**LAPORAN PRAKTIKUM**

**PBO**

**TUGAS 4**

****

**Disusun Oleh :**

Vinsa Qatrunnada

2310264

**PROGRAM STUDI SISTEM INFORMASI KELAUTAN**

**UNIVERSITAS PENDIDIKAN INDONESIA**

**KAMPUS DAERAH SERANG**

**2024**

**Judul**

# Penerapan Polymorphism pada Program Objek-Oriented Programming (OOP) di Javascript.

**Tujuan**

# Memahami konsep pewarisan (inheritance) dalam OOP.

# Mengimplementasikan konsep polymorphism dalam kelas turunan di Javascript.

# Mampu menambah dan menggunakan metode serta properti tambahan pada kelas turunan.

**Dasar Teori**

# Dalam pemrograman berorientasi objek (OOP), terdapat beberapa konsep penting yang perlu dipahami, seperti inheritance dan polymorphism.

# Inheritance (Pewarisan): memungkinkan sebuah kelas untuk mewarisi properti dan metode dari kelas induknya.

# Polymorphism: memungkinkan sebuah metode di kelas induk dapat bekerja berbeda pada kelas turunan tanpa harus mengubah definisi dari metode tersebut.

# Pada kasus ini, metode `infoKapal()` yang ada pada kelas `kapal` bisa di-override oleh kelas-kelas turunan untuk memberikan informasi tambahan spesifik sesuai jenis kapal tersebut.

**Alat dan Bahan**

# Aplikasi Browser.

# Text editor untuk menulis kode Javascript (misalnya, Visual Studio Code, Sublime Text, atau Notepad++).

# HTML untuk menampilkan hasil program.

**Langkah Kerja**

1. Kelas kapal (kelas induk) Kelas kapal (kelas induk) Kelas kapal adalah kelas dasar (kelas induk) yang biasanya digunakan untuk membuat objek kapal. Kelas ini memiliki konstruktor yang menerima parameter.

Nama: Nama kapal.

Jenis: Jenis kapal (kapal pesiar, kapal perang, dll).

Kecepatan: Kecepatan kapal dalam knot.

Kapasitas: Kapasitas penumpang di atas kapal. Panjang: Panjang kapal dalam meter.

Lebar: Lebar kapal dalam meter.

Fungsi kapal info() digunakan untuk mengembalikan informasi kapal dalam bentuk string yang diformat dengan property.

1. Kelas Turunan

Ada beberapa kelas turunan yang mewarisi kelas kapal dan menambahkan atribut tambahan sesuai dengan jenis kapal. Setiap kelas turunan menggunakan keyword extends untuk mewarisi kelas kapal.

* **kapalPesiar (Kapal Pesiar)**: Mewarisi kelas kapal dengan jenis kapal otomatis diatur menjadi 'Kapal Pesiar'. Kelas ini memiliki properti tambahan fasilitas yang merupakan array fasilitas yang dimiliki kapal pesiar. Fungsi infoKapal() di-overriding untuk menampilkan informasi tambahan tentang fasilitas.
* **kapalPerang (Kapal Perang)**: Kelas ini menambahkan atribut persenjataan, yang menyimpan daftar persenjataan kapal perang. Fungsi infoKapal() di-overriding untuk menampilkan persenjataan kapal.
* **kapalSelam (Kapal Selam)**: Menambahkan atribut kedalamanMaksimum, yaitu seberapa dalam kapal selam tersebut bisa menyelam. Fungsi infoKapal() di-overriding untuk menampilkan informasi kedalaman maksimum.
* **kapalPenumpang (Kapal Penumpang)**: Memiliki atribut tambahan rute, yang menggambarkan rute pelayaran kapal penumpang. Fungsi infoKapal() menampilkan informasi tentang rute.
* **kapalKargo (Kapal Kargo)**: Menambahkan properti jenisMuatan, yang menggambarkan jenis muatan yang diangkut kapal kargo. Fungsi infoKapal() menampilkan jenis muatan.
* **kapalLayar (Kapal Layar)**: Menambahkan properti bahanLayar, yang menggambarkan jenis bahan dari layar kapal. Fungsi infoKapal() di-overriding untuk menampilkan bahan layar kapal.

1. Objek Output dan input ke html

* u47: Objek dari kelas kapalSelam dengan nama "U-47", kecepatan 20 knot, kapasitas 40 orang, panjang 67 meter, lebar 7 meter, dan kedalaman maksimum 230 meter.
* queenMary: Objek dari kelas kapalPenumpang dengan nama "Queen Mary 2", kecepatan 25 knot, kapasitas 2700 orang, panjang 345 meter, lebar 41 meter, dengan rute "London - New York".
* everGiven: Objek dari kelas kapalKargo dengan nama "Ever Given", kecepatan 22 knot, kapasitas 200 orang, panjang 400 meter, lebar 59 meter, dengan jenis muatan "Kontainer".
* clipper: Objek dari kelas kapalLayar dengan nama "Cutty Sark", kecepatan 17 knot, kapasitas 35 orang, panjang 85 meter, lebar 11 meter, dan bahan layar "Kapas".

Informasi dari setiap kapal akan ditampilkan dalam elemen HTML dengan id pinse. Fungsi infoKapal() digunakan untuk mendapatkan informasi dari setiap objek dan menampilkannya dalam format HTML.

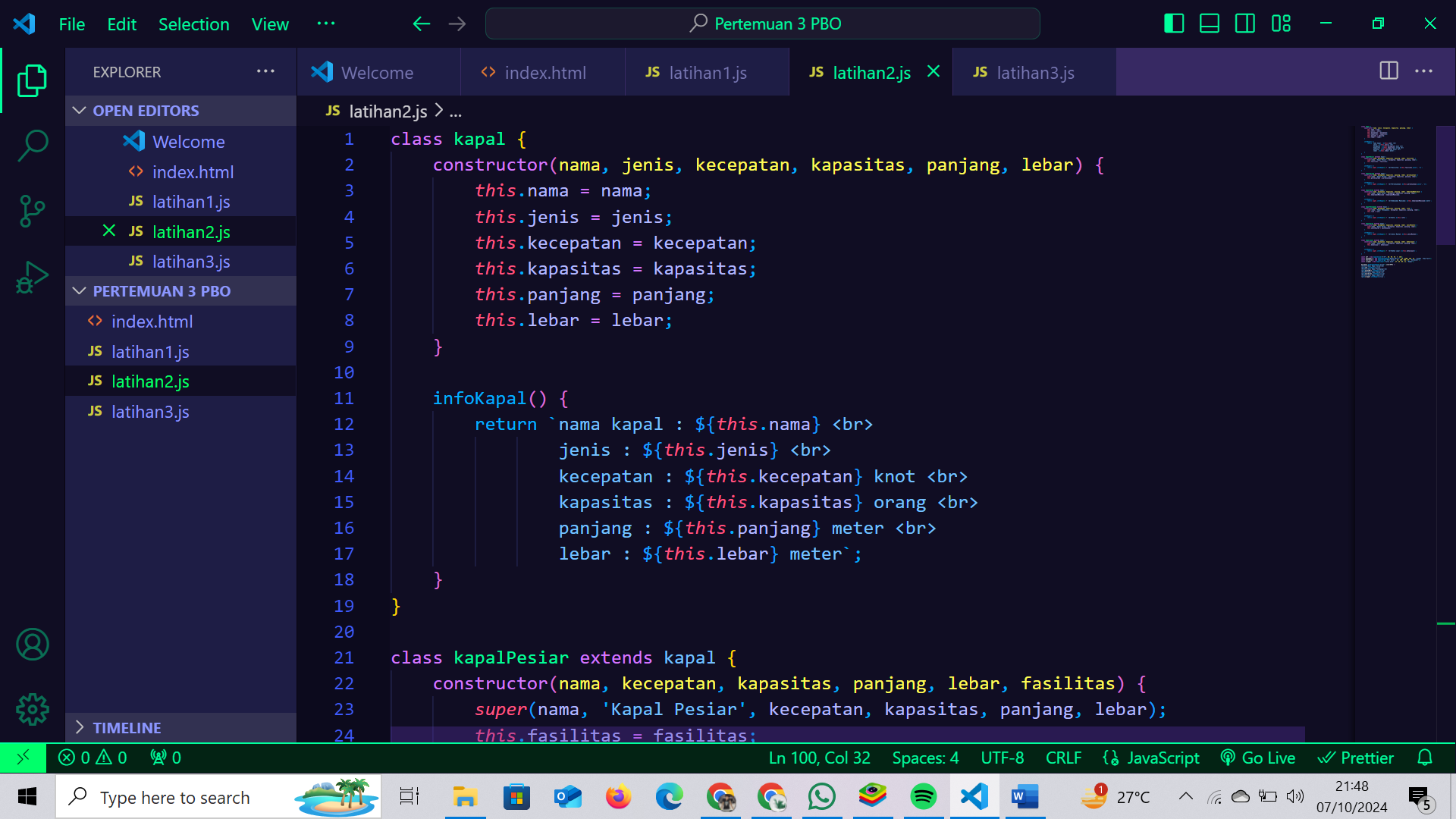
1. Polymorphism

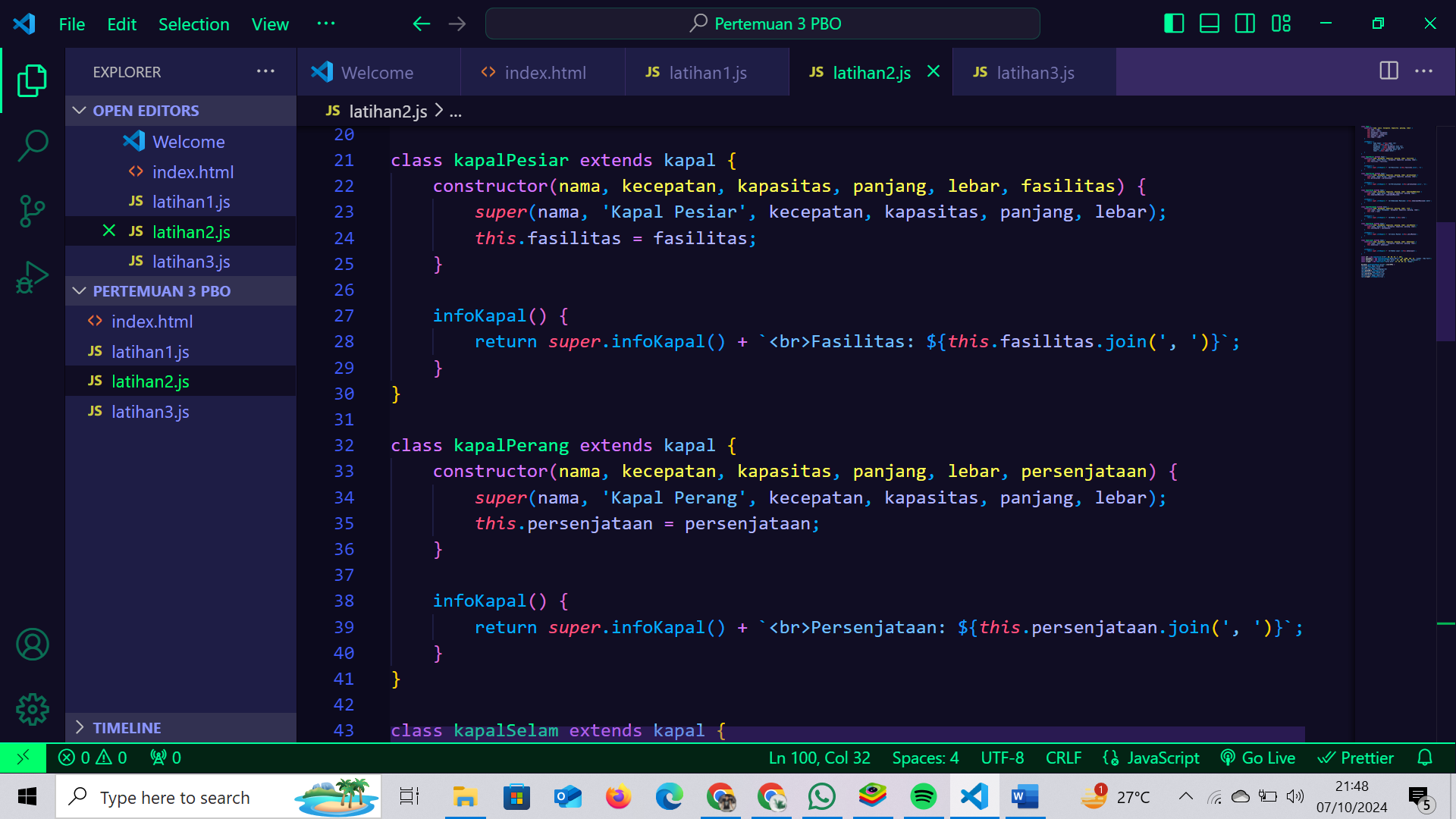
* **Metode infoKapal() di Kelas Induk kapal** Di kelas induk, metode infoKapal() mengembalikan informasi dasar tentang kapal seperti nama, jenis, kecepatan, kapasitas, panjang, dan lebar.
* **Metode infoKapal() Di-override dalam Kelas Turunan** Setiap kelas turunan seperti kapalSelam, kapalPesiar, kapalPerang, kapalPenumpang, kapalKargo, dan kapalLayar meng-override metode infoKapal() untuk menambahkan informasi spesifik yang sesuai dengan jenis kapal mereka.

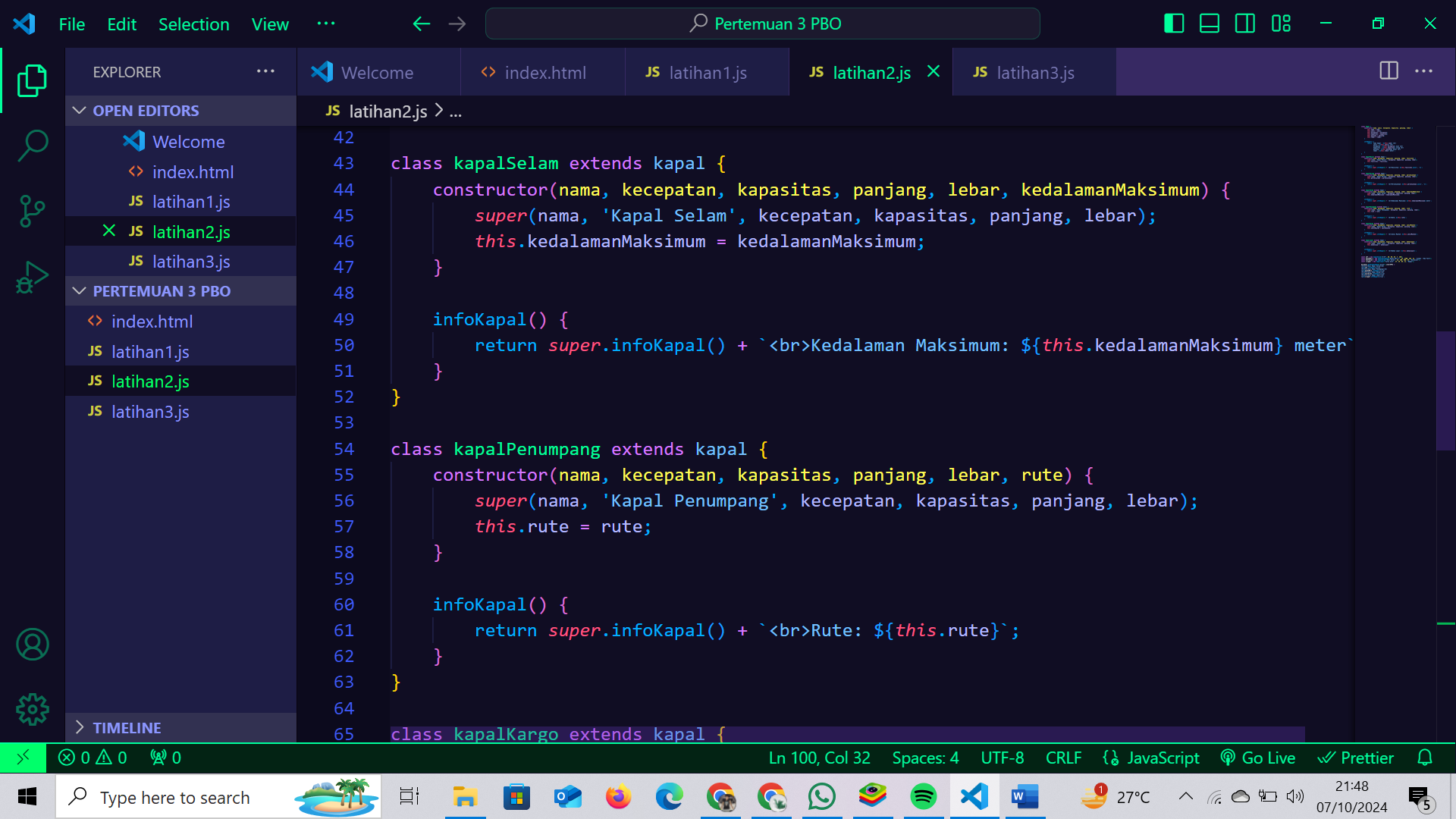
Saat memanggil metode infoKapal() pada objek dari kelas turunan, JavaScript secara otomatis akan mengeksekusi versi metode yang spesifik pada kelas tersebut. Ini adalah contoh polymorphism, di mana metode yang sama (infoKapal()) berperilaku berbeda tergantung pada objek kapal mana yang memanggilnya.

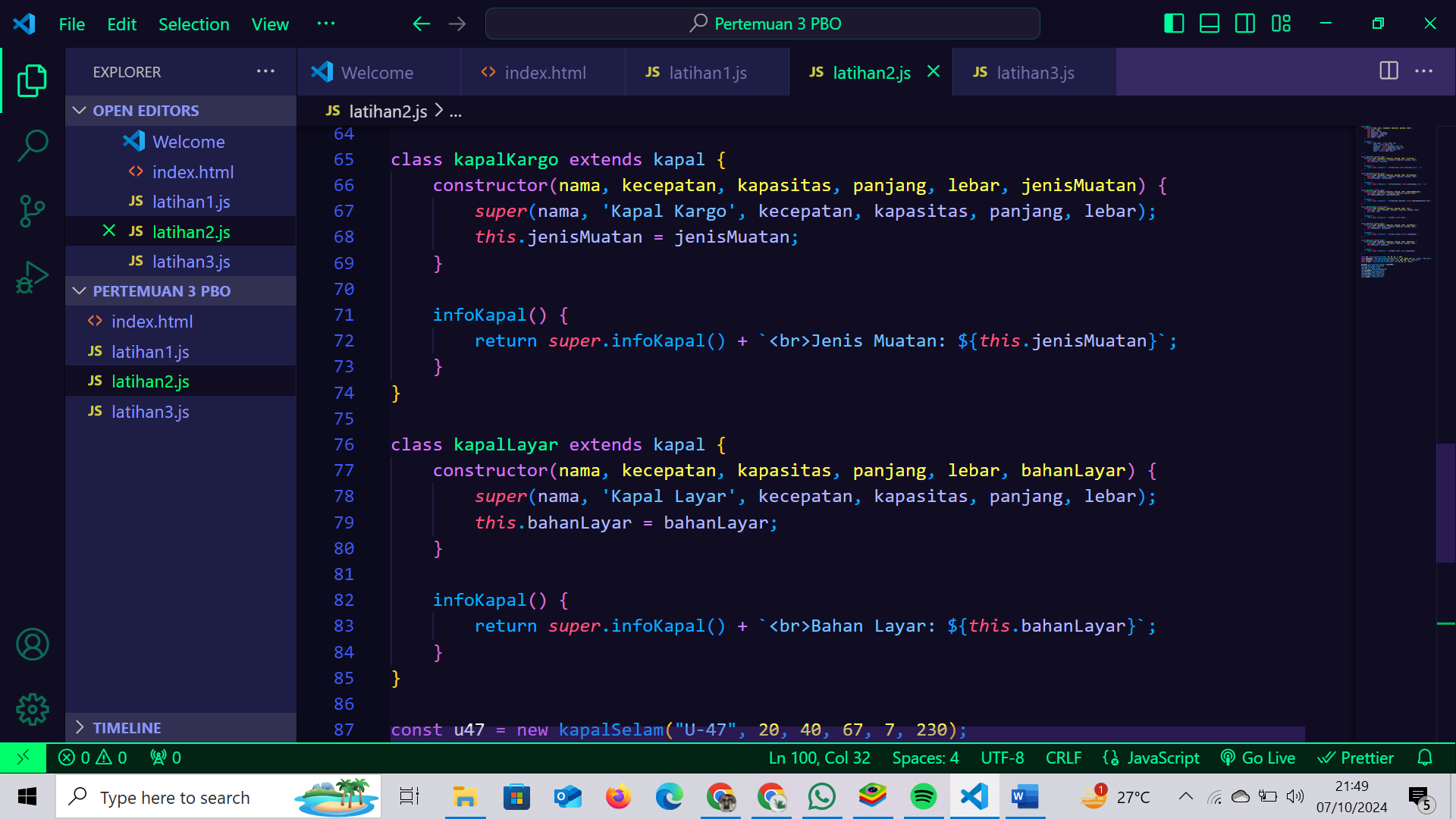
**Kode Program**

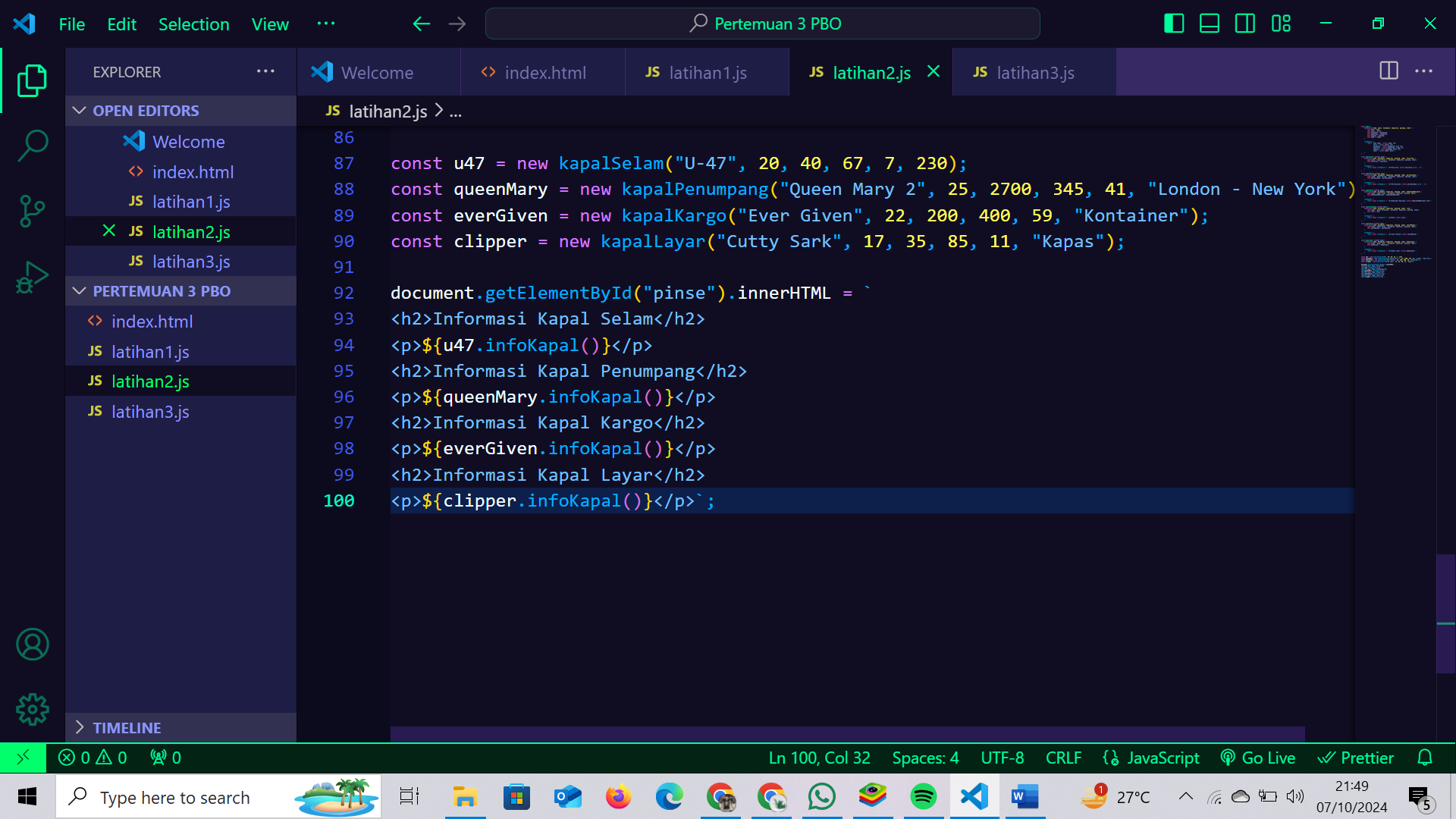
# Javascript :











**Hasil**

# Program di atas menampilkan informasi tentang berbagai jenis kapal menggunakan satu metode `infoKapal()`, namun setiap kelas turunan meng-override metode tersebut untuk menambahkan informasi spesifik. Dengan demikian, meskipun metode yang dipanggil adalah metode yang sama, hasil yang diperoleh berbeda tergantung dari jenis kapal yang digunakan.

**Contoh Keluaran**

# Informasi Kapal Selam:

nama kapal : U-47

jenis : Kapal Selam

kecepatan : 20 knot

kapasitas : 40 orang

panjang : 67 meter

lebar : 7 meter

Kedalaman Maksimum: 230 meter

# Informasi Kapal Penumpang:

nama kapal : Queen Mary 2

jenis : Kapal Penumpang

kecepatan : 25 knot

kapasitas : 2700 orang

panjang : 345 meter

lebar : 41 meter

Rute: London - New York

# Kesimpulan

Dari hasil praktikum ini, kita dapat memahami bahwa

* Konsep pewarisan (inheritance) memungkinkan kelas turunan mewarisi properti dan metode dari kelas induk.
* Polymorphism memungkinkan metode yang sama dapat diimplementasikan dengan perilaku yang berbeda di kelas-kelas turunan.
* Implementasi polymorphism pada OOP di Javascript membantu membuat program lebih modular dan terstruktur dengan baik.

# Link github: