Laporan Praktikum

Mata Kuliah Pemrograman Berorientasi Objek

Pertemuan 5.Praktikum5

“Polimorfisme dengan Javascript”

Dosen Pengampu :

Willdan Aprizal Arifin, S.Pd., M.Kom.

Disusun Oleh :

Siti Nurkholizah

2308807

3A - Sistem Informasi Kelautan

PROGRAM STUDI SISTEM INFORMASI KELAUTAN

UNIVERSITAS PENDIDIKAN INDONESIA

2024

I. PENDAHULUAN

Polimorfisme adalah salah satu konsep penting dalam pemrograman berorientasi objek yang memungkinkan objek-objek yang berbeda untuk merespons dengan cara yang berbeda terhadap panggilan metode yang sama. Dalam praktiknya, polimorfisme dapat meningkatkan fleksibilitas dan keterbacaan kode, memungkinkan pengembang untuk menggunakan antarmuka yang konsisten tanpa harus mengetahui detail spesifik dari implementasi objek tersebut. Praktikum ini bertujuan untuk menerapkan konsep polimorfisme dalam JavaScript dengan menggunakan kelas dan subclass untuk menciptakan berbagai jenis kapal dan mendemonstrasikan cara kerja polimorfisme dalam konteks tersebut.

II. TUJUAN

1. Mempelajari dan memahami konsep polimorfisme dalam pemrograman berorientasi objek menggunakan JavaScript.
2. Membangun hierarki kelas menggunakan superclass dan subclass untuk menggambarkan berbagai jenis kapal.
3. Mengimplementasikan metode dengan nama yang sama pada kelas yang berbeda dan menunjukkan perbedaan respons terhadap metode tersebut.
4. Menggunakan objek untuk menunjukkan penerapan polimorfisme dalam konteks nyata.

III. ALAT DAN BAHAN

- Laptop
- Visual Studio Code
- Browser web (Google Chrome)
- Terminal atau Command Prompt.
- HTML
- Javascript

IV. LANGKAH KERJA

1. Struktur kodenya ada 2 yaitu index.html dan polimorfime.js
    - index.html: Merupakan file HTML yang menyajikan antarmuka pengguna dan memuat skrip JavaScript dari file polymorphism.js.
    - polymorphism.js: Berisi definisi kelas untuk kapal dan subclass-nya, serta implementasi metode untuk mengilustrasikan polimorfisme.

2. Mengimplementasi class kapal dari subclassnya
    - Kelas Kapal: Sebagai superclass, kelas ini memiliki dua properti: nama dan jenis, serta metode berlayar() yang mencetak pesan bahwa kapal sedang berlayar.

- Subclass:
- KapalPenumpang: Menambahkan properti kapasitas dan mengoverride metode berlayar() untuk mencetak informasi tentang jumlah penumpang.
- KapalBarang: Menambahkan properti muatan dan mengoverride metode berlayar() untuk mencetak informasi tentang muatan.
- KapalTanker: Menambahkan properti kapasitasBahanBakar dan mengoverride metode berlayar() untuk mencetak informasi tentang kapasitas bahan bakar.
- KapalPesiar: Menambahkan properti jumlahKabin dan mengoverride metode berlayar() untuk mencetak informasi tentang jumlah kabin.

3. Setelah mendefinisikan kelas-kelas tersebut, objek-objek untuk setiap jenis kapal dibuat. Metode berlayar() dipanggil untuk masing-masing objek untuk menunjukkan penerapan polimorfisme, di mana setiap objek memberikan respons yang sesuai dengan implementasinya masing-masing.

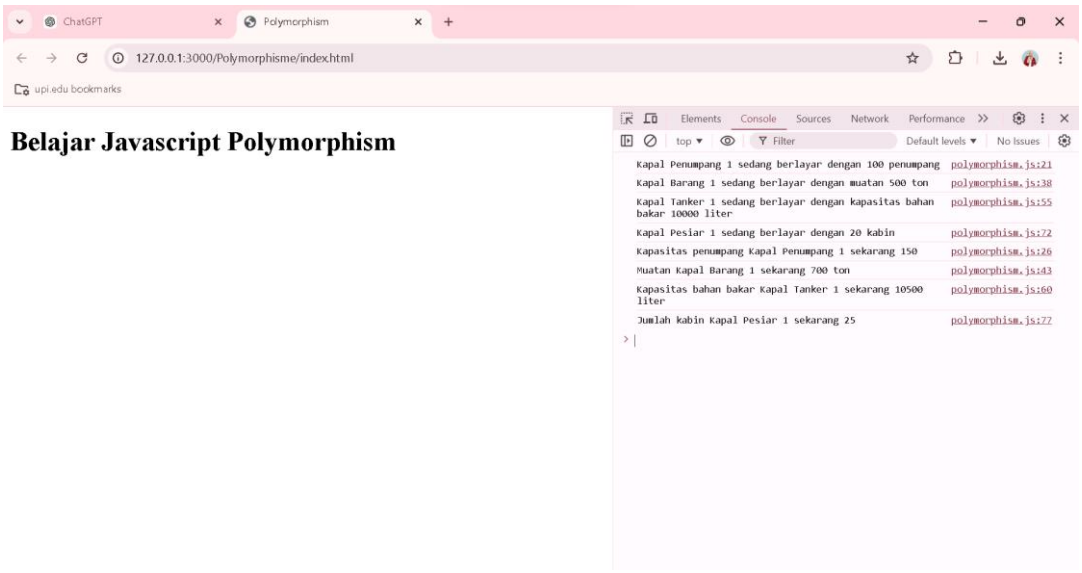
V. HASIL DAN PEMBAHASAN

Setelah menjalankan kode yang diimplementasikan, hasil yang diperoleh menunjukkan bahwa setiap objek kapal berfungsi dengan baik dan dapat memberikan output yang sesuai dengan jenisnya. Berikut adalah contoh output yang dihasilkan:

- Kapal Penumpang: "Kapal Penumpang 1 sedang berlayar dengan 100 penumpang."
- Kapal Barang: "Kapal Barang 1 sedang berlayar dengan muatan 500 ton."
- Kapal Tanker: "Kapal Tanker 1 sedang berlayar dengan kapasitas bahan bakar 10000 liter."
- Kapal Pesiar: "Kapal Pesiar 1 sedang berlayar dengan 20 kabin."

Selain itu, metode tambahan seperti tambahPenumpang(), tambahMuatan(), tambahBahanBakar(), dan tambahKabin() dapat digunakan untuk memperbarui properti kapal sesuai dengan kebutuhan.

HASIL AKHIR



VI. KESIMPULAN

Praktikum ini berhasil menunjukkan penerapan polimorfisme dalam JavaScript melalui pembuatan hierarki kelas yang menggambarkan berbagai jenis kapal. Dengan menggunakan metode yang sama tetapi memberikan respons yang berbeda pada setiap subclass, peserta dapat memahami konsep polimorfisme dengan lebih baik. Implementasi ini tidak hanya meningkatkan fleksibilitas kode, tetapi juga memperjelas struktur dan hierarki data dalam aplikasi yang lebih besar. Penguasaan konsep polimorfisme adalah langkah penting bagi pengembang untuk menciptakan kode yang bersih, efisien, dan mudah dipelihara di masa depan.