

Portfolio Project - Project Closure

Seigneur des Anneaux – Interactive Wiki

1. RÉSUMÉ DES RÉSULTATS

1.1 Vue d'ensemble du projet

Durée totale : 13 août - 31 octobre 2024 (2 mois et demi)

- Phase de documentation : 13 août - 28 septembre
- Phase de développement : 29 septembre - 31 octobre

Équipe : 2 développeurs

1.2 Fonctionnalités principales du MVP

Le MVP livré comprend les fonctionnalités suivantes :

Fonctionnalités implémentées (90% du MVP)

Carte Interactive



- Carte de la Terre du Milieu interactive
- Système de marqueurs cliquables avec popups informatifs
- Intégration de Leaflet avec React

Système d'authentification



- Création de compte utilisateur
- Connexion/Déconnexion sécurisée
- Gestion des sessions avec JWT
- Hashage des mots de passe avec bcrypt
- Mode invité avec limitation aux contenus

Wiki dynamique



- 13 pages fonctionnelles principales
- Architecture basée sur des composants React réutilisables
- Contenu entièrement dynamique (races, personnages, histoires)
- Toutes les données stockées et récupérées depuis PostgreSQL

Galerie de créations artistiques



- Upload d'images pour utilisateurs connectés
- Système de commentaires sur chaque création
- Affichage et navigation dans la galerie

Expérience utilisateur



- Design apprécié par les utilisateurs tests
- Navigation intuitive avec l'ajout d'une barre de navigation pour se diriger sur chaque menu différent rapidement avec l'ajout de logo intuitif

✗ Fonctionnalités non implémentées (10% du MVP initial)

- Mode administrateur pour modification du contenu via interface
- Effets d'animations avancés sur la carte
- Système de filtres par type de contenu
- Responsive complet pour la carte interactive sur mobile

1.3 Comparaison avec les objectifs initiaux

Objectif initial	Statut	Commentaire
Carte interactive Seigneur des Anneaux	Validé	Fonctionnelle avec marqueurs ajoutés au bon endroits et popups apparaissant au clic utilisateur
Wiki informatif dynamique	Validé	Toutes les données sont stockées en base de données, 13 pages créés via react
Galerie de créations artistiques	Validé	Upload d'images et commentaires sur une image fonctionnel
Système d'authentification	Validé	Sécurisé avec JWT et bcrypt
Mode invité	Validé	Accès limité implémenté pour la partie galerie et aucune possibilité pour modifier les textes informatifs
Mode admin	Partiel	Gestion côté back uniquement
Responsive mobile	Partiel	Réalisé excepté pour la carte car trop complexe
Animations carte	Non Validé	Complexité technique et manque de temps
Système de filtres	Non Validé	Complexité technique

1.4 Métrique clé

Technique

- 13 pages fonctionnelles principales
- Architecture complète Frontend (React + Vite) / Backend (Flask)
- Base de données PostgreSQL avec tables relationnelles complexes
- 100% des appels API fonctionnels
- Gestion d'erreurs complète sur les fonctionnalités critiques

Retours utilisateurs

- Design très apprécié
- Navigation intuitive
- Expérience mobile perfectible sur la carte
- Fluidité du système d'upload d'images

2. LECONS APPRISES

2.1 Ce qui a bien fonctionné

Architecture technique solide

Ce qui s'est passé : L'utilisation de Flask avec ses extensions (RestX, SQLAlchemy, JWT) couplée à React a permis de créer une architecture claire et maintenable.

Pourquoi c'était efficace :

- Séparation claire Frontend/Backend
- Composants React réutilisables
- API REST bien structurée avec Flask-RESTX
- ORM SQLAlchemy facilitant la gestion de la base de données

À conserver pour les futurs projets : Cette stack technologique moderne et l'architecture en composants.

Gestion dynamique du contenu

Ce qui s'est passé : Le choix de stocker 100% du contenu en base de données plutôt qu'en HTML statique.

Pourquoi c'était efficace :

- Facilite les mises à jour de contenu
- Évolutivité du projet
- Cohérence des données
- Possibilité future d'ajouter un vrai mode admin

À conserver pour les futurs projets : Toujours privilégier le contenu dynamique dès le départ.

Système d'authentification sécurisé

Ce qui s'est passé : Implémentation complète avec JWT, bcrypt et gestion des sessions.

Pourquoi c'était efficace :

- Sécurité des données utilisateurs
- Expérience utilisateur fluide
- Gestion des erreurs robuste

À conserver pour les futurs projets : Ne pas négliger la sécurité dès le MVP.

Phase de documentation préalable

Ce qui s'est passé : 6 semaines dédiées à la documentation et à l'apprentissage avant de commencer le développement.

Pourquoi c'était efficace :

- Vision claire du projet
- Réduction des hésitations pendant le développement
- Meilleure organisation du travail en binôme
- Apprentissage de React et décision des langages à utiliser

À conserver pour les futurs projets : Investir du temps dans la planification.

2.2 Défis rencontrés et solutions apportées

Défi 1 : Carte interactive avec Leaflet et React

Problème : Intégration complexe de Leaflet dans l'écosystème React, notamment la gestion des marqueurs et des popups.

Comment résolu :

- Apprentissage approfondi de la documentation Leaflet
- Utilisation de l'intelligence artificielle

Impact : Carte fonctionnelle mais responsive mobile limité.

Recommandation future : Prévoir plus de temps pour les fonctionnalités de cartographie

Défi 2 : Apprentissage de React en situation réelle

Problème : Courbe d'apprentissage de React (composants, props, state, hooks) pendant le développement.

Comment résolu :

- Formation progressive par la pratique
- Refactorisation du code au fur et à mesure de la compréhension
- Pair programming pour partager les connaissances
- Demande d'informations avec les autres étudiants

Impact : Quelques ralentissements en début de développement, mais montée en compétence rapide.

Recommandation future : Prévoir une phase de formation technique plus longue avant de démarrer le développement.

Défi 3 : Structure de la base de données évolutive

Problème : Le schéma initial de la base de données nécessitait des ajustements (tables manquant d'informations, tables trop complexes, relations polymorphes difficiles).

Comment résolu :

- Simplification de certaines relations complexes

Impact : Perte de temps car base de données déconstruit plusieurs fois et souvent relancé

Recommandation future : Rester simple dans la conception de la base de données. Pour un projet débutant la base de données était probablement trop complexe.

Défi 4 : Gestion de l'upload d'images avec JavaScript

Problème : Complexité de la gestion des fichiers multipart/form-data entre le frontend et le backend.

Comment résolu :

- Utilisation de l'intelligence artificielle
- Tests exhaustifs avec différents formats d'images

Impact : Système d'upload final fluide et fonctionnel.

Recommandation future : Anticiper la complexité de la gestion des fichiers dans les planifications.

Défi 5 : Tables polymorphes et relations complexes

Problème : Difficulté à comprendre et implémenter des relations polymorphes dans SQLAlchemy.

Comment résolu :

- Simplification de certaines relations
- Schémas (table)alternatifs plus simples

Impact : Perte de temps en reconstruisant une table et modifiant les modèles, routes et façade.

Recommandation future : Évaluer le rapport complexité/bénéfice des relations polymorphes. Parfois, une solution plus simple est préférable.

2.3 Points d'amélioration pour les futurs projets

1. Responsive design dès le départ

Constat : Le responsive a été traité en fin de projet, mais sur certaine page, il aurait été préférable d'y penser au début, étant donné que c'est un prérequis à avoir sur le portfolio.

Amélioration : Tester sur mobile tout au long du développement.

2. Estimation du temps pour les fonctionnalités avancées

Constat : Certaines fonctionnalités prévues (animations, filtres) n'ont pas été implémentées par manque de temps et complexité de code.

Amélioration : Nous avons mis du temps à le développer côté back pour au final ne pas s'en servir en front pour notre site. On aurait du rendre ce type de fonctionnalité optionnelle au départ et le développer qu'à la fin.

3. Synchronisation des diagrammes avec le code

Constat : Les diagrammes initiaux ne correspondent plus au produit final.

Amélioration : Mettre à jour la documentation technique au fil du projet, notamment après chaque modification significative de la base de données.

2.4 Recommandations pour la suite du projet

Court terme

1. Corriger les dernières briques en attente
2. Mettre à jour les diagrammes techniques pour refléter l'état actuel
3. Décider de la stratégie mobile pour la carte (retrait avec message ou carte toujours présente elle sera moins manipulable que sur pc)

Moyen terme

1. Implémenter le mode administrateur avec interface graphique
2. Implémenter un mode utilisateur avec modification de mdp ou autre information personnelle
3. Optimiser les performances de chargement des images
4. Améliorer le responsive sur mobile

Long terme (évolution du projet)

1. Ajouter des animations sur la carte
 2. Système de recherche avancée dans le wiki
 3. Ajouter le système de filtres par type de contenu
-

3. CONCLUSION

Le MVP du Wiki interactif du Seigneur des Anneaux a atteint **90% de ses objectifs initiaux**. Les fonctionnalités core (carte interactive, wiki dynamique, authentification, galerie artistique) sont toutes opérationnelles et appréciées par les utilisateurs tests.

Les principaux défis techniques (intégration Leaflet-React, apprentissage de React, gestion de la base de données) ont été surmontés grâce à la persévérance et un travail assidu de l'équipe et une approche itérative.

Ce projet a permis à l'équipe de développer des compétences solides en développement full-stack moderne et de livrer un produit fonctionnel et esthétiquement réussi.

Points forts du projet :

- Architecture technique robuste et évolutive
- Design apprécié par les utilisateurs
- Système d'authentification sécurisé
- Gestion dynamique complète du contenu

Axes d'amélioration identifiés :

- Responsive mobile sur les composants complexes
- Estimation du temps pour les fonctionnalités avancées
- Tests utilisateurs plus précoce
- Chargement des pages plus courtes (prévoir de la pagination)