, booksGridPanel`: Panel utama dan panel daftar buku.
- `JScrollPane bookScrollPane`: Scroll view untuk menampilkan daftar buku yang panjang.
- `JPanel riwayatPanel`: Panel khusus untuk menampilkan riwayat peminjaman buku.
- `JTextField searchField`: Kolom input pencarian buku.
- `JButton activeButton`: Tombol sidebar yang sedang aktif (terpilih).

### 3. Konstruktor UserDashboard (Baris 40–73)

```
public UserDashboard(User user)
```

Merupakan konstruktor utama untuk inisialisasi antarmuka pengguna setelah login berhasil. Fungsinya antara lain:

- Menyimpan data pengguna yang login ke dalam atribut user.
- Menetapkan properti JFrame: ukuran, lokasi, judul, layout, dan pengaturan close operation.
- Membuat sidebar dengan efek latar gradasi dengan override method paintComponent(Graphics g).
- Menambahkan label “Booktopia” ke sidebar.
- Membuat tiga tombol utama di sidebar:
  - homeBtn: Untuk menampilkan daftar buku.
  - pinjamanBtn: Untuk menampilkan riwayat peminjaman.
  - logoutBtn: Untuk logout dan kembali ke halaman login.
- Mendaftarkan ActionListener pada setiap tombol sidebar untuk menangani klik pengguna.
- Menambahkan kolom pencarian dan tombol “Cari” di bagian atas.
- Menyusun panel utama (contentPanel) dengan BorderLayout.
- Menyusun dan menampilkan daftar buku secara default saat pertama kali class ini dijalankan (homeBtn aktif).

### 4. Method showBooksGrid()

```
private void showBooksGrid()
```

Method ini bertanggung jawab untuk menampilkan daftar buku berdasarkan hasil pencarian pengguna.

- Menghapus semua komponen dari contentPanel dan booksGridPanel agar dapat memuat ulang data.

- Mengambil keyword pencarian dari searchField dan mengubahnya menjadi lowercase.
- Melakukan iterasi terhadap semua data buku dari BookController.getAllBooks().
- Mengecek apakah stok buku masih tersedia dan apakah keyword cocok dengan judul atau penulis. Jika ya, maka ditambahkan ke booksGridPanel dalam bentuk objek BookCard.
- Jika tidak ada buku yang cocok, maka ditampilkan pesan “Tidak ada buku ditemukan.”
- Menambahkan booksGridPanel ke dalam JScrollPane dan kemudian ditampilkan ke dalam contentPanel.

Materi OOP: Method, Object, Enkapsulasi, Event-Driven (kolom pencarian)

#### 5. Method showRiwayatPanel()

```
private void showRiwayatPanel()
```

Digunakan untuk menampilkan panel riwayat peminjaman buku oleh pengguna yang sedang login.

- Menghapus seluruh isi contentPanel agar siap menampilkan panel baru.
- Membungkus riwayatPanel ke dalam JScrollPane untuk mendukung tampilan yang dapat digulir.
- Menambahkan JScrollPane ke contentPanel dengan penempatan di bagian tengah (CENTER).
- Menyegarkan tampilan dengan revalidate() dan repaint() agar perubahan tampak.

Materi OOP: Method, Event-Driven

#### 6. Method refreshBooks()

```
public void refreshBooks()
```

Digunakan untuk memperbarui daftar buku, biasanya dipanggil dari luar class setelah peminjaman atau pengembalian buku.

- Mengecek apakah tombol aktif saat ini adalah “Beranda”. Jika iya, maka showBooksGrid() dipanggil untuk memuat ulang daftar buku.

## Materi OOP: Method, Enkapsulasi

### 7. Method styleSidebarButton(JButton button)

```
private void styleSidebarButton(JButton button)
```

Digunakan untuk mengatur tampilan visual tombol sidebar agar konsisten dan interaktif.

- Mengatur warna latar belakang, jenis font, ukuran, padding, dan cursor.
- Menambahkan MouseListener untuk memberikan efek hover: tombol berubah warna saat diarahkan dan kembali saat mouse keluar.
- Tombol aktif memiliki warna khusus (biasanya biru keunguan).

## Materi OOP: Method, Polymorphism (anonymous inner class MouseAdapter), Enkapsulasi

### 8. Method setActiveButton(JButton button)

```
private void setActiveButton(JButton button)
```

Digunakan untuk menetapkan tombol sidebar mana yang sedang aktif dan menyesuaikan tampilannya.

- Jika ada tombol aktif sebelumnya, maka warnanya dikembalikan ke warna default.
- Menyimpan tombol yang baru diklik sebagai activeButton.
- Mengubah tampilan tombol aktif agar menonjol (warna berbeda).

## Materi OOP: Method, Enkapsulasi

### Kesimpulan Fungsi Class UserDashboard

Class UserDashboard memiliki tanggung jawab utama sebagai tampilan utama bagi pengguna biasa dalam sistem perpustakaan digital. Dengan memanfaatkan komponen GUI berbasis Java Swing, class ini memungkinkan pengguna untuk:

- Melihat dan mencari daftar buku.
- Melihat riwayat peminjaman buku.
- Logout dari aplikasi.

Class ini mengimplementasikan prinsip-prinsip dasar OOP secara konsisten, yaitu:

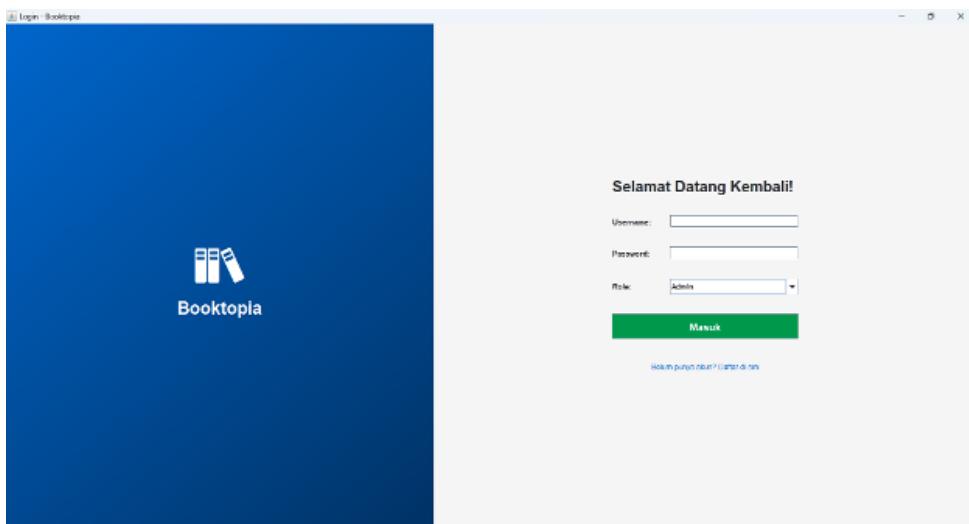
- Inheritance: mewarisi fitur dari JFrame.
- Encapsulation: membatasi akses atribut melalui konstruktor dan method internal.
- Event-Driven: menggunakan action listener untuk menangani aksi pengguna secara interaktif.

## 2.5 Pembahasan Output Program

### 1. Koneksi Database MySQL

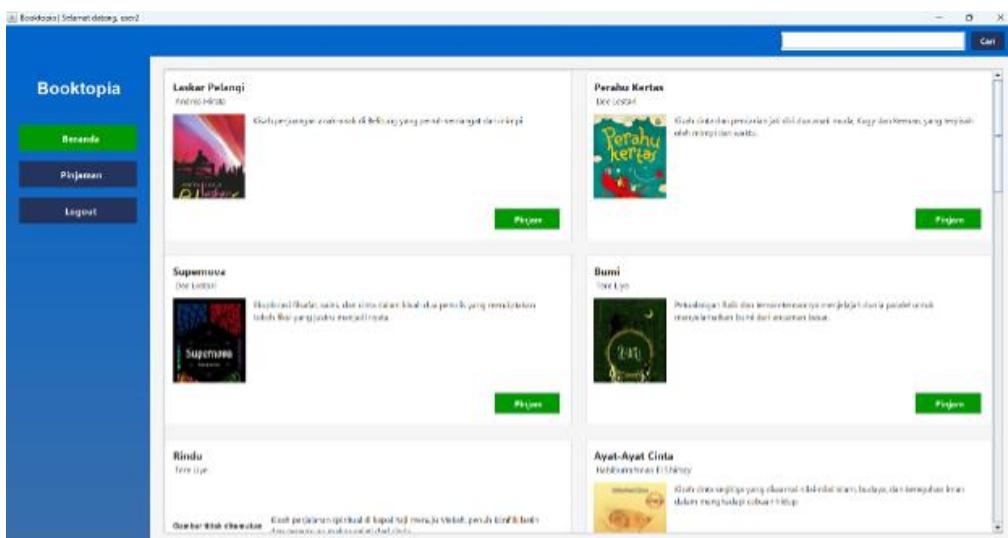
Sistem terhubung dengan database MySQL melalui koneksi JDBC. Seluruh data (user, buku, peminjaman) tersimpan dalam database dan dikelola melalui query SQL. Kelas utilitas DBConnection digunakan untuk mengatur koneksi secara modular dan terpisah dari logika utama aplikasi.

### 2. Login

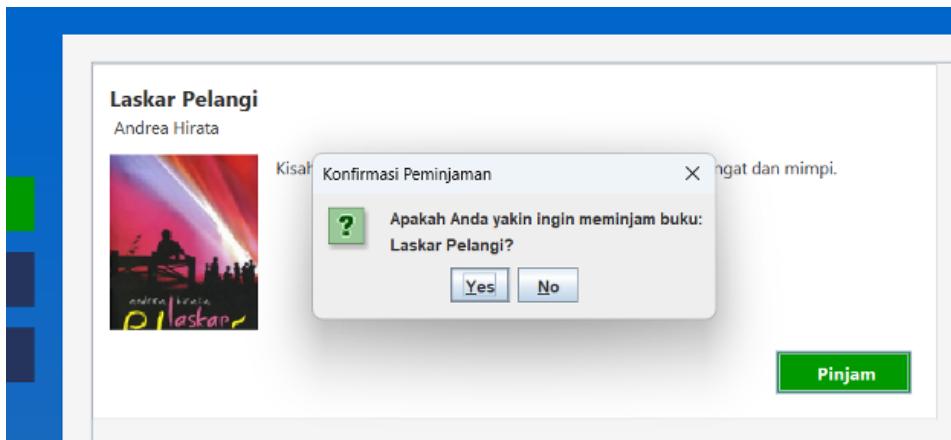


### 3. User

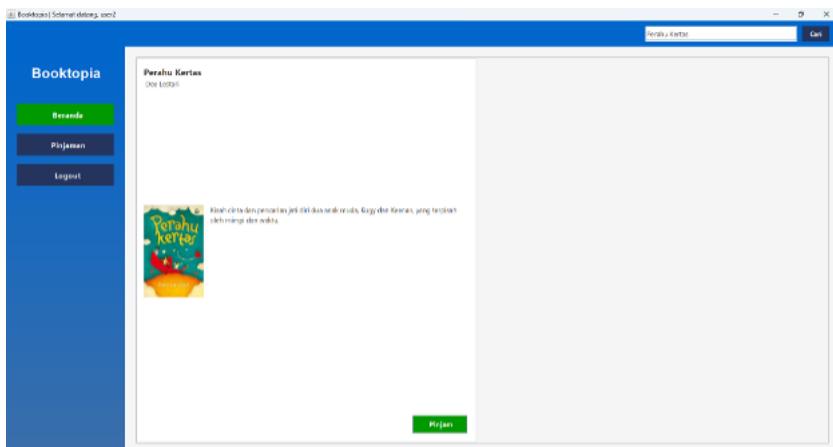
#### a. Dashboard



b. Pinjam Buku



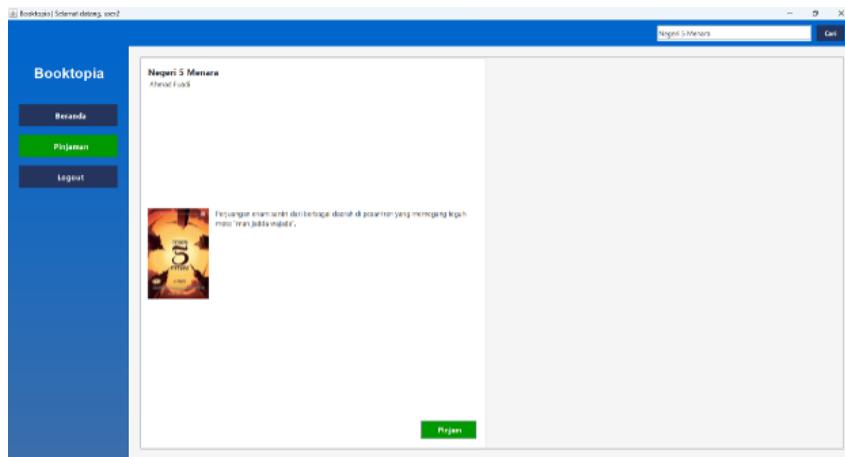
c. Cari Buku



d. Riwayat peminjaman Buku

| Riwayat Peminjaman Anda |         |                       |                    |                 |              |       |
|-------------------------|---------|-----------------------|--------------------|-----------------|--------------|-------|
| ID                      | ID Buku | Nama Buku             | Tanggal Peminjaman | Tanggal Kembali | Status       | Denda |
| 1                       | BD001   | Angela S. Mearns      | 2023-04-10         | 2023-05-10      | Dikembalikan | 0.00  |
| 2                       | BD005   | Khairy                | 2023-05-20         | 2023-06-02      | Dikembalikan | 0.00  |
| 26                      | BD015   | Ketika Cinta Berbalik | 2023-06-01         | 2023-06-10      | Akhir        | 0.00  |
| 29                      | BD015   | Ketika Cinta Berbalik | 2023-06-01         | 2023-06-10      | Akhir        | 0.00  |

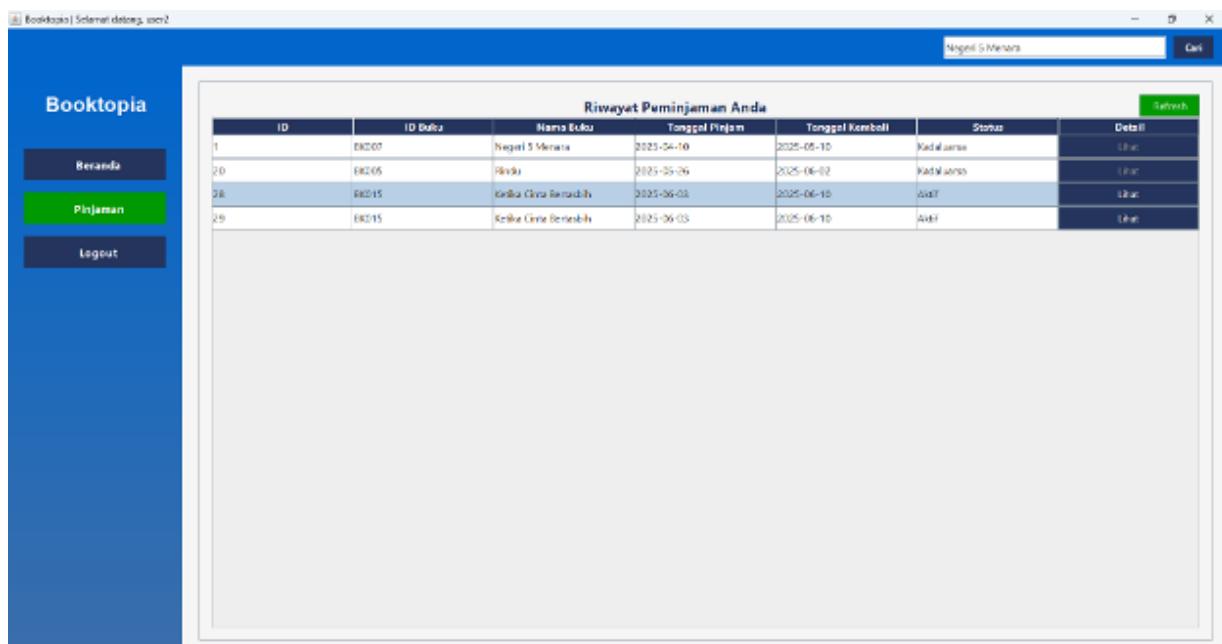
e. Pencarian buku di riwayat peminjaman

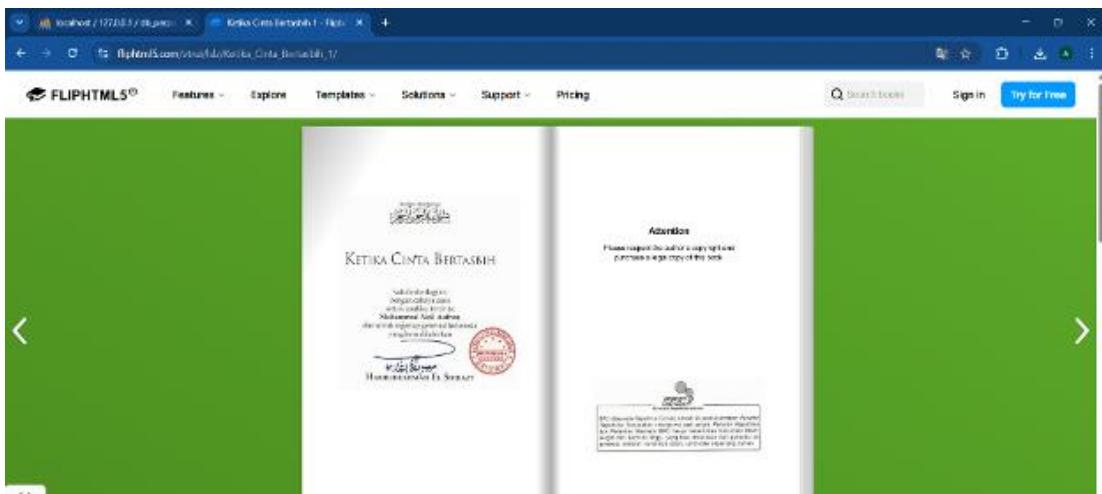


f. Tampilan jika buku sudah lewat batas waktu peminjaman

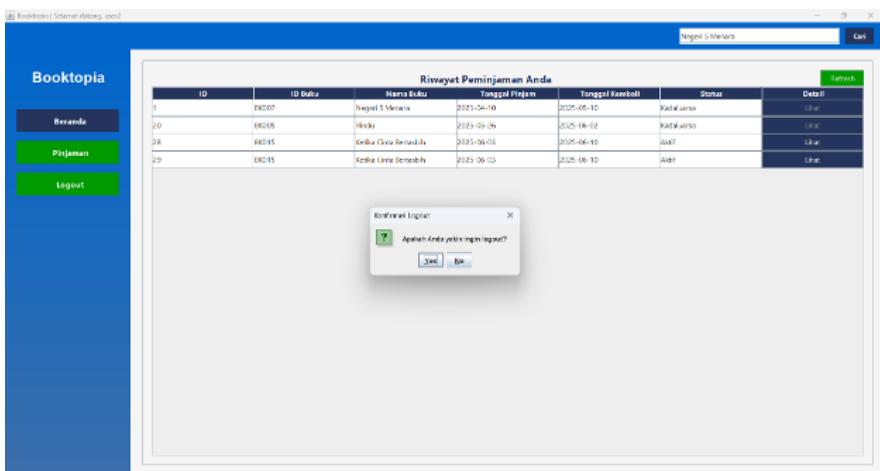
| Riwayat Peminjaman Anda |         |                       |                |                 |            |                       | Refresh |
|-------------------------|---------|-----------------------|----------------|-----------------|------------|-----------------------|---------|
| ID                      | ID Buku | Nama Buku             | Tanggal Pinjam | Tanggal Kembali | Status     | Detail                |         |
| 1                       | BK007   | Negeri 5 Menara       | 2025-04-10     | 2025-05-10      | Kadaluarsa | <a href="#">Lihat</a> |         |
| 20                      | BK005   | Rindu                 | 2025-05-26     | 2025-06-02      | Kadaluarsa | <a href="#">Lihat</a> |         |
| 28                      | BK015   | Kelita Cinta Bertabih | 2025-06-03     | 2025-06-10      | Aleff      | <a href="#">Lihat</a> |         |
| 29                      | BK015   | Kelita Cinta Bertabih | 2025-06-03     | 2025-06-10      | Aleff      | <a href="#">Lihat</a> |         |

g. Tampilan jika buku masih bisa dibaca dalam waktu peminjaman

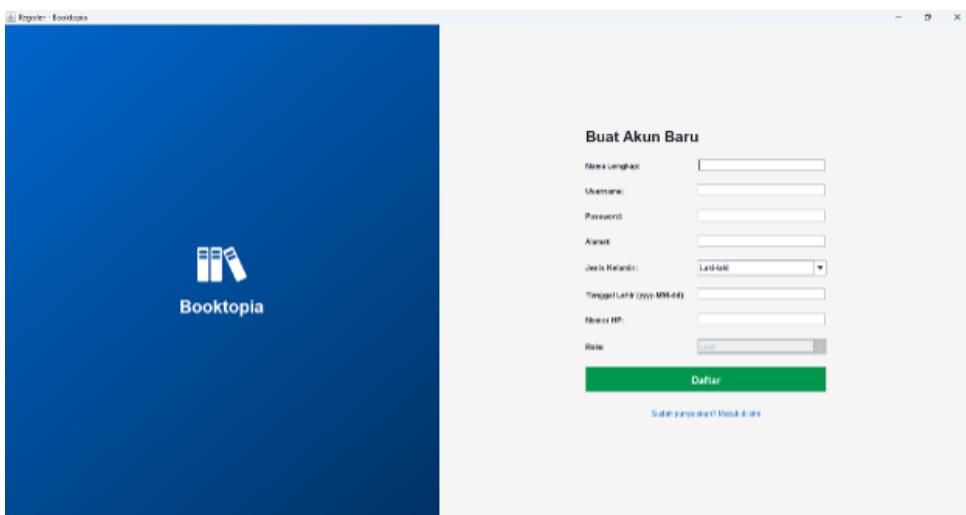


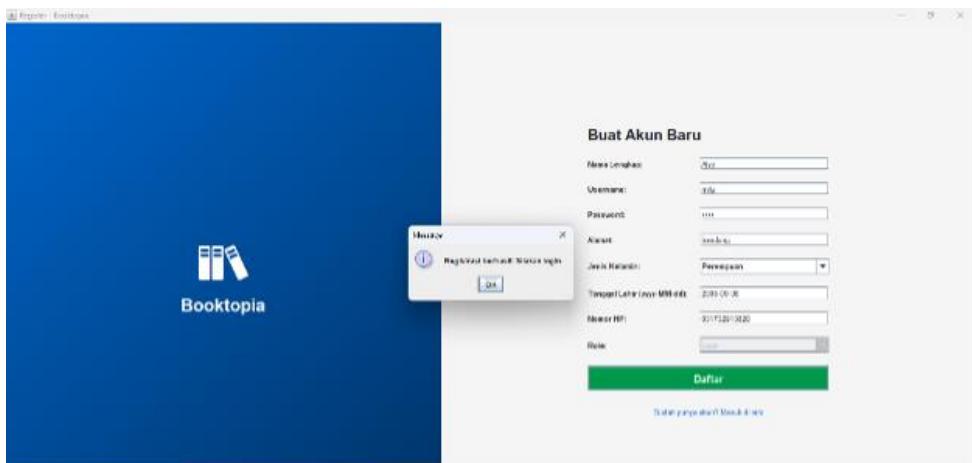


#### h. Logout



### 4. Register (akun baru)





## 5. Admin

### a. Dashboard

**Daftar Buku**

| ID Buku | Judul                  | Pengarang                | Penerbit | Tahun Terbit | Stock |
|---------|------------------------|--------------------------|----------|--------------|-------|
| BK001   | Laskar Pelangi         | Andrea Hirata            | PEN1001  | 2005         | 5     |
| BK002   | Pecatu Kertas          | Irene Lensten            | PEN1002  | 2009         | 3     |
| BK003   | Kapten Keling          | Irene Lensten            | PEN1003  | 2011         | 4     |
| BK004   | Ratu                   | Tessa Tyte               | PEN1002  | 2014         | 6     |
| BK005   | Ratu                   | Tessa Tyte               | PEN1002  | 2015         | 1     |
| BK006   | Ayah-Ayah Cinta        | Habiburrahman El Shafiey | PEN1002  | 2004         | 7     |
| BK007   | Kangen & Memoria       | Ahmad Prod               | PEN1004  | 2006         | 3     |
| BK008   | Pulang                 | Leila S. Chudat          | PEN1005  | 2012         | 2     |
| BK009   | Dilan 1990             | Pidi Baiq                | PEN1004  | 2014         | 9     |
| BK010   | Ruang Kaca untuk Tahan | Agnis Djoko              | PEN1001  | 2008         | 4     |

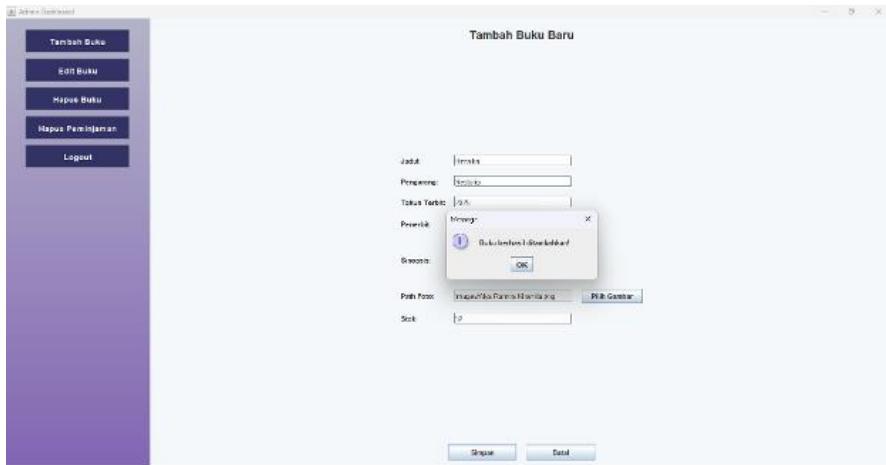
**Daftar Peminjaman**

| ID Peminjaman | User ID | Book ID | Tanggal Pinjam | Jatuh Tempo | Status       |
|---------------|---------|---------|----------------|-------------|--------------|
| 1             | USR005  | BK007   | 2025-04-10     | 2025-05-10  | Dipinjam     |
| 2             | USR010  | BK009   | 2025-04-15     | 2025-05-15  | Dikembalikan |
| 3             | USR012  | BK015   | 2025-04-12     | 2025-05-12  | Dipinjam     |
| 4             | USR016  | BK001   | 2025-04-08     | 2025-05-08  | Dipinjam     |
| 5             | USR010  | BK004   | 2025-04-05     | 2025-05-05  | Dikembalikan |
| 6             | USR012  | BK006   | 2025-04-00     | 2025-05-00  | Dikembalikan |
| 7             | USR006  | BK011   | 2025-04-18     | 2025-05-18  | Dipinjam     |

### b. Tambah Buku

**Tambah Buku Baru**

Judul: \_\_\_\_\_  
 Pengarang: \_\_\_\_\_  
 Tahun Terbit: \_\_\_\_\_  
 Penerbit: **PEN011 - Gramedia**  
 Birokot: \_\_\_\_\_   
 Stock: \_\_\_\_\_



### c. Edit Buku

The screenshot shows a multi-tabbed application interface for managing books and loans.

- Top Tab:** Displays a list of books (Buku) with columns: ID Buku, Judul, Pengarang, Penerbit, Tahun Terbit, and Stok. One book, 'Hans' by 'Hans' (ID BK016), is selected for editing.
- Second Tab:** Displays a list of loan records (Peminjaman) with columns: ID Peminjaman, User ID, and Stok. The same book 'Hans' is selected for editing.
- Third Tab (Active):** An edit dialog for the book 'Hans'. It shows the original data: Judul (Hans), Pengarang (Hans), Tahun Terbit (2003), Penerbit (PEN001 - Gramedia), and Stok (12). The user changes the Stok to 10. A confirmation message box appears: 'Buku berhasil diperbarui!' (Book successfully updated!).
- Bottom Tab:** Displays a list of loans (Peminjaman) with columns: User ID, Book ID, Tanggal Pinjam (Loan Date), Jatuh Tempo (Due Date), and Status. The same loan record for 'Hans' is selected.

#### d. Hapus Buku

Admin Dashboard

|                                  |
|----------------------------------|
| <a href="#">Tambah Buku</a>      |
| <a href="#">Edit Buku</a>        |
| <a href="#">Hapus Buku</a>       |
| <a href="#">Hapus Peminjaman</a> |
| <a href="#">Logout</a>           |

Data Buku

| ID Buku | Judul                         | Pengarang              | Penerbit | Tahun Terbit | Stok |
|---------|-------------------------------|------------------------|----------|--------------|------|
| BK012   | Redovanso                     | Dewi Lestari           | PEN003   | 2008         | 5    |
| BK013   | Seputing Hat yang Baik        | Ismi Syafei            | PEN005   | 2017         | 3    |
| BK014   | 9 Bantuan 10 Kuzanne          | Iwan Setyawan          | PEN001   | 2011         | 2    |
| BK015   | Keluarga Cinta Berpasca       | Mul Hermawan El Sholah | PEN001   | 2007         | 3    |
| BK016   | Garis Waktu                   | Fiersa Besari          | PEN004   | 2016         | 7    |
| BK017   | Langit Merah di Jembatan A... | Remy Sylado            | PEN003   | 2009         | 3    |
| BK018   | Amba                          | Laksmi Pamuntjak       | PEN002   | 2012         | 9    |
| BK019   | Kukila                        | Konstantin Fomic       | X        | 2015         | 6    |
| BK020   | Hana                          |                        | ?        | 2010         | 12   |
| BK021   | Himsika                       |                        | N002     | 2019         | 12   |

Yakin ingin menghapus data ini?

[Yes](#) [No](#)

Data Peminjaman

| ID Peminjaman | User ID | Book ID | Tanggal Pinjam | Jatuh Tempo | Status       |
|---------------|---------|---------|----------------|-------------|--------------|
| 1             | USR003  | BK007   | 2025-04-10     | 2025-05-10  | Dipinjam     |
| 2             | USR008  | BK007   | 2025-04-15     | 2025-05-15  | Dikembalikan |
| 3             | USR002  | BK015   | 2025-04-12     | 2025-05-12  | Dipinjam     |
| 4             | USR001  | BK001   | 2025-04-08     | 2025-05-08  | Dipinjam     |
| 5             | USR003  | BK004   | 2025-04-05     | 2025-05-05  | Dikembalikan |
| 6             | USR002  | BK001   | 2025-04-00     | 2025-05-00  | Dikembalikan |
| 7             | USR003  | BK011   | 2025-04-18     | 2025-05-18  | Dipinjam     |

|                               |                                                               |        |
|-------------------------------|---------------------------------------------------------------|--------|
| Garis Waktu                   | Fiersa Besari                                                 | PEN004 |
| Langit Merah di Jembatan A... | Remy Sylado                                                   | PEN003 |
| Amba                          | Laksmi Pamuntjak                                              | PEN002 |
| Kukila                        | Message                                                       |        |
| Hana                          | <span style="color: #0070C0;">i</span> Buku berhasil dihapus. |        |
| Himsika                       | <span style="color: #0070C0;">OK</span>                       |        |

| User ID | Book ID | Tanggal Pinjam |
|---------|---------|----------------|
| USR003  | BK007   | 2025-04-10     |
| USR008  | BK002   | 2025-04-15     |
| USR012  | BK015   | 2025-04-12     |
| USR006  | BK001   | 2025-04-08     |

### e. Hapus riwayat peminjaman user

Screenshot of the Admin Dashboard showing the 'Hapus Peminjaman' (Delete Loan History) feature.

The interface includes a sidebar with navigation links: Tambah Buku, Edit Buku, Hapus Buku, Hapus Peminjaman, and Logout.

Two tables are displayed:

- Data Buku:**

| ID Buku | Judul                         | Pengarang                             | Penerbit | Tahun Terbit | Stok |
|---------|-------------------------------|---------------------------------------|----------|--------------|------|
| BK011   | Hujan                         | Tere Liye                             | PEN003   | 2015         | 8    |
| BK012   | Ridauweno                     | Dava Lebari                           | PEN001   | 2003         | 8    |
| BK013   | Repotong Hati yang Baru       | Tere Liye                             | PEN005   | 2012         | 3    |
| BK014   | 9 Summers 10 Autumns          | Iwan Setyawan                         | PEN001   | 2013         | 2    |
| BK015   | Ketika Cinta Bertabisih       | Habiburrahman El Shirazy              | PEN001   | 2007         | 5    |
| BK016   | Garis Waktu                   | Fiersa Besari                         | PEN004   | 2016         | 7    |
| BK017   | Langit Merah di Jembatan A... | Remy Sylado                           | PEN003   | 2009         | 2    |
| BK018   | Amba                          | Konfirmasi Hapus                      |          | N002         | 2012 |
| BK019   | Kukila                        | Yakin ingin menghapus peminjaman ini? |          | N004         | 2015 |
| BK020   | Himsika                       | Yakin ingin menghapus peminjaman ini? |          | N002         | 2025 |
- Data Peminjaman:**

| ID Peminjaman | User ID | Book ID | Tanggal Pinjam | Jatuh Tempo | Status       |
|---------------|---------|---------|----------------|-------------|--------------|
| 1             | USR003  | BK007   | 2025-04-10     | 2025-05-10  | Dipinjam     |
| 2             | USR008  | BK002   | 2025-04-15     | 2025-05-15  | Dikembalikan |
| 3             | USR012  | BK015   | 2025-04-12     | 2025-05-12  | Dipinjam     |
| 4             | USR006  | BK001   | 2025-04-08     | 2025-05-08  | Dipinjam     |
| 5             | USR010  | BK004   | 2025-04-25     | 2025-05-25  | Dikembalikan |
| 6             | USR002  | BK008   | 2025-04-30     | 2025-05-30  | Dikembalikan |
| 7             | USR005  | BK011   | 2025-04-18     | 2025-05-18  | Dipinjam     |

A confirmation dialog box is shown for the loan history of book BK018 (Amba), asking "Data peminjaman berhasil dihapus." (Loan history successfully deleted). Buttons: OK.

Below the tables, a message box displays the deleted data: Amba, N002, 2012.

**minjaman**

| ID Peminjaman | User ID | Book ID | Tanggal Pinjam | Jatuh Tempo |
|---------------|---------|---------|----------------|-------------|
| USR003        | BK007   |         | 2025-04-10     | 2025-05-10  |
| USR008        | BK002   |         | 2025-04-15     | 2025-05-15  |

### f. Logout

Screenshot of the Admin Dashboard showing the 'Logout' feature.

The interface includes a sidebar with navigation links: Tambah Buku, Edit Buku, Hapus Buku, Hapus Peminjaman, and Logout.

Two tables are displayed:

- Data Buku:**

| ID Buku | Judul                         | Pengarang                | Penerbit | Tahun Terbit |
|---------|-------------------------------|--------------------------|----------|--------------|
| BK011   | Ketika Cinta Bertabisih       | Habiburrahman El Shirazy | PEN001   | 2007         |
| BK012   | Garis Waktu                   | Fiersa Besari            | PEN004   | 2016         |
| BK013   | Langit Merah di Jembatan A... | Remy Sylado              | PEN003   | 2002         |
| BK014   | Amba                          | Konfirmasi Logout        |          | N002         |
| BK015   | Kukila                        | Yakin ingin logout?      |          | N004         |
| BK016   | Himsika                       | Yakin ingin logout?      |          | N002         |
- Data Peminjaman:**

| ID Peminjaman | User ID | Book ID | Tanggal Pinjam | Jatuh Tempo |
|---------------|---------|---------|----------------|-------------|
| USR003        | BK007   |         | 2025-04-10     | 2025-05-10  |
| USR008        | BK002   |         | 2025-04-15     | 2025-05-15  |

A confirmation dialog box is shown for the logout, asking "Yakin ingin logout?" (Are you sure you want to log out?). Buttons: Yes, No.

## **BAB III**

### **KESIMPULAN**

#### 3.1 Kesimpulan

Pembangunan Sistem Perpustakaan Digital berbasis Java GUI dan MySQL ini berhasil memenuhi tujuan utama, yaitu menciptakan sebuah aplikasi yang mampu menangani pengelolaan koleksi eBook, manajemen pengguna, serta transaksi peminjaman secara efisien dan terstruktur. Sistem ini dirancang dengan pendekatan Pemrograman Berorientasi Objek (OOP), yang terbukti memberikan struktur kode yang modular, fleksibel, dan mudah untuk dikembangkan lebih lanjut.

Dengan penerapan prinsip-prinsip OOP seperti enkapsulasi, pewarisan, polimorfisme, serta exception handling, sistem dapat menangani berbagai skenario dan kesalahan dengan baik. Pemisahan hak akses antara admin dan pengguna umum juga membantu meningkatkan keamanan serta memudahkan pengelolaan peran dalam sistem.

Selain itu, penggunaan antarmuka Java Swing GUI memberikan pengalaman pengguna yang interaktif dan intuitif, bahkan untuk pengguna awam. Fitur-fitur seperti pencarian buku, peminjaman otomatis, validasi masa pinjam, dan riwayat peminjaman memperkaya fungsi sistem secara keseluruhan.

Dengan hadirnya sistem ini, diharapkan proses digitalisasi perpustakaan dapat mendorong peningkatan kualitas literasi, mempermudah akses terhadap informasi, serta menjadi contoh nyata penerapan teknologi dalam dunia pendidikan.