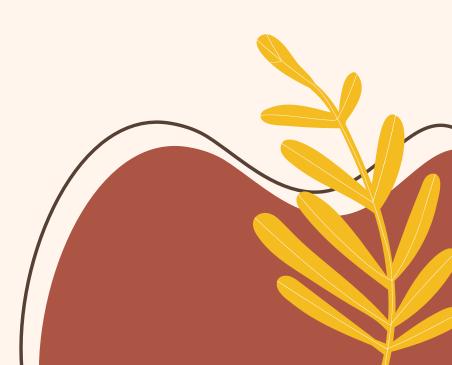
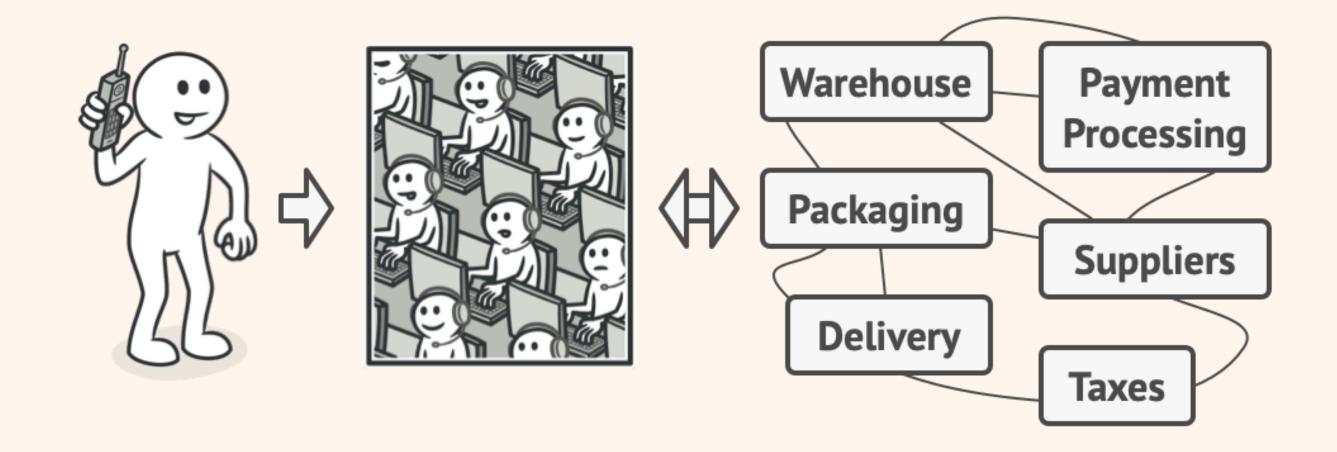
The facade pattern



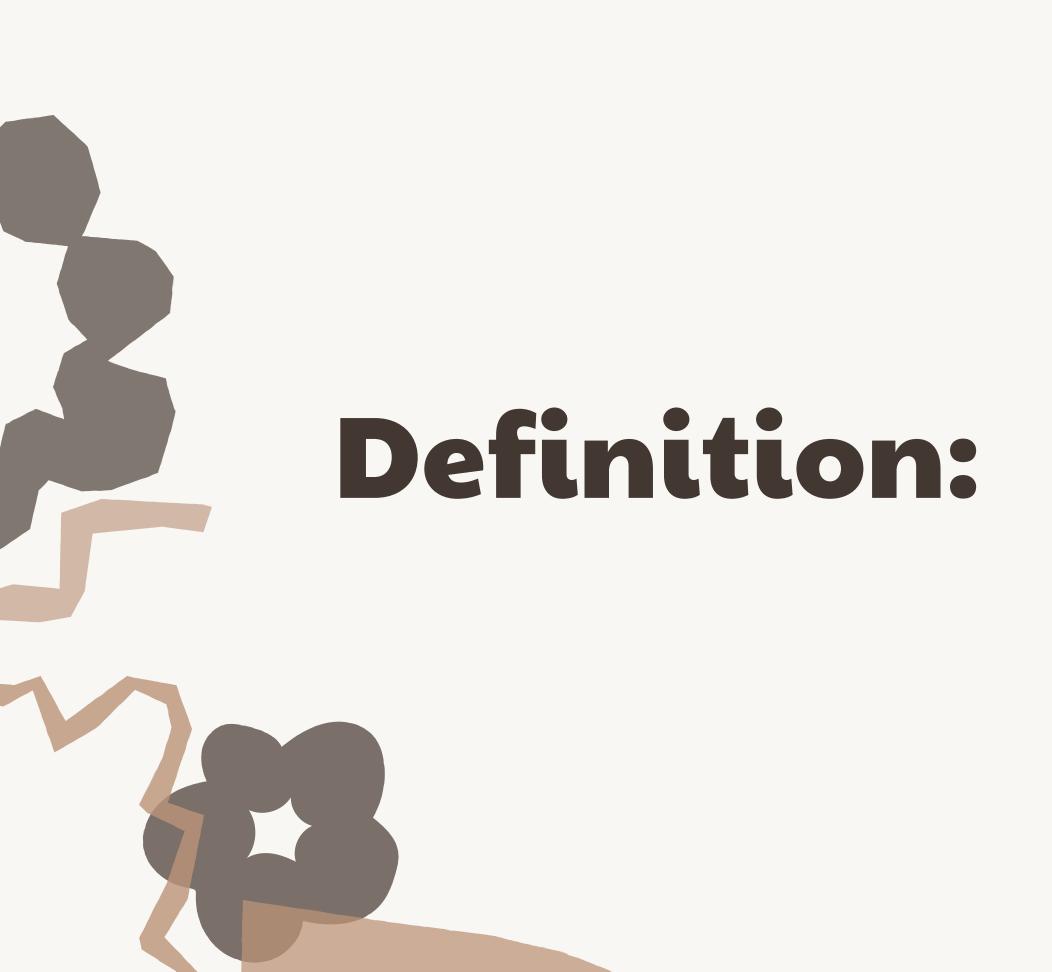
START







Example: Orders Placing by Phone

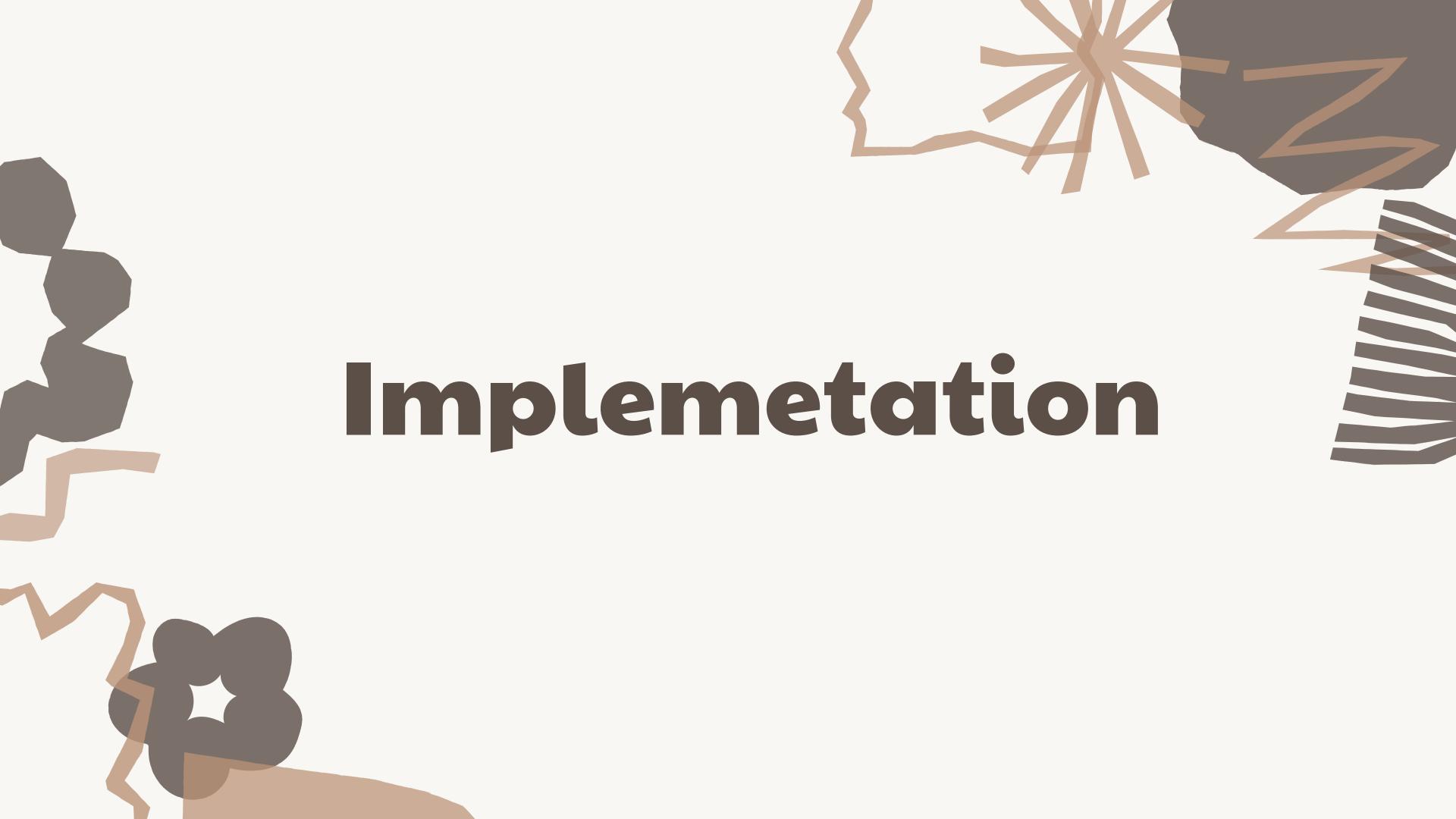


The Facade Pattern is a design pattern that allows you to provide a simplified interface to a complex system. It is a structural pattern because it is concerned with the structure of the code, rather than the behavior of the objects.

When & How to use Facade Pattern

When: The Facade Pattern is useful in situations where you have a complex system that is difficult to use or understand. It can help to hide the complexity of the system behind a simple and easy-to-use interface.

How: The Facade Pattern works by providing a simplified interface to a complex system. It does this by creating a new class that sits between the client and the complex system. This new class is called the Facade, and it provides a simplified interface to the client.



```
// Inventory Management class
class InventoryManagement {
public:
    void checkInventory(int productId) {
        // implementation
};
// Order Processing class
class OrderProcessing {
public:
    void createOrder(int productId, int quantity) {
        // implementation
};
// Payment Processing class
class PaymentProcessing {
                                               private:
public:
    void processPayment(double amount) {
        // implementation
```

};

```
// Purchase Facade class
class PurchaseFacade {
   InventoryManagement inventoryManagement;
   OrderProcessing orderProcessing;
   PaymentProcessing paymentProcessing;
public:
   void purchaseProduct(int productId, int quantity, double amount) {
        inventoryManagement.checkInventory(productId);
        orderProcessing.createOrder(productId, quantity);
       paymentProcessing.processPayment(amount);
};
```



Simplify the code via a single interface

2 Code easier to understand and use



Disavantages

Add an additional layer of abstraction

2 Can hide important details of the system

