Documentation du projet Cinéphoria

Benjamin BALET

Table des matières

1. Informations Générales

- **Nom du projet :** Cinéphoria - Développement d'Applications
- **Date de début :** 01/07/2024
- **Date de fin :** 22/07/2024
- **Méthodologie :** Scrum / Kanban
- **Équipe :**
  - Product Owner : Benjamin BALET
  - Scrum Master : Benjamin BALET
  - Développeurs : Benjamin BALET
  - Testeurs : Benjamin BALET

2. Objectifs du Projet

- **Objectif principal :** Développer une plateforme moderne et performante pour améliorer l'expérience des clients des cinémas Cinéphoria.
- **Livrables :**
  - Application web
  - Application mobile
  - Application bureautique
  - Guide utilisateur
  - Documentation technique
  - Documentation du projet - le présent document
  - Transaction SQL
  - Artefacts SQL (modèle et JDD)

3. Planning

Les éléments clés du planning projet sont présentés dans la table ci-dessous :

Sprint	Dates	Objectifs principaux	Livrables
1	01/07 - 07/07	Configuration initiale et mise en place du CI/CD	Dépôt Git, environnement de développement
2	08/07 - 14/07	Développement des fonctionnalités critiques	Fonctions de réservation, création de compte
3	15/07 - 21/07	Intégration et tests	Tests unitaires et fonctionnels, déploiement
4	22/07	Révision finale et documentation	Documentation complète, manuel utilisateur

Table 1: Éléments clés du planning projet

4. Gestion des Risques

Risque	Probabilité	Impact	Stratégie d'atténuation
Retard dans le développement	Moyenne	Élevé	Révisions de sprint régulières, ajustements flexibles
Problèmes d'intégration avec les composants	Faible	Moyen	Tests réguliers
Choix techniques inappropriés	Moyenne	Élevé	POC, développement au plus tôt

Table 2: Plan de gestion des risques

5. Backlog du Produit

5.1 User Stories

Le Product Backlog est composé des Users Stories synthétisées dans le tableau suivant:

ID	Titre	Description	Priorité
US1	Menu du site web	La page d'accueil du site web doit comporter un menu visible sur toutes les pages	Haute
US2	Page d'accueil	Afficher les derniers films ajoutés le dernier mercredi	Moyenne
US3	Réservation	Sélection du cinéma, film, qualité et choix des sièges	Haute
US4	Création de compte	Formulaire de création de compte avec email et mot de passe	Haute
US5	Connexion	Authentification des utilisateurs	Haute
US6	Visionner les séances du jour	Visionner toutes les séances auxquelles l'utilisateur possède un billet	Moyenne
US7	QR code	Visualiser le QR code servant de billet : permettre à un employé de le scanner	Moyenne
US8	Communication des incidents	Saisir les incidents pour les installations défaillantes	Basse
US9	Bas de page	Pied de page avec adresse, numéro de téléphone et horaires visibles sur toutes les pages	Basse
US10	Les films	Afficher tous les films avec options de filtre par cinéma, genre et jour	Haute
US11	Espace Administrateur	Gestion des films, séances, salles et comptes employés, avec tableau de bord des réservations	Haute
US12	Espace Employé	Gestion des films, séances, salles et modération des avis	Moyenne
US13	Espace Utilisateur	Accès à l'historique des commandes et possibilité de noter les films	Basse
US14	Mot de passe oublié	Réinitialisation du mot de passe avec envoi d'un nouveau mot de passe par email	Moyenne
US15	Contact	Formulaire de contact pour les utilisateurs et visiteurs	Basse

Table 3: Product Backlog

5.2 Analyse des exigences

5.2.1 Exigences Web

ID	Exigence	Description
W1	Menu du site web	Lien vers la page d'accueil, se connecter à son compte, réservation, films, contact
W2	Page d'accueil	Afficher les derniers films ajoutés chaque mercredi
W3	Bas de page	Afficher l'adresse du cinéma, numéro de téléphone, horaires
W4	Réservation	Sélection du cinéma, film, qualité, choix des sièges, affichage du prix, création de compte
W5	Les films	Afficher tous les films avec options de filtre par cinéma, genre, jour
W6	Création de compte	Formulaire demandant email, mot de passe (8 caractères minimum), prénom, nom
W7	Connexion	Formulaire de connexion demandant login et mot de passe
W8	Espace Administrateur	Gestion des films, séances, salles, comptes employés, tableau de bord des réservations
W9	Espace Employé	Gestion des films, séances, salles, validation et suppression des avis
W10	Espace Utilisateur	Visualisation des commandes, possibilité de noter les films
W11	Mot de passe oublié	Génération d'un nouveau mot de passe envoyé par mail
W12	Contact	Formulaire de contact avec nom d'utilisateur, titre de la demande, description

Table 4: Exigences Web

5.2.2 Exigences Mobile

ID	Exigence	Description
M1	Visionner les séances du jour	Afficher les séances pour lesquelles l'utilisateur possède un billet
M2	QR code	Afficher le QR code pour chaque séance
M3	Scanner QR code	Un employé doit pouvoir scanner ce QR code pour valider l'accès

Table 5: Exigences Mobile

5.2.3 Exigences Bureautique

ID	Exigence	Description
B1	Liste des incidents	Les employés peuvent voir la liste des incidents
B2	Communication des incidents	Saisir les installations défaillantes dans une salle

Table 6: Exigences Bureautique

5.2.4 Exigences de sécurité

ID	Exigence	Description
S1	Authentification	Chaque utilisateur a un compte et un mot de passe qui lui sont propres
S2	Rôles	Il existe trois rôles : administrateur, employé, visiteur, anonyme (c.-à-d. non connecté)
S3	Privilèges	Les habilitations liées aux rôles sont basées sur le principe du moindre privilège
S4	Failles	L'application doit être protégée contre les failles de sécurités de tpe XSS, injection SQL, etc.

Table 7: Exigences de sécurité

5.2.5 Environnement de Test

ID	Exigence	Description
T1	Jeu de test	Un environnement de test complet pour au moins une application, incluant tests unitaires et fonctionnels

Table 8: Environnement de Test

5.2.6 Déploiement et Documentation

ID	Exigence	Description
D1	Stack technique possible	Application web : HTML 5, CSS (Bootstrap), JS, PHP (PDO), MySQL/MariaDB/PostgreSQL, MongoDB, déploiement sur fly.io; Application mobile : Flutter; Application bureautique : Python (Tkinter)
D2	Livrables	Lien vers le dépôt GitHub public, lien des applications déployées, lien vers le logiciel de gestion de projet, README.md avec instructions de déploiement en local, manuel d'utilisation en format PDF, charte graphique en format PDF, documentation de gestion de projet, documentation technique
D3	Gestion de projet	Kanban partagé avec colonnes : fonctionnalités prévues, à faire, en cours, terminées, fusionnées dans la branche principale

Table 9: Déploiement et Documentation

6. Utilisation de GitHub Projects

GitHub Projects sera utilisé pour gérer les tâches et suivre l'avancement du projet. La capture d'écran ci-dessous montre une vue de la progression des activités durant le projet.

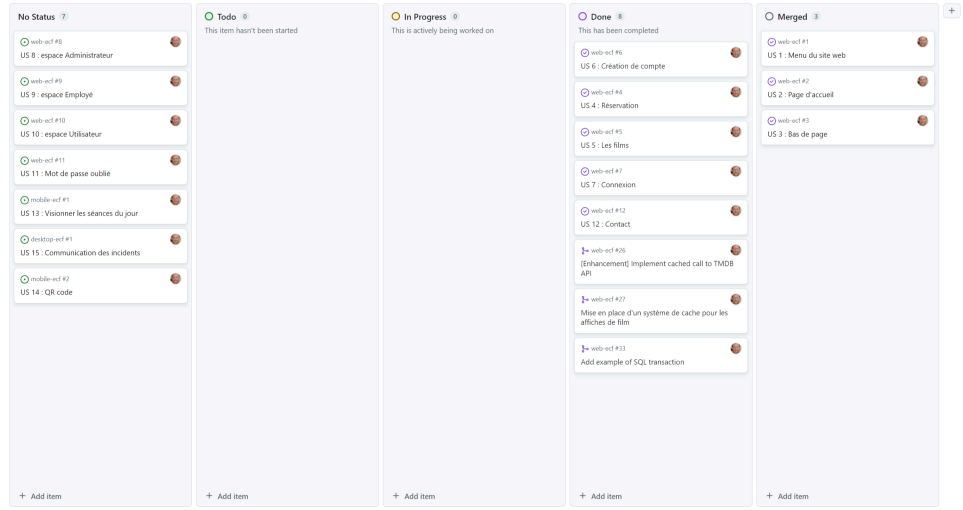

Figure 1: Vue Kanban dans GitHub projects

Les paragraphes suivants expliquent comment il sera structuré et utilisé.

6.1 Colonnes du Kanban

- **Backlog (*No Status*) :**
  - Contient toutes les user stories et tâches à réaliser, ordonnées par priorité.
- **To Do :**
  - Tâches sélectionnées pour le sprint en cours, prêtes à être commencées.
- **In Progress :**
  - Tâches actuellement en cours de développement.
- **Done :**

  - Tâches complètes et validées, prêtes pour le déploiement.
- **Merged :**
  - Tâches intégrées dans la branche principale.

6.2 Workflow

1. **Création des Issues :**
   - Chaque user story sera créée comme une issue dans le dépôt GitHub.
   - Les issues seront étiquetées selon leur type (bug, feature, enhancement) et leur priorité (high, medium, low).
2. **Ajout aux Projects :**
   - Les issues seront ajoutées au projet Kanban correspondant pour le suivi visuel.
3. **Sprint Planning :**
   - Les issues prioritaires seront déplacées de “Backlog” à “To Do” en début de sprint.
4. **Développement :**
   - Les développeurs attribueront les issues à eux-mêmes et les déplaceront vers “In Progress” lorsqu’ils commenceront à travailler dessus.
5. **Revue et Validation :**
   - Une fois une tâche terminée et validée, elle sera déplacée vers “Done” par le Scrum Master ou le Product Owner.
6. **Complétion :**
   - Les tâches validées seront déplacées vers “Done”.
7. **Déploiement :**
   - Les tâches mergées dans la branche *main* seront déplacées vers “Merged”.

7. Réunions et Cérémonies

- **Réunions quotidiennes :** 15 minutes chaque matin pour faire le point sur l’avancement.
- **Sprint planning :** Planification des tâches pour chaque sprint.
- **Sprint review :** Révision des livrables à la fin de chaque sprint.
- **Sprint retrospective :** Retour sur les points d’amélioration et succès du sprint.

8. Suivi et Reporting

- **Outils utilisés :** GitHub Projects pour la gestion des tâches, GitHub Issues pour le suivi des bugs et des fonctionnalités.
- **Fréquence de rapport :** Rapport hebdomadaire de l’avancement à l’équipe et au Product Owner.

9. Plan de Test

- **Tests unitaires :** Écriture et exécution de tests unitaires pour chaque fonctionnalité.
- **Tests fonctionnels :** Vérification des fonctionnalités clés dans un environnement de test.
- **Tests de performance :** Évaluation des performances de l’application sous charge.

10. Déploiement

- **Environnement de production :** fly.io pour le déploiement des applications.
- **Documentation de déploiement :** README.md dans le dépôt Git avec les instructions de déploiement.

11. Documentation

- **Manuel d’utilisation :** PDF détaillant l’utilisation des applications.
- **Charte graphique :** PDF avec palette de couleurs et typographie.

- **Documentation technique :** Explications des choix technologiques, architecture logicielle, MCD, diagrammes de séquence.

12. Gitflow

Le Gitflow est une stratégie de gestion des branches Git qui vise à améliorer le développement collaboratif et à structurer efficacement les cycles de développement et de déploiement. Il repose sur deux branches principales : **main** (ou **master**) et **develop**. La branche **main** contient le code de production stable, tandis que la branche **develop** intègre les nouvelles fonctionnalités en cours de développement.

Les nouvelles fonctionnalités sont développées dans des branches **feature** qui dérivent de **develop** et sont fusionnées dans **develop** une fois terminées. Pour les correctifs de bugs sur le code de production, des branches **hotfix** sont créées à partir de **main** et fusionnées dans **main** et **develop** après correction. Les versions de préparation au déploiement sont gérées via des branches **release**, dérivées de **develop** et fusionnées dans **main** et **develop** une fois prêtes.

Cette structure permet de maintenir un code stable en production tout en facilitant le développement parallèle de nouvelles fonctionnalités et la gestion des correctifs. Elle améliore également la traçabilité des changements et la collaboration entre les membres de l'équipe.

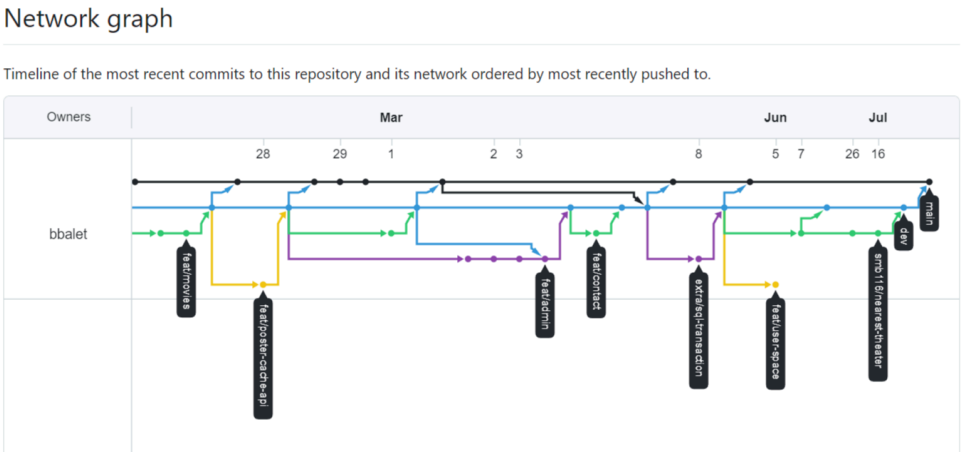

Figure 2: Gitflow sur le projet Cinéphoria

Annexes

Table des illustrations

Table des tableaux