NAMA : I GUSTI AGUNG KRISNA PRASETIA

NIM : 2201010564

UAS

Ide Proyek : aplikasi manajemen pariwisata

1. Access Modifier

Access modifier digunakan untuk mengatur tingkat akses terhadap properti (variabel) dan metode dalam sebuah class. Dengan menggunakan access modifier seperti "**private**, "**public**", "**protected**", dan "**default (package-private**)", kita dapat mengontrol visibilitas dan akses terhadap properti dan metode dalam class tersebut. Misalnya, menggunakan `private` untuk membatasi akses langsung dari luar class, atau `public` untuk memungkinkan akses dari mana saja.

2. Inheritance

Inheritance memungkinkan sebuah class (subclass atau child class) untuk mewarisi properti dan metode dari class lain (superclass atau parent class). Ini memungkinkan untuk memanfaatkan kembali kode, membangun hierarki class yang logis, dan mendefinisikan hubungan antar class.

3. Polymorphism

Polymorphism memungkinkan sebuah objek untuk menunjukkan perilaku yang berbeda tergantung pada konteksnya. Ada dua bentuk utama polymorphism di Java: compile-time polymorphism (method overloading) dan runtime polymorphism (method overriding). Ini memungkinkan untuk menulis kode yang lebih fleksibel dan mudah dimengerti, serta meningkatkan fleksibilitas dalam penggunaan objek.

4. Encapsulation

Encapsulation menggabungkan data dan metode yang beroperasi pada data tersebut dalam sebuah unit tunggal (class), dan menyembunyikan detail implementasi dari pengguna luar. Ini membantu dalam menciptakan kontrol yang lebih baik atas bagaimana data dapat diakses dan dimanipulasi, serta meningkatkan keamanan dan modularitas kode.

Mengapa menggunakan teori tersebut?

Penerapan konsep OOP dalam proyek manajemen pariwisata memberikan beberapa keuntungan:

- Struktur yang Terorganisir : OOP membantu dalam mengatur dan mengelompokkan kode ke dalam class dan objek-objek terkait, mempermudah pemeliharaan dan pengembangan aplikasi.
- Penggunaan Kembali Kode : Dengan inheritance, kode yang sudah ada dapat diwariskan ke subclass, mengurangi duplikasi dan mempercepat pengembangan.

- Fleksibilitas dan Scalability : Polymorphism memungkinkan penggunaan objek secara lebih dinamis, sedangkan encapsulation memungkinkan kontrol yang lebih baik terhadap bagaimana data dapat diakses dan dimanipulasi.

- Keselamatan dan Keamanan : Dengan menggunakan encapsulation, kita dapat menyembunyikan detail implementasi yang tidak perlu diekspos kepada pengguna luar, meningkatkan keamanan aplikasi.

Dengan demikian, penerapan konsep-konsep OOP dalam proyek **manajemen pariwisata** tidak hanya meningkatkan struktur dan efisiensi kode, tetapi juga membuatnya lebih mudah dimengerti, dikembangkan, dan dipelihara dalam jangka panjang.