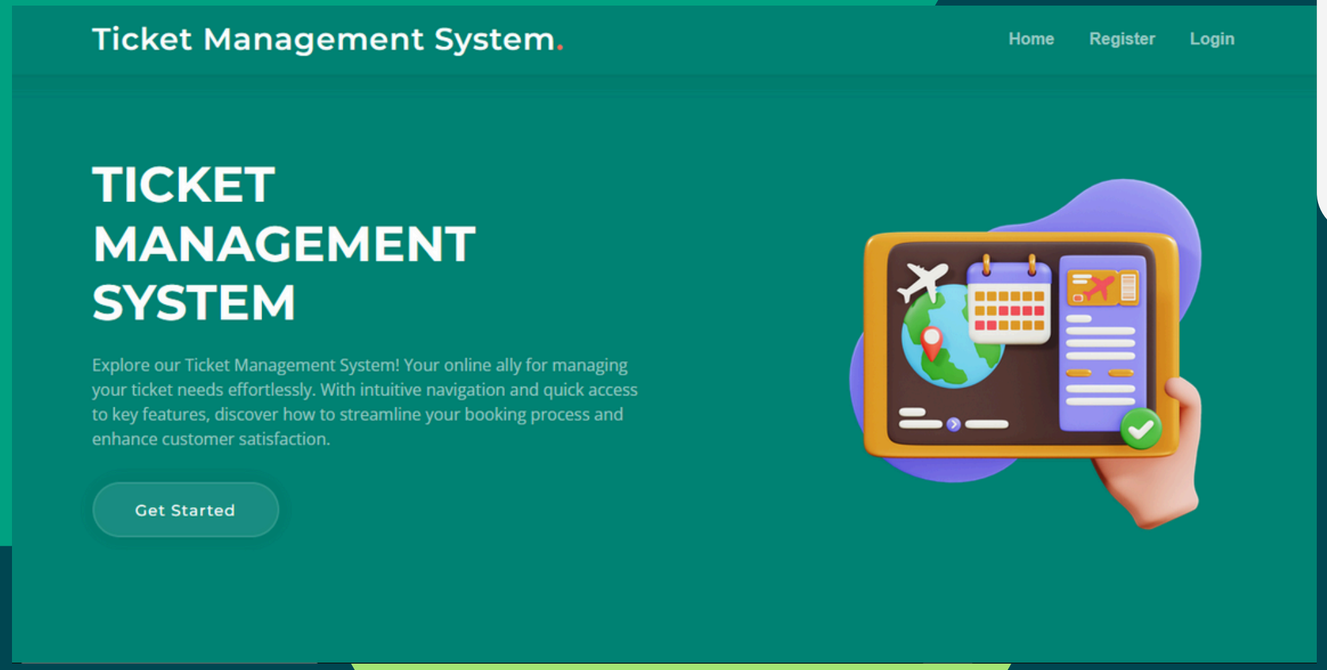
Ticket Management System

Design Pattern Decorator

Encadre Par: Pr.ELHAJJAMY OUSSAMA

Realise Par: BERRADI FATIMA ZOHRA GI2

Gestion de Reservation des Tickets



Dans notre système de gestion des réservations des tickets , les principales tâches comprennent la connexion des utilisateurs, la sélection d'événements et le paiement de billets. Les utilisateurs peuvent se connecter à leur compte, parcourir les événements disponibles et choisir le type de billet qui correspond le mieux à leurs besoins. Une fois leur sélection faite, ils peuvent procéder au paiement en toute sécurité. De plus, si de nouveaux événements sont créés, nous informons automatiquement les clients par e-mail afin qu'ils soient toujours au courant des dernières opportunité.

Technologies utilises

JEE

LE CHOIX DE JAVA EE POUR CE PROJET OFFRE UNE BASE SOLIDE ET FIABLE POUR DÉVELOPPER UN SYSTÈME DE GESTION DES RÉSERVATIONS ROBUSTE, ÉVOLUTIF ET SÉCURISÉ.

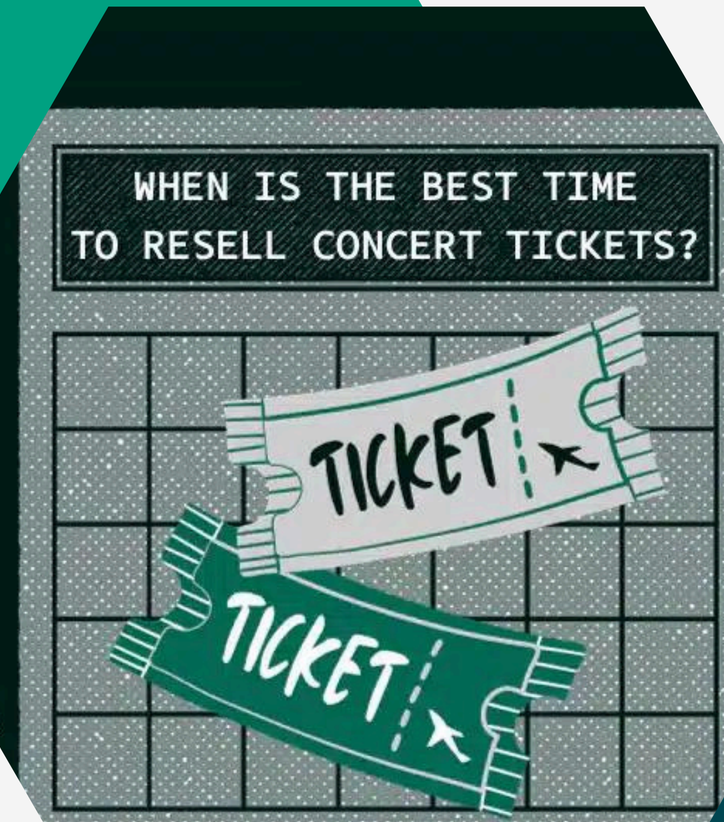
HTML/CSS/BOOTSRAP

PERMETTENT DE CRÉER RAPIDEMENT UNE INTERFACE UTILISATEUR ATTRAYANTE ET ADAPTABLE POUR NOTRE SYSTÈME DE GESTION DES RÉSERVATIONS.

MYSQL

POUR LA GESTION DES DONNÉES DANS NOTRE SYSTÈME DE GESTION DES TICKETS

AUTRES: XAMPP && TOMCAT->SERVEURS



Design Patterns utilises

Strategy

LE PATTERN STRATEGY PERMET AU SYSTÈME DE GÉRER DIFFÉRENTES MÉTHODES DE PAIEMENT POUR LES BILLETS, OFFRANT AINSI UNE FLEXIBILITÉ SANS ALTÉRER LE CODE EXISTANT

Decorator

LE DECORATOR EST UTILISÉ POUR AJOUTER DES FONCTIONNALITÉS SUPPLÉMENTAIRES À CES TICKETS DE MANIÈRE DYNAMIQUE, EN FONCTION DES BESOINS DE L'UTILISATEUR OU DES CARACTÉRISTIQUES DE L'ÉVÉNEMENT.

Observer

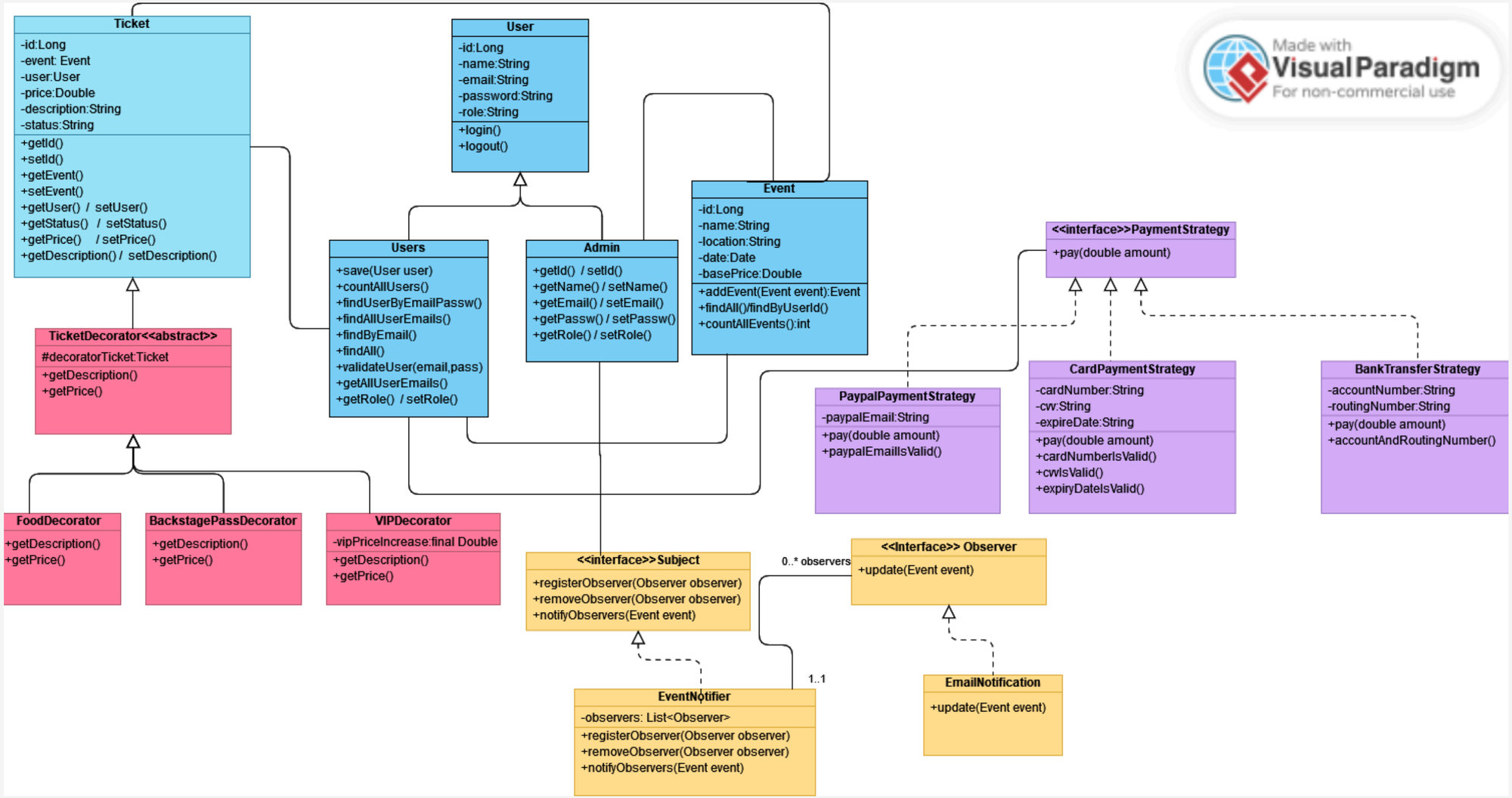
POUR ENVOYER DES E-MAILS AUX UTILISATEURS APRÈS L'AJOUT D'UN NOUVEL ÉVÉNEMENT.

Factory

FACILITE LA CRÉATION DES BILLETS EN FOURNISSANT UNE INTERFACE COMMUNE, EN ENCAPSULANT LES DÉTAILS DE LEUR CRÉATION, ET EN PERMETTANT UNE EXTENSION FACILE DU SYSTÈME.

Ticket Management System

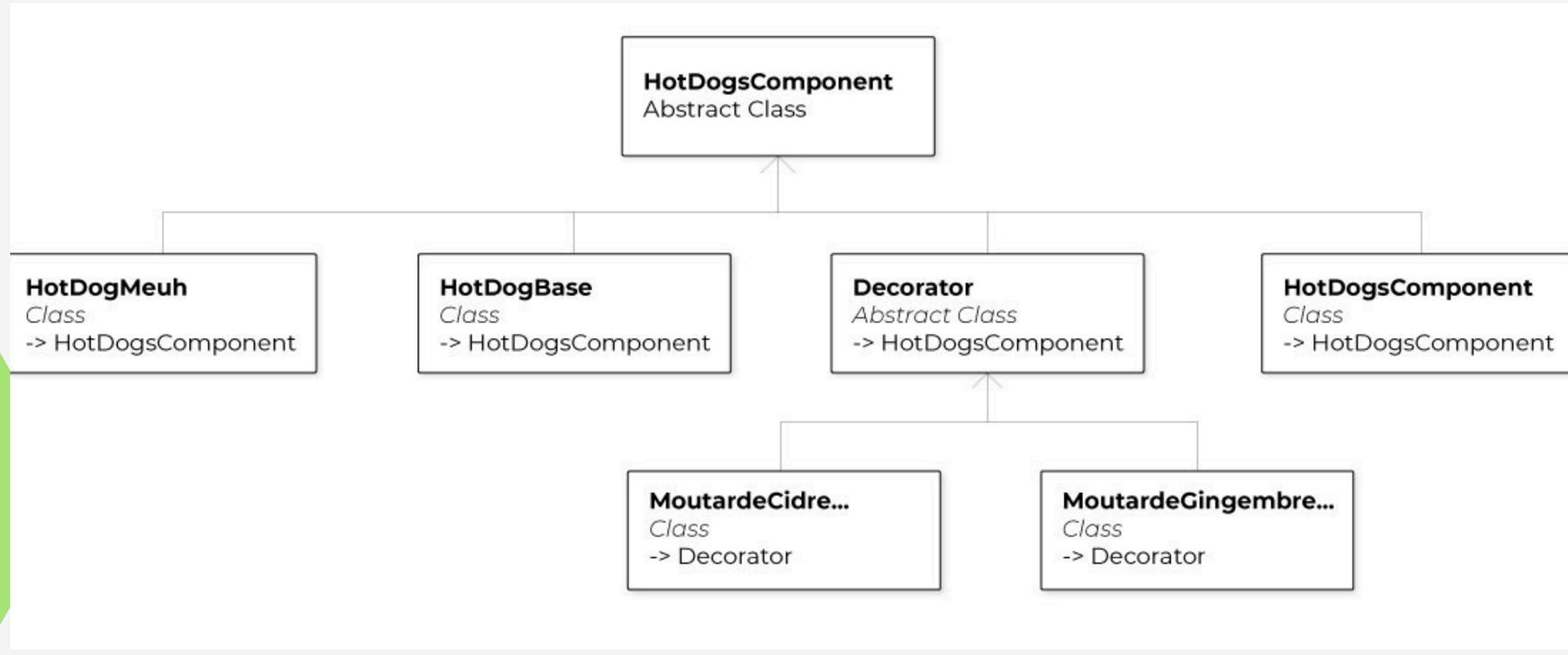
Diagramme De Classe



Implémentation General du Design Pattern Decorator

Decorator

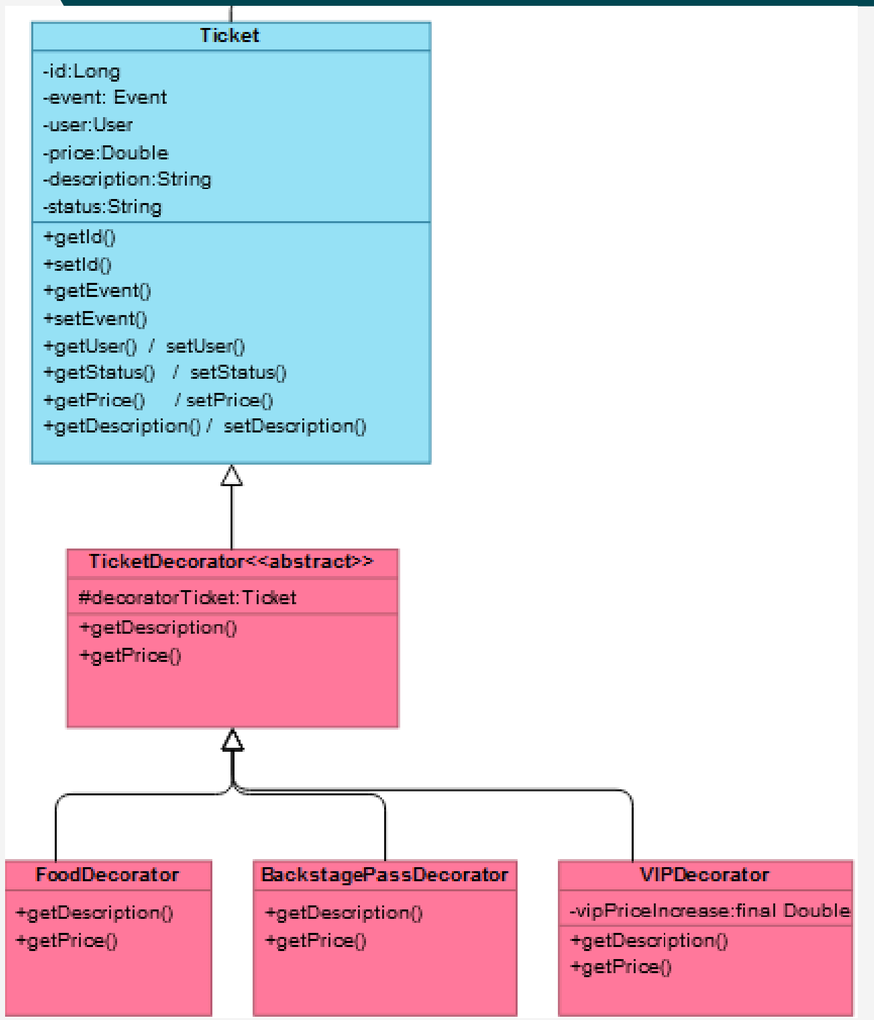
Le pattern décorateur permet d'étendre le comportement d'un objet sans le modifier. Au lieu de changer le code de la classe de base, On peut créer des classes décoratrices qui ajoutent des fonctionnalités sans altérer l'objet de base.



Implementation Decorator

On a le modèle de conception décorateur pour ajouter des fonctionnalités supplémentaires aux billets d'événement tel que chaque décorateur encapsule une instance de Ticket, lui ajoutant ses propres fonctionnalités sans modifier la classe de base Ticket

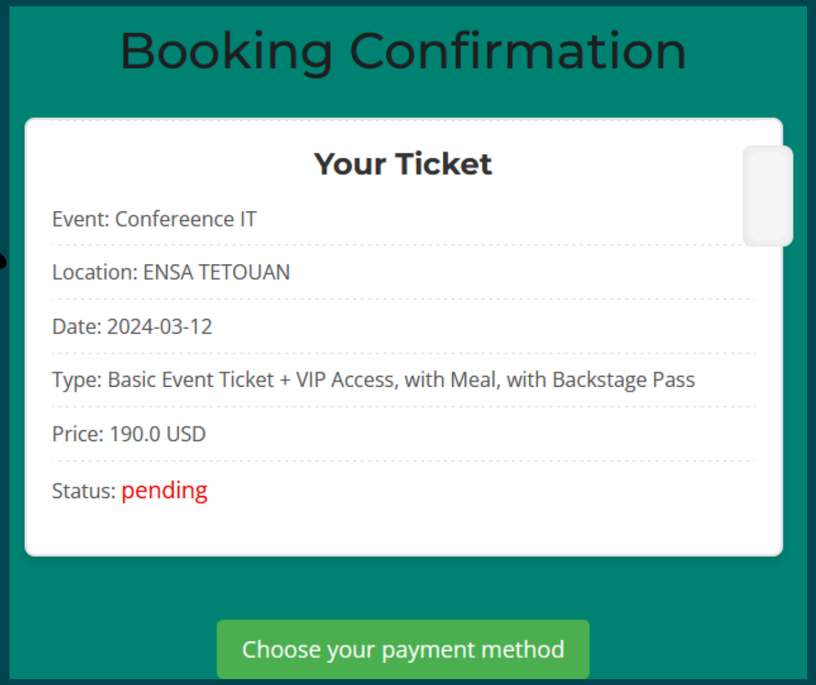
- Le décorateur TicketDecorator est utilisé pour ajouter des fonctionnalités supplémentaires à un objet Ticket existant
- VIPDecorator ajoute l'accès VIP à la description et augmente le prix du billet VIP.
- FoodDecorator ajoute un repas à la description et augmente le prix du billet avec le coût du repas.
- BackstagePassDecorator ajoute un laissez-passer pour les coulisses à la description et augmente le prix du billet avec le coût du laissez-passer.



Fonctionnement Decorator

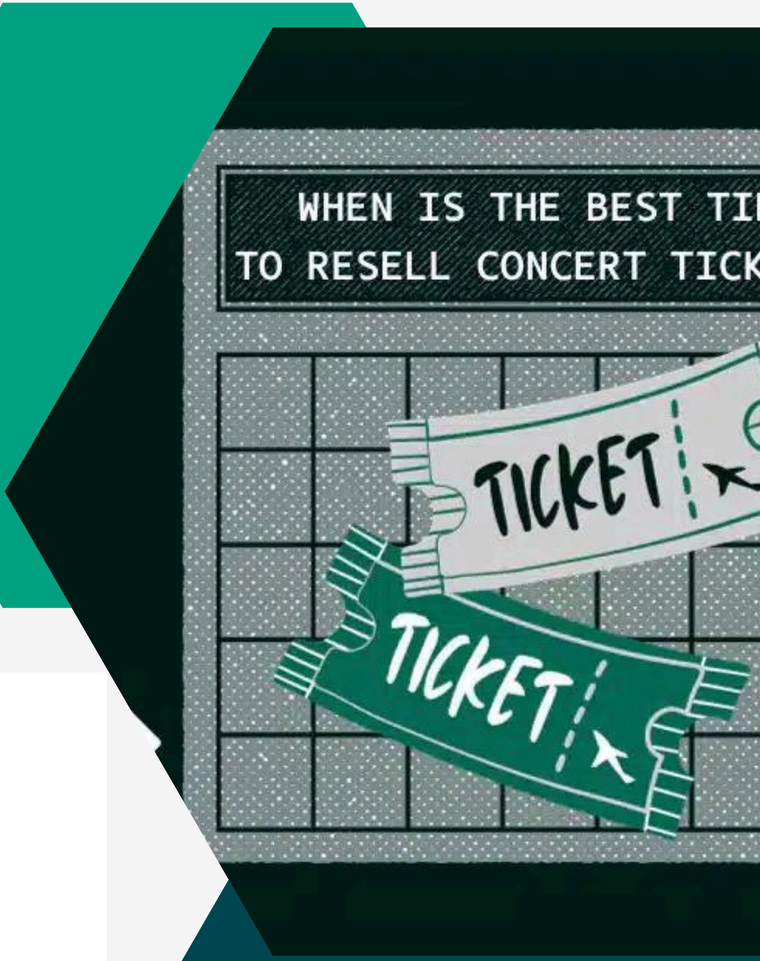
Apres CHoix d'evenement le Client peut choisir le ticket

Il peut reserver Son Ticket selon son Besoins

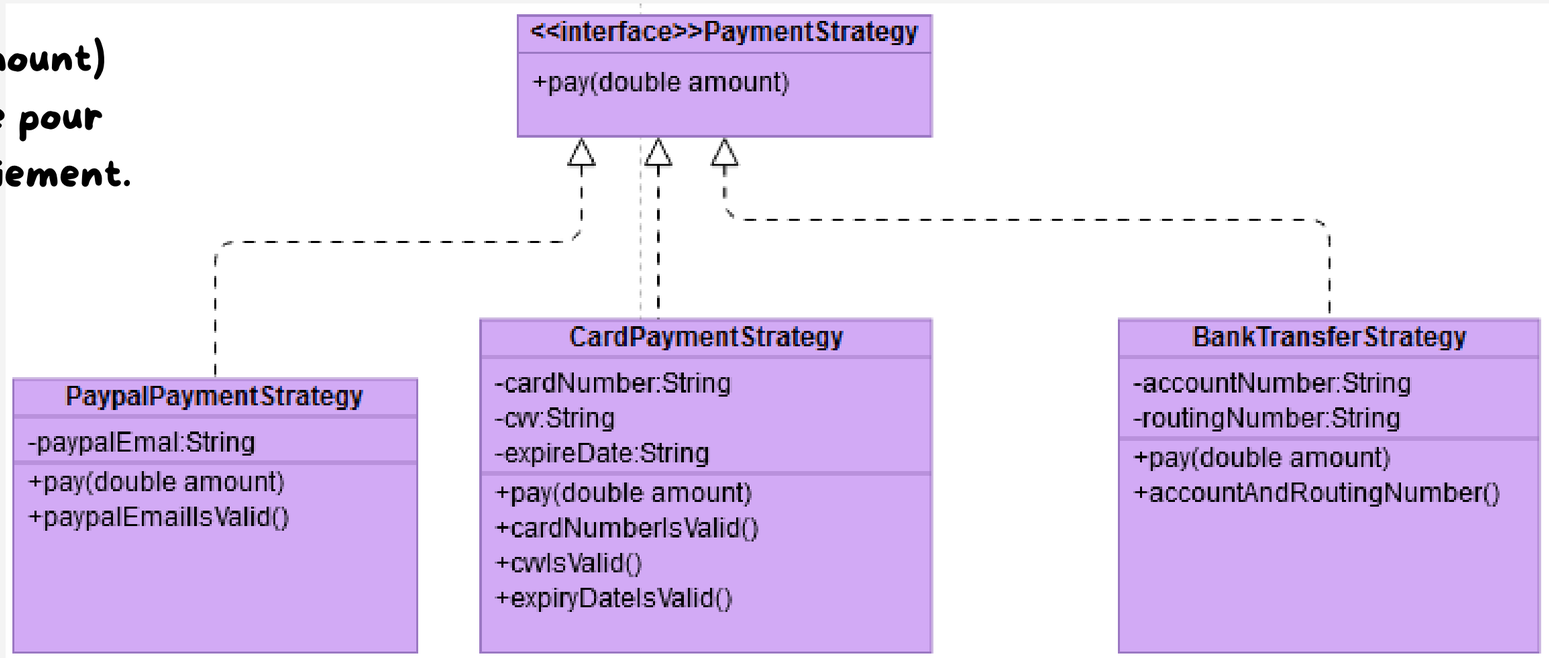


Implementation Strategy

L'utilisateur peut simplement choisir son mode de paiement préféré, et le reste du processus de paiement est géré par le système de manière appropriée, en utilisant la stratégie de paiement correspondante..

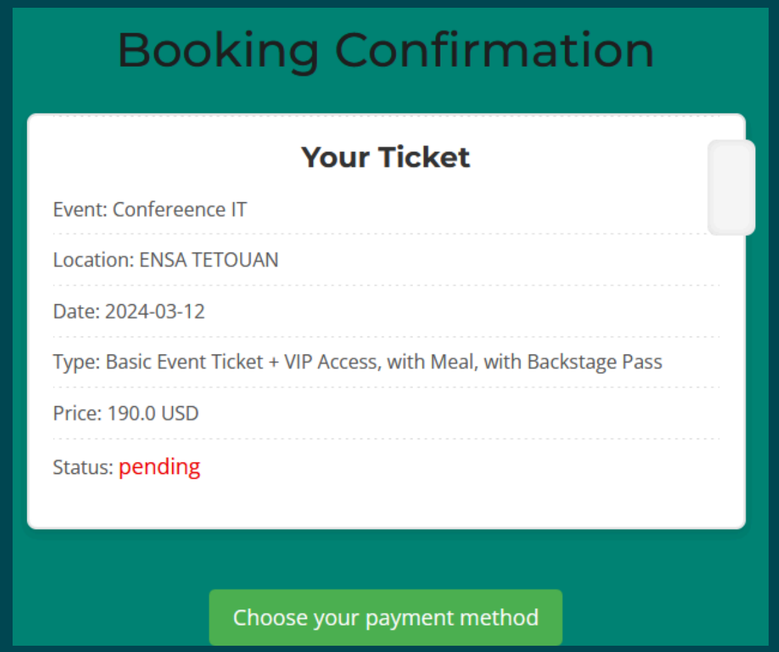


pay(double amount) qui est utilisée pour effectuer le paiement.

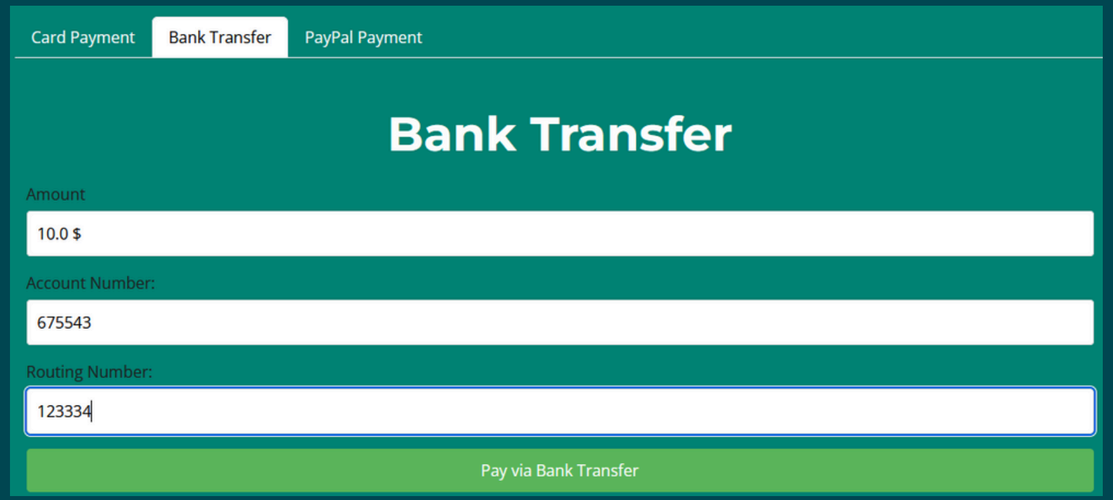


Fonctionnement Strategy

Quand Client Choisit son ticket il va choisir mode de paiement convenable pour lui

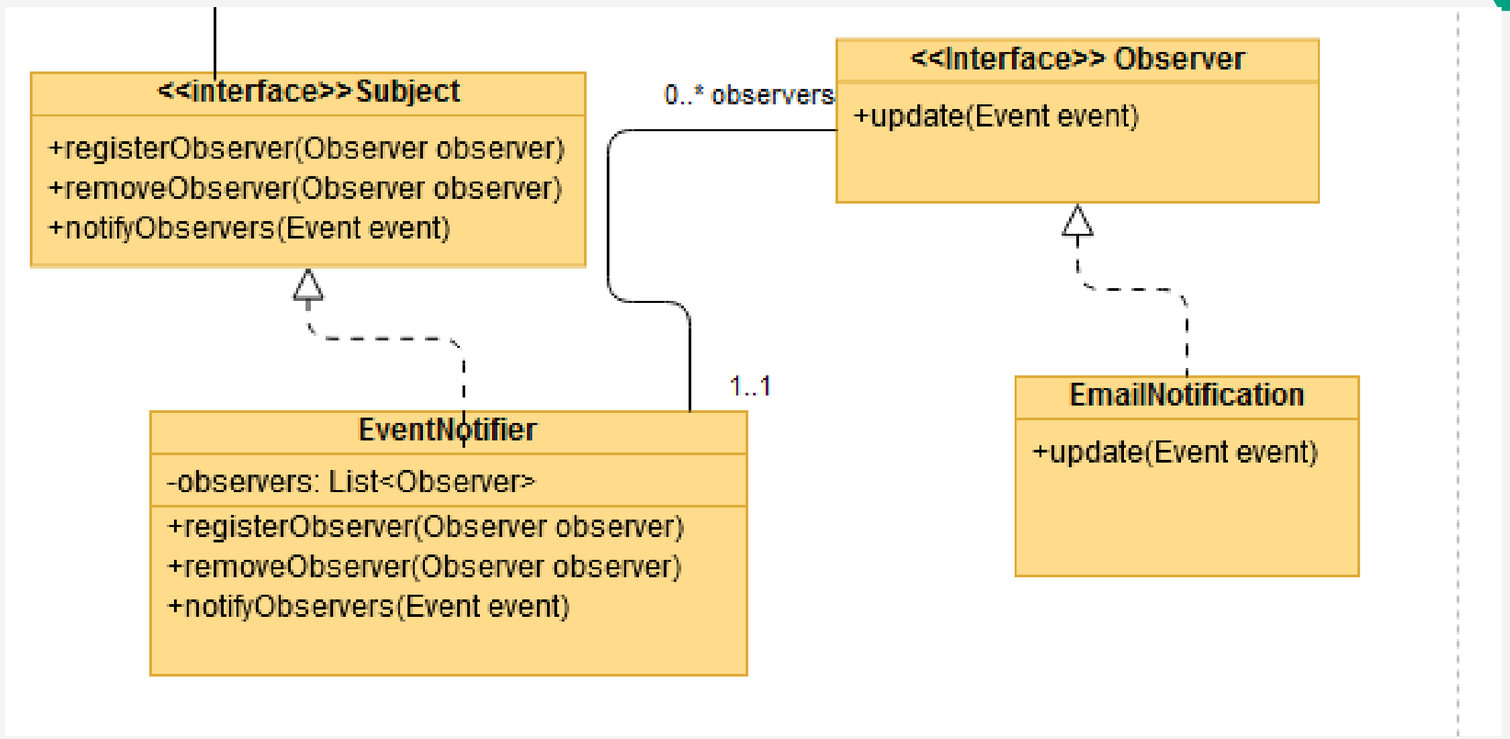


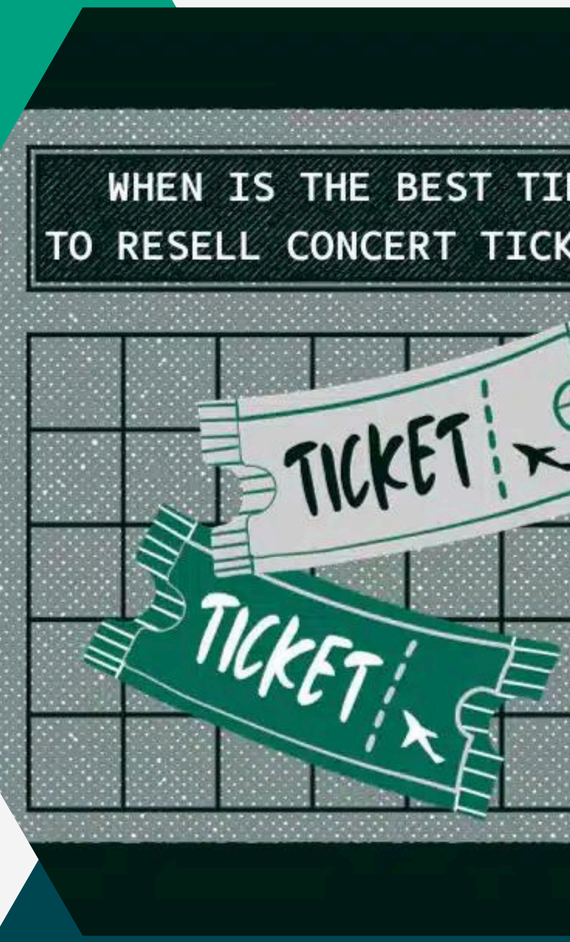
Les clients choisissent le mode de paiement qui leur convient le mieux, sans que le reste du système n'ait à connaître les détails de chaque méthode de paiement.



Implementation Observer

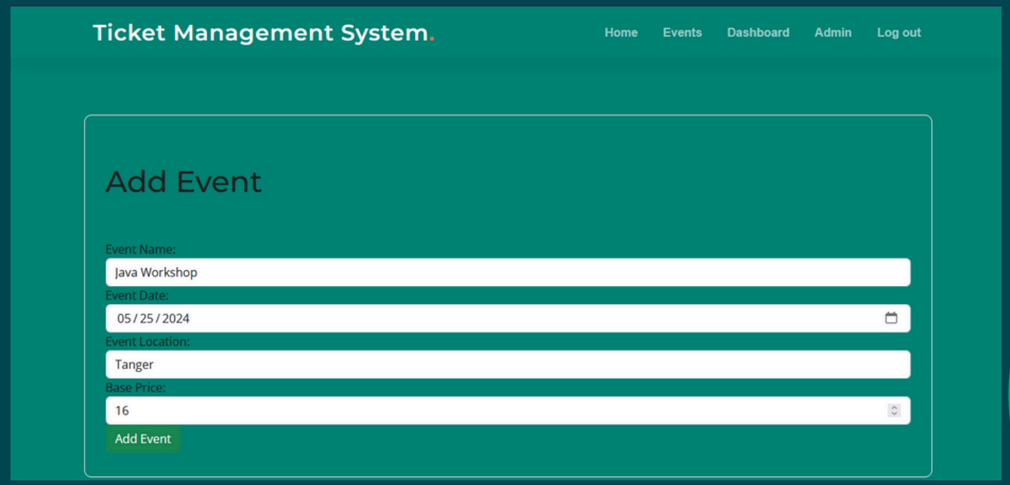
- **Observer dans projet ticket management system permet de déclencher une réaction automatique chaque fois qu'un événement est créé, dans ce cas, l'envoi d'e-mails de notification aux users.**



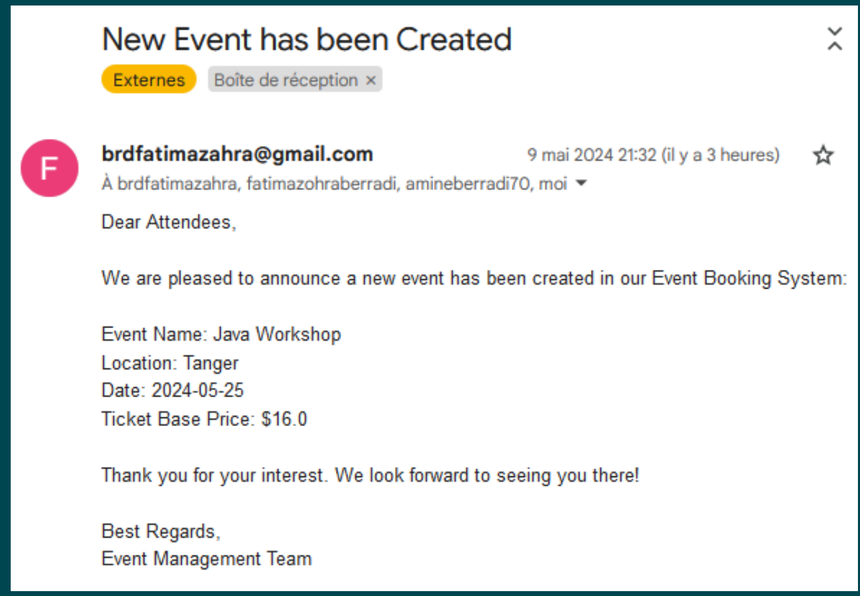


Fonctionnement Observer

L'Admin parmis ses options est d'ajouter evenement

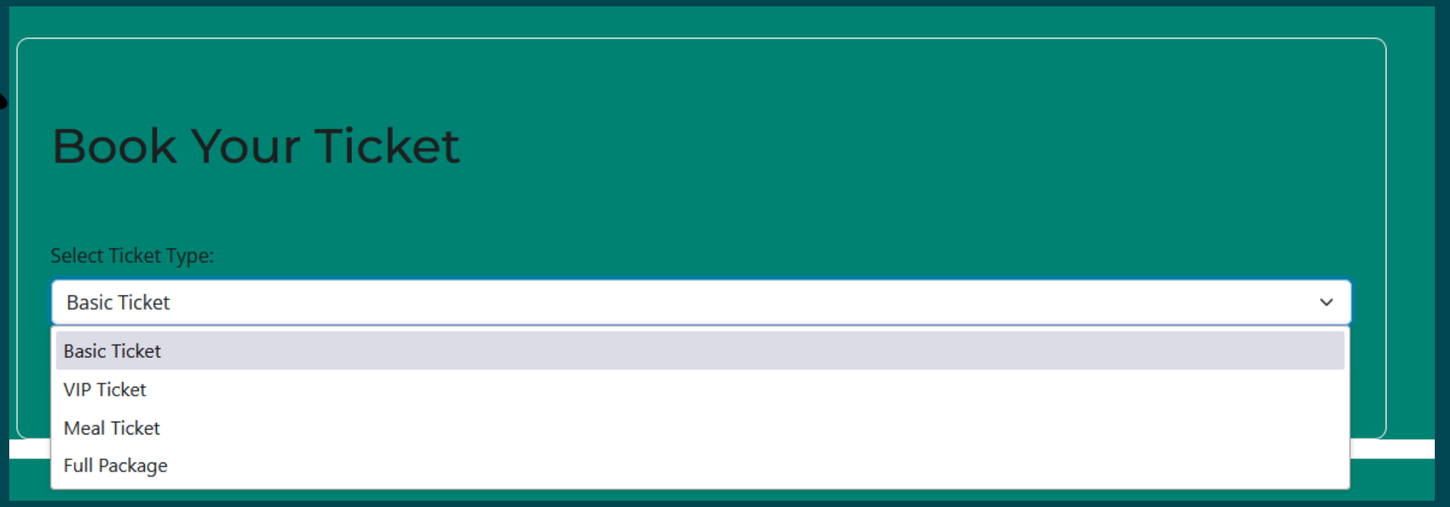


Envoie D'Email pour informer tous les Users->new Event



Fonctionnement Factory

Le pattern de conception "Factory" est utilisé pour créer des objets sans exposer la logique de création au client




```java
package factory;

import Models.Event;
import Models.Ticket;
import Models.User;
import decorator.BackstagePassDecorator;
import decorator.FoodDecorator;
import decorator.VIPDecorator;

public class TicketFactory {

    public static Ticket createBasicTicket
(Event event, User user) {
        return new Ticket(event,user, event.
getBasePrice(), "Basic Event Ticket");
    }

    public static Ticket createVIPTicket(Event
event, User user) {
        Ticket basicTicket = createBasicTicket
(event, user);
        System.out.println(
"basicTicket service toString  "+basicTicket.
toString());
        return new VIPDecorator(basicTicket);
    }

    public static Ticket createMealTicket
(Event event, User user) {
        Ticket basicTicket = createBasicTicket
(event, user);
        return new FoodDecorator(basicTicket);
    }

    public static Ticket
createFullPackageTicket(Event event, User user
) {
        Ticket basicTicket = createBasicTicket
(event, user);
        Ticket vipTicket = new VIPDecorator
(basicTicket);
        Ticket foodTicket = new FoodDecorator
(vipTicket);
        return new BackstagePassDecorator
(foodTicket);
    }

    public static Ticket createTicket(String
type, Event event, User user) {
        switch (type) {
            case "VIP":
                return createVIPTicket
(event, user);
            case "Food":
                return createMealTicket
(event, user);
            case "FullPackage":
                return createFullPackageTicket
(event, user);
            case "Basic":
            default:
                return createBasicTicket
(event, user);
        }
    }

}
```

Grâce au pattern Factory, il est facile d'ajouter de nouveaux types de billets à l'avenir. Vous pouvez simplement étendre la TicketFactory en ajoutant de nouvelles méthodes de création pour ces nouveaux types de billets.

Ticket Management
System