Aplikasi Manajemen Perpustakaan

menggunakan Java Swing dan MySQL.

Table of contents

TEAM STRUCTURE CHART

Abdullah Azzam Rabbani

Achmar Ichsan Anthony

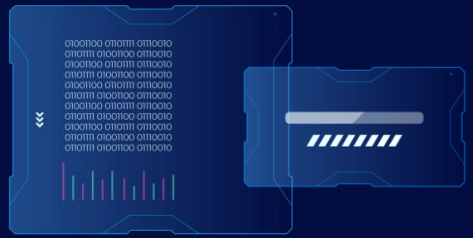
Fauzan Iqbal

Junithan Sehti Aditia

Yudistira Agung Kurnia

La Ode Achmed Sayyed Purnomo

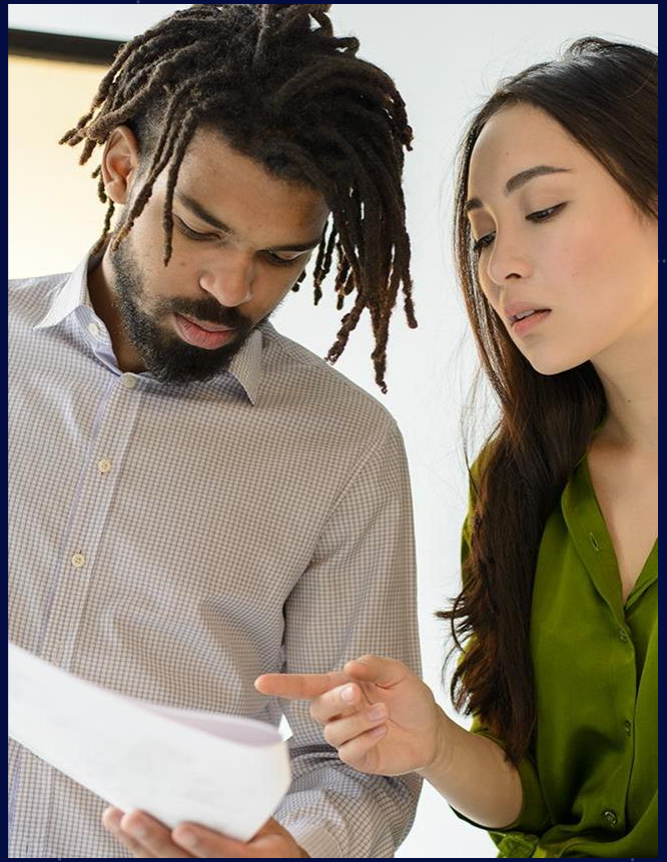
01 Pendahuluan



Latar Belakang

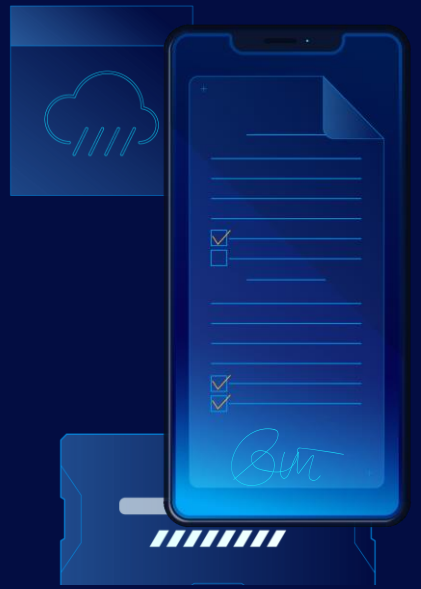
Perpustakaan sebagai pusat sumber informasi masih mengandalkan sistem pencatatan manual (buku fisik/Excel), yang menimbulkan masalah:

- Human Error: Kesalahan input data (misalnya stok tidak terupdate).
- Inefisiensi: Proses peminjaman memakan waktu 5-10 menit per transaksi.
- Pelaporan Terbatas: Kesulitan menghasilkan laporan statistik cepat.
- Keamanan Data: Data rentan hilang/rusak karena tidak terpusat.



Tujuan Pengembangan Sistem

- Membangun aplikasi desktop berbasis Java Swing & MySQL untuk mengotomasi peminjaman buku.
- Memastikan konsistensi stok real-time selama transaksi.
- Menyediakan dashboard analisis data dengan visualisasi grafik interaktif (menggunakan JFreeChart).
- Menerapkan pola MVC (Model-View-Controller) untuk memisahkan logika bisnis, tampilan, dan basis data.

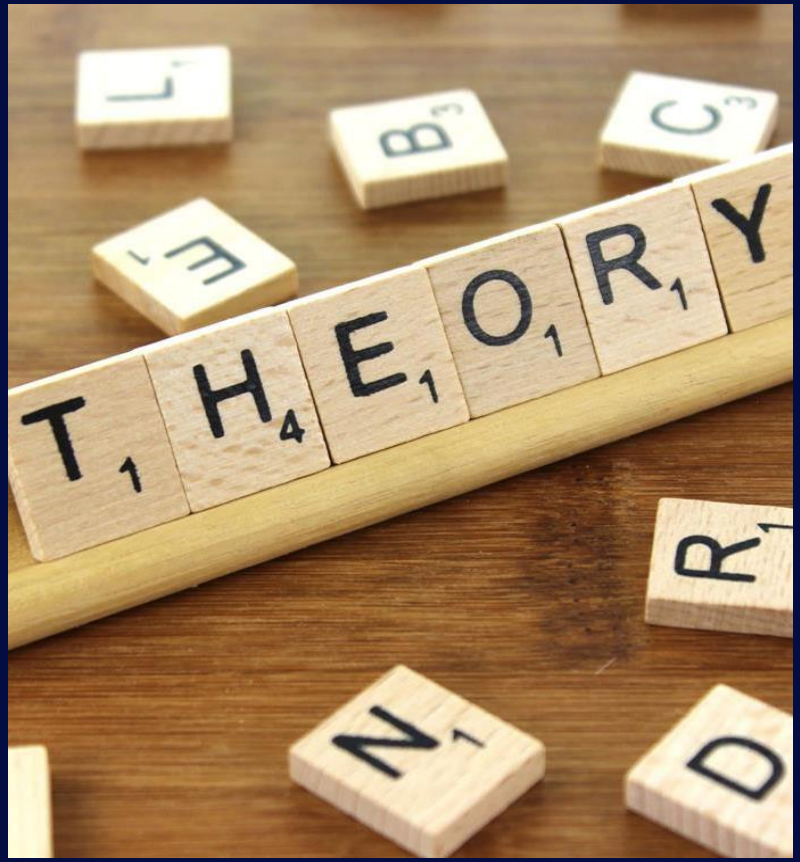


Rumusan Masalah

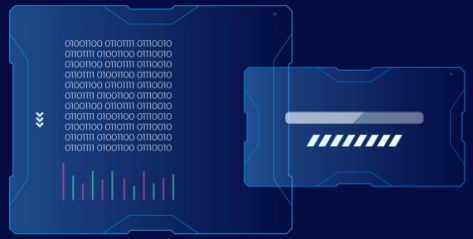
1. Bagaimana membangun sistem autentikasi dengan pembagian hak akses admin dan user?
2. Bagaimana memastikan konsistensi stok buku saat transaksi peminjaman?
3. Bagaimana menyajikan laporan statistik peminjaman secara visual dan interaktif?
4. Bagaimana menyimpan data secara terpusat di database dan mengaksesnya
5. Bagaimana membuat sistem kelola buku yang simple namun mendekati lengkap secara fitur

Batasan Masalah

1. Sistem hanya untuk peminjaman buku fisik (non-elektronik).
2. Masih menerapkan synchronous pada kodingan logic nya
3. Tidak bisa digunakan di multiple device karena belum ada API
4. Fitur ekspor laporan PDF belum diimplementasikan



02
Teori



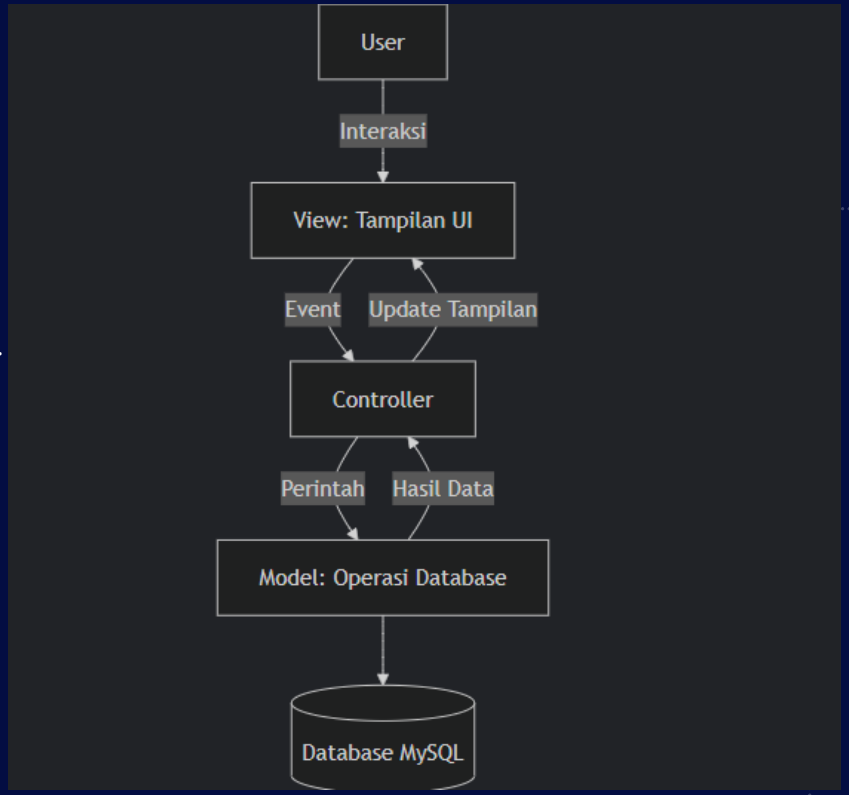
Konsep Dasar

MVC (Model-View-Controller):

- Model: Logika database (contoh: UserDao.java).
- View: Tampilan antarmuka (contoh: LoginFrame.java).
- Controller: Mediator Model-View (contoh: AuthController.java).

Basis Data Relasional (MySQL):

- Skema terstruktur
- Menggunakan normalisasi tahap 3 3NF
- Optimasi Query



Teknologi Pendukung

Java Swing

Framework GUI desktop.

MySQL

Database untuk menyimpan data buku, user, dan transaksi.

JFreeChart

Modul untuk membantu mevisualisasikan data

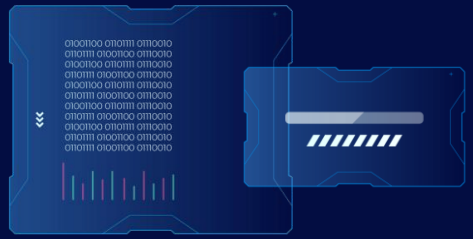
JDBC

Penghubung antara Java dan MySQL agar aplikasi bisa membaca/menulis data.

DOCKER

Untuk me running container mysql database

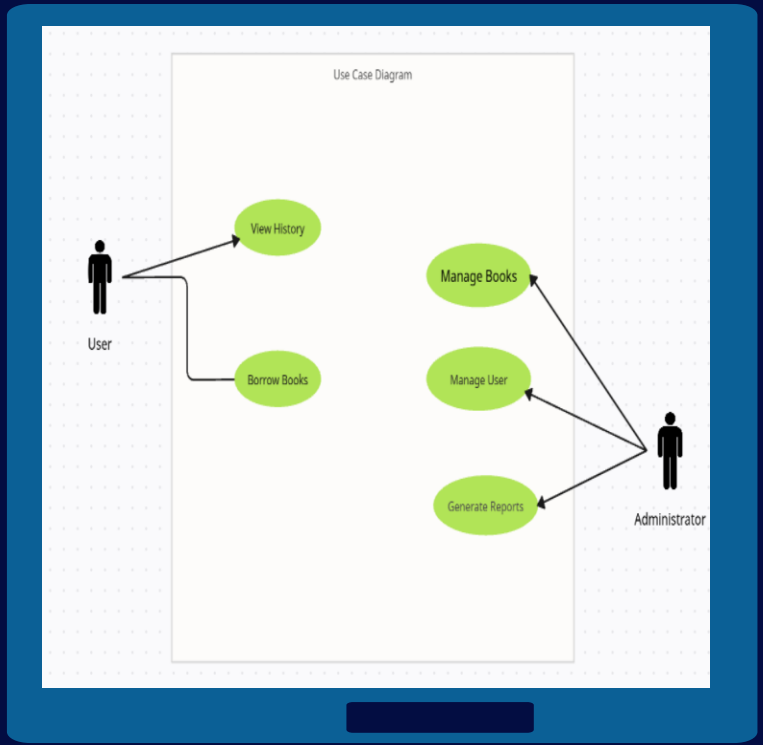
03
Proses Bisnis



Alur Bisnis User dan Admin

Melihat History Peminjaman

Meminjam Buku



Mengatur Buku

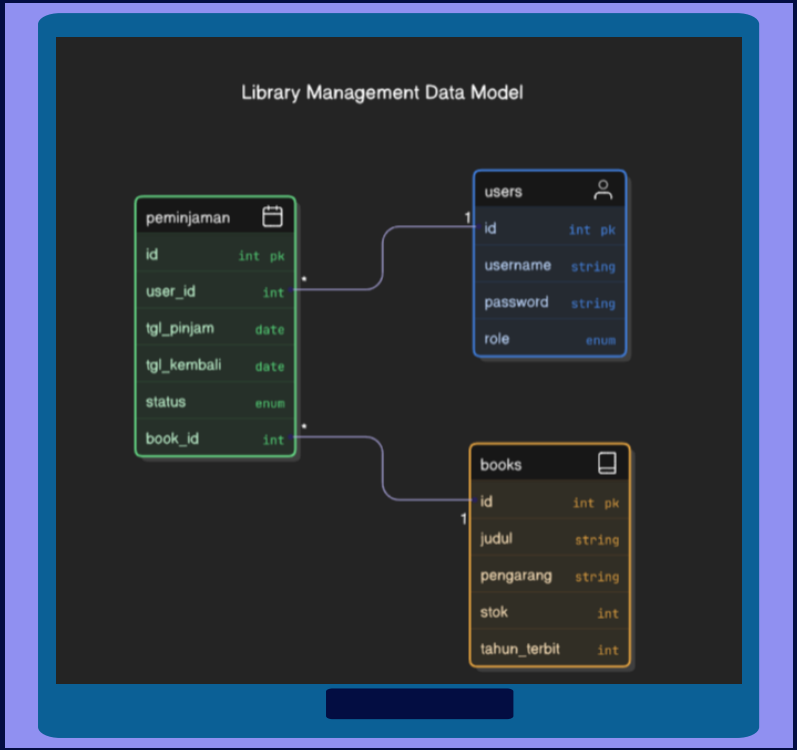
Mengatur User

Analisis Statistik

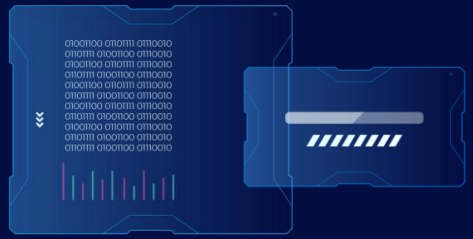
Model Database

Manajemen Data :

- Manajemen buku dan data pengguna, termasuk operasi CRUD.
- Penanganan Transaksi
- Proses peminjaman
- Analisa Statistik peminjaman



04 Program



IMPLEMENTASI SISTEM

Sistem Autentikasi

JFrame, JPasswordField, MySQL SELECT

Vlew

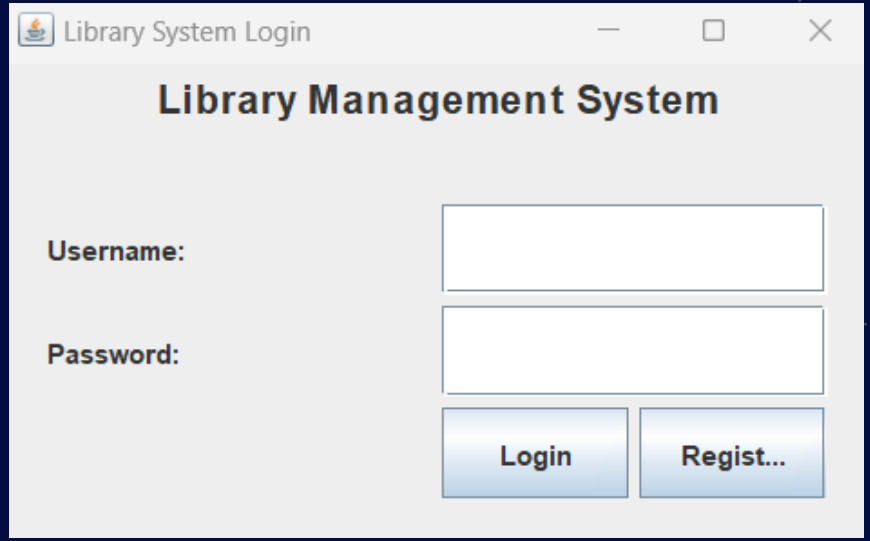
```
User authenticatedUser = authController.authenticate(username,
password);
```

Controller

```
User user = userDAO.getUserByUsername(username);
if (user != null && user.getPassword().equals(password)) {
   return user;
}
```

Dao

```
String sql = "SELECT * FROM users WHERE username = ?";
```



Sistem Registrasi

JFrame, JPasswordField, MySQL SELECT, MySQL INSERT

Vlew

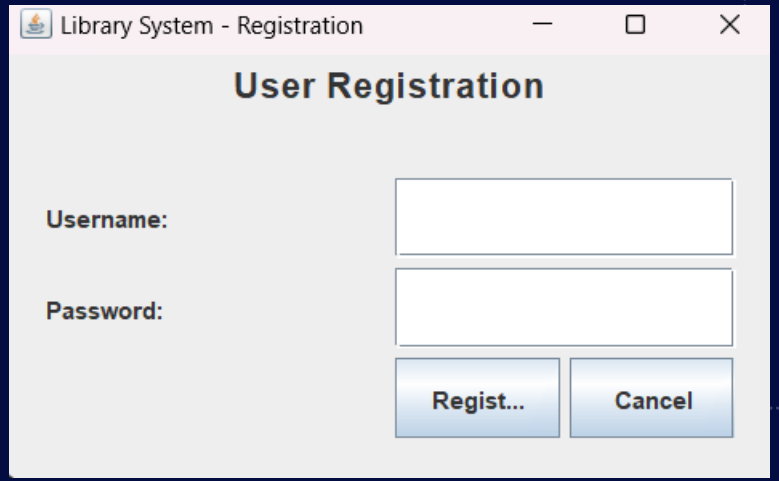
```
User newUser = new User(username, password, "user");
if (authController.register(newUser)) {
  // Registrasi suskses
  new LoginFrame().setVisible(true);
}
```

Controller

```
// Check if username already exists
if (userDAO.getUserByUsername(user.getUsername()) != null) {
   throw new CustomException("Username already exists");
}
// Add new user
return userDAO.addUser(user);
```

Dao

```
String sql = "INSERT INTO users (username, password, role) VALUES (?, ?,
?)";
try (Connection conn = DBConnection.getConnection();
    PreparedStatement stmt = conn.prepareStatement(sql)) {
   stmt.setString(1, user.getUsername());
   stmt.setString(2, user.getPassword());
   stmt.setString(3, user.getRole());
   stmt.executeUpdate();}
```



Sistem Peminjaman

JFrame, JTable, MySQL UPDATE, MySQL SELECT, MySQL INSERT

Alur Peminjaman

User Login → Cari Buku → Konfirmasi Peminjaman dengan klik tombol "Borrow Buku"

View

```
int bookId = (int) booksTable.getValueAt(selectedRow, 0);
try {
   loanController.pinjamBuku(
         bookId
   );}
```

Controller

```
if(bookDAO.getBookStock(bookId) <= 0) {
   throw new CustomException("Stok buku habis!");
}
Peminjaman peminjaman = new Peminjaman(
      Session.getCurrentUser().getId(),
      bookId,
      DateHelper.getCurrentDate(),
      "dipinjam"
);
```

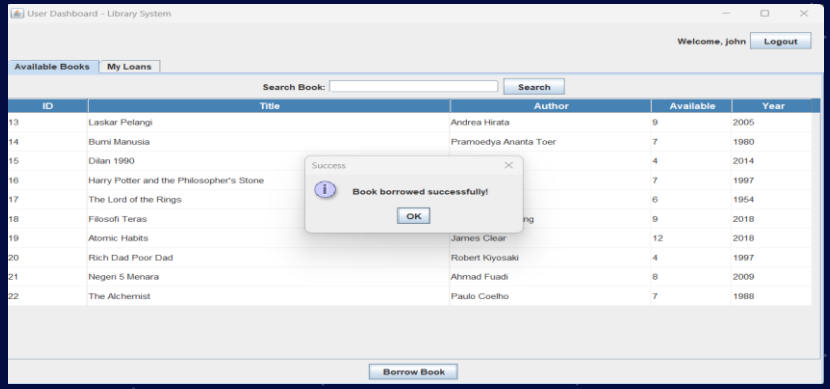
```
if(!peminjamanDAO.addPeminjaman(peminjaman)) {
   throw new CustomException("Gagal memproses peminjaman");
}
// Kurangi stok buku
if(!bookDAO.decreaseStock(bookId)) {
   throw new CustomException("Gagal update stok buku");}
```

Dao

```
String sql = "SELECT stok FROM books WHERE id = ?";
```

```
String sql = "INSERT INTO peminjaman (user_id, book_id, tgl_pinjam, status)
VALUES (?, ?, ?, ?)";
```

```
String sql = "UPDATE books SET stok = stok - 1 WHERE id = ? AND stok > 0";
```



Sistem History

JFrame, JTable, MySQL UPDATE, MySQL SELECT

```
//update stok buku
String sql = "UPDATE books SET stok = stok + 1 WHERE id = ?";
```

Vlew

```
loanController.kembalikanBuku(loanId);
```

Controller

```
if(!peminjamanDAO.isBookDipinjam(peminjaman.getUserId(),peminjaman.getBookId())) {
   throw new CustomException("Pilihlah Buku!\n" +
         "Yang Belum Dikembalikan");}
```

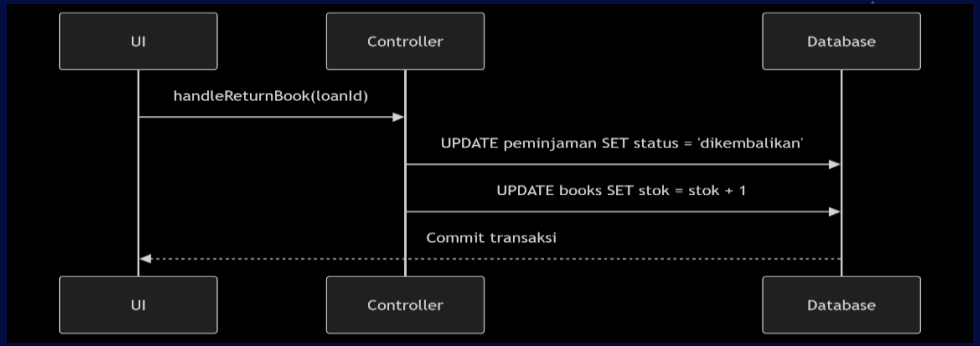
```
if(!peminjamanDAO.kembalikanBuku(peminjamanId)) {
   throw new CustomException("Gagal mengembalikan buku");
}
```

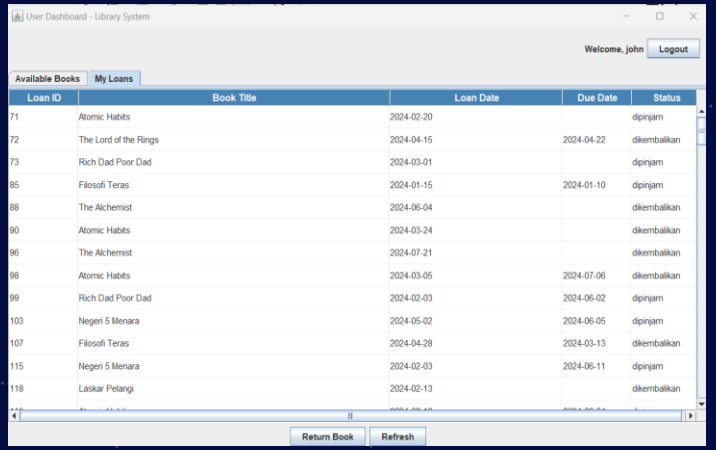
```
// Tambah stok buku
if(!bookDAO.increaseStock(peminjaman.getBookId())) {
   throw new CustomException("Gagal update stok buku");
}
```

Dao

```
// apakah sedang di pinjam
String sql = "SELECT count(*) count  FROM peminjaman p JOIN books b ON p.book_id = b.id WHERE p.user_id = ? and p.book_id = ?  and status = 'dipinjam'";
```

```
// ubah status pengembailian
String sql = "UPDATE peminjaman SET tgl_kembali = CURDATE(), status = 'dikembalikan' WHERE id = ?";
```





Manajemen Buku

JPanel, JTable, MySQL UPDATE, MySQL SELECT, MySQL INSERT, MySQL DELETE

View

```
bookController.addBook(newBook); // add buku
```

```
bookController.updateBook(updatedBook); // edit buku
```

```
Book existingBook = bookController.getBookById(bookId);
bookController.deleteBook(existingBook); hapus buku
```

Controller

```
bookDAO.addBook(book) // tambah buku
```

```
if(bookDAO.isBookExist(book)) {
   if(!bookDAO.updateBook(book)) {
      throw new CustomException("Gagal memperbarui
buku");} // update buku
```

```
if(!bookDAO.deleteBook(book.getId())) {
   throw new CustomException("Gagal menghapus buku");} // hapus buku
```

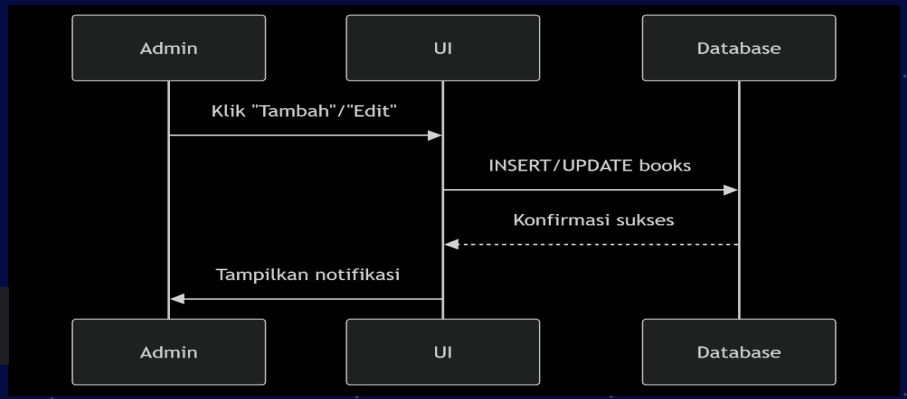
Dao

```
String sql = "INSERT INTO books (judul, pengarang, stok, tahun_terbit) VALUES (?,
?, ?, ?)"; // tambah buku
```

```
String sql = "UPDATE books SET judul=?, pengarang=?, stok=?, tahun_terbit=? WHERE
id=?"; // update book
```

```
String sql = "DELETE FROM books WHERE id = ?";// hapus buku
```

```
// pencarian buku
String sql = "SELECT * FROM books WHERE judul LIKE ?  OR judul LIKE ?" +
      " OR pengarang LIKE ?"+
      "  OR pengarang LIKE ?"+
      "ORDER BY \n" +
      "    CASE \n" +
      "        WHEN judul = ? OR pengarang = ? THEN 1 \n" +
      "        ELSE 2 \n END;";
```

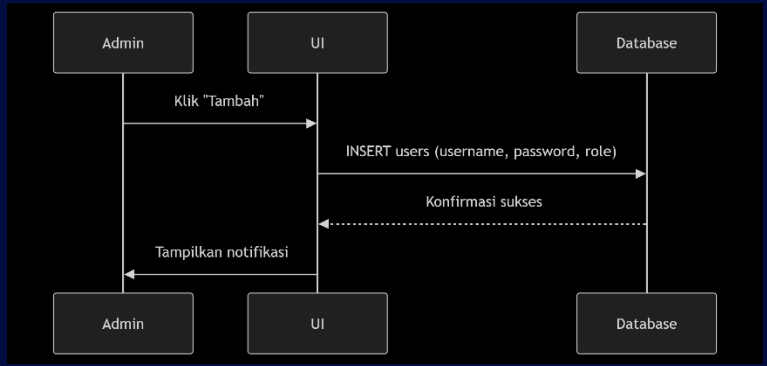


Manajemen User

JPanel, JTable, MySQL UPDATE, MySQL SELECT, MySQL INSERT, MySQL DELETE

View

```
User newUser = dialog.getUser();
userController.addUser(newUser); // tambah user
User updatedUser = dialog.getUser();
updatedUser.setId(userId);
userController.updateUser(updatedUser); // update user
userController.deleteUser(userId); // delete user
User user : userController.searchUsers(keyword) // pencarian user
```



Controller

```
if(userDAO.getUserByUsername(user.getUsername()) != null) {
    throw new CustomException("Username sudah digunakan");}  // add user
if(!userDAO.updateUser(user)) {
    throw new CustomException("Gagal memperbarui data user");} // update user
if(!userDAO.deleteUser(userId)) {throw new CustomException("Gagal menghapus user");} // delete user
List<User> users = userDAO.searchUsers(keyword); // search user
```

Dao

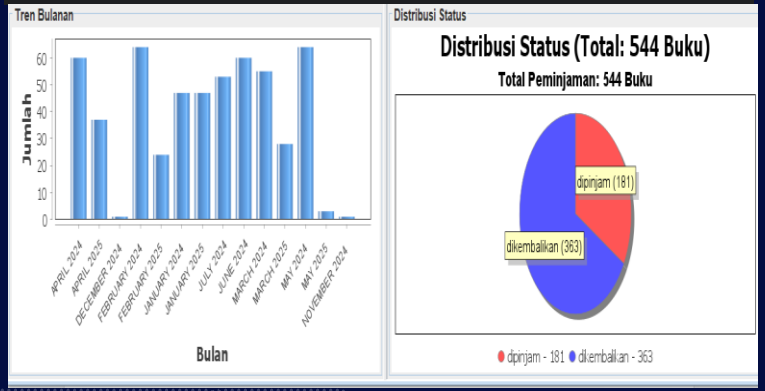
```
"INSERT INTO users (username, password, role) VALUES (?, ?, ?)“ ; // tambah user
"UPDATE users SET username=?, password=?, role=? WHERE id=?"; // update user
"DELETE FROM users WHERE id=?"; // delete user
List<User> users = new ArrayList<>();
String sql = "SELECT * FROM users WHERE username LIKE ?"; // search user
```

Statistik

JPanel, JTable, JFreeChart , MySQL SELECT

View

```
List<Report> data = controller.getFilteredReport(start, end, status, book, user);

// membuat bar chart
DefaultCategoryDataset barData = new DefaultCategoryDataset();
controller.getMonthlyTrend(data).forEach((month, count) ->
    barData.addValue(count, "Peminjaman", month));
JFreeChart barChart = ChartFactory.createBarChart(
    null, "Bulan", "Jumlah", barData,
    PlotOrientation.VERTICAL, false, true, false);
styleBarChart(barChart);
barChartPanel.setChart(barChart);
```




```
// Pie Chart
DefaultPieDataset pieData = new DefaultPieDataset();
controller.getStatusDistribution(data).forEach(pieData::setValue);
// Hitung total buku
int totalBuku = data.stream()
      .mapToInt(Report::getJumlahBuku)
      .sum();

// Buat judul dengan total
String pieTitle = "Distribusi Status (Total: " + totalBuku + " Buku)";
JFreeChart pieChart = ChartFactory.createPieChart(
      pieTitle,
      pieData,
      true,
      true,
      false
);
stylePieChart(pieChart);
// Tambahkan total di legend
pieChart.addSubtitle(new TextTitle("Total Peminjaman: " + totalBuku+" Buku",
      new Font("SansSerif", Font.BOLD, 14)));
pieChartPanel.setChart(pieChart);
```

Statistik

JPanel, JTable, JFreeChart , MySQL SELECT

Controller

```
dao.getFilteredReport(startDate, endDate, status, bookTitle, username); //
filterisasi data
// untuk bar chart
public Map<String, Integer> getMonthlyTrend(List<Report> data)  throws
CustomException {
   try {
      return dao.calculateMonthlyTrend(data);
   }catch(Exception e){
      throw new CustomException("Gagal mengambil data trend buku: " +
e.getMessage());}

// Untuk Pie chart
public Map<String, Integer> getStatusDistribution(List<Report> data) throws
CustomException {
   try {
      return data.stream()
            .collect(Collectors.groupingBy(
                  Report::getStatus,
                  Collectors.summingInt(Report::getJumlahBuku)
            ));
   }catch(Exception e){
      throw new CustomException("Gagal mengambil data distribusi buku: " +
e.getMessage());
   }
}
```

Dao

```
private static final String BASE_REPORT_QUERY = """
   SELECT
      p.id AS peminjaman_id,
      p.tgl_pinjam,
      p.tgl_kembali,
      p.status,
      u.username,
      b.judul AS judul_buku,
      b.pengarang,
      b.tahun_terbit,
      b.stok AS stok_saat_pinjam,
      DATEDIFF(COALESCE(p.tgl_kembali, CURDATE()), p.tgl_pinjam) AS
durasi_hari,
      (SELECT COUNT(*) FROM peminjaman p2 WHERE p2.id = p.id) AS
jumlah_buku

   FROM peminjaman p
   JOIN users u ON p.user_id = u.id
   JOIN books b ON p.book_id = b.id
   """;
```

Buku Terpopuler

JPanel, JTable, JFreeChart, MySQL SELECT

View

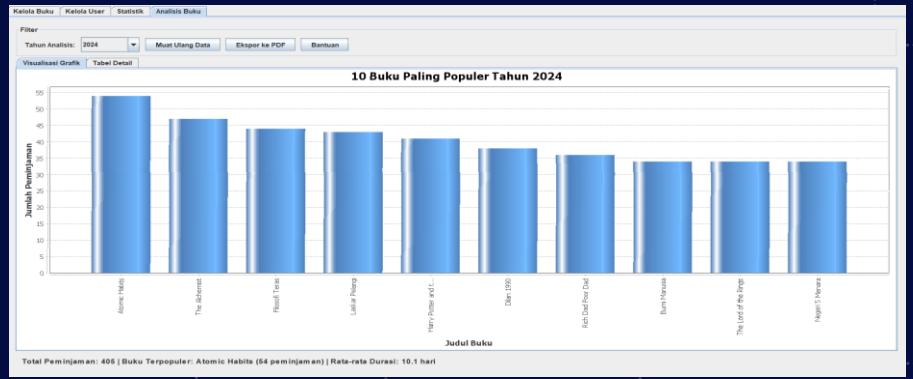
```
int tahun = (Integer) yearCombo.getSelectedItem();
List<PopularBookAnalysis> books = controller.getPopularBooks(tahun, 15);

DefaultCategoryDataset dataset = new DefaultCategoryDataset();
for (PopularBookAnalysis book : books) {
   dataset.addValue(
         book.getTotalPeminjaman(),
         "Jumlah Peminjaman",
         book.getJudul() );}
JFreeChart chart = ChartFactory.createBarChart(
      "10 Buku Paling Populer Tahun " + tahun,
      "Judul Buku",
      "Jumlah Peminjaman",
      dataset,
      PlotOrientation.VERTICAL,
      false, true, false);

chartPanel.setChart(chart);
```

Controller

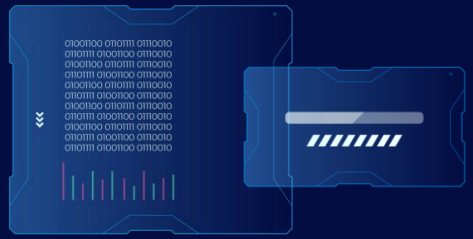
```
dao.getPopularBooksAnalysis(tahun, limit);
```

Dao

```
"""" SELECT
      b.judul,
      b.pengarang,
      COUNT(p.id) AS total_pinjam,
      AVG(DATEDIFF(p.tgl_kembali, p.tgl_pinjam)) AS avg_durasi,
      MAX(p.tgl_pinjam) AS terakhir_pinjam,
      (COUNT(p.id) * 100.0) / (SELECT COUNT(*) FROM peminjaman
   WHERE YEAR(tgl_pinjam) = ?)
    AS persentase
    FROM peminjaman p JOIN books b ON p.book_id = b.id WHERE YEAR(p.tgl_pinjam) = ?
    GROUP BY b.judul, b.pengarang
    ORDER BY total_pinjam DESC
    LIMIT ?""";
```



05 Pengujian



Uji Modul Autentikasi

Skenerio	Input	Output Diharapkan	Hasil
Login admin valid	Username: admin, Password: admin123	Redirect ke dashboard admin	✓
Login user dengan password salah	Password: salah456	Notifikasi "Invalid Username or Password"	✓
Belum input username atau password	Username atau password di kosongkan	Notifkasi “Username or password are required”	✓

Uji Manajemen Buku

Skenerio	Input	Output Diharapkan	Hasil
Tambah buku stok negatif	Stok: -5	Stock akan berubah ke positif : 5	
Edit judul, pengarang, stock dan year	Input semua kolom	Buku terupdate di database dan notifkasi sukses muncul	
Hapus buku yang dipinjam	ID Buku: 11 (sedang dipinjam)	Error "Buku dalam transaksi"	

Uji Manajemen User

Skenerio	Input	Output Diharapkan	Hasil
Tambah user baru dengan password pendek	User : azzam Password : 12345 Role : admin	Muncul notifikasi: password dan username harus lebih dari 6 karakter	
Edit username/role/password	Input semua kolom	Data terupdate di database dan notifkasi sukses muncul	
Hapus User yang sedang login	User : azzam	Notifikasi gagal hapus user yang sedang login	

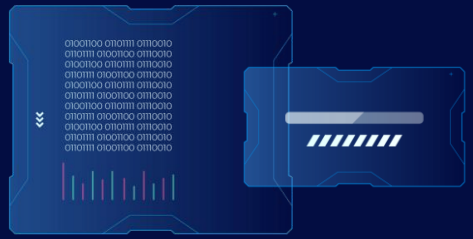
Uji Transaksi Peminjaman

Skenerio	Input	Output Diharapkan	Hasil
Meminjam buku dengan stock 0	Buku ID: 52 (stok=0)	Muncul notifikasi bahwa peminjaman gagal	✓
Mengembalikan buku yang sudah di kembalikan	Peminjaman Id : 192 (sudah di kembalikan)	Muncul notifikasi error “pilih buku yang belum di kembalikan	✓
Pengembalian terlambat	Telat 7 hari	Tampilkan notifikasi denda	✗

Uji Keamanan

Skenerio	Hasil	Kerentanan
SQL Injection pada login	Diblokir	Tidak ditemukan
Enkripsi Password	Belum terimplementasi	ditemukan

06
Evaluasi



Kelebihan VS Kekurangan

Kelebihan

- Efisiensi Waktu Transaksi
- Konsistensi Data
- Analisis Data Real-Time

Kekurangan

- Keamanan (password belum di enkripsi)
- Fitur Belum Lengkap (fitur pencarian lengkap dan fitur export pdf belum tersedia)
- Akses terbatas (belum menerapkan RestfulApi
- Masih synchronous

Thanks!

Do you have any questions?
Ask Us for collaboration

Link App : https://github.com/lildwagz/LBS