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

Puisque le projet requiert deux types d’utilisateurs distincts qui utiliseront le programme de façon différente, deux diagrammes ont donc été conçus : l’un pour l’application de gestion et l’autre pour la borne autonome.

### Diagramme de cas d’utilisation

#### Guichet autonome

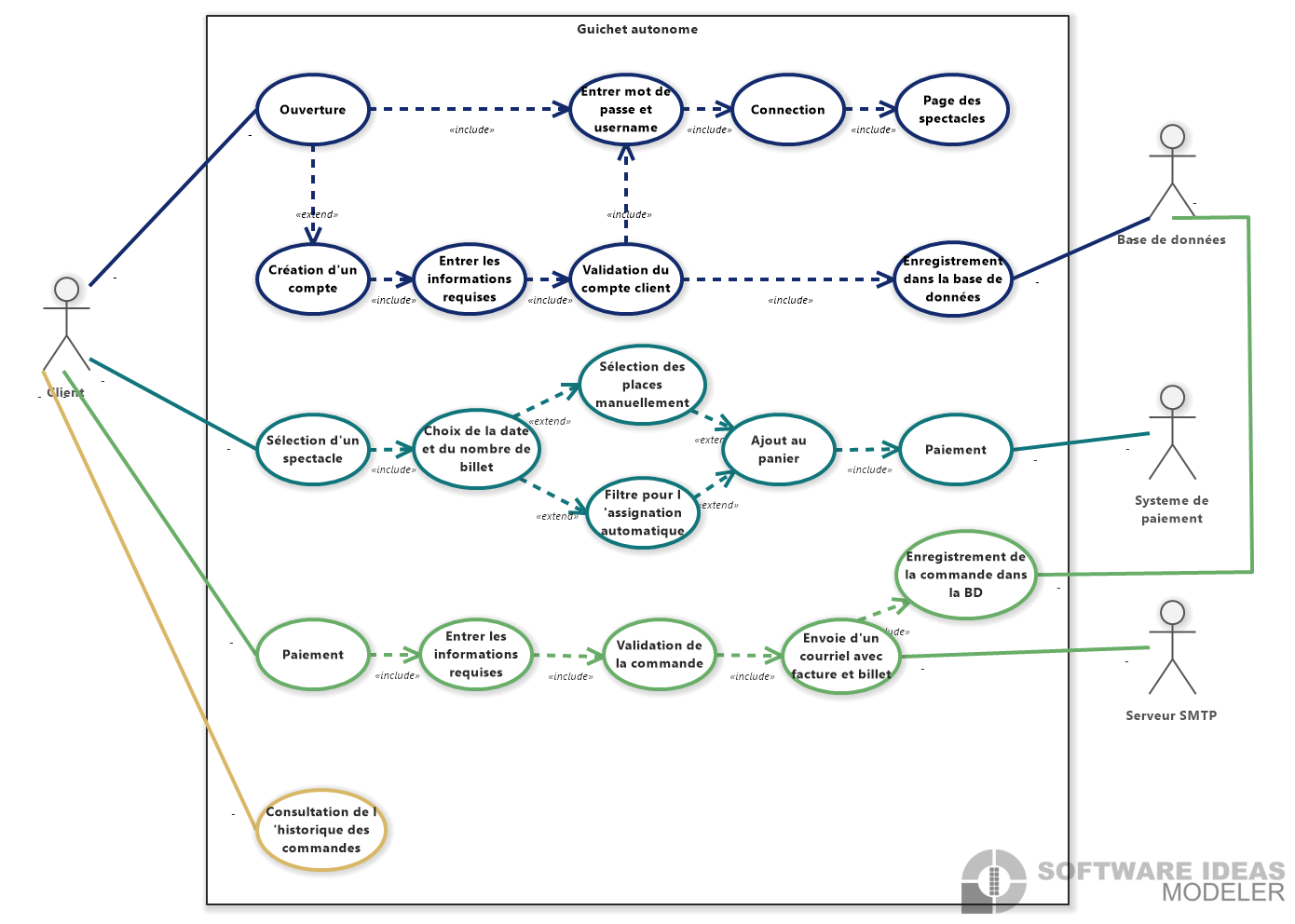
Le diagramme de la figure 1 illustre l'ensemble des actions que l'utilisateur pourra effectuer en utilisant le guichet autonome. Les principaux états de l'application, sont : démarrage/arrêt, Sélection d’un spectacle, paiement, Consultation de l’historique.

Figure 1 : Diagramme de cas d’utilisation pour le guichet autonome

#### Application de gestion

Le diagramme de la figure 1 illustre l'ensemble des actions que l'utilisateur pourra effectuer en utilisant l'application de gestion. L’idée d'adopter un code de couleurs a été retenue et sera appliquée à ce diagramme de cas d'utilisation. Celui-ci permettra de distinguer facilement les principaux états de l'application, à savoir : démarrage/arrêt, gestion des salles, gestion des spectacles, Gestion des employés, Visualisation des rapports et Visualisation en mode lecture seule.

Une image contenant texte, carte, diagramme, capture d’écran

Description générée automatiquement

Figure 2 : Diagramme de cas d’utilisation de l’application de gestion.

### Différents Scénarios d’utilisation

|  |
| --- |
| **Nom du cas** : Ouverture du guichet autonome  **But** : Accéder au spectacle disponible  **Acteur Principal** : Client  **Acteur(s) Secondaire(s)** : Base de données |
| **Séquencement**  Précondition(s)   * L’utilisateur de posséder au compte client pour accéder aux fonctionnalités du guichet autonome.   Séquence Nominale   1. Le client appuie sur l’écran avec un doigt pour retirer l’état de veille de l’ordinateur. 2. Une fenêtre contextuelle apparait demandant à l’utilisateur de se connecter. (Si le client n’a pas de compte voire séquence alternative #1) 3. Le client entre son mot de passe et son nom d’utilisateur. 4. Il appuie sur se connecter. 5. Le système vérifie si le compte existe dans la base de données. 6. Si le système valide le client, Il peut désormais accéder à la page des spectacles du guichet autonome. (Si le système ne valide pas, Voire séquence alternative #2)   Post-condition(s)   * L’utilisateur est connecté |
| **Séquences Alternatives et/ou Exceptions**  Séquence Alternative #1 – Le client n’a pas de compte.   1. **Le client appuie sur le bouton « Créer un compte ».** 2. **Une fenêtre contextuelle apparait lui demandant d’entrer les informations obligatoires et suggérant aussi des informations optionnelles.** 3. **À l’aide du clavier virtuelle, l’utilisateur entre les informations voulue.** 4. **Il valide la création de son compte.** 5. **Le nouveau compte est enregistré dans la base de données.** 6. Une fenêtre contextuelle apparait demandant à l’utilisateur de se connecter. (Continuer la séquence nominale)   Séquence Alternative #2 – L’utilisateur à commet une erreur en entrant son mot de passe ou son nom d’utilisateur.   1. **Un message d’erreur apparait pour avertir le client qu’il a commis une erreur.** 2. **Il peut entrer ses informations à nouveau.** (Continuer la séquence nominale) |

|  |
| --- |
| **Nom du cas** : Ajout d’un spectacle  **But** : Ajouter un nouveau spectacle disponible dans le système  **Acteur Principal** : Admin  **Acteur(s) Secondaire(s)** : Base de données |
| **Séquencement**  Précondition(s)   * L’utilisateur doit-être un administrateur pour ajouter un spectacle.   Séquence Nominale   1. L’administrateur accède à la page de gestion des spectacles. 2. Il sélectionne ajouter un spectacle. 3. Il entre les informations requises. 4. Il sélectionne une salle pour le spectacle. (Si le client n’a pas de compte voire séquence alternative #1) 5. L’administrateur confirme la création du spectacle. |
| **Séquences Alternatives et/ou Exceptions**  Séquence Alternative #1 – Aucune configuration de salle n’est disponible.   1. **L’utilisateur est redirigé vers la page de configuration de salle.** 2. **Il entre les informations requises.** 3. **L’utilisateur est redirigé vers la page de configuration du spectacle pour choisir une salle.** |

### Diagrammes de séquences

Une image contenant capture d’écran, texte

Description générée automatiquement

Figure 3 - Séquence d'ajout d'une entité par l’administrateur dans l’application de gestion Le diagramme de séquences suivant est une séquence basique d’ajout d’une entité au système. Elle fonctionnera pour la plupart des entités du système.

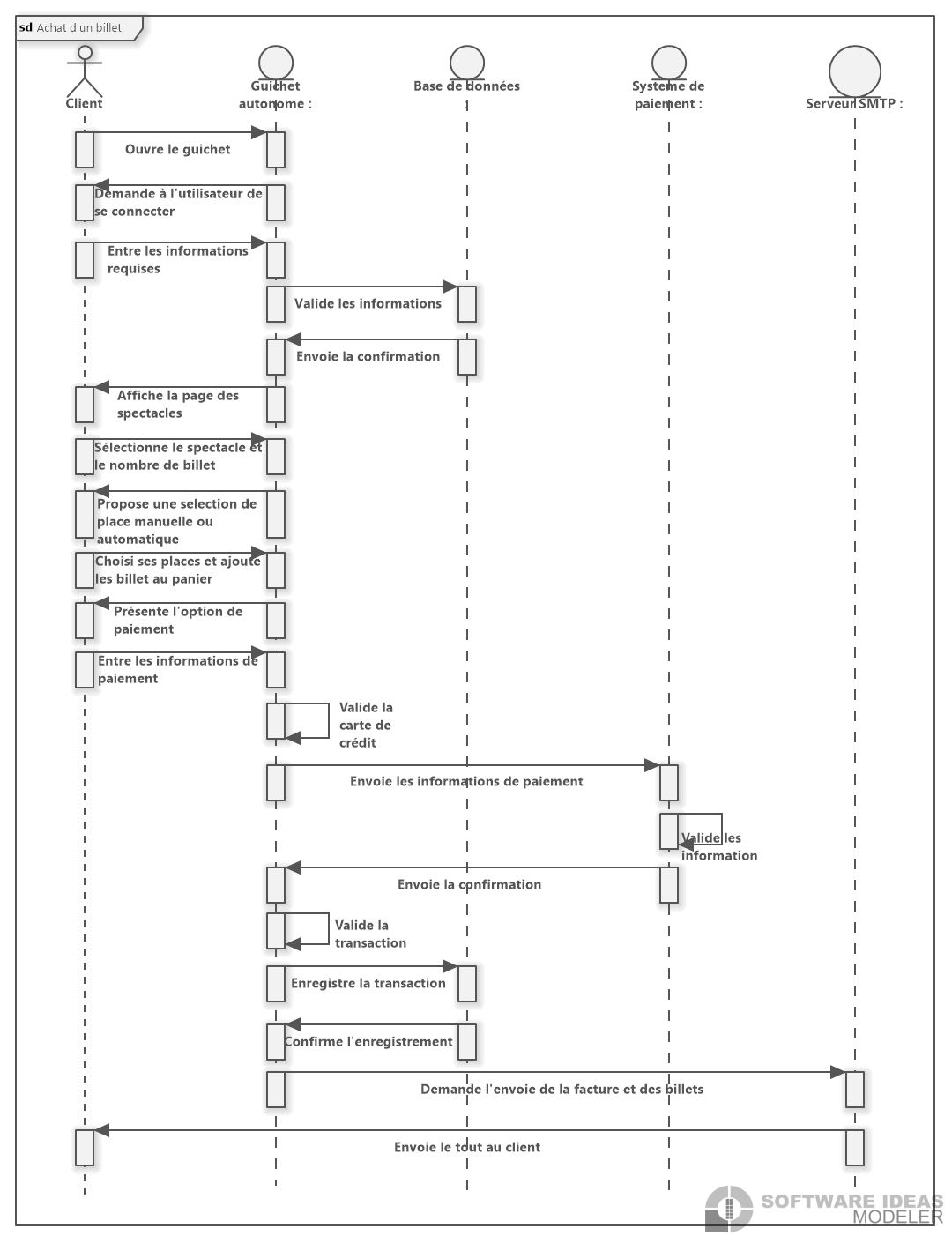


Figure 4 - Séquence d'achat de billet par un client. On peut voir l'envoie de courriel, la validation du système de paiement, etc.

### 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