31/01/2024

SeatSwift – Cahier de conception

Développement d’application ESP

Michael Tremblay

Cégep de chicoutimi

Table des matières

[Cahier de conception 2](#_Toc157630278)

[Rappel 2](#_Toc157630279)

[Rappel du besoin 2](#_Toc157630280)

[Solution retenue 2](#_Toc157630281)

[Modélisation 5](#_Toc157630282)

[Diagramme(s) de cas d’utilisation 5](#_Toc157630283)

[Scénarios d’utilisation de la solution 5](#_Toc157630284)

[Diagrammes de séquences 5](#_Toc157630285)

[Diagramme de classes 6](#_Toc157630286)

[Diagramme de base de données 6](#_Toc157630287)

[Maquettes 6](#_Toc157630288)

[Fenêtre 1 6](#_Toc157630289)

[Fenêtre 2 6](#_Toc157630290)

[Plan de tests 6](#_Toc157630291)

[Tests fontionnels 6](#_Toc157630292)

[Tests unitaires 6](#_Toc157630293)

[Calendrier des échéanciers pour la conception 8](#_Toc157630294)

# Cahier de conception

## Rappel

### Rappel du besoin

L'application de gestion permettra la configuration de la salle de spectacle, la gestion des événements et la production de rapports détaillés. Elle inclura un système de cotation des sièges pour offrir les meilleurs billets disponibles. Les rapports couvriront les ventes et transactions.

Le guichet autonome, devra être une interface tactile intuitive, elle permettra aux clients d'acheter des billets, de sélectionner des événements, des dates, et de choisir des sièges. Un profil client pourra être créé pour faciliter les achats futurs. Le système ne traite pas directement les paiements, cependant, il contiendra un formulaire de saisie des informations de paiement pour simuler cette étape.

### Solution retenue

La solution recommandée est celle modulaire avec deux applications distinctes, développées en utilisant la "Option de Développement Propre avec Logiciels Non Open Source".

#### Fonctionalités de la solution

##### Logiciel de gestion

1. Gestion de la salle :
   1. Sièges (Normal et mobilité réduite)
   2. Sections
   3. Rangées
   4. Si un élément est retiré, on veut quand même garder l’historique
   5. Autre :
2. Gestion des évènements :
   1. Nom de l’évènement
   2. Artiste ou troupe de l’évènement
   3. Image de l’évènement
   4. Type d’évènement : Humour, chant, etc.
   5. Nombre de place
   6. Date et heure de l’évènement (Plusieurs date ou heure pour un même évènement.)
   7. Description de l’élément
   8. Salle pour l’évènement
   9. Nombre de billet maximum par client
   10. Prix de base pour un billet et multiplicateur selon les sections
   11. Autre :
3. Gestion des employés :
   1. Nom
   2. Prénom
   3. Numéro d’employé
   4. Mot de passe
   5. Rôle
   6. Numéro de téléphone
   7. Adresse courriel
   8. Reconnexion après chaque ouverture
   9. Autre :
4. Cotation des sièges
5. Rapport de vente
6. Rapport de transaction
7. Exportation des rapports en PDF (Possibilité d’Excel si trop complexe.)

##### Guichet autonome 4:3

1. Création d’un compte client
   1. Nom
   2. Prénom
   3. Adresse courriel
   4. Mot de passe
   5. Ajouter des champs optionnels comme la ville ou l’âge par exemple
   6. Autre :
2. Connexion à l’ouverture du guichet
3. Déconnexion à la fermeture ou après X seconde d’inactivité
4. Achat de billet
   1. Sélection du spectacle, sélection de la date, nombre de billet.
   2. Autre :
5. Sélection des sièges (Siège pour personne à mobilité réduite inclus)
   1. Meilleurs sièges automatique dû à certain filtre
   2. Possibilité de changer les critères de sélection automatique
   3. Autre :
6. Avertissement si le nombre de place adjacente ne sont pas suffisantes
7. Bannière « complet » sur image du spectacle si celui-ci est complet
8. Validation de la carte de crédit
9. Processus d’achat
   1. Sélection du spectacle, de la date et du nombre de billet
   2. Sélection des places
   3. Possibilité d’ajouter au panier
   4. Validation des achats
   5. Paiement
   6. Remerciement et possibilité d’effectuer d’autre achat ou déconnexion
   7. Autre :
10. Envoie de la facture et du billet par courriel au client
    1. Le billet sera un Code QR

##### Ajout supplémentaire possible

1. Sélection manuelle des sièges
2. Possibilité d’ajouter des configurations de salle
3. Autre :

#### Raisonnement de la Recommandation

La Solution 2, avec des applications séparées pour la billetterie et la gestion des événements, est privilégiée pour plusieurs raisons, renforcées par l'approche de développement propre avec des logiciels non open source :

**Flexibilité :** Le développement distinct offre la possibilité d'adapter et de mettre à jour chaque application de manière indépendante, assurant ainsi une évolution spécifique selon les besoins.

**Mises à Jour Simplifiées :** Avec deux systèmes séparés, les mises à jour peuvent être effectuées individuellement, évitant les interférences entre les fonctionnalités de billetterie et de gestion des événements.

**Spécialisation :** Chaque application peut être finement optimisée pour ses fonctions spécifiques, améliorant ainsi l'efficacité et l'expérience utilisateur.

**Concentration sur l'Interface Utilisateur pour la Billetterie :** Cette séparation permet un focus particulier sur l'interface tactile de la billetterie, la rendant intuitive et agréable pour les utilisateurs.

**Évolutivité :** L'approche modulaire facilite l'adaptation ou l'extension d'une application en fonction de l'évolution des besoins ou des nouvelles technologies, sans impacter l'autre application.

**Sécurité et Contrôle :** En utilisant des logiciels non open source pour le développement, nous bénéficions d'une sécurité renforcée et d'un contrôle total sur les fonctionnalités et l'intégration des systèmes.

**Intégration sur Mesure :** Le développement propre avec des logiciels non open source permet une intégration plus précise avec d'autres systèmes et outils commerciaux, offrant une solution complète et cohérente.

Choisir de développer deux applications séparées avec une approche de logiciels non open source apporte flexibilité, spécialisation, et sécurité, tout en offrant de meilleures possibilités d'évolution pour répondre aux besoins du Théâtre CChic. Bien que le développement de deux applications distinctes puisse sembler plus complexe, cette stratégie simplifie en réalité la gestion des différentes interfaces (Client et Admin), tout en assurant une séparation claire et sécurisée entre elles.

## Modélisation

### Diagramme(s) de cas d’utilisation

### Scénarios d’utilisation de la solution

|  |
| --- |
| **Nom du cas** :  **But** :  **Acteur Principal** :  **Acteur(s) Secondaire(s)** : |
| **Séquencement**  Précondition(s)  Séquence Nominale  Postcondition(s) |
| **Séquences Alternatives et/ou Exceptions**  Séquence Alternative #1 - \*\*Raison de la séquence alternative\*\* |

### Diagrammes de séquences

### Diagramme de classes

### Diagramme de base de données

## Maquettes

### Fenêtre 1

### Fenêtre 2

## Plan de tests

### Tests fontionnels

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Description | Précondition(s) | Scénario(s) | Résultats attendues |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| 6 |  |  |  |  |
| 7 |  |  |  |  |
| 8 |  |  |  |  |
| 9 |  |  |  |  |
| 10 |  |  |  |  |

### Tests unitaires

Nous exécuterons des tests unitaires à chaque ouverture et chaque fermeture de notre programme. Cette méthode assure que chaque fonctionnalité est vérifiée en temps réel pendant le développement, permettant de détecter et de corriger immédiatement les anomalies.

|  |  |  |  |
| --- | --- | --- | --- |
| # | Description | Précondition(s) | Résultats attendues |
| 1 |  |  |  |
| 2 |  |  |  |
| etc. |  |  |  |

# Calendrier des échéanciers pour la conception