LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	: RPL 2
Kelas	: 4IA01
Praktikum ke-	: 4
Tanggal	: 4 November 2025
Materi	: ORM
NPM	: 50422657
Nama	: Harry Mardika
Ketua Asisten	: Satya Bara Justitia
Paraf Asisten	: -
Nama Asisten	: 1.
	2.
	3.
Jumlah Lembar	: 9 Lembar

LABORATORIUM INFORMATIKA

UNIVERSITAS GUNADARMA

2025

Soal

1. Jelaskan apa yang dimaksud dengan ORM dan keuntungannya!
2. Jelaskan kode dan langkah - langkah program yang telah dibuat!

Penjelasan

---

1. ORM (Object Relational Mapping) adalah sebuah teknik atau konsep dalam pemrograman yang digunakan untuk menghubungkan objek dalam kode program dengan tabel di dalam basis data relasional secara otomatis. Dengan ORM, pengembang tidak perlu menulis perintah SQL secara langsung untuk melakukan operasi seperti insert, update, delete, atau select, karena ORM akan menerjemahkan objek dan atribut dalam bahasa pemrograman menjadi query SQL yang sesuai. Contohnya, dalam framework seperti Django (Python), Laravel (PHP), atau Hibernate (Java), ORM digunakan untuk mempermudah pengelolaan data dengan cukup memanggil metode atau properti objek tanpa harus berurusan langsung dengan sintaks SQL yang kompleks.

Keuntungan utama dari ORM adalah efisiensi dan kemudahan pengembangan. Pengembang dapat bekerja menggunakan paradigma berorientasi objek tanpa perlu memahami detail struktur database secara mendalam, sehingga kode menjadi lebih bersih, mudah dibaca, dan cepat dikembangkan. Selain itu, ORM juga meningkatkan portabilitas dan keamanan karena secara otomatis menangani parameterisasi query untuk mencegah SQL Injection serta mendukung berbagai jenis sistem basis data tanpa perlu mengubah banyak kode. ORM juga memudahkan proses pemeliharaan dan pengujian karena setiap tabel direpresentasikan sebagai kelas, yang membuat perubahan pada struktur data dapat dilakukan dengan lebih konsisten dan terpusat dalam kode aplikasi.

2.

- Overview

Program yang telah dibuat merupakan sebuah aplikasi desktop yang berfungsi untuk mengelola data mahasiswa dengan menggunakan teknologi terkini dalam pengembangan Java. Aplikasi ini dibangun dengan memanfaatkan berbagai framework dan library, yaitu Java Swing untuk membuat antarmuka grafis pengguna (GUI), Hibernate sebagai Object-Relational Mapping (ORM) untuk mengelola komunikasi antara aplikasi dan database, MySQL sebagai sistem manajemen database, serta Apache Maven sebagai build automation tool. Penggabungan teknologi-teknologi tersebut memungkinkan aplikasi ini untuk memberikan pengalaman pengguna yang intuitif dengan performa database yang optimal.

Proyek ini menerapkan pola arsitektur Model-View-Controller (MVC) yang merupakan best practice dalam pengembangan aplikasi modern. Pola ini memisahkan aplikasi menjadi tiga komponen utama yang saling berinteraksi. Pertama, Model yang bertanggung jawab mengelola data dan logika bisnis; kedua, Controller yang mengatur aliran data antara Model dan View; dan ketiga, View yang menampilkan antarmuka pengguna. Dengan menerapkan pola MVC, kode menjadi lebih terstruktur, mudah dimaintain, dan dapat dikembangkan lebih lanjut dengan lebih efisien.Arsitektur MVC

- Struktur dan Konfigurasi Proyek
  - Konfigurasi Project dengan Maven (pom.xml)

    File pom.xml merupakan file konfigurasi Maven yang sangat penting dalam proyek ini. File ini mendefinisikan identitas proyek dengan groupId "student.gunadarma.ac.id",

artifactId "Pert4_50422657", dan versi "1.0-SNAPSHOT". Dalam file ini, ditetapkan bahwa proyek menggunakan Java versi 21 sebagai target compiler release. Lebih penting lagi, pom.xml mendeklarasikan dua dependency utama yang diperlukan proyek: pertama adalah Hibernate ORM versi 6.6.0.Final yang berfungsi sebagai framework ORM untuk memetakan objek Java ke tabel database, dan kedua adalah MySQL Connector/J versi 8.0.33 yang berfungsi sebagai JDBC driver untuk memungkinkan aplikasi berkomunikasi dengan database MySQL. Melalui Maven, semua library ini akan secara otomatis diunduh dan diintegrasikan ke dalam project.

- Konfigurasi Database dengan Hibernate (hibernate.cfg.xml)

  File hibernate.cfg.xml berisi konfigurasi penting tentang bagaimana Hibernate akan terhubung ke database dan bagaimana operasi database akan dijalankan. Konfigurasi ini menspesifikkan driver JDBC yang digunakan adalah "com.mysql.cj.jdbc.Driver" yang merupakan driver MySQL terbaru untuk koneksi dengan database. URL koneksi mengarah ke "jdbc:mysql://localhost:3306/pert4_rpl2" yang berarti database bernama "pert4_rpl2" berjalan di mesin lokal (localhost) pada port 3306 (port default MySQL). Username yang digunakan adalah "root" dan tidak ada password yang diset (password kosong). Untuk mengoptimalkan performa dan mengelola koneksi database secara efisien, digunakan C3P0 connection pooling dengan konfigurasi: minimum koneksi yang tersedia adalah 5, maksimum koneksi adalah 20, timeout adalah 300 detik, dan idle test period adalah 3000 ms. Hibernate dialect yang digunakan adalah MySQLDialect agar Hibernate dapat menghasilkan SQL statement yang sesuai dengan syntax MySQL. Mode hbm2ddl.auto diset ke "update" yang berarti Hibernate akan secara otomatis membuat tabel-tabel yang diperlukan jika belum ada, atau melakukan update terhadap struktur tabel yang sudah ada jika ada perubahan. Konfigurasi ini juga mengaktifkan fitur "show_sql" untuk menampilkan semua SQL statement yang dijalankan ke standard output, yang berguna untuk debugging dan monitoring.

- Komponen Model
  - Class ModelMahasiswa

    Class ModelMahasiswa merupakan Entity class yang mewakili struktur tabel "mahasiswa" dalam database. Class ini dilengkapi dengan anotasi-anotasi JPA (Jakarta Persistence API) yang memungkinkan Hibernate untuk secara otomatis memetakan atribut-atribut dalam class ini ke kolom-kolom dalam tabel database. Atribut pertama adalah "id" yang berfungsi sebagai primary key dengan strategi GenerationType.IDENTITY, artinya nilai id akan secara otomatis di-generate oleh database (auto-increment) setiap kali record baru ditambahkan. Atribut kedua adalah "npm" (Nomor Pokok Mahasiswa) yang bertipe String dengan panjang maksimal 10 karakter, dan nullable diset ke false berarti field ini wajib diisi. Atribut ketiga adalah "nama" yang bertipe String dengan panjang 50 karakter dan juga wajib diisi. Atribut keempat adalah "semester" yang bertipe integer, mewakili semester mahasiswa saat ini, dan juga wajib diisi. Atribut kelima adalah "ipk" yang bertipe float, mewakili Indeks Prestasi Kumulatif mahasiswa, dan juga wajib diisi.

    Class ini dilengkapi dengan constructor kosong (no-argument constructor) yang diperlukan oleh Hibernate untuk membuat instance object saat melakukan deserialisasi dari database. Selain itu, class ini juga memiliki constructor yang menerima seluruh parameter

(id, npm, nama, semester, ipk) untuk memudahkan pembuatan object dengan memberikan nilai sekaligus pada saat instantiasi. Untuk setiap atribut, disediakan method getter dan setter yang memungkinkan akses dan modifikasi nilai atribut tersebut. Penggunaan getter dan setter ini merupakan best practice dalam pemrograman Java untuk menjaga enkapsulasi data dan memberikan kontrol terhadap akses atribut-atribut dalam class.

- Class HibernateUtil

Class HibernateUtil adalah utility class yang sangat penting karena berfungsi sebagai manager untuk siklus hidup SessionFactory Hibernate. SessionFactory adalah object yang bertanggung jawab untuk membuat Hibernate Session yang diperlukan untuk melakukan operasi-operasi database. Class ini menggunakan pola Singleton, yang berarti hanya ada satu instance dari SessionFactory yang dibagikan ke seluruh aplikasi, sehingga lebih efisien dalam penggunaan resource. Inisialisasi SessionFactory dilakukan dalam static initializer block, yang berarti inisialisasi ini akan dijalankan sekali saja ketika class HibernateUtil pertama kali dimuat oleh Java Virtual Machine (JVM). Dalam static initializer block ini, Hibernate membaca file konfigurasi hibernate.cfg.xml dan membangun SessionFactory berdasarkan konfigurasi yang ada. Jika terjadi error selama proses inisialisasi, exception akan ditangkap dan ditampilkan di error stream, kemudian dilemparkan kembali sebagai ExceptionInInitializerError yang akan menghentikan jalannya aplikasi.

Class HibernateUtil menyediakan dua method publik yang static. Method pertama adalah getSessionFactory() yang mengembalikan instance SessionFactory yang telah diinisialisasi, sehingga class-class lain dapat memperoleh SessionFactory melalui method ini untuk membuka Session baru. Method kedua adalah testConnection() yang berfungsi untuk menguji apakah koneksi ke database berhasil dilakukan. Method ini membuka sebuah Session, dan jika berhasil akan menampilkan pesan "Connection to the database was successful!" di console. Jika terjadi exception, pesan error akan ditampilkan dan exception akan dicetak ke error stream. Method ini sangat berguna untuk memastikan bahwa konfigurasi database sudah benar dan database server sedang berjalan sebelum aplikasi menjalankan operasi-operasi database yang lebih kompleks.

- Class ModelTableMahasiswa

Class ModelTableMahasiswa adalah sebuah table model yang mengextensi AbstractTableModel dari javax.swing.table. Class ini berfungsi sebagai adapter atau jembatan yang menghubungkan data dari List dengan komponen JTable di GUI. JTable memerlukan sebuah model untuk dapat menampilkan data, dan ModelTableMahasiswa berperan menyediakan data dalam format yang dapat dipahami oleh JTable. Class ini memiliki dua atribut utama: pertama adalah "mahasiswaList" yang merupakan List berisi objek-objek ModelMahasiswa, dan kedua adalah "columnNames" yang merupakan array String berisi nama-nama kolom yang akan ditampilkan di tabel (ID, NPM, Nama, Semester, IPK).

Sebagai subclass dari AbstractTableModel, class ini wajib mengimplementasikan beberapa method abstract. Method getRowCount() mengembalikan jumlah baris dalam tabel, yang sama dengan ukuran list mahasiswa. Method getColumnCount() mengembalikan jumlah kolom, yang sama dengan panjang array columnNames (5 kolom).

Method getValueAt(int rowIndex, int columnIndex) mengembalikan nilai data pada posisi tertentu dengan melakukan switch terhadap columnIndex untuk menentukan atribut mana dari ModelMahasiswa yang akan dikembalikan. Method getColumnName(int column) mengembalikan nama kolom sesuai dengan index yang diberikan. Method isCellEditable() mengembalikan false untuk semua cell, berarti semua cell dalam tabel tidak dapat diedit langsung oleh pengguna. Selain itu, class ini juga memiliki method setMahasiswaList() yang memungkinkan untuk mengganti isi list dengan data baru, dan setiap kali list diganti, method fireTableDataChanged() dipanggil untuk memberitahu JTable bahwa data telah berubah sehingga JTable akan merefresh tampilannya.

- Komponen Controller
  - Interface MahasiswaController

    Interface MahasiswaController mendefinisikan kontrak (contract) untuk semua operasi yang dapat dilakukan terhadap data mahasiswa. Interface ini menggunakan prinsip Object-Oriented Programming yaitu polymorphism, dimana dengan mendefinisikan interface terlebih dahulu, memungkinkan untuk membuat multiple implementations. Interface ini mendefinisikan lima method utama yang merupakan operasi CRUD (Create, Read, Update, Delete) dan beberapa operasi tambahan. Method addMhs(ModelMahasiswa mhs) digunakan untuk menambahkan record mahasiswa baru ke database. Method getMhs(int id) digunakan untuk mengambil satu record mahasiswa berdasarkan ID tertentu. Method updateMhs(ModelMahasiswa mhs) digunakan untuk mengupdate/memodifikasi data mahasiswa yang sudah ada. Method deleteMhs(int id) digunakan untuk menghapus record mahasiswa dari database berdasarkan ID. Method getAllMahasiswa() digunakan untuk mengambil semua record mahasiswa dari database. Dengan adanya interface ini, kode menjadi lebih fleksibel dan memudahkan untuk mengganti implementasi tanpa perlu mengubah code di bagian lain yang menggunakan interface ini.

  - Class MahasiswaControllerImpl

    Class MahasiswaControllerImpl adalah implementasi konkret dari interface MahasiswaController yang berisi logika bisnis sebenarnya untuk mengakses database menggunakan Hibernate. Setiap method dalam class ini melakukan operasi database dengan pola yang konsisten: membuka Session dari SessionFactory, memulai Transaction, melakukan operasi database (save, update, delete, atau query), melakukan commit terhadap transaction jika berhasil, atau melakukan rollback jika terjadi exception. Pola ini sangat penting untuk menjaga konsistensi dan integritas data dalam database.

    Method addMhs(ModelMahasiswa mhs) berfungsi untuk menambahkan record mahasiswa baru ke database. Method ini membuka Hibernate Session dari SessionFactory menggunakan try-with-resources statement, yang memastikan Session akan ditutup secara otomatis setelah selesai. Kemudian dimulai sebuah transaction dengan session.beginTransaction(). Object mahasiswa disimpan ke database menggunakan session.save(mhs), dan transaction di-commit untuk membuat perubahan permanen di database. Jika terjadi exception, transaction akan di-rollback untuk membatalkan operasi tersebut, dan exception akan dicetak ke error stream.

Method updateMhs(ModelMahasiswa mhs) berfungsi untuk mengupdate/memodifikasi data mahasiswa yang sudah ada di database. Pola operasinya sama dengan addMhs, yaitu membuka Session, memulai Transaction, kemudian menggunakan session.update(mhs) untuk mengupdate record yang sesuai dengan object mahasiswa yang diberikan. Jika record dengan ID yang sama ada di database, record tersebut akan diupdate dengan nilai-nilai baru dari object mahasiswa. Commit dan rollback juga dilakukan dengan pola yang sama.

Method deleteMhs(int id) berfungsi untuk menghapus record mahasiswa dari database berdasarkan ID. Method ini membuka Session, memulai Transaction, kemudian menggunakan session.get(ModelMahasiswa.class, id) untuk mengambil object mahasiswa dengan ID tertentu. Jika object ditemukan (tidak null), maka object tersebut akan dihapus dari database menggunakan session.delete(mhs). Setelah penghapusan berhasil, pesan "Berhasil hapus" ditampilkan ke console, dan transaction di-commit. Jika object tidak ditemukan, tidak ada yang akan dihapus namun transaction tetap di-commit. Exception handling dan rollback juga dilakukan sesuai pola yang telah ditentukan.

Method getAllMahasiswa() berfungsi untuk mengambil semua record mahasiswa dari database dan mengembalikan hasilnya sebagai List. Method ini membuka Session, memulai Transaction, kemudian menggunakan Hibernate Query Language (HQL) untuk membuat query dengan statement "from ModelMahasiswa" yang artinya adalah "ambil semua record dari tabel ModelMahasiswa". Query ini kemudian dijalankan dengan query.list() untuk mengambil semua hasil dan menyimpannya dalam sebuah List. Transaction di-commit setelah query selesai, dan List dikembalikan ke caller. Jika terjadi exception atau List masih null, method akan mengembalikan null. Method ini sangat penting karena digunakan untuk menampilkan semua data mahasiswa di table ketika aplikasi startup atau ketika user menekan tombol Refresh.

Method getMhs(int id) saat ini belum diimplementasi dan akan melempar UnsupportedOperationException jika dipanggil. Method ini seharusnya mengambil satu record mahasiswa berdasarkan ID, namun implementasinya belum dibuat.

- Komponen View
  - Class MahasiswaView

Class MahasiswaView adalah kelas utama yang merepresentasikan GUI (Graphical User Interface) aplikasi. Class ini mengextensi javax.swing.JFrame, yang berarti class ini adalah sebuah window aplikasi desktop. Aplikasi ini menggunakan Java Swing sebagai framework untuk membuat komponen-komponen GUI.

- Inisialisasi Aplikasi

Constructor dari MahasiswaView melakukan beberapa inisialisasi penting. Pertama, method initComponents() dipanggil untuk membuat dan mengatur layout semua komponen GUI seperti text field, button, dan table. Kemudian, object MahasiswaControllerImpl dibuat dan disimpan dalam instance variable controller, yang akan digunakan untuk berkomunikasi dengan layer controller ketika user melakukan aksi. Selanjutnya, method HibernateUtil.testConnection() dipanggil untuk melakukan test koneksi ke database,

memastikan bahwa konfigurasi database sudah benar dan database server sedang berjalan sebelum aplikasi melanjutkan eksekusinya. Terakhir, method loadMahasiswaTable() dipanggil untuk memuat data semua mahasiswa dari database dan menampilkannya dalam tabel.

- Komponen-Komponen GUI

Aplikasi menyediakan beberapa komponen input untuk user dapat memasukkan data mahasiswa baru. Terdapat empat JTextField: npmField untuk input NPM (Nomor Pokok Mahasiswa), namaField untuk input nama mahasiswa, semesterField untuk input semester, dan ipkField untuk input IPK (Indeks Prestasi Kumulatif). Aplikasi juga menyediakan tiga button untuk melakukan aksi: btnSimpan untuk menyimpan data mahasiswa baru, btnRefresh untuk merefresh data mahasiswa dari database, dan btnHapus untuk menghapus data mahasiswa tertentu. Komponen utama lainnya adalah dataTable, sebuah JTable yang berfungsi untuk menampilkan semua data mahasiswa dalam bentuk tabel dengan lima kolom: ID, NPM, Nama, Semester, dan IPK.

- Method loadMahasiswaTable()

Method loadMahasiswaTable() berfungsi untuk memuat data mahasiswa dari database dan menampilkannya di dalam JTable. Method ini pertama-tama memanggil controller.getAllMahasiswa() untuk mengambil semua record mahasiswa dari database sebagai List. Kemudian, object ModelTableMahasiswa dibuat dengan memberikan List mahasiswa tersebut sebagai parameter. Object ModelTableMahasiswa ini menjadi model untuk JTable, dan diset ke dataTable menggunakan dataTable.setModel(tableModel). Dengan cara ini, JTable akan menampilkan data mahasiswa dalam bentuk tabel dengan jumlah baris sesuai dengan jumlah data mahasiswa di database.

- Event Handler btnSimpanActionPerformed()

Ketika user menekan tombol "Simpan", event handler btnSimpanActionPerformed() dipanggil untuk memproses input user. Event handler ini pertama-tama mengambil nilai dari semua text field: npmField.getText() untuk mendapatkan NPM, namaField.getText() untuk mendapatkan nama, Integer.parseInt(semesterField.getText()) untuk mendapatkan semester (dan mengkonversi dari String ke int), serta Float.parseFloat(ipkField.getText()) untuk mendapatkan IPK (dan mengkonversi dari String ke float). Setelah semua nilai diambil, sebuah object ModelMahasiswa baru dibuat dengan parameter (0, npm, nama, semester, ipk). Nilai id diset ke 0 karena id akan di-generate secara otomatis oleh database sebagai auto-increment. Kemudian, nilai-nilai input di-print ke console untuk debugging purposes. Selanjutnya, controller.addMhs(mahasiswa) dipanggil untuk menyimpan object mahasiswa ke database. Terakhir, loadMahasiswaTable() dipanggil untuk merefresh tampilan tabel sehingga data mahasiswa yang baru ditambahkan akan terlihat di tabel.

- Event Handler btnRefreshActionPerformed()

Ketika user menekan tombol "Refresh", event handler btnRefreshActionPerformed() dipanggil. Event handler ini hanya memanggil loadMahasiswaTable() untuk merefresh data mahasiswa dari database dan menampilkannya kembali di tabel. Fungsi ini berguna jika ada perubahan data di database dari source lain, atau jika user ingin melihat data yang paling terbaru.

- Event Handler btnHapusActionPerformed()

Ketika user menekan tombol "Hapus", event handler btnHapusActionPerformed() dipanggil untuk memproses penghapusan data. Event handler ini menampilkan sebuah dialog box kepada user menggunakan JOptionPane. Dialog box ini bertanya kepada user untuk memasukkan ID mahasiswa yang ingin dihapus. User diminta untuk memasukkan nilai ID dalam sebuah JTextField yang kemudian ditampilkan dalam sebuah panel dialog dengan dua tombol: OK dan Cancel. Jika user menekan tombol OK, nilai ID yang dimasukkan akan diambil dan dikonversi menjadi integer. Kemudian controller.deleteMhs(id) dipanggil untuk menghapus record mahasiswa dengan ID tersebut dari database. Jika penghapusan berhasil, sebuah message dialog ditampilkan dengan pesan "Data berhasil dihapus." sebagai konfirmasi. Namun jika user memasukkan nilai yang bukan angka (NumberFormatException), sebuah error message dialog akan ditampilkan dengan pesan "ID harus berupa angka." untuk memberitahu user bahwa input tidak valid. Jika user menekan tombol Cancel, tidak ada aksi apapun yang dilakukan. Setelah penghapusan berhasil, loadMahasiswaTable() juga sebaiknya dipanggil untuk merefresh tampilan tabel, meskipun dalam kode saat ini belum terdapat pemanggilan ini (ini adalah bug yang perlu diperbaiki).

- Method main()

Method main(String args[]) adalah entry point dari aplikasi. Method ini mengatur look and feel (tampilan/style) dari GUI menggunakan Nimbus, yang merupakan look and feel default modern untuk Java Swing. Jika Nimbus tidak tersedia, aplikasi akan menggunakan default look and feel. Setelah itu, aplikasi membuat instance baru dari MahasiswaView dan menampilkannya menggunakan setVisible(true) pada Event Dispatch Thread menggunakan EventQueue.invokeLater(). Penggunaan EventQueue.invokeLater() sangat penting karena GUI harus dibuat dan diupdate hanya dari Event Dispatch Thread untuk menghindari race condition dan memastikan thread safety.