Nama : Lodovikus Aldo Charlo

Nim : 2201010705

UAS : OOP

Ide project : Data Mahasiswa

Deskripsi Proyek

Proyek ini akan membuat sistem manajemen data mahasiswa yang mampu mengelola informasi mahasiswa, seperti nim,nama,Alamat,umur dan angkatan. Sistem ini akan menggunakan prinsip-prinsip OOP seperti access modifier, inheritance, polymorphism, dan encapsulation untuk memastikan desain yang bersih, modular, dan dapat diperluas.

Fitur Utama

Pengelolaan Data Pribadi Mahasiswa: Menyimpan informasi dasar seperti nim,nama,Alamat,umur dan Angkatan.

Struktur Kelas dan Hubungan

Kelas Mahasiswa

- Fields: Nim, Nama, Alamat, Umur, Angkatan
- Methods: Getters dan Setters untuk setiap field, metode untuk menampilkan informasi pribadi.

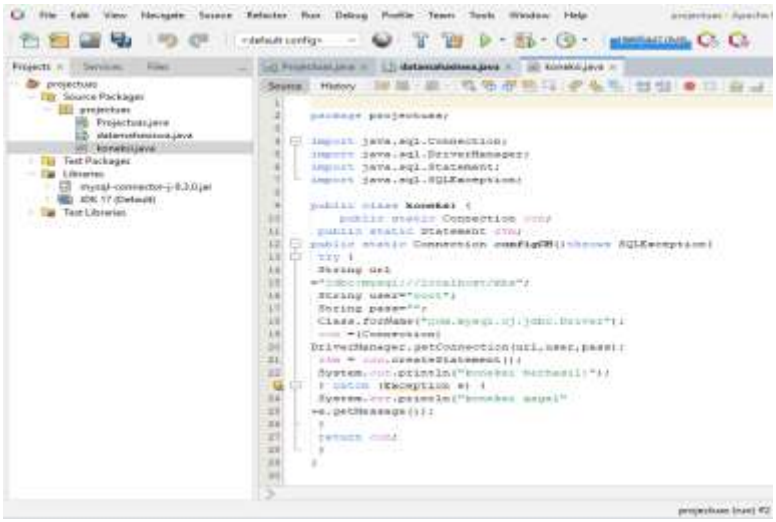
Penerapan OOP

1. Encapsulation
    - Menggunakan access modifiers (private, protected, public) untuk melindungi data dan menyediakan akses melalui getters dan setters.
2. Inheritance
    - Kelas MHS mewarisi properti dan metode dari kelas Mahasiswa.
3. Polymorphism
    - Menggunakan interface Laporan untuk menyediakan berbagai jenis laporan dengan metode generateLaporan().
    - Kelas yang berbeda akan mengimplementasikan metode ini dengan cara yang spesifik.
4. Access Modifiers
    - Menggunakan private untuk fields, protected untuk metode yang bisa diakses oleh subclass, dan public untuk metode yang perlu diakses oleh objek di luar kelas.

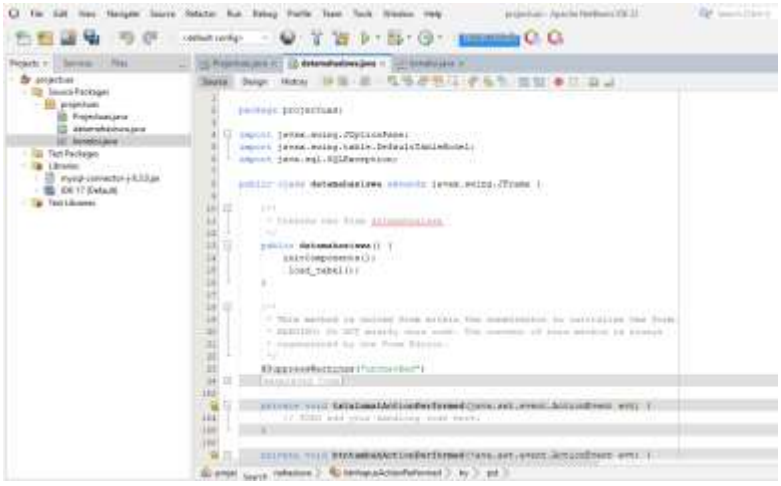
Contoh Implementasi

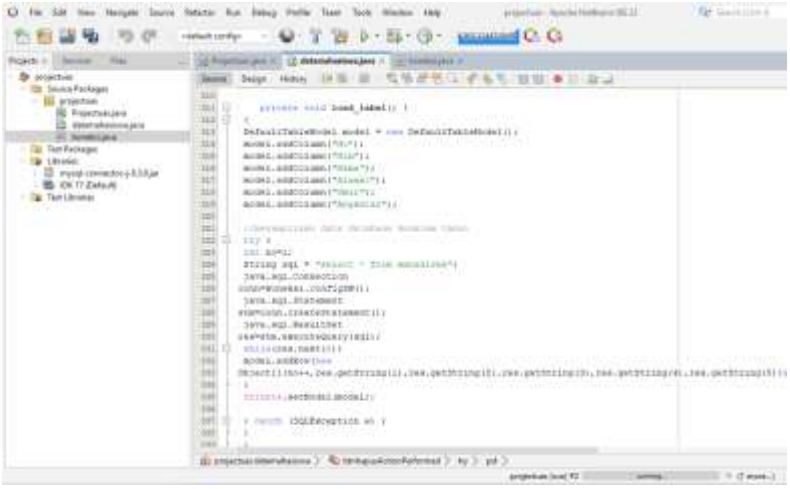
Berikut adalah contoh kode untuk beberapa kelas dan penerapan konsep OOP:

- Kodingan Untuk Koneksi.java



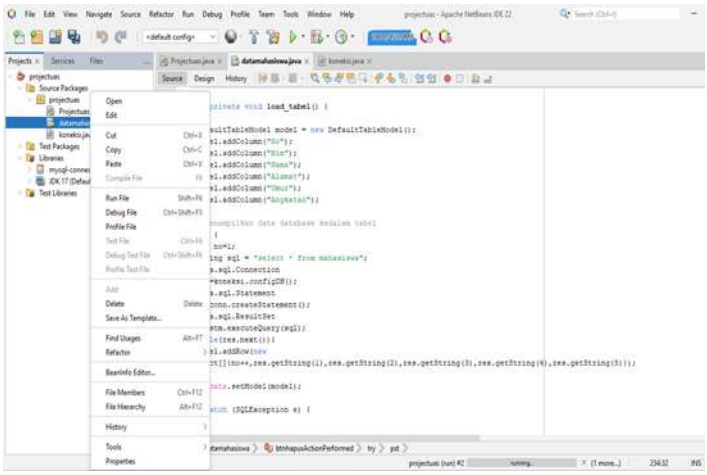
- Kodingan Untuk Datamahasiswa



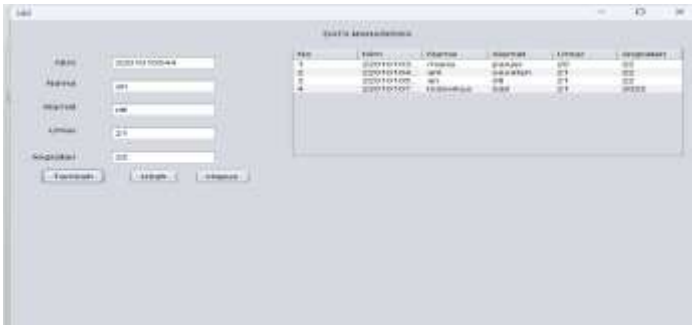


- Run File

  Untuk menjalankan file atau melakukan Run yang pertama klik kanan dibagian datamahasiswa atau seperti gambar dibawah ini.



  Yang kedua setelah melakukan Run file diatas maka akan muncul hasil seperti gambar dibawah ini.



Mengapa Menggunakan Teori-teori OOP di Atas?

1. Encapsulation
   - Alasan**:** Encapsulation melindungi data dari akses langsung luar dan memungkinkan kontrol akses melalui getters dan setters. Hal ini penting untuk menjaga integritas dan keamanan data, terutama dalam sistem yang mengelola informasi sensitif seperti data mahasiswa.
2. Inheritance
   - Alasan**:** Inheritance memungkinkan penggunaan kembali kode dan mengurangi redundansi. Dengan mewarisi properti dan metode dari kelas induk, kelas turunan dapat memanfaatkan fitur yang ada tanpa perlu mendefinisikan ulang, membuat kode lebih efisien dan lebih mudah dipelihara.
3. Polymorphism
   - Alasan**:** Polymorphism memungkinkan satu antarmuka untuk digunakan oleh berbagai tipe objek. Ini meningkatkan fleksibilitas dan memungkinkan kode

untuk menangani objek-objek yang berbeda dengan cara yang seragam. Dalam proyek ini, berbagai jenis laporan dapat dihasilkan melalui metode yang sama, membuat sistem lebih modular dan dapat diperluas.

4. Access Modifiers
    - Alasan: Access modifiers membantu mengontrol aksesibilitas dari kelas, metode, dan variabel. Ini memastikan bahwa hanya bagian yang perlu diakses dari luar kelas yang terbuka, sementara yang lain terlindungi. Penggunaan access modifiers seperti private, protected, dan public membantu dalam menjaga batasan-batasan akses yang tepat dalam kode.

Kesimpulan

Dengan menggunakan prinsip-prinsip OOP seperti encapsulation, inheritance, polymorphism, dan access modifiers, proyek ini memastikan bahwa sistem manajemen data mahasiswa yang dibuat memiliki struktur yang bersih, modular, mudah dipelihara, dan aman. Prinsip-prinsip ini membantu dalam menciptakan kode yang lebih terorganisir, mengurangi redundansi, meningkatkan keamanan, dan memungkinkan perluasan fitur di masa depan dengan lebih mudah.