|  |  |
| --- | --- |
| **420-4B6-LI Applications Web I**  **Cégep Limoilou**  **Département d'Informatique**  **Hiver 2020** | **Tp2(15 %)**  **Application Web ASP.NET MVC** |

Capacité 2 :   Programmer la logique applicative côté serveur

OBJECTIFS

* Créer une application Web (sans BD).
* Planifier le travail à faire et identifier les principales fonctions, objets et interactions à développer.
* Programmer la logique applicative côté client.
* Programmer les interfaces côté client en exploitant Bootstrap.
* Programmer la logique applicative côté serveur avec le framework ASP.NET MVC.
* Programmer en orienté-objet avec C#.
* Valider le fonctionnement de l’application.

**DESCRIPTION**

Créer une application inspirée de Netflix faisant la gestion de produits multimédias (par exemple : film, vidéo, jeux, etc.). Votre application ne comporte aucune donnée provenant d’une base de données. Elle devra disposer de modèles C# et obtenir des données à l’aide d’un fichier XML ou texte pour peupler sa liste de nouveautés de films et conserver l’information sur les utilisateurs. Votre application permettra entre autres :

* S’enregistrer
* Obtenir de l’information sur un produit.
* Voir un extrait.
* Voir au complet.
* Ajouter ou retirer dans une liste de favoris.
* Afficher la liste de favoris.
* Donner son appréciation.

INSTRUCTIONS

* Voici les instructions qu’il est obligatoire de respecter, vous avez la liberté pour l’apparence et pour certaines fonctions. Certaines actions sont présentées, mais seront développées dans le Tp3.
* Au départ de l’application, la page d’accueil principale offre une seule action pour se connecter et fournit de l’information présentant une image en arrière-plan et de l’information générale (utilisez des sites d’images libre de droit comme Pixabay).
* Pour le moment vos utilisateurs sont répertoriés dans un fichier (XML ou texte) qui ne contient que le nom d’utilisateur et le login en clair. Il y a un utilisateur admin avec un mot de passe qui servira pour les opérations d’administration dans le Tp3.
* Une fois connecté, vous arrivez dans la page d’accueil de l’application. La page d’accueil offre une barre de navigation commune (Nom de l’application Accueil, Liste, Ma liste, Suggestions).
* Il y a aussi une barre de navigation verticale qui offrira des options différentes selon le contrôleur qui contrôle les actions. Pour le moment les options sont (Compte avec sous-options Déconnecter et Modifier mot de passe et l’option Rechercher.) Lorsque nous aurons la partie Administration, la barre verticale sera différente.
* La page Accueil présente la liste des nouveautés (au moins 5).
* Liste affiche l’ensemble des produits incluant les nouveautés avec des images et une information sommaire sur le produit.
* Ma Liste affiche la liste des produits pour lesquels l'utilisateur a signifié son intérêt.
* Suggestions offre parmi la liste des produits, des suggestions à l’utilisateur basées sur un algorithme. Cette portion sera complétée dans le Tp3. Contentez-vous de mettre en développement pour le moment.
* Pour chacun des menus Accueil, Liste et Ma Liste, lorsqu’un objet est sélectionné, il permet d’en montrer le détail et d’afficher un autre menu pour :
  + voir ou écouter l’élément sélectionné,
  + voir ou écouter un extrait de l’élément sélectionné,
  + enregistrer l’élément dans ma liste,
  + donner son appréciation du produit sélectionné (étoiles, côte ou aimé ou pas).

Vous avez la liberté de la présentation de cet élément.

* Vous devez avoir un modèle pour enregistrer l’information sur votre produit (nom, description…) et aussi pour emmagasiner l’information sur les utilisateurs, l’inventaire et les différentes listes. Vos modèles auront besoin de méthodes pour enregistrer et lire de l’information, car ce sera à partir de fichiers que nous aurons de la persistance.
* Ayez au moins 2 fichiers de produits contenant au moins 5 éléments et 2 fichiers d’utilisateurs contenant une liste des préférences (2-3) et les appréciations (2-3) pour tester.
* Votre application doit utiliser des dispositions “layout” différentes, au moins une vue partielle, au moins deux contrôleurs, un routage efficace, une mise en page qui exploite des éléments de Bootstrap, au moins un effet en JavaScript ou JQuery, une présentation soignée et ergonomique.
* Portez attention à l’affichage adaptable aux résolutions, à la propreté du code et à ne pas dupliquer le code.

Avant de commencer vous devez fournir un document d’analyse (Word) contenant :

1. Brève description de votre application et de l’apparence (dessin, prototype).
2. Description des modèles à développer (fournir un diagramme de classe de base).
3. Les fichiers qui seront nécessaires (images, document, modèles).
4. Les liens nécessaires (vidéos, extraits).
5. Les routes.
6. Les contrôleurs et les actions prévues (voir Formatif1).
7. Essayez de penser en fonction ce qui s’en vient (Tp3) pour que ce soit facilement évolutif.
8. Vos particularités, suggestions ou ajouts proposés.

**À remettre**

Le travail se réalise en équipe (2 ou 3) sur 3 semaines incluant la démonstration. Le Tp comporte deux remises sur GIT et les dépôts doivent être documentés et alimentés par tous les équipiers.

|  |  |
| --- | --- |
| **Dates de remise :** | **20 février 2020 – Partie analyse**  **6 mars 2020 – Développement complet** |
|  |  |

420-4B6-LI Développement Web 1

**Grille de correction Tp2 – 15 %**

**Noms :**

**/100**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Capacité 1** | **Excellent 100** | **Bien 85** | **Suffisant 60** | **Incomplet 30** |
| Programmer la logique applicative côté client | Le document d’analyse est complet et parfaitement détaillé.  Les vues sont bien exploitées et utilisent des instructions Razor, HTML et CSS.  Au moins 2 dispositions différentes sont utilisées.  Au moins un élément utilise un effet Javascript ou Jquery.  Les barres de navigations sont bien présentées.  Les images et les éléments multimédias sont efficaces et bien présentés.  Le code est bien structuré, propre et complet en respect avec les instructions fournies et les décisions prises dans l’analyse.  Il est facile de se repérer dans l’application et elle est bien documentée.  Les bonnes pratiques au niveau du positionnement des éléments dans les interfaces sont bien utilisées.  L’application est adaptable à différentes résolutions.  L’esthétisme est remarquable.  Toutes les instructions sont respectées. | Le document d’analyse est complet et suffisamment détaillé.  Les vues sont bien exploitées et utilisent des instructions Razor, HTML et CSS.  Au moins 2 dispositions différentes sont utilisées.  Au moins un élément utilise un effet Javascript ou Jquery.  Les barres de navigations sont bien présentées.  Les images et les éléments multimédias sont efficaces et bien présentés.  Le code est bien structuré, propre et complet en respect avec les instructions fournies et les décisions prises dans l’analyse.  Il est généralement facile de se repérer dans l’application et elle est bien documentée.  Les bonnes pratiques au niveau du positionnement des éléments dans les interfaces sont généralement bien utilisées.  L’application est généralement adaptable à différentes résolutions.  L’esthétisme est très bien.  La plupart des instructions sont respectées. | Le document d’analyse est minimal.  Les vues sont bien exploitées et utilisent des instructions Razor, HTML et CSS la plupart du temps  Au moins 1 disposition est utilisée.  Au moins un élément utilise un effet Javascript ou Jquery.  Les barres de navigations sont présentes.  Les images et les éléments multimédias sont présentes.  Le code est généralement bien structuré, propre mais peut présenter une ou deux lacunes mais ne mettant pas en péril l’exécution.  Il peut être difficile de se repérer dans l’application et elle est documentée sommairement.  Les bonnes pratiques au niveau du positionnement des éléments dans les interfaces sont à améliorer.  L’application ne s’adapte pas à différentes résolutions.  L’esthétisme est correct.  La plupart des instructions sont respectées. | Le document d’analyse est absent.  Les vues utilisent des instructions Razor, HTML et CSS la plupart du temps  Au moins 1 disposition est utilisée.  Les barres de navigations sont présentes.  Les images et les éléments multimédias sont présentes.  Le code est généralement bien structuré, propre mais peut présenter plusieurs lacunes.  Il peut être difficile de se repérer dans l’application et elle n’est pas documentée suffisamment.  Les bonnes pratiques au niveau du positionnement des éléments dans les interfaces sont à améliorer.  L’application ne s’adapte pas à différentes résolutions.  L’esthétisme est rudimentaire.  Plusieurs des instructions ne sont pas respectées. |
| **Capacité 2** | **Excellent 100** | **Bien 85** | **Suffisant 60** | **Incomplet 30** |
| Programmer la logique applicative côté serveur | Les contrôleurs sont bien structurés et les actions sont bien définies et complètes en lien avec les instructions.  La navigation est bien soutenue par les routes.  Les fichiers sont bien utilisés pour récupérer les données et sauvegarder les modifications.  Il y a au moins 2 fichiers de produits et 2 fichiers utilisateurs.  Le code est bien structuré, propre et complet en respect avec les instructions fournies et les décisions prises dans l’analyse.  Il est facile de se repérer dans l’application et elle est bien documentée.  Les modèles sont bien définis et sont bien utilisées.  Les outils de débogage sur un navigateur web sont utilisés de manière autonome.  Des cas de tests pour les classes et pour les principales fonctions ont été fournis et appliqués.  L’exécution est impeccable.  Les choix (algo, structure de données, affichage) sont judicieux et efficaces.  Le travail est fait de façon autonome.  Les dépôts sur GIT sont faits de façon régulière, parfaitement repérables et les différentes versions sont clairement identifiées et documentées. | Les contrôleurs sont bien structurés et les actions sont bien généralement définies et complètes en lien avec les instructions.  La navigation est généralement bien soutenue par les routes.  Les fichiers sont généralement bien utilisés pour récupérer les données et sauvegarder les modifications.  Il y a au moins 2 fichiers de produits et 2 fichiers utilisateurs.  Le code est généralement bien structuré, propre et complet en respect avec les instructions fournies et les décisions prises dans l’analyse.  Il est généralement facile de se repérer dans l’application et elle est bien documentée.  Les modèles sont généralement bien définis et sont bien utilisées.  Les outils de débogage sur un navigateur web sont généralement utilisés de manière autonome.  Généralement, des cas de tests pour les classes et pour les principales fonctions ont été fournis et appliqués.  L’exécution est impeccable la plupart du temps.  Les choix (algo, structure de données, affichage) sont généralement judicieux et efficaces.  Le travail est fait de façon autonome la plupart du temps.  Les dépôts sur GIT sont faits de façon régulière, généralement repérables et les différentes versions sont clairement identifiées et documentées. | Les contrôleurs sont bien structurés et les actions sont assez bien définies en lien avec les instructions (il peut manquer 1 ou 2 éléments).  La navigation peut comporter certaines lacunes au niveau de la définition des routes.  Les fichiers sont utilisés pour récupérer les données et sauvegarder les modifications, mais peuvent présenter des lacunes.  Il y a au moins 1 fichier de produits et 1 fichier utilisateurs.  Le code est structuré, mais est à améliorer.  Il peut être difficile ile de se repérer dans l’application et elle est documentée sommairement.  Les modèles sont à améliorer.  Les outils de débogage sur un navigateur web sont utilisés avec de l’aide.  Les cas de tests pour les classes et pour les principales fonctions ont été fournis sommairement et appliqués sommairement.  L’exécution présente des lacunes (moins de 3).  Les choix (algo, structure de données, affichage) sont à raffiner.  Le travail est fait avec de l’aide constante.  Les dépôts sur GIT sont faits de façon irrégulière et peu documentés. | Les contrôleurs sont structurés et les actions sont peu définies en lien avec les instructions (il peut manquer plus de 2 éléments).  La navigation peut comporter certaines lacunes au niveau de la définition des routes ou être absente.  Les fichiers sont utilisés pour récupérer les données et sauvegarder les modifications, mais peuvent présenter plusieurs lacunes.  Il y a au moins 1 fichier de produits et 1 fichier utilisateurs.  Le code est peu structuré.  Il est difficile ile de se repérer dans l’application et elle est peu documentée.  Les modèles sont à améliorer.  Les outils de débogage sur un navigateur web sont peu utilisés.  Les cas de tests pour les classes et pour les principales fonctions ne sont pas fournis.  L’exécution présente des lacunes (plus de 3).  Les choix (algo, structure de données, affichage) sont peu raffinés.  Le travail est fait avec de l’aide constante.  Les dépôts sur GIT sont peu fréquents e et peu documentés. |