ASI - Atelier 1 : Pattern MVC et Web Statique / Service / Dynamique

Pattern MVC :

MVC est l'acronyme de "Modèle-Vue-Contrôleur", un design pattern qui est souvent utilisé pour développer des applications Web et des logiciels en général.

Le principe du Pattern MVC consiste à séparer les différentes couches d'une application en trois composants principaux :

1. Le Modèle : Cette couche est responsable de la gestion des données de l'application. Le modèle peut inclure des classes qui représentent les entités de l'application et les interactions avec la base de données.
2. La Vue : Cette couche est responsable de l'affichage des données de l'application à l'utilisateur. Elle peut inclure des fichiers HTML, des feuilles de style CSS, des fichiers JavaScript et tout autre code nécessaire pour générer l'interface utilisateur.
3. Le Contrôleur : Cette couche est responsable de la gestion des interactions de l'utilisateur avec l'application. Le contrôleur peut inclure des classes qui traitent les requêtes HTTP, récupèrent les données du modèle, et les passent à la vue pour l'affichage.

Le modèle, la vue et le contrôleur sont indépendants les uns des autres et peuvent être développés séparément, ce qui permet une meilleure modularité de l'application. Cette approche facilite également la maintenance et l'évolution de l'application, car chaque couche peut être modifiée sans affecter les autres.

En utilisant le Pattern MVC, les développeurs peuvent également améliorer la qualité et la lisibilité du code en séparant les préoccupations de chaque couche. Par exemple, le code de l'affichage (la vue) ne devrait pas inclure de la logique de traitement ou d'accès aux données (le modèle).

En résumé, le Pattern MVC est une approche de conception qui permet de séparer les différentes couches d'une application en trois composants principaux (modèle, vue et contrôleur), ce qui facilite la maintenance et l'évolution de l'application, ainsi que l'amélioration de la qualité et la lisibilité du code.

MVC dans notre projet :

Notre prototype en Web Statique + Web Service respect le model MVC car il est divisé en trois parties. La première partie est la partie API fournit avec le projet. Elle correspond au Modèle car c'est elle qui est responsable de la gestion des données de l'application, c'est à elle que l'on fait les requêtes en mode « Post » ou « Get » pour respectivement créer ou rechercher une carte. La deuxième partie est composée de nos fichiers HTML, CSS, et aussi des différentes librairies (Templates fournis). Ces pages permettent la mise en place de la Vue pour l'utilisateur. Finalement la gestion des interactions de l'utilisateur avec l'application ainsi que les requêtes HTTP, les récupérations des données du modèle, et leur affichage est géré par nos fichier JavaScripts ou une classe qui forment notre contrôleur. On a donc bien un prototype qui respect le modelé MVC même si les différentes parties ne sont pas complétement dissociés.

Web Statique / Service / Dynamique :

L'approche Web statique implique la création de pages Web avec un contenu qui ne change pas souvent et qui est généralement stocké dans des fichiers HTML, CSS et JavaScript.

Avantages de l'approche Web statique :

1. Vitesse : Les pages Web statiques sont très rapides à charger car elles ne nécessitent pas de traitement côté serveur avant d'être affichées dans le navigateur.
2. Sécurité : Étant donné que les pages sont générées à l'avance, il y a peu de risques de vulnérabilités de sécurité liées à la manipulation de données dynamiques.
3. Faible coût : La création de pages Web statiques est relativement peu coûteuse car elle ne nécessite pas de compétences de développement de serveur ou de base de données.

Inconvénients de l'approche Web statique :

1. Difficulté à maintenir le contenu : Lorsque le contenu doit être modifié, il est souvent nécessaire de modifier manuellement les fichiers HTML, CSS et JavaScript, ce qui peut être fastidieux et source d'erreur.
2. Limitations de la fonctionnalité : Les pages Web statiques ne peuvent pas offrir de fonctionnalités avancées telles que les interactions en temps réel ou la personnalisation dynamique du contenu.
3. Inadapté aux sites de grande envergure : Les sites Web qui ont besoin de beaucoup de contenu dynamique ou qui reçoivent beaucoup de trafic peuvent rencontrer des problèmes de mise à l'échelle avec une approche statique.

L'approche Web dynamique, en revanche, utilise un serveur Web pour générer les pages Web à la volée en fonction des requêtes des utilisateurs. Cette approche est souvent associée à l'utilisation d'un serveur d'applications et d'une base de données pour stocker et gérer les données.

Avantages de l'approche Web dynamique :

1. Facilité de gestion du contenu : Le contenu peut être facilement mis à jour en utilisant un système de gestion de contenu tel que WordPress ou Drupal.
2. Fonctionnalités avancées : Les pages Web dynamiques peuvent offrir des fonctionnalités avancées telles que les formulaires en ligne, les systèmes de réservation en temps réel et les paniers d'achat.
3. Adapté aux sites de grande envergure : Les sites Web qui ont besoin de beaucoup de contenu dynamique ou qui reçoivent beaucoup de trafic peuvent être facilement mis à l'échelle avec une approche dynamique.

Inconvénients de l'approche Web dynamique :

1. Coût : Les coûts de développement et de maintenance d'un site Web dynamique sont généralement plus élevés que ceux d'un site Web statique.
2. Sécurité : Étant donné que les pages Web sont générées à la volée, il y a un risque accru de vulnérabilités de sécurité liées à la manipulation de données dynamiques.
3. Vitesse : Les pages Web dynamiques peuvent être plus lentes à charger car elles nécessitent un traitement côté serveur avant d'être affichées dans le navigateur.

De manière générale, les avantages du Web statique vont correspondre aux inconvénients du Web dynamique et inversement.

Un web service est une application qui permet d'échanger des données avec d'autres applications web, même si ces dernières sont construites dans des langages de programmation différents. Les plus connus sont SOAP, REST et HTTP.