Dossier de projet - TPI

OUTIL DE MONITORING DE SITES / PROJETS DE NOS CLIENTS

Florent Scheibler

FIRSTPOINT SÀRL | GALERIE SAINT-FRANÇOIS B 1003 LAUSANNE

TABLE DES MATIERES

Analyse préliminaire 3
Introduction 3
Objectifs 3
Analyse /conception 4
Phase 1 - Concept 5
Architecture 5
Base de donnée 5
Interface utilisateur 6
Plnaification finale 7
Phase 2 - Concept 8
Phase 2.1 - Installation et configuration initiale de Laravel 8
Phase 2.2 - Implémentation de la base de données 8
Phase 2.3 - Intégration de l'authentification 8
Phase 2.4 - Développement de l'interface utilisateur – layout de base 9
Phase 2.5 - Intégration du services Oh Dear 9
Phase 2.6 - Intégration du service Flare 9
Phase 2.7 - Intégration de Livewire 10
Phase 2.8 – Finalisation de l'interface utilisateur – layout de base 10
Phase 2.8 - Mise en place du cron et fonctionnalité de rafraichissement 10
Phase 2.9 - Corrections et refactorisation 10
Phase 2.10 - Documentation utilisateur 11
Phase 2.11 – Déploiement de la documentation 11
Phase 2.12 - Corrections et refactorisation 11
Phase 2.13 - Evaluation et clôture du projet 11
Risques techniques 12
Glossaire 13
Source & bibliographie 14

ANALYSE PRÉLIMINAIRE

INTRODUCTION

Afin d'assurer un suivi proactif des projets de nos clients, nous avons besoin d'un outil en ligne permettant de monitorer les sites et d'avoir une vue d'ensemble des projets que nous gérons.

Le projet est conçu pour répondre aux besoins croissants de surveillance et de gestion des performances des sites web dans un contexte professionnel.

Réalisé dans le cadre de la formation d'informaticien CFC, ce projet vise à développer une application web utilisant la stack TALL (TailwindCSS, AlpineJS, Laravel, Livewire) pour intégrer des services de monitoring tels que Oh Dear et Flare.

Ce choix s'appuie sur la nécessité d'offrir une solution complète pour le suivi des performances web, la surveillance des certificats SSL, et la détection des liens brisés, tout en proposant une interface utilisateur intuitive et sécurisée.

OBJECTIFS

Phase d'analyse :

- Recueillir et analyser les exigences détaillées du projet.
- Concevoir l'expérience utilisateur (UX) et l'interface utilisateur (UI).
- Élaborer les schémas de la base de données pour supporter efficacement les fonctionnalités prévues.
- Identifier les intégrations nécessaires avec les services tiers (Oh Dear, Flare) et les défis techniques associés.
- Identifier les différentes phases d'implémentation et les stratégies de tests liées.

Phase d'implémentation :

- Concevoir et implémenter les schémas de base de données, en tenant compte des besoins d'évolutivité et de performance.
- Intégrer un système d'authentification sécurisé et flexible pour les utilisateurs de l'application (breeze).
- Créer des composants UI réactifs et accessibles en utilisant Tailwind CSS et Alpine.js, conformément aux maquettes UX/UI.
- Utiliser Livewire pour ajouter des fonctionnalités interactives sans compromettre la performance ou l'accessibilité.
- Intégrer efficacement les APIs d'Oh Dear et de Flare pour le monitoring et la gestion des incidents.
- S'assurant de la modularité et de la maintenabilité du code.
- Configurer des tâches automatisées pour la mise à jour des données.
- Allouer du temps pour l'ajustement basé sur les retours initiaux et pour intégrer des fonctionnalités supplémentaires non prévues.
- Effectuer des tests exhaustifs pour s'assurer de la stabilité, et de la performance de l'application.
- Rédiger une documentation technique complète et des manuels d'utilisation pour faciliter la prise en main de l'application.
- Analyser le déroulement du projet, recueillir les retours, et documenter les leçons apprises pour les projets futurs.

ANALYSE /CONCEPTION

Selon les directives du cahier des charges, le projet sera structuré en plusieurs phases distinctes, chacune soumise à des tests de validation préalables avant de procéder à l'étape suivante. La planification initiale divise le projet en deux grandes sections :

Analyse et Conception : Cette étape initiale se consacre à l'élaboration des objectifs et à la conception globale du projet, y compris l'architecture de l'application et l'interface utilisateur.

Implémentation et Tests : Divisée en plusieurs sous-phases, cette section couvre l'exécution concrète du projet.

Chaque sous-phase comprend l'implémentation, les tests spécifiques à cette étape, et la documentation associée. Une stratégie de test dédiée est déployée pour chaque sous-phase afin de garantir la conformité aux objectifs fixés dans le cahier des charges et de valider le passage à l'étape suivante.

Chaque phase vise à accomplir intégralement ou partiellement les objectifs définis dans le cahier des charges, avec une validation basée sur les résultats des tests de phase.

Aucun budget ne sera défini dans ce projet, cependant, tout coût accessoire engagé (appel API, etc..) sera mentionné.

PHASE 1 - CONCEPT

ARCHITECTURE

Base de données : SQL

Serveur WAMP : PHP 8.2 (min 8.1)

Utilisation du stack TALL :

- Tailwind CSS
- Alpine.js
- Laravel 10
- Livewire

Services tiers :

- Oh Dear
- Flare

IDE : PhpStorm

Gestionnaire de version : Github

Maquettes : WireFrame

BASE DE DONNÉE

Conception de la Base de Données : Modéliser les données à travers des schémas définissant les entités, leurs attributs et les relations.

//TODO

INTERFACE UTILISATEUR

Prototypes de l'interface utilisateur en mode desktop, le responsive et les vue tablettes / mobiles seront fait directement lors de l'implémentation du projet et des tests de validation concernant ce point sont inclus dans les phases d'implémentation et feront l'objet d'une attention particulière tout au long du projet.

Page login : interface de connexion

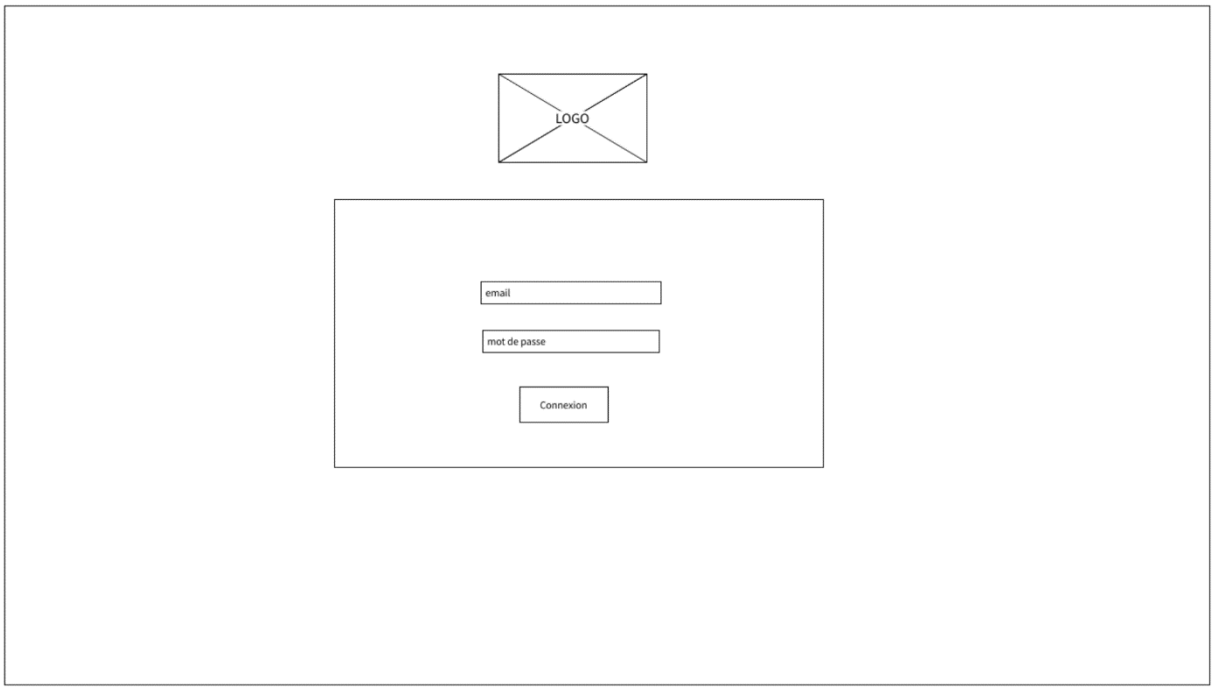

Figure 1Maquette authentification

Page vue d'ensemble : permet d'afficher les projets et voir leur santé en 1 coup d'œil.

Site	Performance OhDear	Code erreur flare	type d'erreur
FirstPoint.ch	USA	Apple Inc , Microsoft	Egregium esse egregium amicitia amicitia parem colebat numquam is anteibat virum per numquam se inferioris.
IKEA	Sweden	IKEA Furnitures , Spotify	
Spotify	Finland	Nokia Communications	Egregium esse egregium amicitia amicitia parem colebat numquam is anteibat virum per numquam se inferioris.
FirstPoint.ch	USA	Apple Inc , Microsoft	
IKEA	Sweden	IKEA Furnitures , Spotify	Egregium esse egregium amicitia amicitia parem colebat numquam is anteibat virum per numquam se inferioris.
IKEA	Sweden	IKEA Furnitures , Spotify	
Spotify	Finland	Nokia Communications	Egregium esse egregium amicitia amicitia parem colebat numquam is anteibat virum per numquam se inferioris.
Spotify	Finland	Nokia Communications	
FirstPoint.ch	USA	Apple Inc , Microsoft	Egregium esse egregium amicitia amicitia parem colebat numquam is anteibat virum per numquam se inferioris.
IKEA	Sweden	IKEA Furnitures , Spotify	

Figure 2 Maquette vue d'ensemble

Page détail : Affichage de données détaillées pour un projet / site.

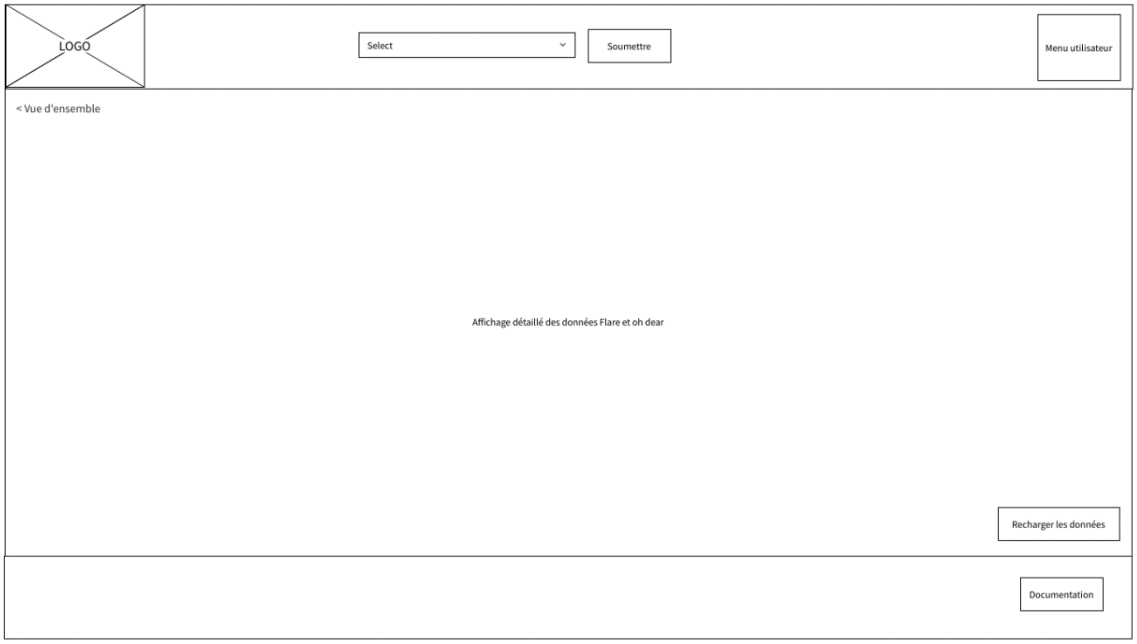

Figure 3 Maquette détails

Animations / effets :

- Boutons : au survol changement de couleur et de curseur
- Boutons : au clic changement de couleur
- ..

Composant livewire :

- Sélecteur de sites/projets et liste déroulante
- ..

PLANIFICATION FINALE

Planification : Etablir la planification détaillée et finale du projet

//TODO

PHASE 2 - CONCEPT

PHASE 2.1 - INSTALLATION ET CONFIGURATION INITIALE DE LARAVEL

Implémentation :

- Configurer l'environnement avec PHP 8.2
- Utiliser Composer pour installer Laravel 10.x (installer composer si besoin)
- Configurer l'environnement (.env) pour la base de données, etc.
- Installer Tailwind CSS via npm et configurer selon les besoins du projet.

Tests :

- S'assurer que l'environnement est correctement configuré pour PHP 8.2.
- Vérifier l'installation de Laravel avec (**php artisan serve**)
- Compiler les assets et vérifier que le css est appliqué (Tailwind CSS)

Documentation : documenter étape par étape sur l'installation de Laravel 10.x, incluant la configuration du fichier .env pour la base de données et d'autres services.

PHASE 2.2 - IMPLÉMENTATION DE LA BASE DE DONNÉES

Implémentation :

- Utiliser les migrations de Laravel pour reproduire le schéma défini dans la phase de conception
- Utiliser un typage fort

Tests :

- Vérifier que les migrations s'effectuent sans erreurs
- Insérer des données tests dans chaque table et lire les données pour tester l'intégrité

Documentation : Documenter le schéma de base de données, en mettant en évidence l'utilisation des types de données de PHP 8.2

PHASE 2.3 - INTÉGRATION DE L'AUTHENTIFICATION

Implémentation :

- Ajouter un système d'authentification (Breeze)
- Modifier les routes Laravel pour n'afficher les données qu'aux utilisateurs authentifiés, qui doivent être redirigé sur la page « vue d'ensemble »

Tests :

- Création d'utilisateurs à l'aide des seeders Laravel authentification
- Création d'utilisateurs à l'aide de php thinker et authentification
- Accéder directement au dashboard sans être authentifié

Documentation : Documenter le système d'authentification (manuel d'utilisation), notamment pour la création d'un utilisateur via php thinker

PHASE 2.4 - DÉVELOPPEMENT DE L'INTERFACE UTILISATEUR – LAYOUT DE BASE

Implémentation :

- Implémenter le layout de base selon les maquettes établies lors de la phase d'analyse
- Utiliser Tailwind CSS pour le style et le responsive

Tests :

- Desktop : Vérifier que le layout s'affiche correctement sur des écrans de bureau de différentes résolutions.
- Tablettes : Tester l'affichage et l'interaction sur des tablettes dans les orientations portrait et paysage.
- Smartphones : S'assurer que le design est responsif sur différents smartphones, en vérifiant en particulier les tailles d'écran les plus courantes.
- Taille et Accessibilité : Confirmer que tous les boutons, liens et éléments interactifs sont facilement cliquables sur mobile, avec une zone de clic suffisante.

Documentation : Documenter les composants développés.

PHASE 2.5 - INTÉGRATION DU SERVICES OH DEAR

Implémentation :

- Utiliser le service container de laravel pour définir un service client personnalisé pour l'API.
- Utiliser des bloc try/catch pour gérer les exceptions
- Intégrer et configurer quelques sites pour le monitoring dans le fichier config

Tests :

- Réaliser des tests d'intégration spécifiques à Oh Dear pour assurer une intégration fiable.
- Vérifier que les données demandées sont bien retournées
- Gestion correcte des erreurs et l'intégration des données dans la base de données.

Documentation : Documenter le processus d'intégration de l'API d'Oh Dear, y compris la configuration et la gestion des erreurs.

PHASE 2.6 - INTÉGRATION DU SERVICE FLARE

Implémentation :

- Utiliser le service container de laravel pour définir un service client personnalisé pour l'API.
- Utiliser des bloc try/catch pour gérer les exceptions
- Intégrer et configurer quelques sites pour le monitoring dans le fichier config

Tests :

- Réaliser des tests d'intégration spécifiques à Flare pour assurer une intégration fiable.
- Vérifier que les données demandées sont bien retournées
- Gestion correcte des erreurs et l'intégration des données dans la base de données.

Documentation : Documenter le processus d'intégration de l'API de Flare, y compris la configuration et la gestion des erreurs.

PHASE 2.7 - INTÉGRATION DE LIVEWIRE

Implémentation :

- Installer Livewire via Composer
- Création de composants et intégration.

Tests :

- Tester les interactions dynamiques pour vérifier leur bon fonctionnement et leur réactivité sur différents navigateurs et appareils.

Documentation : Documenter l'utilisation de Livewire dans le projet, y compris les composants créés, pour faciliter leur réutilisation et maintenance.

PHASE 2.8 – FINALISATION DE L'INTERFACE UTILISATEUR – INTÉGRATION SERVICES TIERS

Implémentation :

- Finaliser le développement de l'interface utilisateur en se basant sur les designs préalablement établis, en accordant une attention particulière aux sections affichant les données provenant des services tiers

Tests :

- Desktop : Vérifier que le tout s'affiche correctement sur des écrans de bureau de différentes résolutions.
- Tablettes : Tester l'affichage et l'interaction sur des tablettes dans les orientations portrait et paysage.
- Smartphones : S'assurer que le design est responsif sur différents smartphones, en vérifiant en particulier les tailles d'écran les plus courantes.
- Taille et Accessibilité : confirmer que tous les boutons, liens, et éléments interactifs sont facilement cliquables sur mobile, avec une zone de clic suffisante.
- Assurer la lisibilité des données de Oh Dear et Flare.
- Confirmer la conformité de l'UI aux standards UX.

Documentation : Mettre en évidence les principes d'UX/UI appliqués pour faciliter la navigation intuitive et présenter efficacement les informations critiques aux utilisateurs.

PHASE 2.8 - MISE EN PLACE DES TÂCHES CRON ET FONCTIONNALITÉ DE RAFRAICHISSEMENT

Implémentation :

- Utiliser la planification de tâches de Laravel (**kernel.php**) pour définir des tâches cron
- Programmer des tâches pour l'exécution régulière de la mise à jour des données depuis les API
- S'assurer que les tâches sont réparties de manière à éviter une surcharge du serveur.

Tests :

- Déclencher manuellement les tâches cron via cli (**php artisan shedule:run**)
- Utiliser Laravel Telescope pour surveiller l'impact des tâches cron sur les performances de l'application.

Documentation : Description des tâches et intervalle de temps recommandé.

PHASE 2.9 - CORRECTIONS ET REFACTORISATION

Implémentation :

- Réviser le code existant pour identifier les éventuelles optimisations / améliorations
- Voir les erreurs qui remonte via l'IDE

Tests :

- Réaliser des tests d'intégration pour s'assurer que les composants fonctionnent toujours
- Tester toutes les fonctionnalités et liens présent sur le site

Documentation : Mettre à jour la documentation pour refléter les changements réalisés pendant cette phase, en se concentrant sur les améliorations apportées. Partager les leçons apprises lors du processus, ce qui a fonctionné ou pas.

PHASE 2.10 - DOCUMENTATION UTILISATEUR

Implémentation :

- Finaliser toute la documentation utilisateur nécessaire pour le projet en vue du déploiement
- Inclure des captures d'écran lorsque c'est utile.

Tests :

- Vérifier la clarté de la documentation, de la structure globale et de l'horthographe.
- Ouvrir les liens pour voir s'ils pointent au bon endroit

Documentation : S'assurer que toute la documentation est bien organisée, facile à naviguer, et accessible aux développeurs et aux utilisateurs finaux.

PHASE 2.11 – DÉPLOIEMENT DE LA DOCUMENTATION

Implémentation :

- Préparer la documentation pour le déploiement, en s'assurant que l'environnement de production est correctement configuré.

Tests :

- Ouvrir le lien depuis l'application pour voir s'il pointe au bon endroit

Documentation : Documenter le processus de déploiement, y compris les étapes spécifiques, les configurations d'environnement, et les procédures de rollback en cas de problème.

PHASE 2.12 - CORRECTIONS ET REFACTORISATION

Implémentation :

- Répondre aux problèmes découverts pendant l'utilisation.
- Ajustements, corrections de bugs, ou ajout de petite fonctionnalité afin de couvrir les objectifs du cahier des charges.

Tests :

- Tester les ajustements et les nouvelles fonctionnalités pour s'assurer qu'ils sont conformes au comportement voulus et n'introduisent pas de nouveaux problèmes.

Documentation : Mettre à jour la documentation pour inclure les changements et les nouvelles fonctionnalités développées durant cette phase.

PHASE 2.13 - EVALUATION ET CLÔTURE DU PROJET

Évaluation : Réaliser une évaluation complète du projet pour s'assurer que tous les objectifs ont été atteints et que le produit final répond aux exigences de qualité, de performance, et d'usabilité.

Documentation : Compléter un rapport final qui résume les réalisations du projet, les leçons apprises, et les recommandations pour les projets futurs. S'assurer que toute la documentation est finalisée et bien organisée.

RISQUES TECHNIQUES

Les principaux risques techniques du projet incluent la complexité inhérente à la conception et au développement de certains tâches, un manque de certaines compétences, ainsi que l'incertitude quant au comportement de l'application en conditions réelles, puisqu'il n'est pas prévu de la déployer dans un environnement de production.

De plus, l'absence de tests unitaires, due à un manque d'expérience et au temps limité, augmente le risque d'erreurs non détectées lors du développement.

Le nombre de tâches inhérente au projet étant élevée, toutes ne pourront pas être réalisée selon les bonnes pratiques du développement.

Priorisation des tâches : Concentration sur les fonctionnalités clés et les composants critiques de l'application pour optimiser l'utilisation des ressources disponibles et minimiser les impacts des risques.

Actions préventives : Mise en place d'une revue régulière du code et d'analyses pour compenser l'absence de tests unitaires. Cela permettra d'identifier et de corriger les vulnérabilités potentielles ou les erreurs de conception dès les premiers stades du développement.

Gestion par phases avec tests : La structuration du projet en phases distinctes, incluant des tests spécifiques avant la transition, contribue positivement à la qualité.

Simulation d'environnement de production : Utilisation d'outils de simulation pour évaluer le comportement de l'application dans un environnement qui se rapproche le plus possible de la production, afin d'identifier les problèmes de performance ou de compatibilité en utilisant un serveur PHP local et intégré à Laravel.

RÉALISATION

GLOSSAIRE

Laravel

TailwindCSS

Alpine.js

Livewire

Oh Dear

Flare

Composer

NPM

MVC

API

GitHub

PHP

SQL

Cron

Dropdown

TABLE DES ILLUSTRATIONS

Figure 1Maquette authentification 6
Figure 2 Maquette vue d'ensemble 6
Figure 3 Maquette détails 7

SOURCE & BIBLIOGRAPHIE

- Wireframe : Edition de maquettes