Module 2: Conception d'applications Web ASP.NET MVC 5

# ****Contenu:****

|  |  |
| --- | --- |
|  | [Aperçu du module](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C02.html#P1) |
| **Leçon 1:** | [Planification dans la phase de conception du projet](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C02.html#P2) |
| **Leçon 2:** | [Conception de modèles, de contrôleurs et de vues](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C02.html#P3) |
| **Laboratoire:** | [Conception d'applications Web ASP.NET MVC 5](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C02.html#P4) |
|  | [Examen du module et points à retenir](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C02.html#P5) |

# ****Aperçu du module****

Microsoft ASP.NET MVC 5 est un modèle de programmation que vous pouvez utiliser pour créer des applications Web puissantes et complexes. Cependant, tous les projets de développement complexes, et les grands projets en particulier, peuvent être difficiles et complexes à comprendre pleinement. Sans une compréhension complète des objectifs d'un projet, vous ne pouvez pas développer une solution efficace au problème du client. Vous devez savoir comment identifier un ensemble de besoins métier et planifier une application Web MVC pour répondre à ces besoins. Le plan de projet que vous créez garantit aux parties prenantes que vous comprenez leurs exigences et communique la fonctionnalité de l'application Web, son interface utilisateur, sa structure et son stockage de données aux développeurs qui la créeront. En rédigeant un plan de projet détaillé et précis,

### ****Objectifs****

Après avoir terminé ce module, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Planifiez l'architecture globale d'une application Web MVC 5 et tenez compte des aspects tels que la gestion de l'état. |
| • | Planifiez les modèles, les contrôleurs et les vues nécessaires pour implémenter un ensemble donné d'exigences fonctionnelles. |

# Leçon 1: Planification dans la phase de conception du projet

Avant que vous et votre équipe de développeurs ne planifiez une application Web Model-View-Controller (MVC) ou écrivez du code, vous devez avoir une compréhension approfondie de deux choses: le problème commercial que vous essayez de résoudre et les composants ASP.NET que vous peut utiliser pour créer une solution. Avant de concevoir une architecture d'application Web et sa base de données, vous devez savoir comment identifier les exigences des utilisateurs potentiels d'une application Web.

## ****Objectifs de la leçon****

Après avoir terminé cette leçon, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Décrivez les différents modèles de développement de projet. |
| • | Décrivez comment collecter des informations sur les exigences du projet lors de la création d'applications Web MVC 5. |
| • | Déterminez les exigences fonctionnelles et les problèmes commerciaux lors de la création d'applications Web. |
| • | Expliquez comment planifier la conception de la base de données lors de la création d'une application Web. |
| • | Décrivez les architectures d'applications distribuées possibles. |
| • | Décrivez les options de planification de la gestion de l'état dans une application Web. |
| • | Décrivez les options de planification de la mondialisation et de la localisation d'une application Web. |
| • | Déterminez les aspects critiques de la conception d'applications Web. |

## ****Méthodologies de développement de projet****

Le développement d'une application Web ou d'une application intranet est souvent un processus complexe qui implique de nombreux développeurs dans différentes équipes exécutant divers rôles. Pour organiser le processus de développement et vous assurer que tout le monde dans un projet travaille ensemble, vous pouvez utiliser un large éventail de méthodologies de développement. Ces méthodologies de développement décrivent les phases du projet de développement, les rôles que les gens jouent, les livrables qui concluent chaque phase et d'autres aspects du projet. Vous devez choisir une méthodologie de développement à un stade précoce d'un projet. De nombreuses organisations ont une méthodologie standard qu'elles utilisent toujours pour le développement de projets.

Certaines méthodologies de développement de projet incluent le modèle en cascade, le modèle de développement itératif, le modèle de prototypage, le modèle de développement logiciel agile, la programmation extrême et le développement piloté par les tests.

**Modèle de cascade**

Le modèle de cascade est une méthodologie précoce qui définit les phases suivantes d'un projet:

|  |  |
| --- | --- |
| • | Analyse de faisabilité.Dans cette phase, les planificateurs et les développeurs étudient et déterminent les approches et les technologies qui peuvent être utilisées pour créer l'application logicielle. |
| • | Analyse des besoins.Au cours de cette phase, les planificateurs et les analystes interrogent les utilisateurs, les gestionnaires, les administrateurs et les autres parties prenantes de l'application logicielle pour déterminer leurs besoins. |
| • | Conception d'applications.Dans cette phase, les planificateurs, les analystes et les développeurs enregistrent une solution proposée. |
| • | Codage et tests unitaires.Dans cette phase, les développeurs créent le code et testent individuellement les composants qui composent le système. |
| • | Intégration et test du système.Dans cette phase, les développeurs intègrent les composants qu'ils ont construits et testent le système dans son ensemble. |
| • | Déploiement et maintenance.Dans cette phase, les développeurs et les administrateurs déploient la solution afin que les utilisateurs puissent commencer à utiliser l'application logicielle. |

Le modèle en cascade classe le projet de développement en phases distinctes avec une définition claire des livrables pour chaque phase. Le modèle met également l'accent sur l'importance des tests. Cependant, le client ne reçoit aucun logiciel fonctionnel pour examen jusqu'à la fin du projet. Cela rend difficile la gestion des modifications apportées à la conception en réponse aux commentaires de la version bêta ou la gestion des circonstances modifiées.

**Modèle de développement itératif**

Lorsque vous utilisez un modèle de développement itératif, vous divisez le projet en petites parties. Pour chaque pièce, vous effectuez les activités liées à toutes les étapes du modèle de cascade. Le projet est construit étape par étape, avec des tests approfondis à chaque étape pour garantir la qualité.

Dans un projet itératif, vous pouvez effectuer une action corrective à la fin de chaque itération. Ces corrections peuvent refléter une meilleure compréhension des problèmes métier, des commentaires des utilisateurs perspicaces ou une meilleure compréhension des technologies que vous avez utilisées pour créer la solution. Étant donné que des exigences sont ajoutées à la fin de chaque itération, les projets itératifs nécessitent beaucoup d'efforts de gestion de projet et comportent souvent un dépassement des efforts et du calendrier planifiés.

**Modèle de prototypage**

Le modèle de prototypage convient à un projet où vous commencez avec quelques exigences commerciales peu ou mal définies. Cette situation se produit lorsque les clients ou parties prenantes n'ont qu'une compréhension vague de leurs besoins et de la manière de les résoudre. Dans cette approche, les développeurs créent une version simplifiée de l'application logicielle, puis sollicitent les commentaires des parties prenantes. Ces commentaires sur le prototype sont utilisés pour définir les exigences détaillées, que les développeurs utiliseront dans la prochaine itération pour créer une solution qui correspond aux besoins des parties prenantes afin de mieux les aider à effectuer leur travail.

Après deux itérations ou plus, lorsque les parties prenantes et les développeurs parviennent à un consensus sur les exigences, une solution complète est construite et testée. Le modèle de prototypage, cependant, peut conduire à une application mal conçue car à aucun stade du projet il n'y a une concentration claire sur l'architecture globale.

**Modèle de développement logiciel agile**

Le modèle en cascade, le modèle de développement itératif et le modèle de prototypage sont basés sur le principe que les exigences métier et d'autres facteurs ne changent pas du début à la fin du projet. En réalité, cette hypothèse est souvent invalide. Le développement logiciel agile est une méthodologie conçue pour intégrer l'évolution des circonstances et des exigences tout au long du processus de développement. Les projets agiles se caractérisent par:

|  |  |
| --- | --- |
| • | Développement incrémental. Le logiciel est développé selon des cycles rapides qui s'appuient sur des cycles antérieurs. Chaque itération est minutieusement testée. |
| • | Accent mis sur les personnes et les interactions.Les développeurs écrivent du code en fonction de ce que les gens font dans leur rôle, plutôt que de ce que font les outils de développement. |
| • | Accent sur les logiciels fonctionnels.Au lieu d'écrire des documents de conception détaillés pour les parties prenantes, les développeurs écrivent des solutions que les parties prenantes peuvent évaluer à chaque itération pour valider si cela répond à une exigence. |
| • | Collaboration étroite avec les clients.Les développeurs discutent quotidiennement avec les clients et les parties prenantes pour vérifier les exigences. |

**Programmation extrême**

La programmation extrême a évolué à partir du développement logiciel agile. Dans la programmation extrême, la phase de conception préliminaire est réduite au minimum et les développeurs se concentrent sur la résolution de quelques tâches critiques. Dès que ces tâches critiques sont finalisées, les développeurs testent la solution simplifiée et obtiennent les commentaires des parties prenantes. Ces commentaires aident les développeurs à identifier les exigences détaillées, qui évoluent au cours du cycle de vie du projet.

La programmation extrême définit une user story pour chaque rôle d'utilisateur. Une user story décrit toutes les interactions qu'un utilisateur avec un rôle spécifique peut effectuer avec l'application terminée. La collection de toutes les user stories pour tous les rôles utilisateur décrit l'ensemble de l'application.

Dans la programmation extrême, les développeurs travaillent souvent par paires. Un développeur écrit le code et l'autre développeur examine le code pour s'assurer qu'il utilise des solutions simples et adhère aux meilleures pratiques. Le développement piloté par les tests est une pratique de base dans la programmation extrême.

**Lecture supplémentaire:**Pour plus d'informations sur le modèle de programmation extrême, accédez à<http://go.microsoft.com/fwlink/?LinkID=288945&clcid=0x409>.

**Développement piloté par les tests**

Dans le développement piloté par les tests (TDD), les développeurs écrivent le code de test comme première tâche dans une itération donnée. Par exemple, si vous souhaitez écrire un composant qui stocke les détails de la carte de crédit, vous commencez par écrire les tests qu'un tel composant réussirait. Il peut s'agir de savoir s'il vérifie correctement les formats numériques, s'il écrit correctement les chaînes dans une table de base de données ou s'il appelle correctement les services bancaires. Après avoir défini les tests, vous écrivez le composant pour réussir ces tests.

Dans les itérations suivantes, les tests de carte de crédit restent en place. Cela garantit que si vous interrompez la fonctionnalité de la carte de crédit, peut-être en refactorisant le code ou en ajoutant un nouveau constructeur, vous le découvrirez rapidement car les tests échouent.

Dans Microsoft Visual Studio 2017, vous pouvez définir un projet de test, dans la même solution que le projet principal, pour stocker et exécuter des tests unitaires. Après avoir écrit les tests, vous pouvez les exécuter individuellement ou en groupes après chaque changement de code. Étant donné que les projets MVC ont le modèle, la vue et le code du contrôleur dans des fichiers séparés, il est facile de créer des tests unitaires pour tous les aspects du comportement de l'application. C'est un avantage majeur de MVC par rapport aux pages Web et aux formulaires Web.

**Langage de modélisation unifié**

Le langage de modélisation unifié (UML) est une notation standard de l'industrie pour enregistrer la conception de toute application qui utilise la technologie orientée objet. UML n'est pas un modèle de développement. Au contraire, les diagrammes UML sont souvent utilisés pour planifier et documenter l'architecture et les composants des applications, dans toutes les méthodologies de développement de projet. Lorsque vous utilisez UML pour concevoir et enregistrer une application, vous créez une plage de diagrammes avec des formes et des connecteurs standard. Ces diagrammes peuvent être divisés en trois classes:

|  |  |
| --- | --- |
| • | Diagrammes de comportement. Ces diagrammes décrivent le comportement des utilisateurs, des applications et des composants d'application. |
| • | Diagrammes d'interaction. Ces diagrammes sont un sous-ensemble de diagrammes de comportement qui se concentrent sur les interactions entre les objets. |
| • | Diagrammes de structure.Ces schémas représentent les éléments d'une application qui sont indépendants du temps. Cela signifie qu'ils ne changent pas pendant la durée de vie de l'application. |

**Question:**Quels aspects du développement logiciel agile et de la programmation extrême pourraient préoccuper les clients? Quels aspects pourraient les rassurer?

## ****Collecte des exigences****

Lorsqu'un projet est mis en service, vous avez besoin d'une vision de la solution. La vision peut souvent être vague et nécessiter une enquête approfondie avant que vous puissiez ajouter des détails et vous assurer que toutes les exigences des parties prenantes sont couvertes par l'application Web que vous créez. Ces exigences peuvent être de deux types:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| • | Exigences fonctionnelles. Ces exigences décrivent comment l'application se comporte et répond aux utilisateurs. Les exigences fonctionnelles sont souvent appelées exigences comportementales. Ils incluent:   |  |  | | --- | --- | | o | Exigences relatives à l'interface utilisateur. Ces exigences décrivent comment l'utilisateur interagit avec une application. | | o | Conditions d'utilisation. Ces exigences décrivent ce qu'un utilisateur peut faire avec l'application. | | o | Besoins de l'entreprise. Ces exigences décrivent comment l'application remplira les fonctions métier. | |
| • | Les pré-requis techniques. Ces exigences décrivent les caractéristiques techniques de l'application et concernent la disponibilité, la sécurité ou les performances. Ces exigences sont parfois appelées exigences non fonctionnelles ou non comportementales. |

Vous recueillez généralement les exigences en interrogeant les parties prenantes telles que les utilisateurs, les administrateurs, les autres développeurs, les membres du conseil d'administration, les titulaires de budget et les responsables d'équipe. Chacun de ces groupes aura un ensemble différent de priorités que l'application doit respecter.

**Scénarios d'utilisation et cas d'utilisation**

Une méthode courante par laquelle vous pouvez créer un ensemble d'exigences d'interface utilisateur, d'exigences métier et d'exigences d'utilisation consiste à demander aux utilisateurs ce qu'ils feront de l'application Web que vous créez. Vous pouvez enregistrer ces actions sous forme de scénarios d'utilisation et de cas d'utilisation.

Un scénario d'utilisation est un exemple concret spécifique, avec des noms et des valeurs d'entrée suggérées, d'une interaction entre l'application et un utilisateur. Voici un exemple simple:

|  |  |
| --- | --- |
| 1. | Roger Lengel clique sur le lien Ajouter une photo dans le menu principal du site. |
| 2. | Roger fournit l'entrée, RogerL, dans la case Nom d'utilisateur et le mot de passe dans la case Mot de passe pour s'authentifier sur le site. |
| 3. | Roger tape le titre, Sunset, pour la photo. |
| 4. | Roger parcourt le fichier JPEG de sa nouvelle photo. |
| 5. | Roger clique sur le bouton Télécharger. |
| 6. | L'application Web stocke la nouvelle photo et affiche la galerie de photos sur Roger. |

Un cas d'utilisation est similaire à un scénario d'utilisation, mais il est plus généralisé. Les cas d'utilisation n'incluent pas les noms d'utilisateur ni les valeurs d'entrée. Ils décrivent plusieurs chemins d'une interaction, qui dépendent de ce que l'utilisateur fournit comme entrée ou d'autres valeurs. Voici un exemple simple:

|  |  |
| --- | --- |
| 1. | L'utilisateur clique sur le lien Ajouter une photo dans le menu principal du site. |
| 2. | Si l'utilisateur est anonyme, la page de connexion s'affiche et l'utilisateur fournit des informations d'identification. |
| 3. | Si les informations d'identification sont correctes, la vue CreatePhoto s'affiche. |
| 4. | L'utilisateur tape un titre. |
| 5. | L'utilisateur spécifie le fichier photo à télécharger. |
| 6. | L'utilisateur tape éventuellement une description pour la photo. |
| 7. | L'utilisateur clique sur le bouton Télécharger. |
| 8. | L'application Web stocke la nouvelle photo et affiche la galerie de photos à l'utilisateur. |

**Noter:**À l'instar des descriptions verbales, vous pouvez utiliser des diagrammes de cas d'utilisation UML pour enregistrer les cas d'utilisation de votre application Web.

En analysant les scénarios d'utilisation et les cas d'utilisation, vous pouvez identifier les exigences fonctionnelles de tous types. Par exemple, à partir du cas d'utilisation précédent, vous pouvez identifier l'exigence d'interface utilisateur suivante: La page Web qui permet aux utilisateurs de créer une nouvelle photo doit inclure des zones de texte Titre et Description, un contrôle d'entrée de fichier pour le fichier photo et un bouton Télécharger pour enregistrez la photo.

**Modélisation des exigences agiles**

Dans un modèle de cascade traditionnel ou un modèle de développement itératif, les développeurs et les analystes étudient et enregistrent les exigences fonctionnelles et techniques précises et détaillées à un stade précoce du projet, qui ne changent pas par la suite. En revanche, dans un projet basé sur un modèle de développement agile, les développeurs reconnaissent que les exigences peuvent changer à tout moment au cours du développement. L'analyse des besoins est donc caractérisée comme suit:

|  |  |
| --- | --- |
| • | Modélisation des besoins initiaux. Dans la phase de conception initiale, les développeurs identifient et enregistrent quelques cas d'utilisation généraux de manière informelle sans détails complets. |
| • | Modélisation juste à temps.Avant d'écrire du code qui implémente un cas d'utilisation, un développeur en discute avec les utilisateurs concernés. À ce stade, le développeur ajoute des détails complets au cas d'utilisation. Dans un projet de développement agile, les développeurs parlent aux utilisateurs et aux autres parties prenantes à tout moment, et pas seulement au début et à la fin du projet. |
| • | Test d'acceptation. Un test d'acceptation est un test que l'application doit réussir pour que toutes les parties prenantes acceptent et approuvent l'application. Lorsque vous identifiez une exigence fonctionnelle, vous pouvez également spécifier un test d'acceptation correspondant qui doit être exécuté pour garantir que les exigences sont satisfaites. |

**Témoignages d'utilisateurs dans la programmation extrême**

Dans les projets de programmation extrêmes, les développeurs effectuent encore moins d'analyse des exigences fonctionnelles au début du projet, par rapport aux autres modèles de développement. Ils créent des user stories au lieu de cas d'utilisation ou de scénarios utilisateur. Une user story est un exemple très large d'interaction entre l'application et un utilisateur, et elle est souvent énoncée en une seule phrase comme l'illustre l'exemple suivant:

|  |  |
| --- | --- |
| • | Les utilisateurs peuvent télécharger des photos et donner aux nouvelles photos un titre et une description. |

Les user stories ne contiennent que le minimum de détails pour permettre aux développeurs d'estimer l'effort impliqué dans leur développement. Les programmeurs extrêmes discutent de chaque user story avec les parties prenantes juste avant d'écrire du code pour implémenter chaque user story.

**Question:**Si un client vous demande de garantir une disponibilité de 95%, s'agit-il d'une exigence fonctionnelle ou technique?

## ****Planification de la conception de la base de données****

Lorsque vous avez une bonne compréhension des exigences fonctionnelles et des exigences techniques de l'application Web proposée, vous pouvez commencer à concevoir l'implémentation physique de l'application. L'un des objets physiques les plus importants à planifier est une ou plusieurs bases de données. Bien que toutes les applications Web n'utilisent pas de bases de données pour le stockage des informations, elles constituent un objet sous-jacent pour une majorité de sites et vous les utiliserez dans la plupart de vos projets.

**Modélisation logique**

Vous pouvez commencer la conception de vos données à un niveau élevé en créant des diagrammes de modèle de domaine UML et des diagrammes de modèle de données logique (LDM).

Un diagramme de modèle de domaine, également appelé modèle de données conceptuel, montre les objets conceptuels de haut niveau gérés par votre application Web. Par exemple, dans une application Web de commerce électronique, le modèle de domaine comprend les concepts de clients, de paniers d'achat et de produits. Le modèle de domaine n'inclut pas les détails des propriétés de chaque concept, mais montre les relations entre les concepts. Utilisez le modèle de domaine pour enregistrer vos conversations initiales avec les parties prenantes.

En substance, un LDM est un diagramme de modèle de domaine avec des détails supplémentaires ajoutés. Vous pouvez utiliser des diagrammes LDM pour fournir plus de détails, tels que les propriétés et les types de données, pour les concepts que vous avez définis dans le modèle de domaine. Notez que les objets du LDM ne correspondent pas aux tables de la base de données. Par exemple, l'objet de panier d'achat peut afficher des données à la fois de la base de données client et des tables de base de données de produits.

**Structure de la base de données physique**

Vous devez prendre en compte les objets de base de données suivants dans votre plan de projet:

|  |  |
| --- | --- |
| • | Les tables.Ce sont les objets de stockage fondamentaux dans une base de données. Lorsque vous définissez une table, vous devez spécifier les colonnes de chaque table. Pour chaque colonne, vous devez définir un type de données tel qu'un entier, une chaîne, généralement le type nvarchar dans SQL Server, ou la date et l'heure. Vous devez également définir la clé primaire de la table - la valeur de cette colonne identifie de manière unique chaque enregistrement et est essentielle pour définir les relations avec les enregistrements dans d'autres tables. |
| • | Vues. Ce sont des présentations courantes de données dans des tableaux et sont basées sur des requêtes. Par exemple, une vue peut joindre deux tables, comme une table de produits et une table de niveaux de stock. |
| • | Procédures stockées. Il s'agit de séquences courantes d'opérations de base de données que vous définissez dans la base de données. Certaines opérations sont complexes et peuvent impliquer une transformation complexe des données. Vous pouvez définir une procédure stockée pour implémenter une routine aussi complexe. |
| • | Sécurité.Vous devez considérer comment l'application Web s'authentifiera auprès du serveur de base de données et comment vous autoriserez l'accès à chaque table de base de données. |

Dans UML, un modèle de données physique est un diagramme qui représente des tables, des colonnes, des types de données et des relations entre des tables.

**Travailler avec les administrateurs de base de données**

Parfois, l'équipe de développeurs a un contrôle total sur la base de données qui sous-tend l'application Web. Cela se produit, par exemple, lorsque l'organisation est petite ou lorsque l'application Web dispose d'un serveur de base de données distinct sans données critiques pour l'entreprise. Cependant, dans les grandes organisations ou dans les projets où la base de données stocke des informations commerciales critiques, il peut y avoir une équipe dédiée d'administrateurs de base de données (DBA). Ces administrateurs de base de données sont généralement hautement qualifiés dans la conception et l'administration de bases de données et leur travail consiste à garantir l'intégrité des données en fonction de la politique de stockage des données de l'organisation.

Si la base de données de votre projet est administrée par l'équipe DBA, il est essentiel de bien communiquer avec elle. Vous devez consulter les administrateurs de base de données pour connaître leurs besoins. Ils imposent souvent une liste d'exigences techniques que d'autres parties prenantes peuvent ne pas comprendre. Lorsque vous créez et déployez l'application Web, les administrateurs de base de données sont responsables de la création de bases de données sur les bons serveurs ou clusters et de l'attribution des autorisations. Les administrateurs de base de données sont des contributeurs essentiels à la livraison de l'application Web.

**Conception de bases de données dans le développement Agile et la programmation extrême**

Le développement agile et la programmation extrême sont caractérisés par une quantité relativement faible de planification initiale et de documentation, et reconnaissent que les exigences sont susceptibles de changer tout au long du projet de développement. Si vous utilisez ces méthodologies de développement, vous ne créerez des modèles de domaine qu'au cours de la phase de planification initiale de votre projet. Vous ne développez pas de LDM ou de modèles de données physiques tant que vous n'avez pas écrit de code qui implémente les exigences fonctionnelles. Au cours de la phase de développement, vous discuterez des exigences avec les utilisateurs et les administrateurs de base de données et vous créerez des LDM et des modèles de données physiques juste avant d'écrire le code.

Dans le développement agile et la programmation extrême, la conception de la base de données change tout au long du projet jusqu'au déploiement. Par conséquent, les développeurs doivent être en mesure de modifier la base de données chaque fois que cela est nécessaire sans consulter les administrateurs de base de données ou se conformer à des politiques de données complexes. Pour cette raison, vous devez utiliser une base de données distincte hébergée sur un serveur de développement dédié.

Dans certains cas, le projet de développement fonctionne avec une base de données qui existe déjà. Par exemple, vous développez peut-être une application Web qui présente et modifie les informations de la base de données des employés de l'entreprise sur l'intranet. Dans de tels cas, la base de données ne change pas au fur et à mesure que le code est développé, mais les exigences fonctionnelles et techniques peuvent encore changer. Vous devez copier la base de données sur un serveur de base de données de développement isolé pour vous assurer que votre code de développement ne modifie pas par erreur les données critiques pour l'entreprise.

**Question:**Vous souhaitez implémenter un panier dans votre application Web. Combien de modèles de données logiques sont nécessaires? Combien de tables de base de données sont nécessaires?

## ****Planification des applications distribuées****

Pour les petites applications Web avec moins de trafic utilisateur, vous pouvez choisir d'héberger tous les composants de votre application Web sur un seul serveur. Cependant, à mesure que votre application Web se développe, un déploiement distribué, dans lequel différents serveurs hébergent des composants distincts de l'application, est souvent utilisé. Les applications Web distribuées utilisent souvent une architecture en couches:

|  |  |
| --- | --- |
| • | Couche de présentation. Les composants de cette couche implémentent l'interface utilisateur et la logique de présentation. Si vous créez une application Web MVC, les vues et les contrôleurs constituent votre couche de présentation. |
| • | Couche de logique métier. Les composants de cette couche implémentent des objets métier de haut niveau tels que des produits ou des clients. Si vous créez une application Web MVC, les modèles constituent votre couche de logique métier. |
| • | Couche d'accès aux données. Les composants de cette couche implémentent des opérations d'accès à la base de données et des objets de base de données abstraits, tels que des tables, à partir d'objets métier. Par exemple, un objet métier produit peut inclure des données des tables de base de données Products et StockLevels. Si vous créez une application Web MVC, les modèles constituent souvent à la fois la logique métier et les couches d'accès aux données. Cependant, avec une conception et des pratiques de codage soignées, il est possible de refactoriser le code pour séparer ces couches. |
| • | Couche de base de données. Cette couche contient la base de données elle-même. |

Si vous implémentez une telle architecture en couches dans votre application Web, vous pouvez héberger chaque couche sur des serveurs distincts. Souvent, par exemple, la couche de présentation est hébergée sur un ou plusieurs serveurs IIS, la logique métier et les couches d'accès aux données sont hébergées sur des serveurs de niveau intermédiaire dédiés et la base de données est hébergée sur un serveur SQL dédié. Cette approche présente les avantages suivants:

|  |  |
| --- | --- |
| • | Vous pouvez spécifier du matériel serveur qui correspond étroitement à chaque rôle. Par exemple, un serveur qui héberge des composants de logique métier nécessite de bonnes ressources de mémoire et de traitement. |
| • | Vous pouvez dédier plusieurs serveurs à chaque rôle pour garantir qu'une seule panne de serveur n'entraîne pas d'interruption de service. |
| • | Seuls les serveurs Web doivent se trouver sur le réseau de périmètre. Les serveurs de niveau intermédiaire et les serveurs de base de données peuvent être protégés par deux pare-feu sans accès direct depuis Internet. |
| • | Vous pouvez également héberger des couches et des bases de données de niveau intermédiaire sur un service cloud, tel que Windows Azure. |

**Communication entre les couches**

Lorsqu'un serveur unique héberge tous les composants d'une application Web, les composants de présentation, de logique métier et d'accès aux données s'exécutent en un seul processus dans la mémoire du serveur Web. La communication entre les composants n'est pas un problème. Cependant, lorsque vous exécutez différentes couches sur différents serveurs, vous devez tenir compte de deux facteurs:

|  |  |
| --- | --- |
| • | Comment chaque couche échange-t-elle des informations et des messages? |
| • | Comment chaque serveur authentifie-t-il et sécurise-t-il les communications avec les autres serveurs? |

La communication des informations et de la sécurité s'effectue de différentes manières entre les différentes couches:

|  |  |
| --- | --- |
| • | Entre le navigateur et le serveur Web de la couche présentation. Dans toute application Web, le navigateur Web, où s'exécute la couche de présentation, communique avec le serveur Web à l'aide de HTTP. Si l'authentification est requise, elle est souvent effectuée en échangeant des informations d'identification en texte brut. Vous pouvez également utiliser Secure Sockets Layer (SSL) pour crypter cette communication sensible. |
| • | Entre le serveur Web et le serveur de niveau intermédiaire. Les mécanismes de communication et de sécurité utilisés pour la communication entre le serveur Web et le serveur de niveau intermédiaire dépendent de la technologie que vous utilisez pour créer les composants de logique métier:   |  |  | | --- | --- | | o | Services Web: si vous implémentez des objets métier et des classes d'accès aux données en tant que services Web, les composants de la couche de présentation communiquent avec les services Web à l'aide de HTTP. Vous pouvez effectuer l'authentification à l'aide du protocole Kerberos qui fait partie de l'authentification intégrée de Windows ou en utilisant du texte brut chiffré avec SSL. | | o | Services Windows Communication Foundation (WCF): si vous implémentez des objets métier et des classes d'accès aux données en tant que services WCF, vous pouvez choisir entre deux mécanismes d'hébergement. Vous pouvez héberger les services WCF dans IIS, auquel cas HTTP est le mécanisme de transport et SSL est le mécanisme de sécurité. Vous pouvez également héberger les services WCF dans un service d'activation de processus Windows (WAS), auquel cas vous pouvez utiliser TCP, Microsoft Message Queuing (MSMQ) ou des canaux nommés comme mécanisme de transport. | |
| • | Entre serveur de niveau intermédiaire et base de données. Le serveur de niveau intermédiaire envoie des requêtes T-SQL au serveur de base de données, qui s'authentifie par rapport à la base de données en utilisant les informations d'identification requises qui sont souvent incluses dans la chaîne de connexion. |

**Noter:**Les services Web et les services WCF sont traités en détail dans le cours 20487C: Développement de Windows Azure et des services Web.

**Question:**Quels sont les avantages de l'écriture de composants de niveau intermédiaire en tant que services WCF et non services Web?

## ****Gestion de l'état de la planification****

Dans le développement d'applications, l'état de l'application fait référence aux valeurs et aux informations conservées au cours de plusieurs opérations. Le protocole HTTP (Hypertext Transfer Protocol) est fondamentalement un protocole sans état, ce qui indique qu'il ne dispose d'aucun mécanisme pour conserver les informations d'état sur plusieurs requêtes de page. Cependant, il existe de nombreux scénarios, tels que les suivants, qui nécessitent que l'état soit préservé:

|  |  |
| --- | --- |
| • | Préférences de l'utilisateur.Certains sites Web permettent aux utilisateurs de spécifier des préférences. Par exemple, une application Web de partage de photos peut permettre aux utilisateurs de choisir une taille préférée pour les photos. Si ces informations de préférence sont perdues entre les demandes de page, les utilisateurs doivent continuellement réappliquer la préférence. |
| • | Identité de l'utilisateur.Certains sites authentifient les utilisateurs pour fournir un accès au contenu réservé aux membres. Si l'identité de l'utilisateur est perdue entre les demandes de page, l'utilisateur doit ressaisir les informations d'identification pour chaque page. |
| • | Paniers.Si le contenu d'un panier est perdu entre les demandes de page, le client ne peut rien acheter à partir de votre application Web. |

Dans presque toutes les applications Web, le stockage d'état est une exigence fondamentale. ASP.NET fournit plusieurs emplacements où vous pouvez stocker des informations d'état et des moyens simples d'accéder aux informations d'état. Cependant, vous devez planifier soigneusement l'utilisation de ces mécanismes. Si vous utilisez le mauvais emplacement, vous ne pourrez peut-être pas récupérer une valeur au moment prévu. En outre, une mauvaise planification de la gestion de l’État entraîne souvent de mauvaises performances.

En général, vous devez faire attention à conserver de grandes quantités de données d'état car elles consomment la mémoire du serveur, si elles sont stockées sur le serveur, ou ralentissent le transfert de la page Web vers le navigateur, si elles sont incluses dans une page Web. Si vous avez besoin de stocker des valeurs d'état, vous pouvez choisir entre le stockage d'état côté client ou le stockage d'état côté serveur.

**Stockage d'état côté client**

Lorsque vous stockez des informations d'état sur le client, vous vous assurez que les ressources du serveur ne sont pas utilisées. Cependant, vous devez considérer que toutes les informations d'état côté client doivent être envoyées entre le serveur Web et le navigateur Web, et ce processus peut ralentir le temps de chargement de la page. Utilisez le stockage d'état côté client uniquement pour de petites quantités de données:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| • | Biscuits. Les cookies sont de petits fichiers texte que vous pouvez transmettre au navigateur pour stocker des informations. Un cookie peut être stocké:   |  |  | | --- | --- | | o | Dans la mémoire de l'ordinateur client, dans ce cas, il ne conserve les informations que pour une seule session utilisateur. | | o | Sur le disque dur de l'ordinateur client, auquel cas, il préserve les informations sur plusieurs sessions. |   La plupart des navigateurs ne peuvent stocker des cookies que jusqu'à 4 096 octets et n'autoriser que 20 cookies par site Web. Par conséquent, les cookies ne peuvent être utilisés que pour de petites quantités de données. En outre, certains utilisateurs peuvent désactiver les cookies à des fins de confidentialité, vous ne devez donc pas vous fier aux cookies pour des fonctions critiques. |
| • | Chaînes de requête. Une chaîne de requête est la partie de l'URL après le point d'interrogation et est souvent utilisée pour communiquer des valeurs de formulaire et d'autres données au serveur. Vous pouvez utiliser la chaîne de requête pour conserver une petite quantité de données d'une demande de page à une autre. Tous les navigateurs prennent en charge les chaînes de requête, mais certains imposent une limite de 2 083 caractères sur la longueur de l'URL. Vous ne devez placer aucune information sensible dans les chaînes de requête car elle est visible pour l'utilisateur, toute personne observant la session ou toute personne surveillant le trafic Web. |

**Noter:**Dans les applications Web Forms ASP.NET, l'état d'affichage, l'état du contrôle et les champs masqués peuvent être utilisés pour stocker les informations d'état dans le code HTML rendu que le serveur envoie au client. Ces mécanismes ne sont pas disponibles dans les applications Web MVC car ils n'utilisent pas de contrôles Web Forms.

**Stockage d'état côté serveur**

Les informations d'état stockées sur le serveur consomment des ressources serveur, vous devez donc faire attention à ne pas abuser du stockage d'état côté serveur ou à ne pas risquer de mauvaises performances.

Les emplacements suivants stockent les informations d'état dans la mémoire du serveur:

|  |  |
| --- | --- |
| • | TempData. Il s'agit d'un emplacement de stockage d'état que vous pouvez utiliser dans les applications MVC pour stocker des valeurs entre une demande et une autre. Vous pouvez stocker des valeurs en les ajoutant à la collection TempData. Ces informations sont conservées pour une seule demande et sont conçues pour aider à conserver les données à travers une redirection de page Web. Par exemple, vous pouvez l'utiliser pour transmettre un message d'erreur à une page d'erreur. |
| • | État de l'application. Il s'agit d'un emplacement de stockage d'état que vous pouvez utiliser pour stocker des valeurs pendant toute la durée de vie de l'application. Les valeurs stockées dans l'état de l'application sont partagées entre tous les utilisateurs. Vous pouvez stocker des valeurs en les ajoutant à la collection Application. Si le serveur Web ou l'application Web est redémarré, les valeurs sont détruites. La procédure Application\_Start () dans le fichier Global.asax est un endroit approprié pour initialiser les valeurs d'état de l'application. L'état de l'application n'est pas un endroit approprié pour stocker des valeurs spécifiques à l'utilisateur, telles que des préférences, car si vous stockez une préférence dans l'état de l'application, tous les utilisateurs partagent la même préférence, au lieu d'avoir leur propre valeur unique. |
| • | État de la session. La collection Session stocke des informations pour la durée de vie d'une seule session de navigateur et les valeurs stockées ici sont spécifiques à une seule session utilisateur; ils ne sont pas accessibles aux autres utilisateurs. Par défaut, si le serveur Web ou l'application Web est redémarré, les valeurs sont détruites. Cependant, vous pouvez configurer ASP.NET pour stocker l'état de session dans une base de données ou un serveur d'état. Si vous faites cela, l'état de la session peut être conservé lors des redémarrages. L'état de session est disponible pour les utilisateurs authentifiés et les utilisateurs anonymes. Par défaut, l'état de session utilise des cookies pour identifier les utilisateurs, mais vous pouvez configurer ASP.NET pour stocker l'état de session sans utiliser de cookies. |

Si vous choisissez d'utiliser ces emplacements de mémoire de serveur, assurez-vous d'estimer le volume total de données d'état pouvant être requis pour tous les utilisateurs simultanés que vous prévoyez de gérer. Les valeurs d'état d'application ne sont stockées qu'une seule fois, mais les valeurs d'état de session sont stockées une fois pour chaque utilisateur simultané. Spécifiez le matériel du serveur qui peut facilement gérer cette charge ou déplacez les données d'état vers les emplacements suivants basés sur les disques durs du serveur.

|  |  |
| --- | --- |
| • | Propriétés du profil. Si votre site utilise un fournisseur de profils ASP.NET, vous pouvez stocker les préférences utilisateur dans des profils. Les propriétés de profil sont conservées dans la base de données des membres, elles seront donc conservées même si l'application Web ou le serveur Web redémarre. |
| • | Tables de base de données. Si votre site utilise une base de données sous-jacente, comme le font la plupart des sites, vous pouvez stocker des informations d'état dans ses tables. C'est un bon endroit pour stocker de gros volumes de données d'état qui ne peuvent pas être placés dans la mémoire du serveur ou sur l'ordinateur client. Par exemple, si vous souhaitez stocker un grand volume d'informations d'état spécifiques à la session, vous pouvez stocker une valeur d'ID simple dans la collection Session et l'utiliser pour interroger et mettre à jour un enregistrement dans la base de données. |

N'oubliez pas que les données d'état ne sont qu'une forme d'informations qu'une application ASP.NET place dans la mémoire du serveur. Par exemple, les caches doivent partager la mémoire avec les données d'état.

**Question:**Vous montrez aux visiteurs de votre site Web un choix de pays. Lorsqu'ils choisissent un état, vous souhaitez les rediriger vers une page qui affiche une carte de cet état. Vous n'utiliserez pas le nom du pays choisi que l'utilisateur a sélectionné par la suite. Quel emplacement devez-vous utiliser pour stocker le nom du pays choisi?

## ****Planification de la mondialisation et de la localisation****

Internet est un réseau international, et à moins que vous ne soyez sûr que le public de votre application Web parle une seule langue, vous devez vous assurer que tout le monde peut lire votre contenu. Vous limitez l'audience potentielle du site si vous affichez les pages uniquement en anglais. Le processus par lequel vous rendez une application Web disponible dans plusieurs langues est appelé mondialisation ou internationalisation. Le processus par lequel vous rendez une application Web disponible dans une langue et une culture spécifiques est appelé localisation.

**Gestion des navigateurs pour les langues et les régions**

Il existe un ensemble internationalement reconnu de codes de langue qui spécifient une culture sur Internet. Ces codes sont en deux parties:

|  |  |
| --- | --- |
| 1. | La langue. Par exemple, l'anglais est «en» et le russe est «ru». |
| 2. | La région. Cela spécifie les variations régionales au sein d'une langue et affecte l'orthographe et les formats. Par exemple, en anglais américain, «Globalize» est correct et les dates sont écrites au format mm / jj / aa, alors qu'en anglais britannique, «Globalize» est correct et les dates sont écrites au format jj / mm / aa.  Le code de langue et de région complet pour l'anglais américain est «en-US» et le code complet de langue et de région pour l'anglais britannique est «en-UK». |

La langue préférée que les utilisateurs choisissent est disponible comme code de langue dans l'en-tête HTTP du navigateur de l'utilisateur. C'est la valeur à laquelle vous répondez, afin de globaliser votre site. Vous pouvez également fournir un contrôle, tel qu'une liste déroulante, dans laquelle les utilisateurs peuvent choisir leur langue préférée. Ceci est un bon exemple de préférence utilisateur que vous pouvez stocker dans l'état de session.

**Utilisation des fichiers de ressources**

Lorsque l'utilisateur spécifie une langue et une région préférées, vous devez répondre en affichant des pages pour cette culture. Une méthode pour fournir une réponse localisée consiste à utiliser des fichiers de ressources pour insérer du texte dans la langue appropriée dans la page au moment de l'exécution. Un fichier de ressources est un simple dictionnaire de termes dans une langue donnée. Pour chaque terme du fichier, vous devez spécifier un nom, une valeur et éventuellement un commentaire. Le fichier a une extension .resx. Le nom de fichier inclut également le code de langue auquel les ressources s'appliquent. Par exemple, si vous créez un fichier de ressources pour une vue appelée Index, qui stocke des valeurs en espagnol chilien, vous nommez le fichier Index.es-CL.resx.

Les fichiers de ressources peuvent également avoir une culture neutre. Cela signifie que le fichier s'applique à n'importe quelle région dans une langue donnée. Par exemple, le fichier Index.es.resx applique les termes espagnols, quel que soit le code régional choisi.

Vous devez également créer les fichiers de ressources par défaut correspondants, sans aucun code de langue dans le nom de fichier, tel que Index.resx. Ces fichiers sont utilisés lorsqu'aucune langue préférée n'est spécifiée.

Lorsque vous utilisez des fichiers de ressources pour localiser un site, chaque vue s'applique, quelle que soit la langue préférée. Vous devez insérer du code Razor supplémentaire dans la vue pour prendre des valeurs de texte à partir d'un fichier de ressources. Cela peut réduire la lisibilité des fichiers de vue car tout le texte rendu provient de fichiers de ressources. Cependant, la prise en charge de nouvelles langues est plus facile, car il vous suffit d'ajouter un nouveau fichier de ressources pour chaque langue qui peut être créée par un traducteur professionnel.

**Utilisation de vues séparées**

Certains développeurs préfèrent utiliser des ensembles de vues parallèles séparés pour chaque code de langue pris en charge. Si vous utilisez cette approche, vous devez insérer du code dans les contrôleurs pour détecter la langue préférée que l'utilisateur a spécifiée. Ensuite, vous pouvez utiliser cette valeur pour rendre la vue correcte.

Lorsque vous utilisez des vues séparées pour globaliser et localiser un site, les vues sont plus lisibles, car la plupart du texte et des étiquettes restent dans le fichier de vue. Cependant, vous devez créer des fichiers de vue, ce qui vous oblige, vous ou les membres de votre équipe, à maîtriser la langue cible.

**Question:**Quelle langue pensez-vous qu'il serait approprié de spécifier dans le fichier de ressources par défaut?

# Leçon 2: Conception de modèles, de contrôleurs et de vues

Les modèles, les contrôleurs et les vues sont les éléments de base fondamentaux d'une application Web MVC 5. Dans un site complexe, il peut y avoir des centaines de modèles, vues et contrôleurs. Vous devez gérer ces objets et bien planifier votre application, de sorte qu'il soit facile de gérer l'organisation et la structure interne pendant le développement. Un plan détaillé garantit que vous détectez rapidement tout code incorrect et déboguez les problèmes.

## ****Objectifs de la leçon****

Après avoir terminé cette leçon, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Décrivez comment concevoir des modèles. |
| • | Décrivez comment concevoir des contrôleurs. |
| • | Décrivez comment concevoir des vues. |

## ****Conception de modèles****

Une activité fondamentale du processus de conception MVC est la conception d'un modèle. Chaque classe de modèle dans le modèle représente un type d'objet géré par votre application. Vous ne pouvez pas planifier les contrôleurs et les vues tant que vous n'avez pas compris la structure et la conception du modèle.

**Identification des classes et des propriétés du modèle**

Les cas d'utilisation, les scénarios d'utilisation ou les user stories que vous avez rassemblés au cours de la phase d'analyse du projet doivent vous permettre de déterminer les classes de modèle que vous devez créer. Chaque classe de modèle possède une plage de propriétés. Par exemple, considérons l'exemple de cas d'utilisation suivant partagé précédemment.

|  |  |
| --- | --- |
| 1. | L'utilisateur clique sur le lien Ajouter une photo dans le menu principal du site. |
| 2. | Si l'utilisateur est anonyme, la page de connexion s'affiche et l'utilisateur fournit des informations d'identification. |
| 3. | Si les informations d'identification sont correctes, la vue CreatePhoto s'affiche. |
| 4. | L'utilisateur tape un titre. |
| 5. | L'utilisateur spécifie le fichier photo à télécharger. |
| 6. | L'utilisateur tape éventuellement une description pour la photo. |
| 7. | L'utilisateur clique sur le bouton Télécharger. |
| 8. | L'application Web stocke la nouvelle photo et affiche la galerie de photos à l'utilisateur. |

Cet exemple inclut les objets suivants, dont chacun nécessite une classe de modèle:

|  |  |
| --- | --- |
| • | Utilisateur. La classe de modèle User doit inclure des propriétés pour stocker les informations d'identification, telles que le nom d'utilisateur et le mot de passe. |
| • | photo. La classe de modèle Photo doit inclure les propriétés Titre et Description. |

D'autres cas d'utilisation vous aident également à ajouter de nouvelles classes de modèles ou de nouvelles propriétés aux classes de modèles User et Photo.

**Diagrammes de modèle de domaine et de modèle de données logiques**

Vous pouvez utiliser des diagrammes de modèle de domaine et des diagrammes de modèle de données logiques (LDM) pour analyser les informations gérées par votre site Web et suggérer un modèle de données physique ou une conception de base de données. Vous pouvez revenir à ces diagrammes pour planifier des classes de modèle. Chaque objet de votre modèle de domaine ou diagramme LDM doit être implémenté en tant que classe de modèle MVC. Le diagramme de modèle de domaine comprend non seulement les noms de ces classes, mais également le diagramme LDM, si vous en avez créé un, avec les noms de propriétés et les types de données.

**Relations et agrégats**

Lorsque vous identifiez les classes de modèle que vous implémenterez dans votre site Web, vous devez également tenir compte des relations entre elles. Par exemple, dans le cas d'utilisation de l'exemple d'application de partage de photos, chaque photo est associée à un et un seul utilisateur. C'est ce qu'on appelle une relation individuelle. Cependant, chaque utilisateur peut créer plusieurs photos. C'est ce qu'on appelle une relation un-à-plusieurs.

Les diagrammes de modèle de domaine et les diagrammes LDM incluent des relations telles que des liens entre des objets. Les nombres aux extrémités de chaque lien indiquent si la relation est un-à-un, un-à-plusieurs ou plusieurs-à-plusieurs.

Les agrégats imposent des limites supplémentaires au comportement des classes modèles et clarifient les relations. Par exemple, dans une application de partage de photos, une photo est créée par un seul utilisateur. Les autres utilisateurs peuvent faire plusieurs commentaires sur chaque photo. Si un utilisateur supprime une photo, tous les commentaires sur cette photo doivent également être supprimés de la base de données. Cependant, l'utilisateur qui a créé la photo ne doit pas être supprimé avec la photo car il ou elle peut ajouter d'autres photos ou commentaires sur le site. Dans ce cas, les commentaires et les photos doivent être placés dans un agrégat, mais les utilisateurs doivent être en dehors des agrégats. La photo est «l'entité racine» de l'agrégat - cela signifie que la suppression d'une photo supprime tous les commentaires associés, mais la suppression d'un commentaire ne supprime pas la photo associée.

**Cadre d'entité**

Entity Framework est un framework ORM (Object Relational Mapping) pour les applications basées sur .NET Framework. Un framework ORM lie les tables et vues de base de données à des classes sur lesquelles un développeur peut programmer, en créant des instances ou en appelant des méthodes. Entity Framework fait partie d'ADO.NET depuis .NET Framework 3.5.

Lorsque vous utilisez Entity Framework dans votre application Web MVC, il lie les tables et les vues aux classes de modèle que vous avez planifiées. Vous n'avez pas besoin d'écrire du code SQL pour interroger ou mettre à jour les tables de base de données, car Entity Framework le fait pour vous. Entity Framework est bien intégré au langage LINQ (Language Integrated Query).

Si vous prévoyez d'utiliser Entity Framework pour l'accès aux données, vous devez décider de la manière dont la base de données sera créée pendant la phase de planification:

|  |  |
| --- | --- |
| o | Base de données d'abord. Utilisez Entity Framework en mode base de données d'abord lorsque vous disposez d'une base de données préexistante avec laquelle travailler. Cela peut se produire parce que vous avez déjà des données d'un système antérieur ou parce qu'un DBA a conçu la base de données pour vous. Vous pouvez également choisir ce mode si vous êtes familiarisé avec la création de bases de données dans un outil d'administration de base de données, tel que Microsoft SQL Server Management Studio. Lorsque vous utilisez ce mode, vous devez spécifier la chaîne de connexion à la base de données. Entity Framework se connecte à la base de données et examine le schéma de la base de données. Il crée un ensemble de classes que vous pouvez utiliser pour l'accès aux données. |
| o | Modèle d'abord. Utilisez Entity Framework en mode modèle d'abord lorsque vous n'avez pas de base de données préexistante et que vous préférez concevoir votre modèle dans Visual Studio. Vous pouvez utiliser l'outil Entity Designer pour nommer, configurer et lier vos classes de modèle. Cela crée des fichiers XML qu'Entity Framework utilise à la fois pour créer des classes de modèle et pour créer la base de données avec ses tables et ses relations. |
| o | Code d'abord. Utilisez Entity Framework en mode code d'abord lorsque vous n'avez pas de base de données préexistante et que vous préférez concevoir vos modèles entièrement en code C #. Votre code doit inclure des objets DBContext et DBSet - ceux-ci correspondent à la base de données et à ses tables. Lorsque vous exécutez l'application pour la première fois, Entity Framework crée la base de données pour vous. |

**Conception en programmation agile et extrême**

Les projets Agile et Extreme Programming se caractérisent par de courtes phases de conception au cours desquelles les modèles de données ne sont pas terminés. Au lieu de cela, une conception simple, avec peu de détails, est créée et les développeurs remplissent les détails au fur et à mesure qu'ils construisent le code en discutant en permanence des exigences avec les utilisateurs et les autres parties prenantes.

Dans un projet MVC, cela signifie que vous identifiez les noms de modèle et les relations pendant la phase de conception. Vous pouvez les enregistrer sur un diagramme UML de modèle de domaine. Cependant, vous pouvez laisser les détails tels que les noms de propriétés et les types de données à finaliser dans la phase de développement, ainsi que les diagrammes LDM complets.

Entity Framework vous permet de travailler dans les styles de programmation Agile ou Extreme. Par exemple, le framework peut mettre à jour la base de données lorsque le modèle change. Sinon, le framework peut mettre à jour le modèle lorsque la base de données change. Entity Framework peut effectuer ces mises à jour dans n'importe quel mode Entity Framework.

## ****Conception de contrôleurs****

Dans une application Web ASP.NET MVC, les contrôleurs sont des classes basées sur .NET Framework qui héritent de la classe de base System.Web.Mvc.Controller. Ils implémentent une logique d'entrée - cela signifie qu'ils reçoivent des entrées de l'utilisateur sous la forme de requêtes HTTP et sélectionnent à la fois le modèle et la vue appropriés à utiliser pour formuler une réponse.

**Identifier les contrôleurs et les actions**

Dans une application Web ASP.NET MVC, il existe généralement un contrôleur pour chaque classe de modèle. Classiquement, si la classe de modèle s'appelle «Photo», le contrôleur s'appelle «PhotoController». Si vous suivez cette convention dans votre conception, vous pouvez utiliser le comportement de routage par défaut MVC pour sélectionner le contrôleur approprié pour une demande.

Cependant, pour chaque contrôleur, il peut y avoir de nombreuses actions - chaque action est implémentée en tant que méthode dans le contrôleur et renvoie généralement une vue. Vous avez souvent besoin d'actions distinctes pour les verbes de requête HTTP GET et POST. Comme pour la conception d'un modèle, vous pouvez identifier les actions à écrire dans chaque contrôleur en examinant les cas d'utilisation que vous avez rassemblés lors de l'analyse. Par exemple, considérez le cas d'utilisation suivant partagé précédemment.

|  |  |
| --- | --- |
| 1. | L'utilisateur clique sur le lien Ajouter une photo dans le menu principal du site. |
| 2. | Si l'utilisateur est anonyme, la page de connexion s'affiche et l'utilisateur fournit des informations d'identification. |
| 3. | Si les informations d'identification sont correctes, la vue CreatePhoto s'affiche. |
| 4. | L'utilisateur tape un titre. |
| 5. | L'utilisateur spécifie le fichier photo à télécharger. |
| 6. | L'utilisateur tape éventuellement une description pour la photo. |
| 7. | L'utilisateur clique sur le bouton Télécharger. |
| 8. | L'application Web stocke la nouvelle photo et affiche la galerie de photos à l'utilisateur. |

Vous avez déjà identifié les classes de modèles Photo et User à partir de ce cas d'utilisation. En adhérant à la convention MVC, vous devez créer un PhotoController et un UserController. Le cas d'utilisation montre que les actions suivantes sont nécessaires pour chaque contrôleur.

| **Manette** | **action** | **La description** |
| --- | --- | --- |
| photo | **AddPhoto (GET)** | L'action AddPhoto pour les demandes GET crée une nouvelle instance de la classe de modèle Photo, définit des valeurs par défaut telles que la date de création et la transmet à la vue correcte. |
| **AddPhoto (POST)** | L'action AddPhoto pour les demandes POST appelle les méthodes de classe de modèle Photo pour enregistrer les valeurs de photo dans la base de données et redirige le navigateur vers l'action DisplayGallery. |
| **DisplayGallery (GET)** | L'action DisplayGallery pour les demandes GET affiche toutes les photos stockées dans la base de données. |
| Utilisateur | **Connexion (GET)** | L'action d'ouverture de session pour les demandes GET affiche une vue dans laquelle un utilisateur anonyme peut entrer des informations d'identification. |
| **Connexion (POST)** | L'action d'ouverture de session pour les demandes POST vérifie les informations d'identification de l'utilisateur par rapport à la base de données des membres. Si les informations d'identification sont correctes, l'action d'ouverture de session authentifie et redirige l'utilisateur vers la page demandée à l'origine. |

**Conception en programmation agile et extrême**

Semblable à la conception de modèles, vous ne ferez des plans généralisés pour les contrôleurs que pendant la phase de conception d'un projet de développement agile ou de programmation extrême. Étant donné que vous avez identifié les classes de modèles nécessaires, vous pouvez planifier les contrôleurs requis. Cependant, vous ne devez spécifier que quelques actions à ce stade.

## ****Conception de vues****

L'interface utilisateur est un composant essentiel de tout système car c'est la partie que les utilisateurs, les responsables du budget et les autres parties prenantes voient et avec laquelle interagissent. Les utilisateurs sont plus intéressés par la bonne exécution de cette partie de l'application et ont souvent le plus à dire sur sa conception. En tant que développeur, vous avez la possibilité d'impressionner vos utilisateurs en concevant et en implémentant une interface utilisateur sophistiquée, ce qui peut générer plus d'activité.

Dans une application Web ASP.NET MVC, l'interface utilisateur est créée en créant des vues.

**Vues, ​​vues de modèle et vues partielles**

Il existe une relation plusieurs à un entre les contrôleurs MVC et les vues. Par exemple, un contrôleur peut utiliser une vue pour afficher une seule photo, une autre vue pour afficher plusieurs photos et une troisième vue pour permettre aux utilisateurs de télécharger de nouvelles photos. Chaque vue correspond à une page Web que l'application peut afficher à l'utilisateur, bien qu'elle puisse afficher des données différentes. Par exemple, la vue PhotoDetails peut afficher différentes photos, en fonction du paramètre ID qui lui est passé.

Lorsque vous planifiez des vues, vous devez également prendre en compte les parties de l'interface utilisateur qui apparaissent sur toutes les pages. Par exemple, le logo de l'entreprise, le menu principal du site, des liens vers des informations juridiques et des contrôles de connexion peuvent devoir apparaître sur chaque page. Vous pouvez placer ces composants d'interface utilisateur dans une vue modèle pour créer une apparence cohérente sur toutes les pages.

**Noter:**Les vues de modèle dans les applications Web ASP.NET MVC jouent le même rôle que les pages maîtres dans les applications Web Forms ASP.NET.

Certains composants de l'interface utilisateur n'apparaissent pas sur toutes les pages, mais sont réutilisés sur plusieurs pages. Par exemple, les commentaires peuvent être affichés dans un seul affichage de photo, sur la galerie ou sur d'autres pages. En créant une vue partielle, vous pouvez créer un élément d'interface utilisateur réutilisable qui peut apparaître à de nombreux endroits de cette manière, sans dupliquer le code.

**Création de wireframes**

Une technique courante pour discuter et planifier l'interface utilisateur de votre application consiste à créer des diagrammes filaires. Ce sont des mises en page simplifiées qui montrent où les éléments apparaîtront sur les pages Web finales. Ils sont destinés à communiquer aux utilisateurs les parties fonctionnelles essentielles de l'interface utilisateur, mais n'incluent pas les graphiques, logos ou couleurs. Il n'est pas nécessaire de créer un modèle filaire pour chaque vue de votre application, mais uniquement pour les plus importantes.

Vous pouvez commencer un diagramme filaire en l'esquissant sur un tableau blanc, en conversation avec un utilisateur. De nombreux outils sont disponibles pour créer des versions plus formelles. Par exemple, Microsoft Visio possède d'excellentes capacités de dessin filaire.

**Conception en programmation agile et extrême**

Vous ne concevez pas de nombreuses parties de l'interface utilisateur pendant les phases initiales des projets de développement agile ou de programmation extrême. Au lieu de cela, vous concevez des vues et des vues partielles en étroite consultation avec les utilisateurs pendant la phase de développement. Cela s'applique même à la vue modèle qui affiche les composants communs de votre interface utilisateur, bien qu'il soit probable que la vue modèle soit l'un des premiers composants de l'interface utilisateur qui est conçu et créé. Vous créerez la vue modèle au cours des premières itérations de la phase de développement du projet.

# ****Atelier: Conception d'applications Web ASP.NET MVC 5****

### ****Scénario****

Votre équipe a choisi ASP.NET MVC 5 comme modèle de programmation ASP.NET le plus approprié pour créer l'application de partage de photos pour l'application Web Adventure Works. Vous devez créer une conception de projet détaillée pour l'application et vous avez reçu un ensemble d'exigences fonctionnelles et techniques avec d'autres informations. Vous devez planifier:

|  |  |
| --- | --- |
| • | Un modèle MVC que vous pouvez utiliser pour implémenter la fonctionnalité souhaitée. |
| • | Un ou plusieurs contrôleurs et actions de contrôleur qui répondent aux actions des utilisateurs. |
| • | Un ensemble de vues pour implémenter l'interface utilisateur. |
| • | Les emplacements d'hébergement et de stockage des données. |

### ****Objectifs****

Après avoir terminé cet atelier, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Concevez une application Web ASP.NET MVC 5 qui répond à un ensemble d'exigences fonctionnelles. |
| • | Enregistrez la conception de manière précise, précise et informative. |

##### ****Configuration du laboratoire****

Durée estimée: 40 minutes

Vous trouverez les étapes de haut niveau sur la page suivante:<https://github.com/MicrosoftLearning/20486-DevelopingASPNETMVCWebApplications/blob/master/Instructions/20486C/20486C_MOD02_LAB_MANUAL.md>.

Vous trouverez les étapes détaillées sur la page suivante:<https://github.com/MicrosoftLearning/20486-DevelopingASPNETMVCWebApplications/blob/master/Instructions/20486C/20486C_MOD02_LAK.md>.

### ****Exercice 1: Planification des classes de modèle****

##### ****Scénario****

Vous devez recommander un modèle MVC requis pour implémenter une application de partage de photos. Vous proposerez des classes de modèles basées sur les résultats d'une première enquête sur les exigences.

### ****Exercice 2: Planification des contrôleurs****

##### ****Scénario****

Vous devez recommander un ensemble de contrôleurs MVC requis pour implémenter une application de partage de photos. Vous proposerez des contrôleurs sur la base des résultats d'une première enquête sur les exigences.

### ****Exercice 3: Vues de planification****

##### ****Scénario****

Vous devez recommander un ensemble de vues MVC requises pour implémenter une application de partage de photos. Vous proposerez des vues basées sur les résultats d'une enquête initiale sur l'exigence.

### ****Exercice 4: Architecture d'une application Web MVC****

##### ****Scénario****

Vous devez recommander une configuration de serveur Web et de serveur de base de données requise pour implémenter une application de partage de photos. Vous proposerez des détails basés sur les résultats d'une première enquête sur les exigences.

### ****Question (s) de révision****

**Vérifiez vos connaissances**

**Découverte**

**Quelles classes de modèles doivent être créées pour l'application de partage de photos en fonction de l'enquête initiale?**

Afficher la solution Réinitialiser

**Vérifiez vos connaissances**

**Découverte**

**Quels contrôleurs doivent être créés pour l'application de partage de photos sur la base de l'enquête initiale?**

Afficher la solution Réinitialiser

**Vérifiez vos connaissances**

**Découverte**

**Quelles vues doivent être créées pour l'application de partage de photos?**

Afficher la solution Réinitialiser

# ****Examen du module et points à retenir****

Dans ce module, vous avez vu comment des équipes de développeurs, d'architectes logiciels et d'analystes métier collaborent pour concevoir une application Web MVC qui répond aux besoins des utilisateurs. Vous pouvez rassembler les exigences fonctionnelles et techniques en discutant avec les parties prenantes et en créant des cas d'utilisation, des scénarios d'utilisation et des user stories. Le modèle, la vue, le contrôleur et d'autres aspects de la conception dépendent de ces exigences. Vous avez également vu comment ces activités de conception sont réalisées dans des projets qui utilisent la méthodologie agile ou la programmation extrême.

**Meilleur entrainement:**Dans les projets de développement agile et de programmation extrême, les développeurs discutent avec les utilisateurs et les parties prenantes tout au long du développement pour s'assurer que leur code répondra aux exigences changeantes. Même si vous n'utilisez pas officiellement ces méthodologies, il est recommandé de communiquer régulièrement avec les utilisateurs.

**Meilleur entrainement:**Lorsque vous concevez une application Web ASP.NET MVC, commencez par le modèle, puis planifiez les contrôleurs, les actions et les vues. Les contrôleurs, actions et vues que vous créez dépendent chacun du modèle.

**Problèmes courants et conseils de dépannage**

| **Problème commun** | **Astuce de dépannage** |
| --- | --- |
| Lorsque vous créez un plan de projet très détaillé, une grande partie de votre travail est gaspillée lorsque les exigences changent tard dans le projet. | Veuillez consulter le contenu du compagnon étudiant pour ce cours. |

### ****Question (s) de révision****

**Vérifiez vos connaissances**

**Découverte**

**Vous souhaitez prendre en charge l'anglais et l'espagnol dans votre application Web. Vous avez à la fois des développeurs hispanophones et anglophones et vous voulez vous assurer que les vues restent lisibles aussi facilement que possible. Devez-vous utiliser plusieurs fichiers de vue ou plusieurs fichiers de ressources pour globaliser votre site?**

Afficher la solution Réinitialiser

### ****Problèmes et scénarios du monde réel****

Vous devez garder à l'esprit que lorsque vous sélectionnez une méthodologie de projet, peu de projets suivent un plan soigné dans des situations réelles. Parmi les méthodologies décrites dans ce module, le développement agile et la programmation extrême sont les plus flexibles et répondent lorsque les plans changent au milieu du développement. Cependant, même avec ces méthodologies, des circonstances changeantes entraînent une perte de temps de développement et le budget de votre projet doit inclure une réserve pour faire face à de tels changements.

De plus, lorsqu'ils travaillent avec des projets de développement agile et de programmation extrêmes, les chefs de projet doivent veiller à éviter le fluage du projet ou le dérapage de la portée. Cela se produit lorsque les utilisateurs ajoutent de nouvelles exigences lors du développement. Le fluage des projets se traduit par des projets qui dépassent le budget et qui sont en retard.

### ****Outils****

Microsoft Office Visio: vous pouvez utiliser Visio pour créer tous les types de diagrammes de conception de logiciels UML, y compris les diagrammes de modèle de domaine et les LDM. Vous pouvez également l'utiliser pour créer des wireframes.

Visual Studio 2017: vous pouvez créer des diagrammes de classes tels que des LDM dans Visual Studio 2017.