CINEMANIA DOCUMENTATION

07/12/2018

PRÉSENTATION

1. Introduction au projet

Pour commencer, Le projet « Cinémania » reprend le concept d'Allo ciné. C'est-à-dire le référencement des films et séries avec toutes les informations les concernant (Titre, Réalisateur, Catégorie, Date de sortie, Bande annonce, ...) avec différentes options telles que consulter les bandes annonces, trier les films par catégorie, noter les films, ajouter des films aux favoris sous forme de playlist, …

Cinémania avait pour but premier d'exporter un site web sous forme de logiciel plus accessible, ici nous avons utilisé un projet WPF qui utilise le XAML pour le côté frontend et le C# pour le côté backend.

2. L'accessibilité du logiciel

Le logiciel fourni une accessibilité numérique pour plusieurs raisons :

- Possibilité pour les utilisateurs de manipuler les interfaces graphiques avec le clavier. Par exemple avec la tabulation pour passer d'un élément à un autre ou bien de la touche entrée pour valider les formulaires comme le formulaire de login
- Le logiciel s'assure que tous les utilisateurs puissent naviguer entre les différents éléments d'une page avec le clavier sans être « piégés » dans un endroit de l'application, puisque pour chaque page ouverte il y a la possibilité d'avoir soit un bouton Retour (sous forme d'une icône de flèche) en haut de la page ou bien d'utiliser le menu de navigation qui reste figer à droite tout au long de l'utilisation de l'application
- Les limitations techniques comme la connexion internet ne pose pas de problème car le logiciel dispose d'une base de données en local qui permet de stocker les films sans connexion, cependant pour une mise à jour des films automatique (sans ajouter à la main) une connexion sera nécessaire.

DIAGRAMMES

1. Diagramme de cas d'utilisation

Vous pouvez trouver le diagramme ici : Cinemania UseCase.jpg

Le diagramme décrit pour chaque acteur chacune des interactions qu'il peut réaliser :

Pour le visiteur :

- Il peut soit consulter les films directement ou bien il peut consulter les films par catégories tout simplement parce que la page de Catégorie se présente sous forme de rectangle sur lequel il faut cliquer pour accéder aux films triés, mais l'utilisateur peut aussi bien décider de ne pas consulter film à partir des catégories.
- Il peut s'inscrire tout simplement, cependant s'il termine l'inscription correctement (sans champs vides, …), il sera automatiquement connecté

Pour l'utilisateur :

- L'utilisateur est un visiteur car il peut quand même consulter les films, catégories, ... Mais pour accéder à d'autres options, il a besoin d'être authentifier
- Consulter son compte (nom, prénom, image, mot de passe, …) et il peut possiblement modifier certaines de ces informations
- Le fait d'être authentifier permet à l'utilisateur en consultant les films de possiblement noter un film ou l'ajouter/retirer à ses favoris

Pour l'administrateur :

- L'administrateur est simplement un utilisateur spécial qui peut modifier la base de données directement depuis l'application

2. Diagramme de classe

Vous pouvez trouver le diagramme ici : Diagramme de classe.jpg

Pour la partie personnelle, le « StockageBDD » est une classe statique qui s'occupe de gérer la persistance sous forme d'une base de données SQLLITE mais aussi d'encapsuler toutes les requêtes sous forme de fonction qui permettent d'avoir un contrôle assez global sur les données sérialisées.

Le diagramme dispose de pas mal d'élément je ne vais donc pas pouvoir détailler tout mais je vais aller à l'essentiel. Pour la partie « VUE » toutes les classes implémente l'interface ISwitch qui leur demande d'implémenter une fonction qui permet de gérer la navigation dynamique de l'application en passant par la classe ControlSwitcher (qui a besoin d'être instanciée dans la MainView puisque la MainView a le rôle de gérer et stocker la navigation). La modélisation ne plaît pas en l'état puisque l'interface ne sert à rien en elle-même puisque la classe ControlSwitcher est statique, mais l'usage d'interface aide à avoir des points ☹

Sinon chaque classe correspond à une page dynamique, ce qui permet en cas de maintenance de ne pas impacter l'application en entière mais seulement une page en particulier. C'est aussi plus propre car chaque UserControl a sa propre responsabilité et ne fait quasiment jamais a une responsabilité d'une autre vue.

Enfin, l'EnumExtension est une classe qui permet de créer de Attribute et de les récupérer grâce à la fonction EnumAttr implémenté. Par exemple la classe Catégorie utilise cette classe pour associer à chaque Catégorie une image en attribut, ce qui permet un binding plus simple.

3. Diagramme de paquetage

Vous pouvez trouver le diagramme ici : Diagramme de paquetage.jpg

Le diagramme représente comment les classes sont réparties dans l'application, il y a 3 différents grands paquetages (Managers, Models, Views) :

- Managers : S'occupe de la gestion des données, en l'occurrence ici la base de données mais aussi des tests unitaires
- Models : Comme son nom l'indique, c'est la partie métier de l'application ou l'on gère les objets (Utilisateurs, Films, ..) et les différents concepts objets comme les convertisseurs ou bien les interfaces, dépend de Managers
- Views : Gère toute la la partie graphique, dépend de Models

4. Diagramme de séquence

Vous pouvez trouver le diagramme ici : Diagramme de séquence.jpg

Ce diagramme décrit le mécanisme de connexion de l'application, lorsque l'utilisateur rentre ses informations c'est-à-dire son pseudo et son mot de passe, si les informations sont valides (ex : non nulles) on questionne la base de donnée si le doublon pseudo/mdp existe, si oui elle retourne l'utilisateur en question ou bien null. Si l'utilisateur n'est pas null, alors on peut connecter l'utilisateur et changer de UserControl.

APPROBATION ET AUTORISATION

Nous approuvons le projet tel qu'il est décrit ci-dessus, et autorisons l'équipe à le mettre en œuvre.

Nom	Titre	Date
VARA Clément G5	Manager, Développeur	07 / 06 / 2018