

LAPORAN UAS
PEMROGRAMAN BERORIENTASI OBJEK

"Perancangan dan Implementasi Sistem Informasi Perpustakaan Berbasis GUI."

Dosen Pengampu :
Taufik Ridwan, M. T.



Disusun oleh:
Kelompok 4
4A Sistem Informasi

- | | |
|-------------------------------|---------------|
| 1. Dwi Ayu Meilinda Maharani | 2310631250047 |
| 2. Chamila Meyra Aziza | 2310631250008 |
| 3. Nisrina Dwi Devina Sari | 2310631250071 |
| 4. Akbar Abigail Rachman | 2310631250085 |
| 5. Mochammad Delvin Farhan A. | 2310631250063 |

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS SINGAPERBANGSA KARAWANG
2025

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga kami, kelompok 4, dapat menyelesaikan laporan Ujian Akhir Semester (UAS) ini dengan baik dan tepat waktu. Laporan ini disusun sebagai salah satu syarat untuk menyelesaikan tugas pada mata kuliah Sistem Informasi dengan judul "*Perancangan dan Implementasi Sistem Informasi Perpustakaan Berbasis GUI.*"

Penyusunan laporan ini bertujuan untuk memberikan gambaran yang komprehensif mengenai proses perancangan, implementasi, dan pengujian sistem informasi perpustakaan berbasis GUI. Laporan ini mencakup berbagai aspek penting, mulai dari pengertian perancangan basis data, penggunaan Entity-Relationship Diagram (ERD) dalam database perpustakaan, implementasi database di phpMyAdmin, hingga perancangan UML (class diagram dan activity diagram) serta implementasi dan pengujian kode program.

Selama proses penyusunan laporan ini, kami telah memperoleh banyak bantuan dan dukungan dari berbagai pihak. Oleh karena itu, kami ingin menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Bapak Taufik Ridwan, S.T., M.T., selaku dosen mata kuliah Pemrograman Berbasis Objek yang telah memberikan pembelajaran, bimbingan, dan arahan selama penyusunan laporan ini.
2. Teman-teman kelompok 4, yang telah memberikan dukungan dan kerja sama yang sangat baik dalam menyelesaikan laporan ini secara menyeluruh.

Kami menyadari bahwa laporan ini masih jauh dari sempurna dan masih terdapat kekurangan yang perlu diperbaiki. Oleh karena itu, kami sangat mengharapkan saran dan kritik yang membangun dari berbagai pihak untuk perbaikan laporan ini di masa mendatang. Akhir kata, kami berharap laporan ini dapat bermanfaat bagi para pembaca dan menjadi referensi yang berguna dalam pengembangan sistem informasi perpustakaan berbasis GUI di masa yang akan datang.

Karawang, 09 Juni 2025

Kelompok 4

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI.....	ii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan.....	2
BAB II PEMBAHASAN.....	3
2.1 Perancangan Basis Data	3
A. Struktur tabel yang terdapat di dalam database sistem perpustakaan:	6
2.2. Unified Modelling Language (UML)	12
1. CLASS DIAGRAM.....	12
2. ACTIVITY DIAGRAM.....	14
3. USE CASE DIAGRAM.....	17
4. SEQUENCE DIAGRAM.....	18
2.3 Implementasi Kode Program dan Pengujian.....	22
1. Implementasi Kode Program.....	22
2. Konsep OOP Beserta Penerapannya Pada Sistem Informasi Perpustakaan	34
3. Fitur dan Pengujian Sistem Informasi Perpustakaan	37
BAB III PENUTUP	45
3.1 Kesimpulan.....	45
3.2 Saran	46

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam perkembangan teknologi informasi yang semakin pesat dewasa ini, kebutuhan akan sistem pengelolaan perpustakaan yang efisien dan terstruktur menjadi sangat penting guna menunjang kualitas layanan kepada pengguna. Pemanfaatan sistem informasi berbasis antarmuka grafis (Graphical User Interface/GUI) menawarkan solusi yang inovatif dalam mengelola data perpustakaan secara sistematis, cepat, dan mudah diakses. Untuk mewujudkan sistem informasi perpustakaan berbasis GUI yang andal, diperlukan tahapan perancangan basis data yang tepat dan terstruktur. Perancangan ini mencakup proses identifikasi entitas, atribut, serta hubungan antarentitas guna membentuk struktur database yang efisien dan mendukung kinerja sistem secara optimal. Salah satu alat yang berperan penting dalam tahapan ini adalah Entity-Relationship Diagram (ERD), yang berguna dalam memetakan hubungan logis antar elemen data sehingga mempermudah proses perancangan dan pengembangan sistem.

Dalam implementasinya, perancangan basis data umumnya diwujudkan melalui penggunaan phpMyAdmin, sebuah perangkat lunak berbasis web yang mendukung sistem manajemen basis data MySQL. Dengan phpMyAdmin, proses pembuatan, pengelolaan, dan pengolahan data menjadi lebih praktis dan efisien. Struktur database yang baik akan mendukung kegiatan utama perpustakaan seperti pengelolaan data buku, data anggota, serta transaksi peminjaman dan pengembalian buku. Selain perancangan basis data, penyusunan diagram berbasis Unified Modeling Language (UML), seperti class diagram dan activity diagram, juga memiliki peran strategis dalam merancang sistem. Class diagram digunakan untuk menggambarkan struktur dan hubungan antar kelas dalam sistem, sedangkan activity diagram menjelaskan alur proses dan aktivitas yang terjadi dalam sistem. Kedua jenis diagram ini diperlukan untuk memastikan sistem dirancang secara terstruktur, logis, dan mudah diimplementasikan.

Tahap implementasi dan pengujian kode program menjadi fase akhir dalam proses pengembangan sistem informasi. Implementasi melibatkan penerapan rancangan sistem ke dalam bentuk kode program yang dapat dijalankan, sementara pengujian dilakukan untuk memastikan sistem berfungsi sesuai spesifikasi dan bebas dari kesalahan. Pengujian yang dilakukan secara menyeluruh akan sangat membantu dalam mendeteksi serta memperbaiki kesalahan sebelum sistem digunakan oleh pengguna secara luas. Dengan demikian, penyusunan laporan ini bertujuan untuk memberikan gambaran yang komprehensif mengenai proses perancangan, implementasi, dan pengujian sistem informasi perpustakaan berbasis GUI, sekaligus menjadi referensi dalam pengembangan sistem pengelolaan perpustakaan berbasis digital yang efektif dan efisien.

1.2 Rumusan Masalah

Mengacu pada latar belakang masalah yang dikemukakan, maka rumusan masalah pada makalah ini adalah:

1. Bagaimana ERD dapat diaplikasikan pada database perpustakaan untuk sistem informasi perpustakaan?
2. Apa saja struktur database yang dimiliki oleh database perpustakaan di PHPMyAdmin?
3. Bagaimana cara merancang class diagram dan activity diagram untuk sistem informasi perpustakaan berbasis GUI?
4. Bagaimana proses implementasi dan pengujian kode program untuk sistem informasi perpustakaan berbasis GUI?

1.3 Tujuan

Dengan laporan ini, penulis memiliki tujuan yang ingin dicapai, yaitu:

1. Menjelaskan bagaimana ERD dapat diaplikasikan untuk mendesain database perpustakaan.
2. Mengidentifikasi dan menjelaskan struktur database yang diperlukan untuk sistem informasi perpustakaan.
3. Menjelaskan konsep UML, khususnya menyusun class diagram dan activity diagram yang relevan untuk sistem informasi perpustakaan berbasis GUI.
4. Menguraikan implementasi kode program dan juga menganalisis hasil pengujian untuk sistem informasi perpustakaan berbasis GUI.

BAB II PEMBAHASAN

2.1 Perancangan Basis Data

a. Entity Relation Ship (ERD)

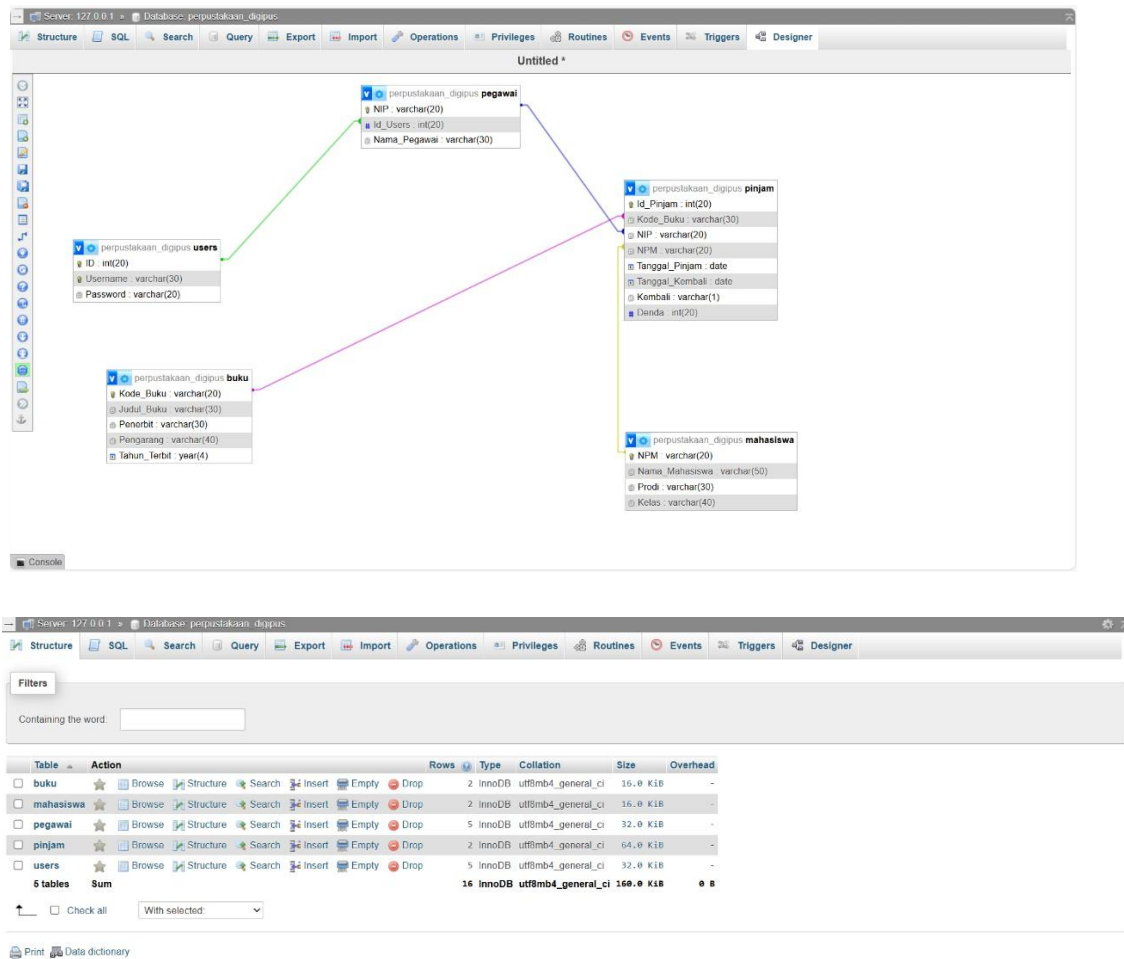
Entity Relationship Diagram (ERD) adalah suatu diagram yang digunakan untuk merancang suatu basis data, dipergunakan untuk memperlihatkan hubungan atau relasi antar entitas atau objek yang terlihat beserta atributnya. Diagram ERD dipisahkan berdasarkan keterkaitan antara entitas ke entitas, atau entitas ke atribut data. Dengan menggunakan ERD, sistem database yang sedang dibentuk dapat digambarkan dengan lebih terstruktur dan terlihat rapi. Selain digunakan dalam perancangan database, ERD sendiri sering digunakan untuk debugging database jika terjadi masalah pada database. Untuk melakukan debug pada database bukanlah hal yang mudah, terlebih lagi jika database yang mengalami masalah memiliki banyak tabel dan memerlukan penulisan SQL yang kompleks. Dengan menggambarkan skema database menggunakan ERD, kamu menjadi lebih mudah untuk menemukan permasalahan yang terjadi dalam database dan menyelesaikan masalah dengan mudah.



Entity Relationship Diagram (ERD) di atas menggambarkan hubungan antar entitas dalam sistem peminjaman buku Digidaw. Terdapat lima entitas utama, yaitu users, pegawai, mahasiswa, buku, dan pinjam. Entitas users menyimpan data login untuk pegawai. Entitas

pegawai dan mahasiswa mewakili pihak yang terlibat langsung dalam proses peminjaman. Entitas buku menyimpan data koleksi buku yang tersedia, sedangkan entitas pinjam mencatat seluruh transaksi peminjaman. Relasi antar entitas menunjukkan keterkaitan logis, seperti satu pegawai dapat menangani banyak peminjaman, dan satu mahasiswa dapat meminjam banyak buku. Struktur ini dirancang untuk mempermudah pengelolaan data secara sistematis dan efisien.

a. Database PHPMyAdmin



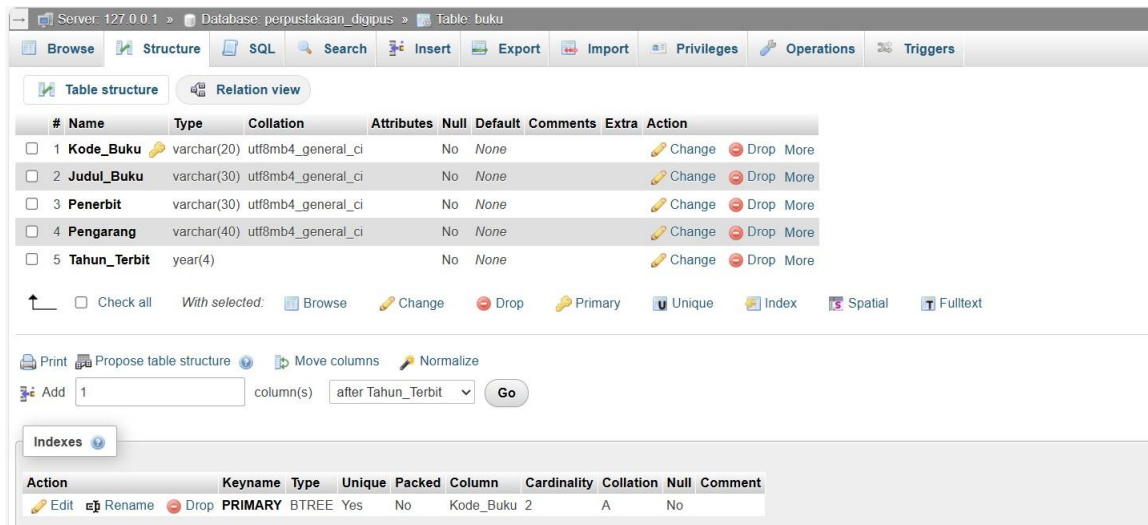
Sebuah desain database struktur relasional dari sistem perpustakaan digital sederhana yang telah kami berikan nama 'perpustakaan_digipus'. Skema ini memperlihatkan bagaimana data mengenai buku, pengguna sistem, dan aktivitas peminjaman saling terhubung dan dikelola. Dalam desain ini, terdapat lima tabel utama, yaitu 'users', 'pegawai', 'mahasiswa', 'buku', dan 'pinjam', yang masing-masing memiliki

peran penting dalam operasional sistem. Tabel `users` menyimpan informasi akun login pengguna, khususnya untuk pegawai perpustakaan. Tabel ini berisi data seperti `ID`, `Username`, dan `Password`, serta terhubung ke tabel `pegawai` melalui kolom `id_Users`, yang berfungsi sebagai kunci tamu (*foreign key*). Tabel `pegawai` menyimpan informasi pegawai atau pustakawan, termasuk `NIP` sebagai identitas utama, nama lengkap, dan referensi ke akun `users`.

Untuk data mahasiswa yang menjadi peminjam buku, disimpan di tabel `mahasiswa`, yang mencakup `NPM`, `Nama_Mahasiswa`, `Prodi`, dan `Kelas`. Sementara itu, data koleksi buku dikelola dalam tabel `buku`, dengan atribut seperti `Kode_Buku`, `Judul_Buku`, `Penerbit`, `Pengarang`, dan `Tahun_Terbit`. Semua aktivitas peminjaman dicatat di tabel `pinjam`, yang menjadi pusat transaksi dalam sistem. Tabel ini berisi kolom `id_Pinjam`, serta kunci tamu yang menghubungkannya ke tabel `buku`, `pegawai`, dan `mahasiswa`, ditambah informasi tanggal pinjam, tanggal kembali, status pengembalian, dan jumlah denda bila terjadi keterlambatan. Relasi antar tabel digambarkan melalui garis yang menunjukkan hubungan kunci tamu. Tabel `users` terhubung satu-satu dengan `pegawai`, sedangkan `buku`, `pegawai`, dan `mahasiswa` memiliki hubungan satu-ke-banyak dengan tabel `pinjam`, karena satu entitas dalam masing-masing tabel tersebut dapat terlibat dalam banyak transaksi peminjaman. Dengan struktur seperti ini, sistem perpustakaan digital dapat berjalan dengan baik. Mahasiswa meminjam buku melalui pegawai yang sudah masuk ke sistem, lalu pegawai mencatat data peminjaman yang tersimpan di tabel `pinjam`. Nantinya, saat pengembalian dilakukan, data tersebut akan diperbarui, termasuk pencatatan denda jika berlaku. Skema ini menunjukkan pengelolaan data perpustakaan yang terstruktur dan cukup lengkap untuk kebutuhan sistem manajemen perpustakaan tingkat dasar.

A. Struktur tabel yang terdapat di dalam database sistem perpustakaan:

1. Tabel Buku



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Kode_Buku	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
2	Judul_Buku	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Penerbit	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
4	Pengarang	varchar(40)	utf8mb4_general_ci		No	None			Change Drop More
5	Tahun_Terbit	year(4)			No	None			Change Drop More

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	Kode_Buku	2	A	No	

Memperlihatkan tampilan struktur tabel buku dalam database perpustakaan_digipus, yang diakses menggunakan phpMyAdmin. Tampilan ini memberikan informasi detail mengenai susunan kolom-kolom dalam tabel tersebut, serta bagaimana database mengelola indeks yang berfungsi mempercepat proses pencarian data. Dalam tabel buku, terdapat beberapa kolom penting seperti Kode_Buku, Judul_Buku, Penerbit, Pengarang, dan Tahun_Terbit, masing-masing dengan tipe data yang sesuai kebutuhan, seperti varchar untuk teks dan year untuk tahun terbit buku. Salah satu hal penting yang ditunjukkan adalah bahwa kolom Kode_Buku telah ditetapkan sebagai Primary Key, yang ditandai dengan ikon kunci berwarna emas di sebelahnya. Ini berarti setiap entri buku di database harus memiliki Kode_Buku yang unik dan tidak boleh kosong. Penetapan Kode_Buku sebagai Primary Key memberikan jaminan bahwa tidak akan ada dua buku dengan kode yang sama, menjaga integritas dan keunikan data buku dalam sistem. Di bagian bawah tampilan struktur, phpMyAdmin juga menampilkan informasi tentang indeks yang digunakan dalam tabel ini. Terlihat bahwa indeks utama atau primary index memakai algoritma BTREE, yang umum digunakan karena efisien untuk pencarian data. Indeks ini bersifat unik dan tidak memperbolehkan nilai kosong, sesuai dengan aturan dari Primary Key. Kolom yang digunakan sebagai dasar indeks adalah Kode_Buku. Keseluruhan struktur ini menunjukkan bahwa sistem database dirancang dengan baik, karena menggunakan Kode_Buku sebagai identifikasi utama setiap buku. Hal ini tidak

hanya memudahkan pencarian dan pengelolaan data, tetapi juga menjadi dasar untuk menghubungkan tabel ini dengan tabel lain seperti tabel pinjam yang mencatat transaksi peminjaman. Dengan begitu, setiap transaksi peminjaman bisa dipastikan merujuk ke buku yang valid dan tercatat secara resmi dalam sistem perpustakaan digital tersebut.

2. Tabel Mahasiswa

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	NPM	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
2	Nama_Mahasiswa	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
3	Prodi	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
4	Kelas	varchar(40)	utf8mb4_general_ci		No	None			Change Drop More

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	NPM	2	A	No	

Menunjukkan tampilan struktur dari tabel 'mahasiswa' dalam database 'perpustakaan_digipus', yang diakses melalui antarmuka phpMyAdmin. Tampilan ini menyajikan informasi teknis mengenai isi tabel, termasuk daftar kolom beserta tipe datanya. Tabel 'mahasiswa' terdiri dari beberapa kolom penting, yaitu 'NPM', 'Nama_Mahasiswa', 'Prodi', dan 'Kelas'. Masing-masing kolom menggunakan tipe data 'varchar', menandakan bahwa isinya berupa teks. Salah satu hal yang langsung terlihat adalah ikon kunci emas di sebelah kolom 'NPM', yang berarti kolom ini dijadikan sebagai Primary Key atau kunci utama untuk tabel ini. Di bagian bawah struktur, phpMyAdmin menampilkan detail tentang indeks yang berlaku dalam tabel tersebut. Indeks utama, yang diberi nama 'PRIMARY', diterapkan pada kolom 'NPM'. Indeks ini bersifat unik, artinya setiap nilai 'NPM' harus berbeda satu sama lain dan tidak boleh kosong (null). Jenis indeks yang digunakan adalah 'BTREE', yang umum dipakai karena mampu mempercepat proses pencarian data dalam skala besar secara efisien. Penetapan 'NPM' sebagai Primary Key memiliki peran penting dalam menjaga struktur dan keandalan database. Pertama, ia menjamin bahwa setiap mahasiswa memiliki identitas yang unik, sehingga tidak terjadi data ganda atau bentrok antar identitas. Kedua, hal ini juga membantu menjaga integritas data karena tidak mungkin menyimpan informasi mahasiswa tanpa NPM. Terakhir, 'NPM'

berfungsi sebagai penghubung antar tabel, terutama dalam relasi dengan tabel `pinjam`, yang mencatat aktivitas peminjaman buku. Dengan begitu, sistem dapat mencocokkan data peminjaman dengan identitas mahasiswa yang sah dan terdokumentasi dalam sistem.

3. Tabel Pegawai

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	NIP	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
2	Id_Users	int(20)			No	None			Change Drop More
3	Nama_Pegawai	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	NIP	5	A	No	
Edit Rename Drop	Id_Users	BTREE	No	No	Id_Users	5	A	No	

Memperlihatkan struktur dari tabel pegawai yang berada dalam database perpustakaan_digipus. Tabel ini terdiri dari tiga kolom utama, yaitu NIP dengan tipe data varchar(20), id_Users bertipe int(20), dan Nama_Pegawai bertipe varchar(30). Pada bagian struktur, terlihat dua ikon penting yang menunjukkan keberadaan dua jenis kunci dalam tabel ini. Ikon kunci emas yang berada di samping kolom NIP menunjukkan bahwa kolom tersebut berfungsi sebagai primary key, yaitu kunci utama yang memastikan setiap data pegawai memiliki identitas unik. Sementara itu, ikon kunci perak atau abu-abu di samping kolom id_Users menunjukkan bahwa kolom tersebut memiliki indeks dan dalam konteks desain basis data, kolom ini berperan sebagai foreign key atau kunci tamu yang menghubungkan ke tabel lain. Lebih jauh, bagian indeks di bawah struktur tabel memberikan informasi teknis terkait pengaturan dua kunci tersebut. Pada baris pertama, terlihat bahwa kolom NIP menjadi primary key dengan karakteristik unique dan not null, yang artinya setiap nilai pada kolom tersebut harus unik dan tidak boleh kosong. Ini menjadikan NIP sebagai identitas utama untuk setiap pegawai dalam sistem. Baris kedua menampilkan indeks pada kolom id_Users. Meskipun bukan primary key karena tidak

bersifat unik, kolom ini juga diatur agar tidak boleh kosong. Pemberian indeks pada id_Users bertujuan untuk meningkatkan performa saat terjadi relasi antara tabel pegawai dan tabel users. Adanya dua jenis indeks ini memiliki fungsi yang berbeda namun saling melengkapi. Primary key NIP digunakan untuk identifikasi internal dalam tabel pegawai, sedangkan indeks pada id_Users digunakan untuk menjaga efisiensi dan integritas saat menghubungkan tabel pegawai dengan tabel users melalui relasi foreign key. Dengan kata lain, sistem dapat dengan cepat mencari dan mencocokkan data login pengguna berdasarkan hubungan antar tabel tersebut. Oleh karena itu, gambar ini menunjukkan bahwa desain struktur tabel pegawai telah menerapkan praktik yang baik dalam pengelolaan relasi dan integritas data di dalam basis data perpustakaan digital tersebut.

4. Tabel Pinjam

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Id_Pinjam	int(20)			No	None		AUTO_INCREMENT	Change Drop More
2	Kode_Buku	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	NIP	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
4	NPM	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
5	Tanggal_Pinjam	date			No	current_timestamp()			Change Drop More
6	Tanggal_Kembali	date			Yes	NULL			Change Drop More
7	Kembali	varchar(1)	utf8mb4_general_ci		No	None			Change Drop More
8	Denda	int(20)			No	0			Change Drop More

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	Id_Pinjam	2	A	No	
Edit Rename Drop	NIP	BTREE	No	No	NIP	2	A	No	
Edit Rename Drop	NPM	BTREE	No	No	NPM	2	A	No	
Edit Rename Drop	Kode_Buku	BTREE	No	No	Kode_Buku	2	A	No	

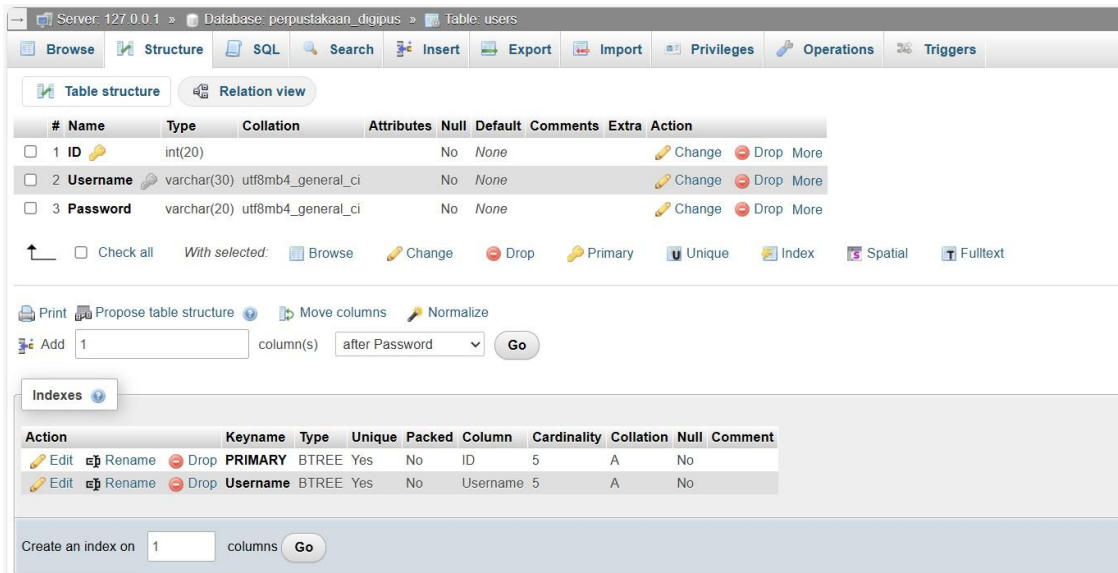
Gambar yang ditampilkan menunjukkan struktur tabel pinjam yang terdapat dalam database perpustakaan_digipus. Tabel ini merupakan tabel transaksi utama dalam sistem, yang menjadi penghubung antara data buku, mahasiswa, dan pegawai. Tabel ini memiliki sejumlah kolom penting, dimulai dari kolom id_Pinjam yang berfungsi sebagai primary key. Kolom ini diberi fitur auto_increment, artinya setiap transaksi baru akan secara otomatis mendapat nomor urut yang unik dari sistem, seperti 1, 2, 3, dan seterusnya. Selain

itu, terdapat kolom-kolom seperti Kode_Buku, NIP, dan NPM yang masing-masing menjadi foreign key yang terhubung ke tabel buku, pegawai, dan mahasiswa. Kolom-kolom ini memiliki indeks untuk mempercepat proses pencarian data terkait peminjaman.

Selanjutnya, terdapat kolom Tanggal_Pinjam yang mencatat waktu peminjaman. Jika tidak diisi secara manual, kolom ini otomatis terisi dengan waktu saat data dibuat karena adanya nilai default `current_timestamp()`. Kolom Tanggal_Kembali mencatat waktu pengembalian, namun nilainya diizinkan kosong atau null karena ketika transaksi peminjaman baru dibuat, tanggal kembalinya belum diketahui. Kolom Kembali menunjukkan status apakah buku sudah dikembalikan atau belum, dengan nilai awal 0 sebagai tanda buku belum kembali. Terakhir, kolom Denda mencatat jumlah denda dengan nilai awal 0, yang nantinya dapat berubah tergantung lamanya keterlambatan pengembalian buku. Bagian indeks pada tabel ini menunjukkan bahwa terdapat empat jenis indeks, jumlah yang wajar untuk sebuah tabel transaksi. Indeks pertama adalah primary key pada kolom `id_Pinjam` yang berfungsi sebagai identitas utama setiap transaksi. Indeks kedua berada pada kolom NIP yang menjadi foreign key ke tabel pegawai, memungkinkan sistem dengan cepat menampilkan transaksi yang ditangani oleh pegawai tertentu. Indeks ketiga berada pada kolom NPM, yang mempercepat pencarian riwayat peminjaman berdasarkan mahasiswa tertentu. Indeks keempat berada pada kolom Kode_Buku, yang mempercepat pencarian status atau riwayat peminjaman sebuah buku.

Keseluruhan struktur tabel pinjam ini menunjukkan desain database yang baik dan logis. Dengan adanya primary key yang otomatis, foreign key yang saling terhubung, serta indeks yang menunjang performa, sistem perpustakaan digital ini mampu mengelola transaksi dengan efisien. Nilai default dan pengaturan null juga mencerminkan logika bisnis di dunia nyata, seperti keterlambatan pengembalian atau status peminjaman. Hal ini membuat tabel pinjam menjadi pusat dari seluruh alur peminjaman dalam sistem perpustakaan yang dibangun.

5. Tabel Users



Server: 127.0.0.1 » Database: perpustakaan_digipus » Table: users

Table structure

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(20)			No	None			Change Drop More
2	Username	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Password	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	ID	5	A	No	
Edit Rename Drop	Username	BTREE	Yes	No	Username	5	A	No	

Create an index on 1 columns Go

Menunjukkan struktur tabel 'users' yang berada dalam database 'perpustakaan_digipus'. Tabel ini berfungsi untuk menyimpan data akun pengguna yang dapat digunakan untuk login ke dalam sistem. Struktur tabel ini terdiri dari tiga kolom utama, yaitu 'ID' dengan tipe data 'int(20)', 'Username' dengan tipe data 'varchar(30)', dan 'Password' dengan tipe data 'varchar(20)'. Kolom 'ID' memiliki ikon kunci emas yang menandakan bahwa kolom ini adalah Primary Key. Sementara itu, kolom 'Username' memiliki ikon kunci perak atau abu-abu yang menandakan bahwa kolom tersebut merupakan Unique Key. Informasi indeks yang ditampilkan pada bagian bawah gambar menjelaskan bahwa terdapat dua indeks pada tabel ini. Indeks pertama adalah Primary Key yang diterapkan pada kolom 'ID'. Kunci ini berfungsi sebagai identitas utama untuk setiap baris data dan tidak boleh kosong atau duplikat. Kolom 'ID' juga digunakan dalam relasi dengan tabel lain, seperti menghubungkan tabel 'users' dengan tabel 'pegawai'. Indeks kedua adalah Unique Key yang diterapkan pada kolom 'Username'. Indeks ini memastikan bahwa tidak ada dua akun yang memiliki nama pengguna yang sama. Selain bersifat unik, kolom ini juga tidak boleh kosong karena setiap pengguna wajib memiliki username yang berbeda.

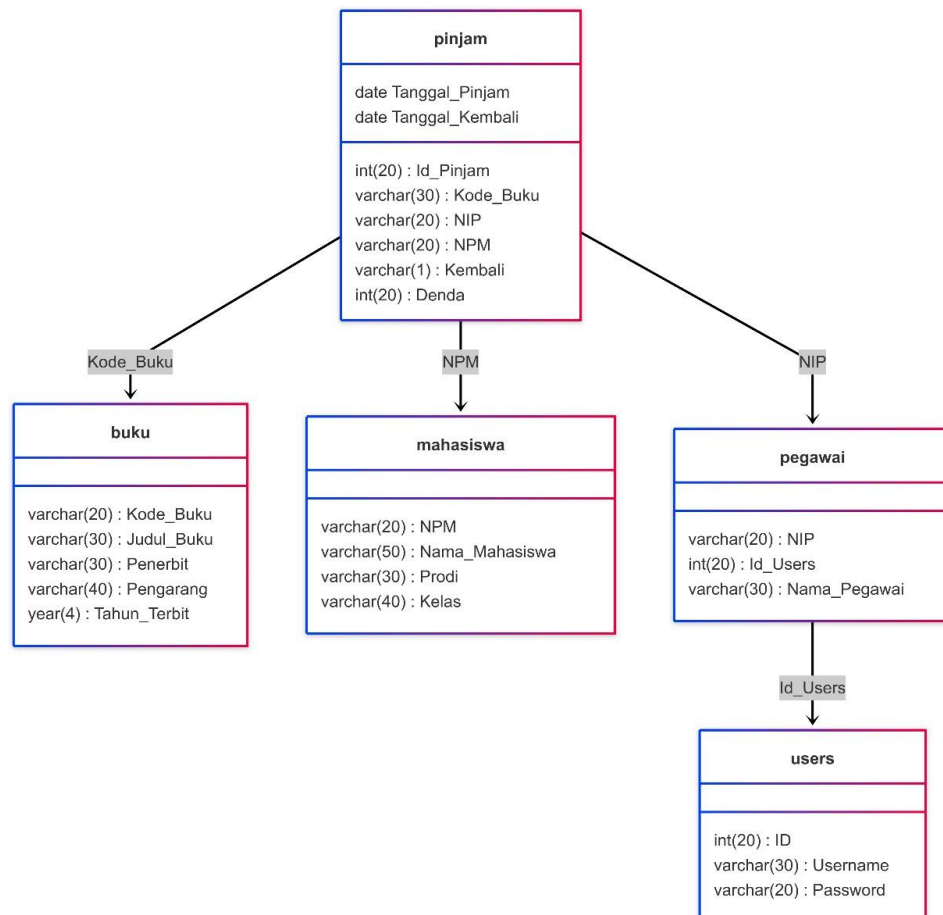
Perbedaan antara Primary Key dan Unique Key di dalam tabel ini terletak pada fungsinya. Primary Key pada kolom 'ID' digunakan sebagai identitas internal utama sistem

dan hanya boleh ada satu dalam sebuah tabel. Sementara Unique Key pada kolom 'Username' digunakan sebagai aturan bisnis untuk menjamin bahwa setiap pengguna memiliki username yang berbeda, dan sebuah tabel diperbolehkan memiliki lebih dari satu Unique Key jika diperlukan. Secara keseluruhan, struktur tabel 'users' ini sudah dirancang dengan baik dan sesuai standar keamanan sistem login. Dengan adanya kombinasi Primary Key pada 'ID' dan Unique Key pada 'Username', sistem dapat memastikan setiap akun pengguna memiliki identitas yang unik dan tidak ada duplikasi username. Hal ini sangat penting untuk menjaga integritas data dan menghindari kesalahan saat proses otentikasi pengguna dalam sistem perpustakaan digital.

2.2. Unified Modelling Language (UML)

1. CLASS DIAGRAM

a. Class Diagram Umum

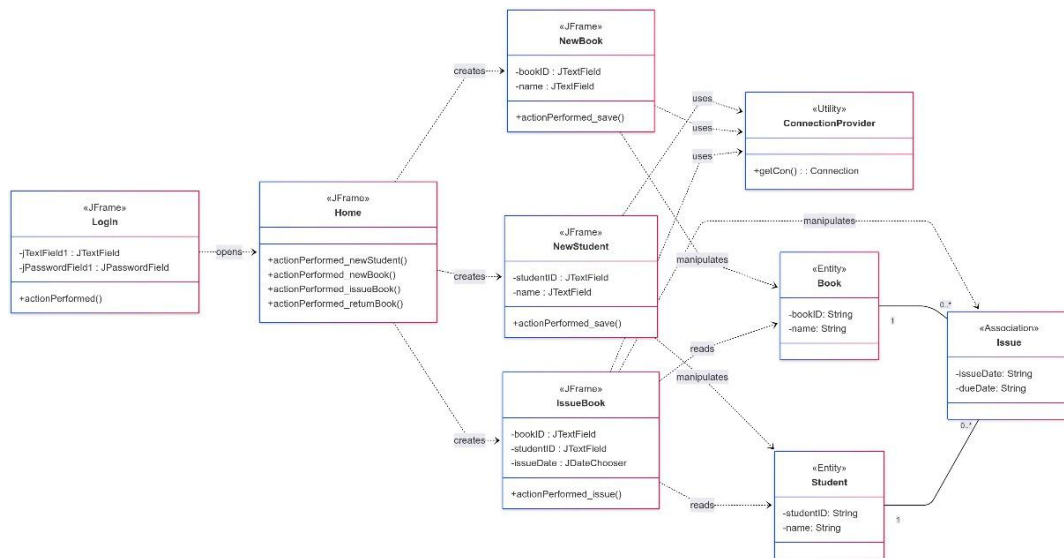


Class diagram ini digunakan untuk merepresentasikan struktur statis dari sistem, yaitu kelas-kelas beserta atribut dan relasi antar kelas. Beberapa kelas utama dalam sistem DigiPus antara lain:

- **User:** menyimpan data login seperti id, username, password, dan role.
- **Mahasiswa:** memiliki atribut seperti NIM, nama, jurusan, dan berelasi dengan kelas Peminjaman.
- **Buku:** atribut mencakup id_buku, judul, pengarang, kategori, dan stok.
- **Peminjaman:** relasi antara Mahasiswa dan Buku, mencatat tanggal peminjaman dan batas waktu.
- **Pengembalian:** menyimpan informasi pengembalian buku, termasuk tanggal kembali dan denda.

Relasi antar kelas meliputi asosiasi (seperti Mahasiswa → Peminjaman), agregasi, dan komposisi. Diagram ini penting untuk desain database dan struktur program.

b. Class Diagram OOP



Versi *Object-Oriented* dari class diagram menambahkan method atau operasi yang dapat dilakukan oleh masing-masing kelas. Dengan pendekatan ini, diagram menjadi acuan dalam proses pemrograman berorientasi objek.

2. ACTIVITY DIAGRAM

a. Manajemen User

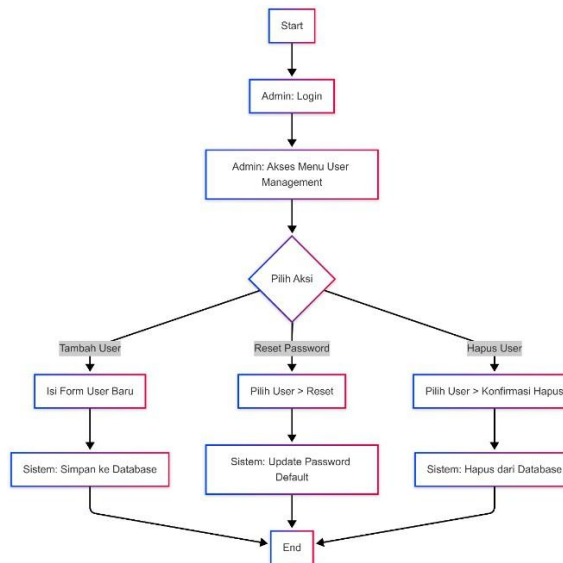
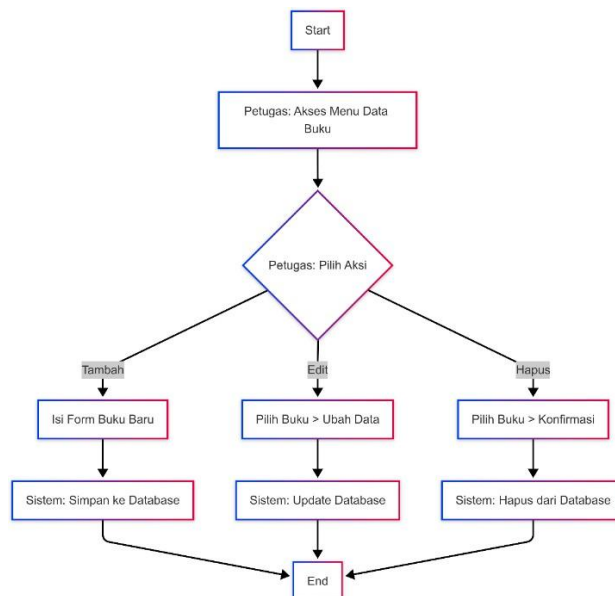


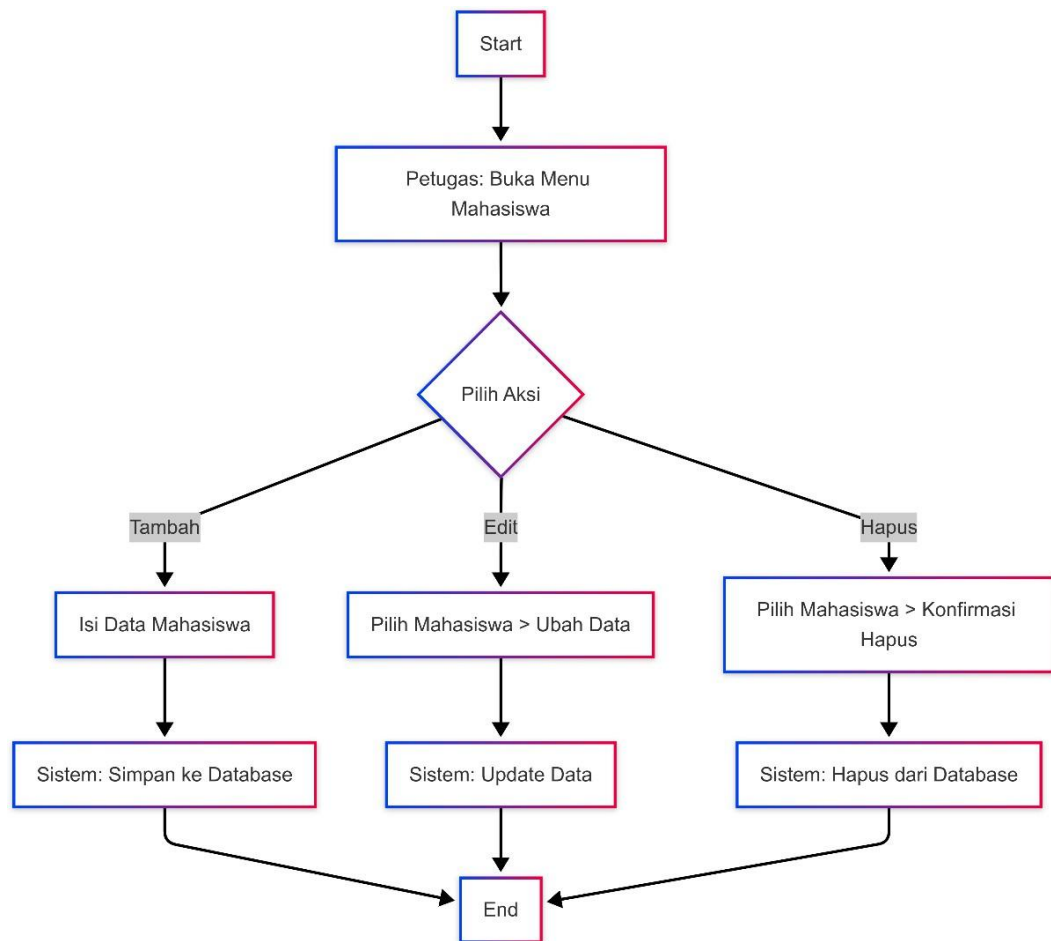
Diagram ini menggambarkan alur kegiatan dalam pengelolaan data pengguna oleh admin. Proses dimulai dari pemilihan menu “Manajemen User”, dilanjutkan dengan input data user (seperti username, password, dan role). Sistem kemudian memvalidasi data tersebut. Jika valid, data disimpan dalam database; jika tidak valid, pengguna akan diarahkan untuk memperbaiki input.

b. Mengelola Data Buku



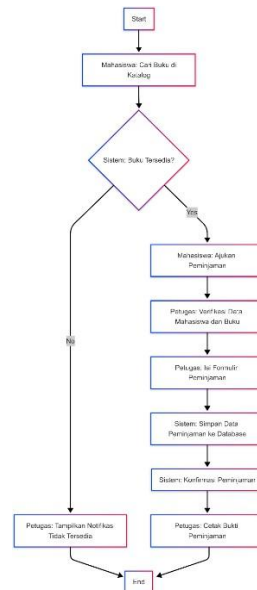
Menjelaskan langkah-langkah yang dilakukan saat petugas mengelola data buku, baik menambahkan buku baru, mengedit data buku lama, maupun menghapus buku. Alur dimulai dari login, lalu memilih menu buku, memilih aksi yang diinginkan, memasukkan atau mengubah data, hingga sistem menyimpan data ke database.

c. Mengelola Data Mahasiswa



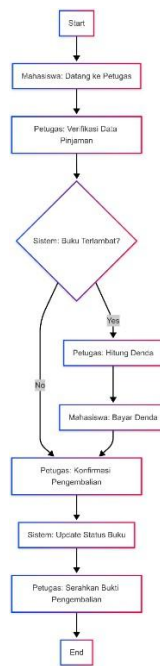
Menggambarkan proses input, update, dan penghapusan data mahasiswa oleh admin. Proses mencakup validasi data seperti NIM, nama, dan jurusan. Jika data telah diverifikasi, maka akan disimpan atau diperbarui dalam sistem.

d. Peminjaman Buku

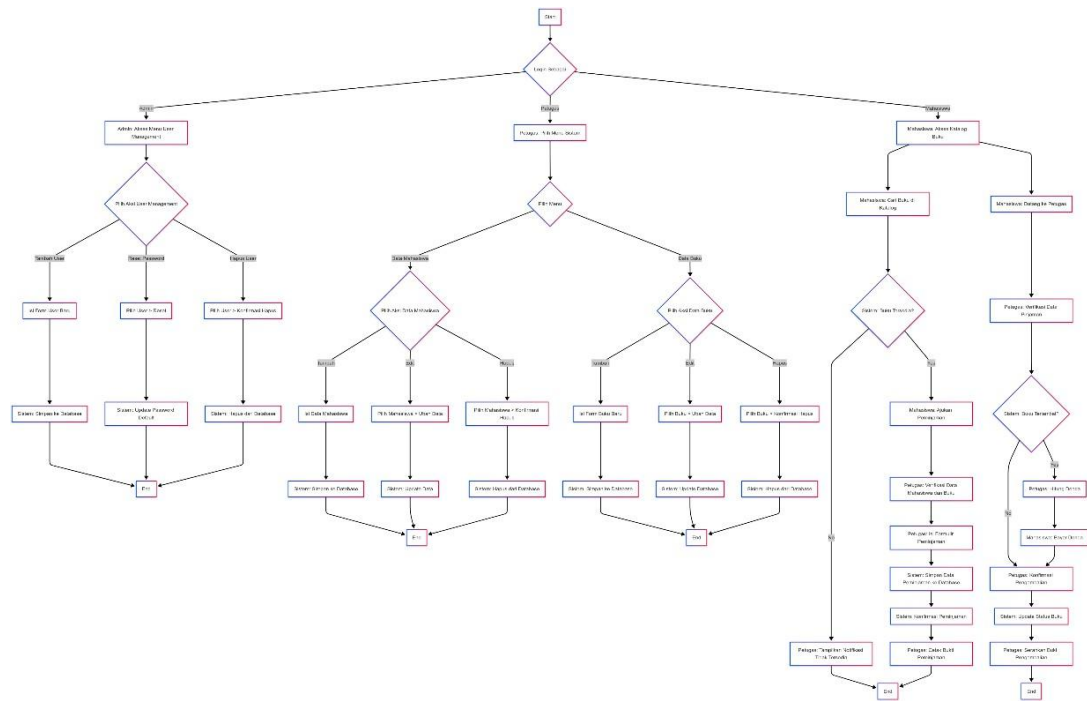


Aktivitas peminjaman dimulai dari mahasiswa memilih buku yang diinginkan, kemudian sistem mengecek apakah stok tersedia. Jika tersedia, sistem mencatat data peminjaman dan mengurangi jumlah stok buku. Jika tidak tersedia, proses dihentikan dan notifikasi diberikan ke user.

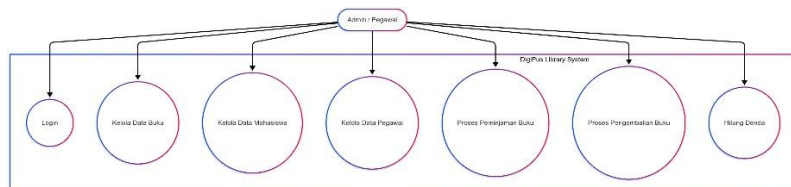
e. Pengembalian Buku



f. Diagram Lengkap



3. USE CASE DIAGRAM



17

dapat diakses oleh setiap jenis pengguna serta batasan sistem. Pada sistem DigiPus, aktor utama yang terlibat meliputi:

- **Admin/Petugas:** memiliki hak akses penuh untuk mengelola data buku, mahasiswa, dan proses peminjaman/pengembalian.
- **Mahasiswa:** pengguna yang dapat mencari buku, meminjam, dan mengembalikan buku.

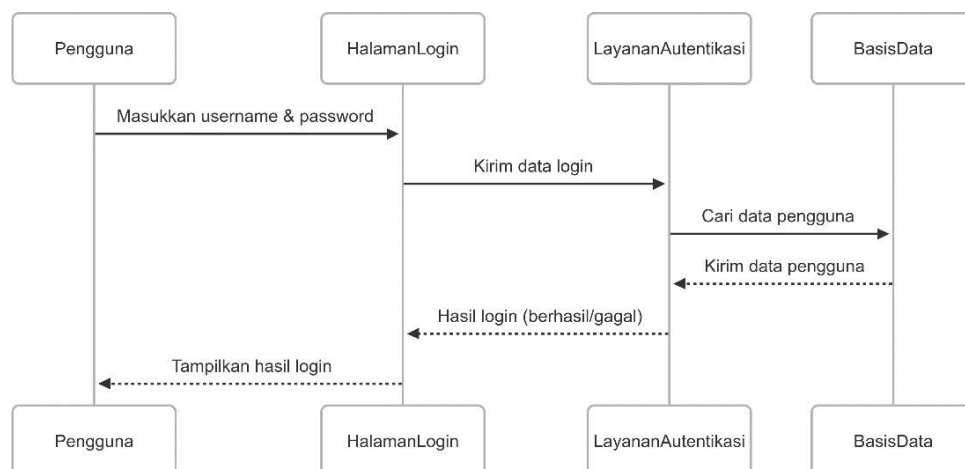
Use case utama dalam sistem meliputi:

- Autentikasi pengguna (login dan logout)
- Manajemen data buku (tambah, ubah, hapus)
- Manajemen data mahasiswa
- Peminjaman dan pengembalian buku
- Pencarian buku

Diagram ini memudahkan stakeholder memahami fungsi sistem secara menyeluruh sebelum implementasi teknis dilakukan.

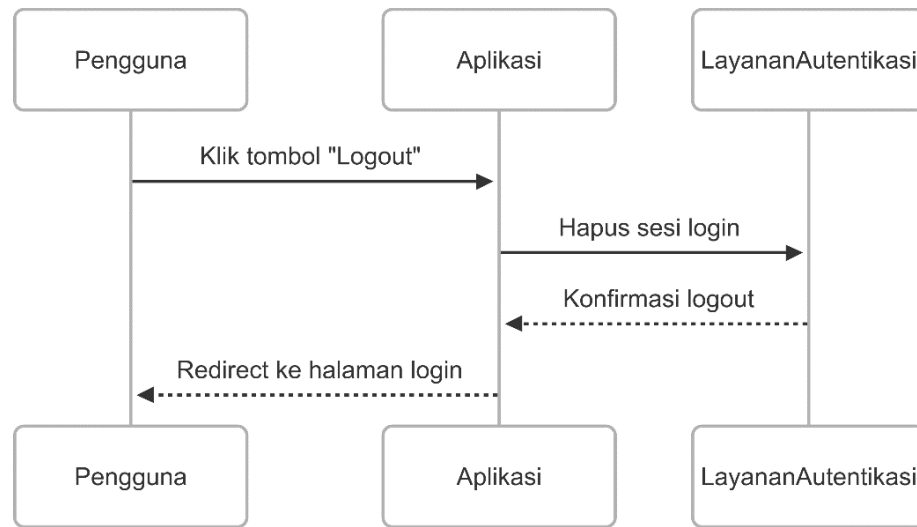
4. SEQUENCE DIAGRAM

a. Login User



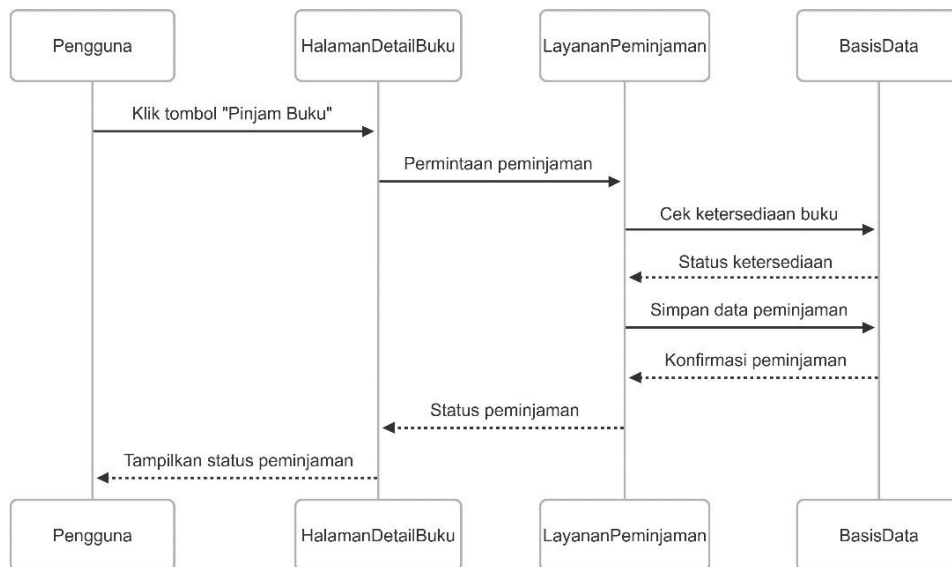
Menggambarkan rangkaian interaksi antara user dan sistem saat proses login terjadi. Proses dimulai dari user menginput username dan password. Sistem kemudian memverifikasi kredensial tersebut terhadap database. Jika sesuai, akses diberikan; jika tidak, sistem menampilkan pesan kesalahan.

b. Logout User



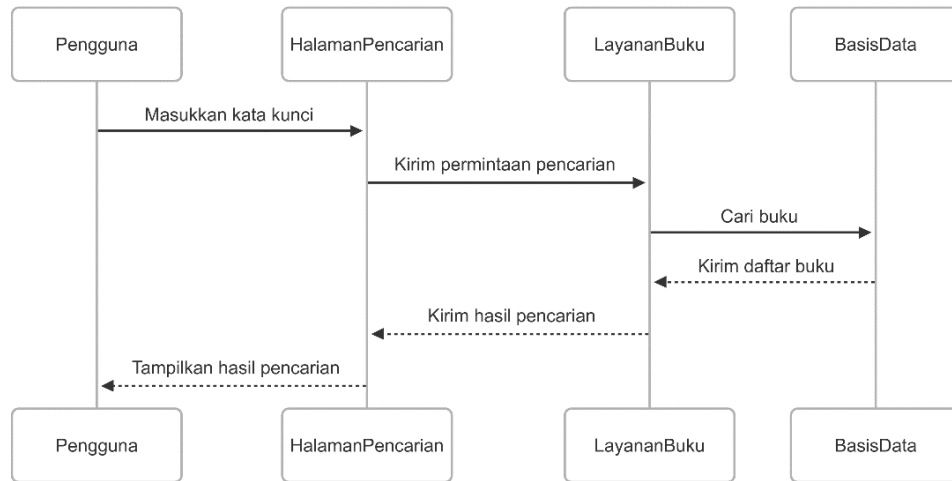
Menjelaskan interaksi ketika user keluar dari sistem. Setelah user menekan tombol logout, sistem akan menghapus sesi pengguna dan mengembalikan tampilan ke halaman login utama.

c. Peminjaman Buku



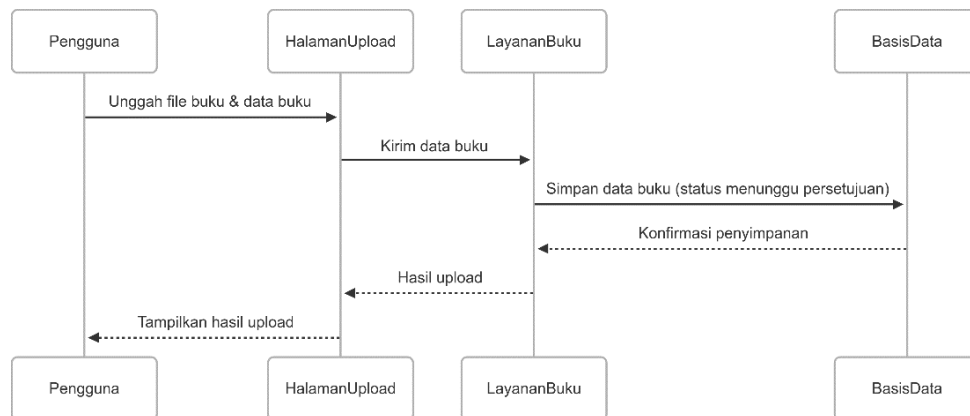
Menunjukkan bagaimana user (mahasiswa) melakukan peminjaman. Dimulai dari pencarian buku, pengecekan ketersediaan, pengisian formulir peminjaman, hingga proses penyimpanan data peminjaman dan update stok buku.

d. Pencarian Buku



Proses ini dimulai ketika user memasukkan kata kunci pada kolom pencarian. Sistem melakukan query terhadap database dan menampilkan daftar buku yang sesuai dengan kata kunci tersebut.

e. Tambah Buku



Memperlihatkan langkah-langkah saat admin menambahkan buku baru ke dalam sistem. Admin memasukkan detail buku (judul, penulis, penerbit, jumlah stok), sistem memvalidasi input, dan jika benar, menyimpan data ke database.

f. Pengembalian Buku

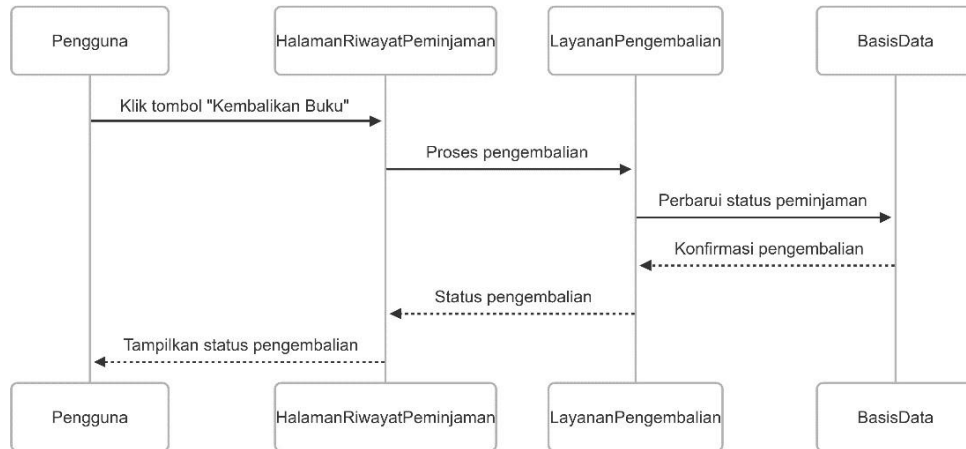
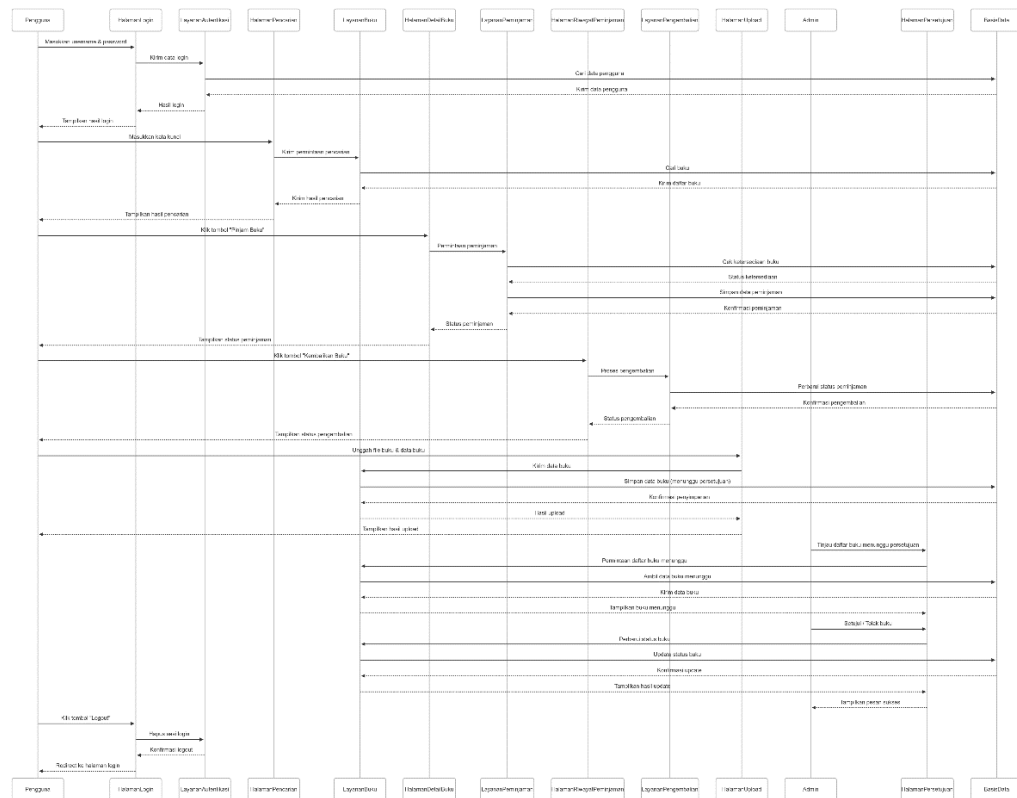


Diagram ini menunjukkan interaksi saat user melakukan pengembalian buku. Sistem mengecek status peminjaman, menghitung denda (jika ada), dan memperbarui status serta stok buku di database.

g. Diagram Lengkap



Merangkum seluruh interaksi penting dalam sistem mulai dari login, pengelolaan data, peminjaman, pengembalian, dan logout. Diagram ini memberikan gambaran alur komunikasi antar komponen sistem secara menyeluruh.

2.3 Implementasi Kode Program dan Pengujian

Implementasi kode program mengacu pada proses pengembangan atau penulisan kode sumber dari sebuah sistem atau aplikasi berdasarkan desain yang telah dibuat sebelumnya. Implementasi ini melibatkan penerjemahan kebutuhan dan spesifikasi sistem ke dalam bahasa pemrograman sehingga menghasilkan perangkat lunak yang dapat dijalankan dan digunakan oleh pengguna.

1. Implementasi Kode Program

1) Login

```
import java.sql.*;
import Project.ConnectionProvider;
import javax.swing.JOptionPane;

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */

/**
 *
 * @author MyBook Nype AMD
 */

import javax.swing.JOptionPane;

public class login extends javax.swing.JFrame {

    /**
     * Creates new form login
     */
    public login() {
        initComponents();
    }

    private void jButtonActionPerformed(java.awt.event.ActionEvent evt) {
        String username = jTextField1.getText();
        String password = new String(jPasswordField1.getPassword());

        // Cek apakah username atau password kosong
        if (username.isEmpty() || password.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Username dan Password harus diisi.");
            return; // Berhenti eksekusi jika kosong
        }

        try (Connection con = ConnectionProvider.getConnection()) {
            // 1. Siapkan query sql yang akan menggunakan PreparedStatement
            String sql = "SELECT * FROM users WHERE Username = ? AND Password = ?";

            // 2. Det parameter query dengan input dari user
            PreparedStatement pst = con.prepareStatement(sql);
            pst.setString(1, username);
            pst.setString(2, password);

            // 3. Eksekusi query
            try (ResultSet rs = pst.executeQuery()) {
                // 4. Cek apakah ada hasil yang cocok
                if (rs.next()) {
                    // Jika ada, berarti login berhasil
                    setVisible(false);
                    new home().setVisible(true);
                } else {
                    // Jika tidak ada, berarti username atau password salah
                    JOptionPane.showMessageDialog(null, "Username atau Password salah.");
                }
            }
        } catch (Exception e) {
            // Tangani jika ada error koneksi atau error lainnya
            JOptionPane.showMessageDialog(null, "Terjadi kesalahan pada database: " + e.getMessage());
            e.printStackTrace();
        }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        System.exit(0);
    }

    private void jPasswordField1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new login().setVisible(true);
            }
        });

        // Variables declaration - do not modify
        private javax.swing.JButton jButton1;
        private javax.swing.JButton jButton2;
        private javax.swing.JLabel jLabel1;
        private javax.swing.JLabel jLabel2;
        private javax.swing.JLabel jLabel3;
        private javax.swing.JPanel jPanel1;
        private javax.swing.JPanel jPanel2;
        private javax.swing.JPasswordField jPasswordField1;
        private javax.swing.JTextField jTextField1;
        // End of variables declaration
    }
}
```

Kode program login.java ini berfungsi sebagai gerbang utama atau jendela otentikasi untuk seluruh sistem manajemen perpustakaan. Saat aplikasi pertama kali dijalankan, JFrame ini akan muncul, menampilkan antarmuka grafis yang meminta pengguna memasukkan Username dan Password pada field yang telah disediakan

2) Home

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO and your handling code here:  
    new newweek().setVisible(true);  
}  
  
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO and your handling code here:  
}  
  
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO and your handling code here:  
    new issueBook().setVisible(true);  
}  
  
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO and your handling code here:  
    setVisible(false);  
    new login().setVisible(true);  
}  
  
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO and your handling code here:  
    new Statistics().setVisible(true);  
}
```

```

139 }
140
141 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
142     // TODO add your handling code here:
143     new NewStudent().setVisible(true);
144 }
145
146 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
147     // TODO add your handling code here:
148     new Page401().setVisible(true);
149 }
150
151 /**
152  * Spawns args the command (line arguments)
153  */
154 public static void main(String args[]) {
155     // Get the Window Code and Feel.
156     LookAndFeelSettingCode(Optional);
157
158     /* Create and display the form */
159     java.awt.EventQueue.invokeLater(new Runnable() {
160         public void run() {
161             new Home().setVisible(true);
162         }
163     });
164 }

```

```

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton4;
private javax.swing.JButton jButton5;
private javax.swing.JButton jButton6;
private javax.swing.JButton jButton7;
private javax.swing.JLabel jLabel1;
// End of variables declaration
}

```

Kode program home.java ini berfungsi sebagai menu utama atau dasbor navigasi pusat untuk keseluruhan aplikasi sistem manajemen perpustakaan. Setelah pengguna berhasil login, jendela ini akan muncul dan secara otomatis memaksimalkan ukurannya untuk memenuhi seluruh layar, menampilkan antarmuka yang terdiri dari beberapa tombol di atas sebuah gambar latar belakang. Setiap tombol pada jendela ini dirancang sebagai pintu gerbang ke modul fungsionalitas yang berbeda; misalnya, menekan tombol "Mahasiswa" akan membuka jendela manajemen data mahasiswa (newStudent), tombol "Buku" akan membuka modul manajemen buku (newBook), tombol "Pegawai" membuka modul manajemen pegawai (pegawai), dan begitu pula untuk "Peminjaman" (issueBook) serta "Statistics" (statistics). Selain itu, terdapat tombol "Logout" yang memungkinkan pengguna untuk kembali ke layar login tanpa harus menutup aplikasi sepenuhnya. Perlu dicatat bahwa berdasarkan kode yang diberikan, tombol "Pengembalian" sudah ada secara visual namun belum memiliki fungsionalitas yang diimplementasikan. Secara keseluruhan, kelas home ini bertindak sebagai pusat kendali yang memudahkan pengguna untuk mengakses berbagai fitur utama sistem.

3) New Student

```

import java.sql.*;
import Project.ConnectionProvider;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame to edit this template
 */
package com.mycompany.newstudent;

import javax.swing.*;

public class newStudent extends javax.swing.JFrame {

    /**
     * Creates new form newStudent
     */
    public newStudent() {
        initComponents();
        buatDataTabel(); // Panggil metode ini

        // Atur kondisi awal tombol
        btnSimpan.setEnabled(false);
        btnKirim.setEnabled(false);
    }

    private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jTextField2 = new javax.swing.JTextField();
        jTextField3 = new javax.swing.JTextField();
        jTextField4 = new javax.swing.JTextField();
        btnKirim = new javax.swing.JButton();
        btnSimpan = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTable1 = new javax.swing.JTable();
        btnMahasiswa = new javax.swing.JButton();
        btnPeminjaman = new javax.swing.JButton();
        btnBuku = new javax.swing.JButton();
        btnPegawai = new javax.swing.JButton();
        btnLogout = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setLocation(new java.awt.Point(325, 125));
        setUndecorated(true);
        getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

        jLabel1.setText("ID");
        getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteLayout(160, 200, 30, 1));
    }
}

```

```

jTextField1.setText("Nama");
getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(160, 240, -1, -1));

jTextField1.setText("NPM");
getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(160, 280, 1, 1));

jLabel5.setText("Kelas");
getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(160, 320, -1, -1));

jTextField1.setBackground(new java.awt.Color(204, 255, 255));
getContentPane().add(jTextField1, new org.netbeans.lib.awtextra.AbsoluteConstraints(270, 200, 204, -1));

jTextField12.setBackground(new java.awt.Color(204, 255, 255));
jTextField12.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jTextField1ActionPerformed(evt);
    }
});

getContentPane().add(jTextField12, new org.netbeans.lib.awtextra.AbsoluteConstraints(270, 240, 204, -1));

jTextField13.setBackground(new java.awt.Color(204, 255, 255));
jTextField13.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jTextField1ActionPerformed(evt);
    }
});
}
}

```

```

getContentPane().add(jTextField13, new org.netbeans.lib.awtextra.AbsoluteConstraints(270, 280, 204, -1));

jTextField14.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jTextField1ActionPerformed(evt);
    }
});

getContentPane().add(jTextField14, new org.netbeans.lib.awtextra.AbsoluteConstraints(270, 320, 204, -1));

btnTambah.setText(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
btnTambah.setText("Tambah");
btnTambah.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnTambahActionPerformed(evt);
    }
});

getContentPane().add(btnTambah, new org.netbeans.lib.awtextra.AbsoluteConstraints(270, 360, -1, -1));

btnTutup.setText(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
btnTutup.setText("Tutup");
btnTutup.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnTutupActionPerformed(evt);
    }
});

getContentPane().add(btnTutup, new org.netbeans.lib.awtextra.AbsoluteConstraints(490, 360, -1, -1));
}
}

```

```

tabelMahasiswa.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {
        {null, null, null, null},
        {null, null, null, null},
        {null, null, null, null},
        {null, null, null, null},
        {null, null, null, null}
    },
    new String [] {
        "Title 1", "Title 2", "Title 3", "Title 4"
    }
));

tabelMahasiswa.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        tabelMahasiswaMouseClicked(evt);
    }
});

jScrollPane1.setViewportView(tabelMahasiswa);

getContentPane().add(jScrollPane1, new org.netbeans.lib.awtextra.AbsoluteConstraints(0, 700, 160, 160));

btnSimpan.setText("Simpan Mahasiswa");
btnSimpan.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSimpanActionPerformed(evt);
    }
});
}
}

```

```

}

jLabel1.setText("NPM");
getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(170, 400, -1, -1));

btnSimpan.setText("Simpan");
btnSimpan.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSimpanActionPerformed(evt);
    }
});

getContentPane().add(btnSimpan, new org.netbeans.lib.awtextra.AbsoluteConstraints(130, 360, -1, -1));

btnDetail.setText("Detail Form");
btnDetail.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnDetailActionPerformed(evt);
    }
});

getContentPane().add(btnDetail, new org.netbeans.lib.awtextra.AbsoluteConstraints(420, 400, 1, -1));

btnExit.setText("Exit");
}
}

```

```

private void muatDataTabel() {
    DefaultTableModel model = (DefaultTableModel) tabelMahasiswa.getModel();
    model.setColumnIdentifiers(new Object[]{"NPM", "Nama Mahasiswa", "Prodi", "Kelas"});
    model.setRowCount(0); // Mengosongkan tabel

    try (Connection con = ConnectionProvider.getConnection();
        Statement st = con.createStatement();
        ResultSet rs = st.executeQuery("SELECT * FROM mahasiswa ORDER BY Nama_Mahasiswa")) {

        while (rs.next()) {
            model.addRow(new Object[]{
                rs.getString("NPM"),
                rs.getString("Nama_Mahasiswa"),
                rs.getString("Prodi"),
                rs.getString("Kelas")
            });
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal memuat data mahasiswa: " + e.getMessage());
    }
}

```

```

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void jTextField2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void jTextField4ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void btnTutupActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dispose();
}

private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String npm = jTextField1.getText();
    String nama = jTextField2.getText();
    String prodi = jTextField3.getText();
    String kelas = jTextField4.getText();
}

```

```

try {
    Connection con = ConnectionProvider.getConnection();
    String sql = "INSERT INTO mahasiswa (NPM, Nama Mahasiswa, Prodi, Kelas) VALUES (?, ?, ?, ?)";
    PreparedStatement pst = con.prepareStatement(sql);
    pst.setString(1, npm);
    pst.setString(2, nama);
    pst.setString(3, prodi);
    pst.setString(4, kelas);
    pst.executeUpdate();

    JOptionPane.showMessageDialog(null, "Mahasiswa baru berhasil ditambahkan");

    // Ganti logika lama dengan yang baru
    muatDataTabel();
    btnDetail.doClick();
} catch (SQLException e) {
    JOptionPane.showMessageDialog(null, "Gagal menyimpan NPM " + npm + " sudah terdaftar.");
} catch (Exception e) {
    JOptionPane.showMessageDialog(null, "Terjadi kesalahan: " + e.getMessage());
    e.printStackTrace();
}
}
}

```

```

private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String npm = jTextField1.getText();
    String nama = jTextField2.getText();
    String prodi = jTextField3.getText();
    String kelas = jTextField4.getText();

    String sql = "UPDATE mahasiswa SET Nama_Mahasiswa = ?, Prodi = ?, Kelas = ? WHERE NPM = ?";
    try (Connection con = ConnectionProvider.getConnection();
        PreparedStatement pst = con.prepareStatement(sql)) {

        pst.setString(1, nama);
        pst.setString(2, prodi);
        pst.setString(3, kelas);
        pst.setString(4, npm); // WHERE clause

        pst.executeUpdate();
        JOptionPane.showMessageDialog(this, "Data mahasiswa berhasil diperbarui");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal memperbarui data: " + e.getMessage());
    }

    muatDataTabel();
    btnDetail.doClick();
}
}

```

```

btnSimpan.setEnabled(false);
btnSimpan.setEnabled(true);
btnTambah.setEnabled(false);
jTextField1.setEditable(false); // NPM (Primary Key) tidak boleh diubah

private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String npm = jTextField1.getText();
    int a = JOptionPane.showMessageDialog(this, "Ada yang ingin menghapus mahasiswa dengan NPM " + npm + "?");

    if (a == JOptionPane.YES_OPTION) {
        try (Connection con = ConnectionProvider.getConnection();
            PreparedStatement pst = con.prepareStatement("DELETE FROM mahasiswa WHERE NPM = ?")) {
            pst.setString(1, npm);
            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Data berhasil dihapus");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, "Gagal menghapus data: " + e.getMessage());
        }

        muatDataTabel();
        btnDetail.doClick();
    }
}
}

```

```

public static void main(String args[]) {
    // Set the window look and feel
    //Editor: Old default state "collapsed" does not look and feel getting code (optional)
    // To Windows (introduced in Java SE 6 is not available, stay with the default look and feel).
    // For details see http://download.oracle.com/javase/6/tutorial/lookandfeel/plaf.html

    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Windows".equals(info.getClassName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
}
}

```

```
java.awt.EventQueue.invokeLater(new Runnable() {  
    public void run() {  
        new HelloWorld().setVisible(true);  
    }  
});  
  
// Variable declaration - do not modify  
private javax.swing.JButton jButton1;  
private javax.swing.JButton jButton2;  
private javax.swing.JButton jButton3;  
private javax.swing.JButton jButton4;  
private javax.swing.JLabel jLabel1;  
private javax.swing.JLabel jLabel2;  
private javax.swing.JLabel jLabel3;  
private javax.swing.JLabel jLabel4;  
private javax.swing.JScrollPane jScrollPane1;  
private javax.swing.JTextField jTextField1;  
private javax.swing.JTextField jTextField2;  
private javax.swing.JTextField jTextField3;  
private javax.swing.JTextField jTextField4;  
  
// End of variable declaration
```

Kode program `newStudent.java` ini merupakan modul mandiri yang dirancang untuk melakukan manajemen data mahasiswa secara lengkap, mencakup fungsi CRUD (Create, Read, Update, Delete). Saat jendela ini dibuka, ia secara otomatis menjalankan `method` `muatDataTabel` untuk mengambil dan menampilkan seluruh data mahasiswa dari database ke dalam sebuah `JTable` (fungsi `Read`). Pengguna dapat mengklik salah satu baris pada tabel, yang akan memicu data mahasiswa tersebut (NPM, Nama, Prodi, dan Kelas) untuk dimuat ke dalam form input yang tersedia, sekaligus mengaktifkan mode edit dengan menyalakan tombol "Simpan Perubahan" dan "Hapus". Dari form ini, pengguna dapat membuat mahasiswa baru menggunakan tombol "Tambah" yang akan mengeksekusi perintah `INSERT` ke database (fungsi `Create`), memperbarui data mahasiswa yang dipilih melalui tombol "Simpan Perubahan" (fungsi `Update`), atau menghapus data setelah konfirmasi melalui tombol "Hapus" (fungsi `Delete`). Untuk meningkatkan pengalaman pengguna, modul ini juga dilengkapi dengan tombol "Bersihkan Form" untuk mengosongkan semua field input dan mengembalikan form ke kondisi awal, serta penanganan error spesifik untuk mencegah duplikasi NPM.

4) New Book

```
import java.sql.*;
import Project.ConnectionProvider;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit
 */
/*
 *
 * @author MyBook Hype AMD
 */
public class newBook extends javax.swing.JFrame {

    /**
     * Creates new form newBook
     */
    public newBook() {
        initComponents();
        muatDataTabel(); // Memuat data saat form pertama kali dibuka

        // Atur kondisi awal tombol
        btnSimpan.setEnabled(false);
        btnHapus.setEnabled(false);
    }
}
```

```

ProgressWarnings("Success")
Generated Code
}

private void mmatDataTable1 () {
DefaultTableModel model = (DefaultTableModel) tabelBuku.getModel();
model.setColumnIdentifiers(new Object[] { "Kode Buku", "Judul", "Penerbit", "Pengerang", "Tahun Terbit" });
model.setRowCount(0);

try {
Connection con = ConnectionProvider.getConnection();
Statement st = con.createStatement();
ResultSet rs = st.executeQuery("SELECT * FROM buku ORDER BY Judul_Buku");

while (rs.next()) {
model.addRow(new Object[] {
rs.getString("Kode_Buku"),
rs.getString("Judul_Buku"),
rs.getString("Penerbit"),
rs.getString("Pengerang"),
rs.getString("Tahun_Terbit")
});
}

} catch (Exception e) {
JOptionPane.showMessageDialog(this, "Gagal memuat data buku: " + e.getMessage());
}
}

private void txtField3ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
}
}

```



```

private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dispose();
}

private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String kode_buku = jTextField1.getText();
    String judul_buku = jTextField2.getText();
    String penerbit = jTextField3.getText();
    String pengarang = jTextField4.getText();
    String tahun_terbit = jTextField5.getText();

    try {
        Connection con = DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/newBook", "root", "password");
        PreparedStatement pst = con.prepareStatement(
            "INSERT INTO buku (kode_buku, judul_buku, penerbit, pengarang, tahun_terbit) VALUES (?, ?, ?, ?, ?)");
        pst.setString(1, kode_buku);
        pst.setString(2, judul_buku);
        pst.setString(3, penerbit);
        pst.setString(4, pengarang);
        pst.setString(5, tahun_terbit);
        pst.executeUpdate();

        JOptionPane.showMessageDialog(null, "Buku baru berhasil ditambahkan");
        // tutup koneksi yang membuka koneksi buku
        pst.close();
        con.close();
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Gagal menambahkan buku: " + e.getMessage());
    }
}

```

```

jTextField1.setText(model.getValueAt(selectedRow, 0).toString()); // Kode Buku
jTextField2.setText(model.getValueAt(selectedRow, 1).toString()); // Judul
jTextField3.setText(model.getValueAt(selectedRow, 2).toString()); // Penerbit
jTextField4.setText(model.getValueAt(selectedRow, 3).toString()); // Pengarang
jTextField5.setText(model.getValueAt(selectedRow, 4).toString()); // Tahun Terbit

// Mengatur kondisi tombol dan field
btnSimpan.setEnabled(true);
btnHapus.setEnabled(true);
btnTambah.setEnabled(false);
jTextField1.setEditable(false); // Kode Buku (Primary Key) tidak boleh diubah
}

private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String kode_buku = jTextField1.getText();
    String judul_buku = jTextField2.getText();
    String penerbit = jTextField3.getText();
    String pengarang = jTextField4.getText();
    String tahun_terbit = jTextField5.getText();

    String sql = "UPDATE buku SET judul_buku = ?, penerbit = ?, pengarang = ?, tahun_terbit = ? WHERE kode_buku = ?";
    try {
        Connection con = DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/newBook", "root", "password");
        PreparedStatement pst = con.prepareStatement(sql);
        pst.setString(1, kode_buku);
        pst.setString(2, penerbit);
        pst.setString(3, pengarang);
        pst.setString(4, tahun_terbit);
        pst.setString(5, kode_buku);
        pst.executeUpdate();
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Gagal menghapus buku: " + e.getMessage());
    }
}

```

```

// Mengosongkan semua field
jTextField1.setText("");
jTextField2.setText("");
jTextField3.setText("");
jTextField4.setText("");
jTextField5.setText("");

// Mengembalikan kondisi tombol dan field
btnSimpan.setEnabled(false);
btnHapus.setEnabled(false);
btnTambah.setEnabled(true);
jTextField1.setEditable(true); // Kode Buku bisa diisi lagi
tabelBuku.clearSelection();
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new newBook().setVisible(true);
        }
    });
}

```

```

modelBuku();

tblBukuMouseClicked() // Menampilin data yang sudah tersimpan
} catch (SQLException e) {
    JOptionPane.showMessageDialog(null, "Gagal menampilkan data: " + e.getMessage());
}

private void tblBukuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}

private void tblBukuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}

private void tblBukuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}

private void tblBukuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}

DefaultTableModel model = (DefaultTableModel) tblBuku.getModel();

```

```

private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String kode_buku = jTextField1.getText();
    String judul_buku = jTextField2.getText();
    String penerbit = jTextField3.getText();
    String pengarang = jTextField4.getText();
    String tahun_terbit = jTextField5.getText();

    String sql = "DELETE FROM buku WHERE kode_buku = ?";
    try {
        Connection con = DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/newBook", "root", "password");
        PreparedStatement pst = con.prepareStatement(sql);
        pst.setString(1, kode_buku);
        pst.executeUpdate();

        JOptionPane.showMessageDialog(null, "Buku berhasil dihapus");
        // tutup koneksi yang membuka koneksi buku
        pst.close();
        con.close();
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Gagal menghapus buku: " + e.getMessage());
    }
}

```

```

// Variables declaration - do not modify
private javax.swing.JButton btnBersih;
private javax.swing.JButton btnHapus;
private javax.swing.JButton btnSimpan;
private javax.swing.JButton btnTambah;
private javax.swing.JButton btnTutup;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTextField jTextField1;
private javax.swing.JTextField jTextField2;
private javax.swing.JTextField jTextField3;
private javax.swing.JTextField jTextField4;
private javax.swing.JTextField jTextField5;
private javax.swing.JTextField jTextField6;
private javax.swing.JTable tblBuku;

// End of variables declaration
}

```

Program Java ini, bernama newBook, adalah sebuah jendela aplikasi yang memungkinkan petugas perpustakaan untuk mengelola data buku. Saat jendela dibuka, program akan langsung menampilkan semua buku yang sudah ada dari database ke dalam sebuah tabel di bagian atas. Di bagian bawah, ada kolom-kolom untuk memasukkan detail buku baru seperti kode buku, judul, penerbit, pengarang, dan tahun terbit, serta beberapa tombol aksi. Secara bawaan, tombol "Simpan Perubahan" dan "Hapus" akan nonaktif karena belum ada data yang dipilih atau diubah. Pengguna bisa menambahkan buku baru dengan mengisi semua kolom lalu mengklik tombol "Tambah". Setelah itu, program akan menyimpan data buku ke database. Jika pengguna mengklik salah satu baris di tabel, detail

5) Issue Book (Peminjaman Buku)

[illegible]

```

// Create and display the form
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new issueBook().setVisible(true);
    }
});
}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private com.toedter.calendar.JDateChooser jDateChooser1;
private com.toedter.calendar.JDateChooser jDateChooser2;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JTextField jTextField2;
private javax.swing.JTextField jTextField3;
private javax.swing.JTextField jTextField4;
// End of variables declaration

```

Kode program issueBook.java ini mengelola fungsionalitas untuk mencatat transaksi peminjaman buku. Antarmukanya (JFrame) menyediakan field untuk memasukkan Kode Buku, NIP pegawai yang melayani, dan NPM mahasiswa yang meminjam. Salah satu fitur utamanya adalah penggunaan komponen kalender (JDateChooser) yang interaktif; saat petugas memilih Tanggal Pinjam, program secara otomatis akan menghitung dan menampilkan Tanggal Kembali (jatuh tempo 7 hari kemudian) pada field kalender kedua. Logika terpenting ada pada tombol "Pinjam", di mana sebelum data disimpan, program akan melakukan validasi data ke database terlebih dahulu. Sistem akan memeriksa apakah NPM yang dimasukkan benar-benar ada di tabel mahasiswa dan apakah Kode_Buku ada di tabel buku. Jika salah satu data tidak ditemukan, aplikasi akan menampilkan pesan error yang mudah dimengerti (misalnya, "NPM tidak ditemukan") dan menghentikan proses. Hanya jika semua data valid, program akan melanjutkan untuk menyimpan data transaksi ke dalam tabel pinjam menggunakan PreparedStatement yang aman untuk mencegah SQL Injection.

6) Statistic (Riwayat Peminjaman dan Pengembalian Buku)

<pre> import java.sql.*; import javax.swing.*; import javax.swing.table.DefaultTableModel; // Class untuk koneksi ke database class DatabaseConnection { private static Connection conn; public static Connection getConnection() { try { conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/issue_book", "root", ""); } catch (SQLException e) { e.printStackTrace(); } return conn; } } // Class untuk Statistic class Statistic extends JFrame { private void initComponents() { // Model untuk tabel data DefaultTableModel model = new DefaultTableModel() { // Kolom-kolom yang akan ditampilkan String[] columnNames = {"ID Pegawai", "Kode Buku", "Nama Mahasiswa", "Tanggal Pinjam", "Tanggal Kembali", "Denda", "Status"}; // Menambahkan baris data ke tabel model.addRow(columnNames); }; // Koneksi ke database Connection conn = DatabaseConnection.getConnection(); // Query untuk mengambil data String sql = "SELECT p.ID_Pegawai, b.Kode_Buku, m.Nama_Mahasiswa, p.Tanggal_Pinjam, p.Tanggal_Kembali, p.Denda, p.Status FROM pinjam p JOIN buku b ON p.Kode_Buku = b.Kode_Buku JOIN mahasiswa m ON p.ID_Pegawai = m.ID_Pegawai"; try { Statement stmt = conn.createStatement(); ResultSet rs = stmt.executeQuery(sql); // Menambahkan data ke tabel while (rs.next()) { model.addRow(new Object[]{rs.getString(1), rs.getString(2), rs.getString(3), rs.getString(4), rs.getString(5), rs.getString(6), rs.getString(7)}); } } catch (SQLException e) { e.printStackTrace(); } } } </pre>	<pre> // Method untuk menghitung denda private void calculateDenda() { // Mengambil tanggal pinjam dan tanggal kembali String tanggalPinjam = tx.getText("Tanggal_Pinjam"); String tanggalKembali = tx.getText("Tanggal_Kembali"); // Menghitung selisih hari int selisihHari = tanggalKembali.getDate() - tanggalPinjam.getDate(); // Menghitung denda double denda = selisihHari * 10000; // 10.000 per hari // Menampilkan denda ke field tx.setText("Denda", String.valueOf(denda)); } // Method untuk menampilkan status private void showStatus() { // Mengambil status dari database String status = tx.getText("Status"); // Menampilkan status ke field tx.setText("Status", status); } // Method untuk menampilkan pesan error private void showError(String message) { JOptionPane.showMessageDialog(this, message, "Error", JOptionPane.ERROR_MESSAGE); } </pre>
--	--


```

}

/**
 * Creates new form statistics
 */
public statistics() {
    initComponents();
    setSize(700, 450);
    setLocation(325, 125);
    muatDataTabel();
}

/**
 * This method is called from within the constructor to initialize the form
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
Generated Code

private void formComponentShown(java.awt.event.ComponentEvent evt) {
    // TODO add your handling code here:
}

private void tombolTutupActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dispose();
}

```

```

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new statistics().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel3;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTable tabelRiwayat;
private javax.swing.JButton tombolTutup;
// End of variables declaration
}

```

```

} else {
    JOptionPane.showMessageDialog(this, "Data peminjaman tidak ditemukan lagi. Mungkin sudah diproses.");
}

} catch (Exception e) {
    JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage());
    e.printStackTrace(); // Ini membantu melihat error detail di console output
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    setVisible(false);
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new returnBook().setVisible(true);
        }
    });
}

```

Program ini menciptakan sebuah aplikasi jendela (*frame*) yang berfungsi untuk menampilkan statistik atau riwayat peminjaman dan pengembalian buku dalam sebuah sistem perpustakaan. Saat jendela ini dibuka, program akan langsung mengisi sebuah tabel dengan data yang diambil dari database. Tabel ini memiliki kolom untuk "ID Pinjam", "Judul Buku", "Nama Mahasiswa", "Tanggal Pinjam", "Tanggal Kembali", "Denda", dan "Status". Proses pengisian tabel dimulai dengan mengosongkan tabel terlebih dahulu, kemudian program akan terhubung ke database dan menjalankan *query* (permintaan data) untuk mengambil semua informasi peminjaman dan pengembalian dari tabel pinjam, buku, dan mahasiswa secara terhubung (*join*). Setiap baris data yang diambil dari database akan diproses, misalnya tanggal kembali yang kosong akan ditampilkan sebagai tanda "-", jumlah denda akan diformat dengan "Rp", dan status peminjaman ("Y" atau "N") akan diubah menjadi teks yang lebih mudah dibaca, yaitu "Sudah Kembali" atau "Masih Dipinjam". Setelah data diproses, setiap baris akan ditambahkan ke tabel. Jika ada masalah saat mengambil data dari database, sebuah pesan kesalahan akan ditampilkan. Terakhir,

program ini juga menyediakan tombol "Close" yang memungkinkan pengguna untuk menutup jendela tampilan riwayat ini.

7) ReturnBook (Pengembalian Buku)

```
import java.sql.*;
import Project.ConnectionProvider;
import javax.swing.JOptionPane;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.concurrent.TimeUnit;

// Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to edit
// Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit

/**
 * Author HyBook Nya AND
 */
public class returnBook extends javax.swing.JFrame {

    /**
     * Creates new form returnBook
     */
    public returnBook() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // Generated Code

```

```
    @Override
    public void initComponents() {
        // Initialize the database connection
        Connection con = ConnectionProvider.getConnection();
        Statement st = con.createStatement();

        // Query to fetch book details
        String sql = "SELECT * FROM pinjaman WHERE Kode_Buku = '" + Kode_Buku.getText() + "' AND NPM = '" + NPM.getText() + "'";
        ResultSet rs = st.executeQuery(sql);

        // Check if data is found
        if (rs.next()) {
            // Fetch book details
            String judul = rs.getString("Judul");
            String tanggal_kembali = rs.getString("Tanggal_Kembali");

            // Set text fields
            jTextField1.setText(judul);
            jTextField2.setText(tanggal_kembali);

            // Enable buttons
            jButton1.setEnabled(true);
            jButton2.setEnabled(true);
            jButton3.setEnabled(true);
        } else {
            JOptionPane.showMessageDialog(this, "Data not found!");
        }
    }

    // Close button action
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here
        dispose();
    }

    // Return button action
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here
        // Query to fetch book details
        String sql = "SELECT * FROM pinjaman WHERE Kode_Buku = '" + Kode_Buku.getText() + "' AND NPM = '" + NPM.getText() + "'";
        ResultSet rs = st.executeQuery(sql);

        // Check if data is found
        if (rs.next()) {
            // Fetch book details
            String judul = rs.getString("Judul");
            String tanggal_kembali = rs.getString("Tanggal_Kembali");

            // Set text fields
            jTextField1.setText(judul);
            jTextField2.setText(tanggal_kembali);

            // Enable buttons
            jButton1.setEnabled(true);
            jButton2.setEnabled(true);
            jButton3.setEnabled(true);
        } else {
            JOptionPane.showMessageDialog(this, "Data not found!");
        }
    }

```

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here
    // Close button action
    dispose();
}

// Return button action
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here
    // Query to fetch book details
    String sql = "SELECT * FROM pinjaman WHERE Kode_Buku = '" + Kode_Buku.getText() + "' AND NPM = '" + NPM.getText() + "'";
    ResultSet rs = st.executeQuery(sql);

    // Check if data is found
    if (rs.next()) {
        // Fetch book details
        String judul = rs.getString("Judul");
        String tanggal_kembali = rs.getString("Tanggal_Kembali");

        // Set text fields
        jTextField1.setText(judul);
        jTextField2.setText(tanggal_kembali);

        // Enable buttons
        jButton1.setEnabled(true);
        jButton2.setEnabled(true);
        jButton3.setEnabled(true);
    } else {
        JOptionPane.showMessageDialog(this, "Data not found!");
    }
}

```

```
// Close button action
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here
    dispose();
}

// Return button action
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here
    // Query to fetch book details
    String sql = "SELECT * FROM pinjaman WHERE Kode_Buku = '" + Kode_Buku.getText() + "' AND NPM = '" + NPM.getText() + "'";
    ResultSet rs = st.executeQuery(sql);

    // Check if data is found
    if (rs.next()) {
        // Fetch book details
        String judul = rs.getString("Judul");
        String tanggal_kembali = rs.getString("Tanggal_Kembali");

        // Set text fields
        jTextField1.setText(judul);
        jTextField2.setText(tanggal_kembali);

        // Enable buttons
        jButton1.setEnabled(true);
        jButton2.setEnabled(true);
        jButton3.setEnabled(true);
    } else {
        JOptionPane.showMessageDialog(this, "Data not found!");
    }
}

```

```

}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JTextField jTextField1;
private javax.swing.JTextField jTextField2;
private javax.swing.JTextField jTextField3;
private javax.swing.JTextField jTextField4;
// End of variables declaration
}

```

Program returnBook adalah bagian dari sistem perpustakaan yang berfungsi untuk mengelola proses pengembalian buku secara efisien. Melalui antarmuka grafis yang sederhana, petugas perpustakaan dapat mencari detail peminjaman dengan memasukkan kode buku dan NPM mahasiswa, lalu sistem akan menampilkan tanggal pinjam dan secara otomatis mengisi tanggal kembali dengan tanggal hari ini. Setelah data ditemukan, tombol "Kembalikan" akan aktif, memungkinkan sistem untuk menghitung denda berdasarkan keterlambatan pengembalian (Rp 10.000 per hari jika lebih dari 7 hari), dan kemudian

memperbarui status peminjaman di database menjadi "sudah kembali" dengan mencatat tanggal pengembalian serta denda. Program ini juga dilengkapi dengan fitur validasi untuk memastikan proses berjalan lancar dan menampilkan pesan jika data tidak ditemukan atau terjadi kesalahan, serta tombol "Close" untuk menutup jendela.

8) Pegawai

[illegible][illegible]

```

110 try {
111     // MessageLog file - target write file should be deleted first
112     try {
113         // catch exception if it
114         OptionsPane.showMessageLogToLogDir("Log messages data prepared: " + e.getMessage());
115     }
116     //
117 }
118
119 //
120 private void WriteLogValidationPerformed(java.awt.event.ActionEvent evt) {
121     // TODO add your handling code here
122 }
123
124 //
125 private void WriteValidationPerformed(java.awt.event.ActionEvent evt) {
126     // TODO add your handling code here
127 }
128
129 //
130 String sip = JOptionPane.getText();
131 String url = JOptionPane.getText(); // need to have url of target log local name variable code
132 String username = JOptionPane.getText();
133 String password = JOptionPane.getText();
134
135 try {
136     // help.loggng() // name.loggng() // username.loggng() // password.loggng() {
137         OptionsPane.showMessageLogToLogDir("Some field have error:");
138     }
139     return;
140 }
141
142 //
143
144 Connection con = null;
145 try {
146     con = ConnectionProvider.getConnection();
147     // MessageLog.loggng()
148     con.close();
149 }

```

```
// 1. Ambil ke tabel 'user' dan return id user  
long users = 0;  
  
String query = "SELECT ID FROM Users WHERE (1=?)";  
try {PreparedStatement pstmt = con.prepareStatement(query); pstmt.setString(1, users);} catch (SQLException e) {}  
pstmt.setString(1, users);  
pstmt.executeQuery(); pstmt // Di eksekusi juga, akan password limit  
pstmt.executeUpdate();  
  
// (Kembalikan hasil query ke pstmt.getResultSet())  
if (getResultSet().isEmpty())  
    user = pstmt.getLong(1);  
else  
    throw new SQLException("Data tidak ada");  
}  
  
// 2. Ambil ke tabel 'pegawai' menggunakan ID user  
String query2 = "SELECT ID FROM pegawai (ID, ID_User, Nama_Pegawai) VALUES (?, ?, ?)";  
try {PreparedStatement pstmt2 = con.prepareStatement(query2); pstmt2.setString(1, user); pstmt2.setString(2, user); pstmt2.setString(3, user);} catch (SQLException e) {}  
pstmt2.setString(1, user);  
pstmt2.setString(2, user);  
pstmt2.setString(3, user);  
pstmt2.executeQuery();
```

```

231 // kita punya beberapa commit transaksi
232 con.commit();
233 JOptionPane.showMessageDialog(rootPane, "Bergas! baru berhasil ditransaksikan!");
234
235 } catch (SQLException e) {
236     // kita ada error, roll back transaksi
237     try {
238         if (con != null) con.rollback();
239     } catch (SQLException ex) {}
240     System.err.println("Rollback gagal: " + ex.getMessage());
241
242     JOptionPane.showMessageDialog(rootPane, "Gagal membatalkan operasi: " + e.getMessage());
243 } finally {
244     // koneksi ke db mode auto commit
245     try {
246         if (con != null) {
247             con.setAutoCommit(true);
248             con.close();
249         }
250     } catch (SQLException e) {
251         System.err.println("Gagal menutup koneksi: " + e.getMessage());
252     }
253 }

```

[illegible][illegible]

```

1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int n;
8      cin >> n;
9      int a[n];
10     for (int i = 0; i < n; i++)
11     {
12         cin >> a[i];
13     }
14     int sum = 0;
15     for (int i = 0; i < n; i++)
16     {
17         sum += a[i];
18     }
19     cout << sum << endl;
20     return 0;
21 }

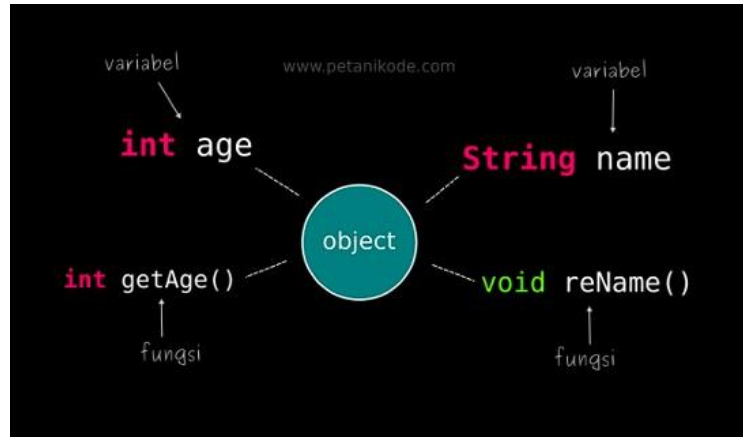
```

<pre> 3 private void btnBersihActionPerformed(java.awt.event.ActionEvent evt) { // TODO add your handling code here: txtNIP.setText(""); txtPegawai.setText(""); txtUsername.setText(""); txtPassword.setText(""); btnHapus.setEnabled(false); btnHapus.setEnabled(false); btnTambah.setEnabled(true); txtNIP.setEditable(true); tabelPegawai.clearSelection(); } 3 private void btnTutupActionPerformed(java.awt.event.ActionEvent evt) { // TODO add your handling code here: dispose(); } </pre>	<pre> // Variables Declaration - do not modify private javax.swing.JButton btnBersih; private javax.swing.JButton btnHapus; private javax.swing.JButton btnHapus; private javax.swing.JButton btnTambah; private javax.swing.JButton btnTutup; private javax.swing.JLabel jLabel5; private javax.swing.JLabel jLabel6; private javax.swing.JLabel jLabel7; private javax.swing.JLabel jLabel8; private javax.swing.JScrollPane jScrollPane1; private javax.swing.JTable tabelPegawai; private javax.swing.JTextField txtNIP; private javax.swing.JPasswordField txtPassword; private javax.swing.JTextField txtPegawai; private javax.swing.JTextField txtUsername; // End of variables declaration </pre>
---	--

Kode program pegawai.java ini merupakan sebuah modul manajemen data pegawai yang lengkap dengan fungsionalitas CRUD (Create, Read, Update, Delete) yang ditampilkan dalam satu jendela JFrame. Saat jendela pertama kali dibuka, program akan langsung menjalankan method muatDataTabel() untuk mengambil semua data pegawai dari database dengan menggabungkan tabel pegawai dan users, lalu menampilkannya dalam sebuah JTable (fungsi Read). Pengguna dapat berinteraksi dengan tabel tersebut; saat sebuah baris diklik, data lengkap pegawai yang dipilih (termasuk password) akan dimuat ke dalam form input di bawah tabel, dan tombol "Simpan Perubahan" serta "Hapus" akan menjadi aktif. Dari form ini, pengguna bisa melakukan beberapa aksi: "Tambah" akan mengambil input dari form untuk membuat data baru di tabel users dan pegawai secara bersamaan dalam sebuah transaksi database yang aman (fungsi Create); "Simpan Perubahan" akan memperbarui data pegawai yang ada berdasarkan NIP-nya (fungsi Update); dan "Hapus" akan menghapus data pegawai yang dipilih setelah mendapatkan konfirmasi dari pengguna (fungsi Delete). Selain itu, terdapat tombol utilitas seperti "Bersihkan Form" untuk mengosongkan input dan "Tutup" untuk keluar dari jendela, menjadikan modul ini sebuah sistem manajemen data yang mandiri dan interaktif.

2. Konsep OOP Beserta Penerapannya Pada Sistem Informasi Perpustakaan

OOP (Object Oriented Programming) atau dalam bahasa Indonesia dikenal dengan pemrograman berorientasi objek (PBO) merupakan sebuah paradigma atau teknik pemrograman yang berorientasi objek. Pada OOP, Fungsi dan variabel dibungkus dalam sebuah objek atau class yang dapat saling berinteraksi sehingga membentuk sebuah program.



1) Library

Library adalah kumpulan dari class, object, method seperti : `Javax.swing`, `java.sql`

```
import java.sql.*;
import Project.ConnectionProvider;
import javax.swing.JOptionPane;
```

2) Class

Class bertugas untuk mengumpulkan prosedur/fungsi dan variabel dalam satu tempat. Class merupakan blueprint dari sebuah objek atau cetakan untuk membuat objek. Class akan merepresentasikan objek yang mau dibuat. Jadi dalam membuat nama kelas harus disesuaikan dengan objek yang akan dibuat.

```
public class login extends javax.swing.JFrame {
```

3) Object

Objek adalah instance dari sebuah kelas. Objek adalah entitas konkret yang memiliki nilai untuk atribut-atribut yang didefinisikan dalam kelasnya, dan dapat melakukan perilaku yang ditentukan oleh metode-metode kelasnya. Setiap objek dari sebuah kelas akan memiliki salinan atributnya sendiri, namun berbagi metode yang sama.

```
java.awt.EventQueue.invokeLater(new Runnable() {  
    public void run() {  
        new login().setVisible(true);  
    }  
});
```

4) Atribut

Atribut merupakan bagian dari sebuah kelas yang masih berhubungan erat dari kelas tersebut. Atribut bisa juga disebut sebagai properti atau properties dari sebuah class.

```
private javax.swing.JButton jButton1;  
private javax.swing.JButton jButton2;  
private javax.swing.JButton jButton3;  
private javax.swing.JButton jButton4;  
private javax.swing.JButton jButton5;  
private javax.swing.JButton jButton6;  
private javax.swing.JLabel jLabel1;
```

5) Methods

Metode adalah blok kode yang melakukan tugas tertentu dalam sebuah kelas. Mereka mendefinisikan perilaku objek.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String kode_Buku=jTextField1.getText();  
    String judul_Buku=jTextField2.getText();  
    String penerbit=jTextField4.getText();  
    String pengarang=jTextField5.getText();  
    String tahun_Terbit=jTextField6.getText();
```

6) Constructor

konstruktor adalah metode khusus yang ada di dalam sebuah kelas. Fungsinya adalah untuk menginisialisasi objek yang baru dibuat dari kelas tersebut.

```
public newBook() {  
    initComponents();  
}
```

7) Encapsulasi

Encapsulation adalah proses menyembunyikan detail data atau implementasi dari luar dan hanya memperbolehkan akses melalui method yang telah ditentukan.

```
private javax.swing.JButton jButton1;  
private javax.swing.JButton jButton2;  
private javax.swing.JButton jButton3;  
private javax.swing.JButton jButton4;  
private javax.swing.JButton jButton5;  
private javax.swing.JButton jButton6;  
private javax.swing.JButton jButton7;  
private javax.swing.JLabel jLabel1;
```

8) Inheritance

Inheritance pada java dapat ditandai dengan menggunakan keyword `extends` seperti dibawah ini.

```
public class home extends javax.swing.JFrame {
```

```
public class login extends javax.swing.JFrame {
```

```
public class statistics extends javax.swing.JFrame {
```

```
public class returnBook extends javax.swing.JFrame {
```

```
public class pegawai extends javax.swing.JFrame {
```

```
public class issueBook extends javax.swing.JFrame {
```

```
public class newBook extends javax.swing.JFrame {
```

```
public class newStudent extends javax.swing.JFrame {
```

Yang dimana class home, login, statistics, returnBook, Pegawai, IssueBook, newBook dan newBook merupakan subclass dari class JFrame, serta mendapatkan warisan atribut dan object yang dimiliki oleh class JFrame.

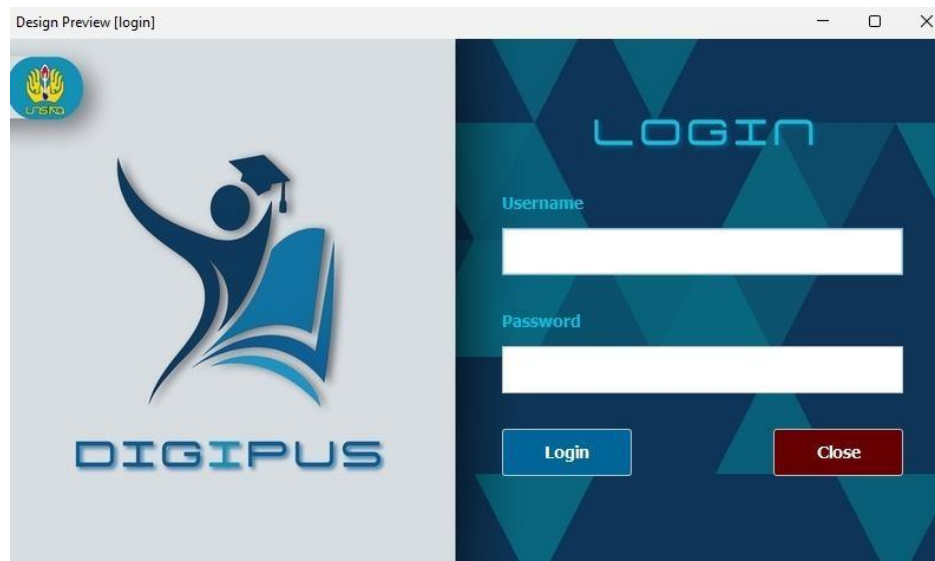
9) Polymorphism

Polymorphism adalah kemampuan objek untuk memiliki banyak bentuk yaitu method yang sama bisa berperilaku berbeda tergantung objeknya.

```
jButton7.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        jButton7ActionPerformed(evt);  
    }  
});
```

3. Fitur dan Pengujian Sistem Informasi Perpustakaan

1) Login



Halaman Home merupakan antarmuka pertama yang ditampilkan ke pengguna. Jika pengguna berhasil login ke dalam sistem maka akan langsung masuk ke halaman home. Jika gagal atau password salah akan direct akan ada pesan muncul “Incorrect Username dan Password”.

2) Home



Halaman **Home** merupakan antarmuka utama yang ditampilkan setelah pengguna berhasil login ke dalam sistem. Tampilan ini menyajikan berbagai tombol navigasi utama yang memudahkan pengguna (seperti petugas perpustakaan) untuk mengakses fitur-fitur sistem secara cepat.

a. Book

Mengarahkan ke halaman manajemen buku, seperti menambah data buku baru, melihat daftar buku, mengedit, atau menghapus informasi buku dari database perpustakaan.

b. Issue Buku

Digunakan untuk memproses pengembalian buku yang telah dipinjam oleh siswa/anggota. Sistem akan memverifikasi data peminjaman dan mengupdate status buku agar tersedia kembali.

c. Return Book

Digunakan untuk memproses pengembalian buku yang telah dipinjam oleh siswa/anggota. Sistem akan memverifikasi data peminjaman dan mengupdate status buku agar tersedia kembali.

d. New Student

Digunakan untuk memproses peminjaman buku oleh siswa. Admin akan mencari buku yang tersedia dan mencatat peminjamannya ke sistem.

e. Pegawai

Fitur ini Untuk mendukung pengelolaan pengguna sistem yang lebih baik, ditambahkan fitur "Tambah Pegawai", "Hapus Pegawai", "View Pegawai" pada halaman utama (Home Page). Fitur ini memungkinkan admin atau pengguna yang berwenang untuk mendaftarkan pegawai baru ke dalam sistem perpustakaan.

f. Statistics

Menampilkan data statistik penting yang berkaitan dengan operasional perpustakaan. Misalnya:

- Jumlah total buku yang tersedia.
- Jumlah buku yang sedang dipinjam.
- Jumlah pengembalian dalam periode tertentu.
- Jumlah anggota aktif (siswa).
- Grafik atau visualisasi tren peminjaman buku.

Statistik ini membantu petugas untuk memantau performa dan aktivitas perpustakaan secara menyeluruh.

g. Logout

Digunakan untuk keluar dari akun pengguna dan kembali ke halaman login. Menjaga keamanan sistem agar tidak diakses oleh pihak yang tidak berwenang.

3) Input buku (NewBook)

Design Preview [newBook]

Title 1	Title 2	Title 3	Title 4
---------	---------	---------	---------

Kode Buku

Judul Buku

Penerbit

Pengarang

Tahun Terbit

Tambah

Hapus

Simpan Perubahan

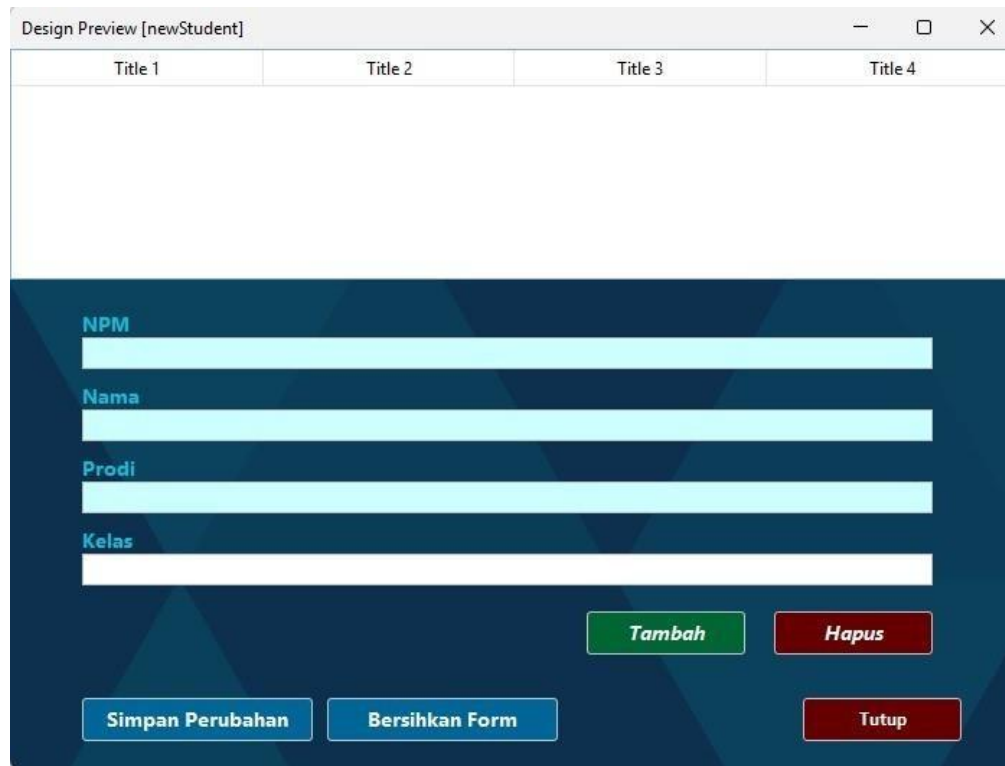
Bersihkan Form

Tutup

Digunakan untuk menambahkan data buku baru ke dalam sistem.
Fungsinya mencakup:

- Mengisi data seperti judul buku, pengarang, penerbit, tahun terbit, kategori, dan jumlah stok.
- Menyimpan informasi ke database agar buku bisa tersedia untuk dipinjam.

4) Input Mahasiswa (NewStudent)



Design Preview [newStudent]

Title 1 Title 2 Title 3 Title 4

NPM

Nama

Prodi

Kelas

Tambah Hapus

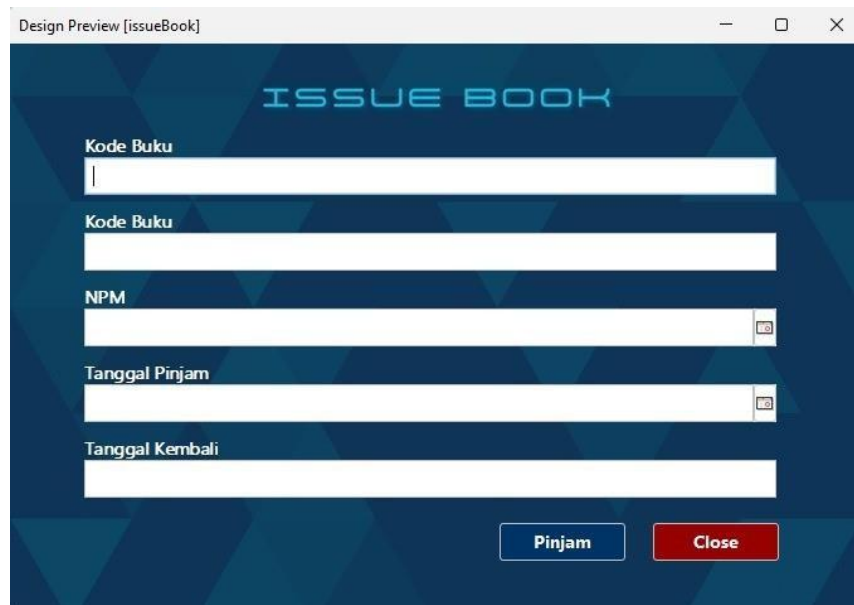
Simpan Perubahan Bersihkan Form Tutup

Digunakan untuk mendaftarkan mahasiswa atau siswa baru sebagai anggota perpustakaan.

Fungsinya mencakup:

- Mengisi data seperti nama lengkap, NIM/NPM, kelas, jurusan, dan kontak.
- Data ini digunakan untuk mencatat aktivitas peminjaman dan pengembalian buku.

5) Pinjam Buku (IssueBook)



The screenshot shows a web form titled "ISSUE BOOK" within a "Design Preview [issueBook]" window. The form has a dark blue background with a geometric pattern. It contains five input fields: "Kode Buku" (twice), "NPM", "Tanggal Pinjam", and "Tanggal Kembali". Each field has a small calendar icon on the right. At the bottom right, there are two buttons: "Pinjam" (blue) and "Close" (red).

Digunakan untuk memproses peminjaman buku oleh mahasiswa

Fungsinya mencakup:

- Memilih nama mahasiswa yang meminjam.
- Memilih buku yang akan dipinjam dari daftar buku tersedia.
- Mencatat tanggal peminjaman dan tanggal jatuh tempo pengembalian.

6) Kembalikan (ReturnBook)



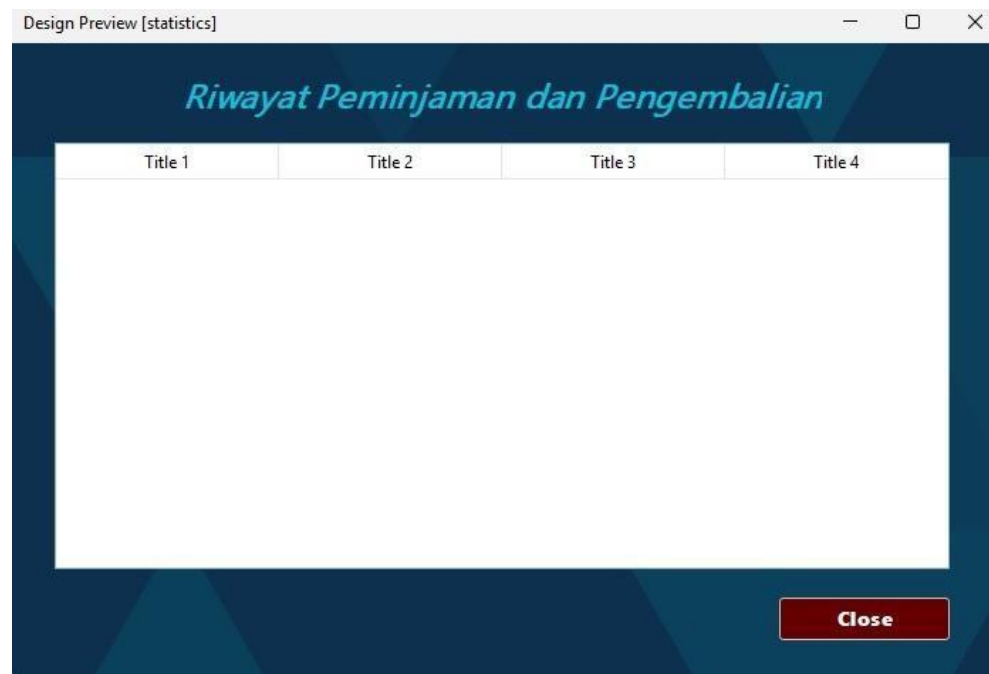
The screenshot shows a web form titled "Riwayat Peminjaman dan Pengembalian" (Borrowing and Return History). It has a dark blue background with a geometric pattern. The form contains four input fields: "Kode Buku", "NPM", "Tanggal Peminjaman", and "Tanggal Pengembalian". At the bottom right, there are three buttons: "Cari" (blue), "Kembalikan" (green), and "Close" (red).

Digunakan untuk mencatat proses pengembalian buku.

Fungsinya mencakup:

- Memasukkan data mahasiswa dan buku yang dikembalikan.
- Mengupdate status buku menjadi "tersedia".
- Menentukan apakah terlambat atau tidak, untuk perhitungan denda.

7) Cek Denda (statistic)

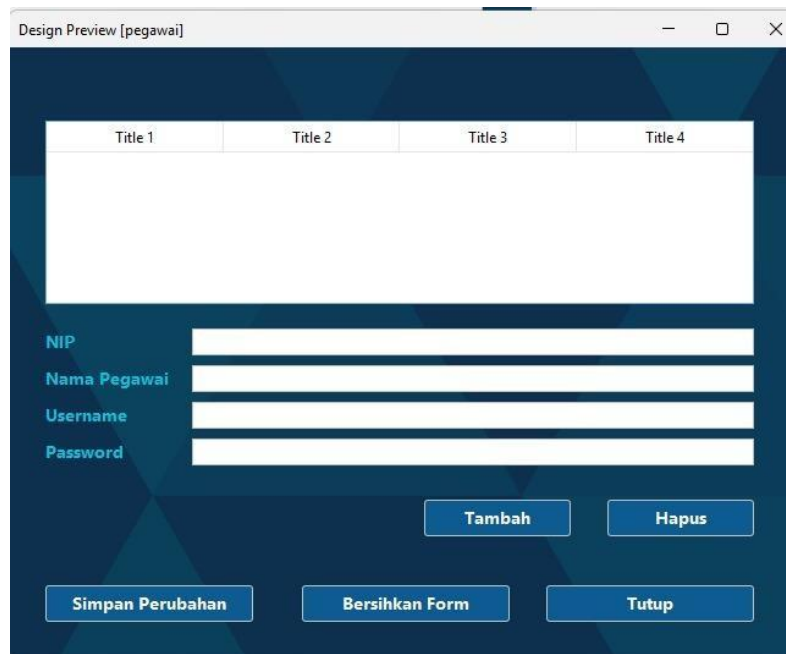


Fitur ini digunakan untuk melihat apakah mahasiswa dikenakan denda atas keterlambatan pengembalian buku.

Fungsinya mencakup:

- Menampilkan daftar peminjaman yang terlambat.
- Menghitung jumlah hari keterlambatan.
- Menampilkan nominal denda (misal: Rp1.000/hari keterlambatan).
- Bisa juga digunakan sebagai dasar pembayaran denda.

8) Pegawai



Title 1	Title 2	Title 3	Title 4

NIP

Nama Pegawai

Username

Password

Digunakan untuk menampilkan data pegawai yang sudah ada dalam bentuk tabel, serta menyediakan form untuk menambah, mengubah, atau menghapus data pegawai tersebut, termasuk informasi NIP, Nama Pegawai, Username, dan Password.

BAB III

PENUTUP

3.1 Kesimpulan

Berdasarkan perancangan dan implementasi yang telah dilakukan terhadap sistem informasi perpustakaan berbasis GUI, dapat disimpulkan bahwa sistem ini berhasil dikembangkan secara terstruktur dengan pendekatan pemrograman berorientasi objek. Sistem telah mencakup berbagai komponen penting, seperti perancangan basis data menggunakan ERD, implementasi database dengan phpMyAdmin, serta penggunaan diagram UML seperti class diagram dan activity diagram sebagai acuan pengembangan. Fitur-fitur utama seperti login, pengelolaan data buku dan mahasiswa, peminjaman dan pengembalian buku, serta perhitungan denda telah diimplementasikan dengan baik. Penggunaan Java Swing pada antarmuka pengguna memungkinkan interaksi yang lebih mudah dan responsif. Walaupun sistem telah berjalan sesuai fungsi dasarnya, masih terdapat ruang pengembangan lebih lanjut, khususnya dalam aspek keamanan, manajemen stok otomatis, penghitungan denda otomatis, serta perbaikan struktur kode melalui pola desain seperti DAO. Dengan peningkatan dan pemeliharaan berkelanjutan, sistem ini memiliki potensi besar untuk menjadi solusi digital yang andal dan efisien dalam pengelolaan perpustakaan secara menyeluruh.

3.2 Saran

Untuk pengembangan selanjutnya, sangat disarankan agar sistem informasi perpustakaan ini dilengkapi dengan peningkatan aspek keamanan, seperti penerapan enkripsi password dan kontrol akses yang lebih ketat, guna melindungi data pengguna dan mencegah potensi penyalahgunaan. Selain itu, fitur otomatisasi dalam manajemen stok buku dan perhitungan denda peminjaman sebaiknya dikembangkan agar proses operasional menjadi lebih efisien dan minim kesalahan. Dari sisi teknis, penggunaan pola desain perangkat lunak seperti DAO dapat membantu memperbaiki struktur kode sehingga sistem lebih mudah untuk dikembangkan dan dipelihara. Di samping itu, pengembangan antarmuka pengguna dengan teknologi GUI yang lebih modern dan responsif juga perlu dipertimbangkan untuk meningkatkan kenyamanan dan kemudahan penggunaan. Terakhir, pemeliharaan sistem secara berkala serta pengujian yang terus menerus sangat penting untuk memastikan sistem selalu berfungsi dengan baik dan mampu beradaptasi terhadap kebutuhan pengguna di masa mendatang.