**Présentation personnelle**

Mon intérêt pour l’informatique a débuté dès mon adolescence, que ce soit en stage de première lors de la création d'un logiciel avec Access, ou encore à travers mes cours d'informatique en BTS d’assistante de gestion où j’ai appris la modélisation de tables de données, entre autres. Cependant, on m’a souvent dit que je pouvais, parfois, ne pas avoir assez de patience pour ce domaine. Manquant de confiance en moi, ces remarques m’ont fait prendre d’autres chemins. J’ai alors exploré d'autres univers tout aussi intéressants et variés. Ces expériences m'ont permis de développer une confiance en moi et une persévérance qui, aujourd'hui, me permettent de me reconvertir avec légitimité dans ce domaine. Pour cela, j’ai suivi une formation de découverte du numérique puis une formation où j’ai acquis les bases de la programmation ce qui m’a permis d’intégrer sereinement le cursus actuel d’Elan Formation.

**Cahier des charges**

Présentation du projet

Je me suis rendue compte que de nombreux conflits restent dans l'ombre, souvent parce qu'ils sont jugés peu historiques ou peu glorieux pour certains pays. Parfois, certains événements sont même passés sous silence dans les cours d'histoire, alors que certains combats méritent pourtant d'être connus. Bien que je ne sois pas spécialiste en Histoire, j'ai souhaité partager quelques-unes de mes recherches, même si mon site n'a pas vocation à voir le jour.

Regards de guerre est une association fictive dédiée à l’organisation d’expositions sur le thème de la guerre. Chaque exposition met en lumière un conflit particulier, en utilisant le pouvoir de l'art pour susciter l'intérêt et la réflexion.

Organisation du site

Header :

L'en-tête, situé en haut de chaque page, joue un rôle crucial dans l'expérience utilisateur et l'identité visuelle du site. Il doit être à la fois fonctionnel, esthétique et adapté à tous les appareils.

Logo et liens de connexion/inscription : Placés à gauche on les trouve aisément.

Menu de navigation : Clair et intuitif permet aux utilisateurs d'accéder aux pages principales du site (accueil, expositions, tickets, blog, panier).

Pages :

* Accueil : Permet de connaitre les 3 prochaines expositions grâce à un carrousel, de connaitre l’association (local, café et magasin), d’avoir accès à l’agenda de toutes les expositions, de présenter l’accessibilité au lieu d’exposition.
* Exposition : Permet de connaitre le détail de l’exposition (sujet, artistes/type d’art, salle où sera représenté l’artiste)
* Ticket : Permet de connaitre les tickets disponibles pour chaque exposition et de les réserver.
* Panier : Permet de réserver l’ensemble des tickets.
* Back-office : Permet aux administrateurs de gérer l’ensemble du site à un seul endroit et sans avoir besoin de passer par le développeur (gestion des utilisateurs, expositions, artistes, tickets, stock)
* Blog : Permet aux utilisateurs de partager leur expérience et leur ressenti en écrivant des commentaire sur la page de l’exposition.

Footer :

Le pied de page, situé en bas de chaque page, regroupe les informations importantes et les liens utiles. Il contribue à la crédibilité et à la transparence du site.

Formulaire d'inscription à la newsletter : Un formulaire incitera les visiteurs à s'abonner pour recevoir les actualités de l'association.

Liens vers les réseaux sociaux : Des icônes discrètes mènent aux pages des réseaux sociaux de l'association.

Informations de contact : Un lien permet d’envoyer directement un mail à un administrateur.

Liens vers les pages légales : Des liens mènent aux mentions légales et à la politique de confidentialité.

Copyright : La mention du type d’association sera affichée.

Champs d’action

Les champs d’action de l’utilisateur en tant que :

* **Visiteur** : Un visiteur est un utilisateur non connecté qui peut naviguer librement sur le site et consulter les informations mises à disposition.

Consultation des pages publiques :

* Accueil : Présentation du site, agenda des expositions, plan d'accès et coordonnées.
* Blog : Accès aux articles des expositions passées et lecture des commentaires associés.
* Tickets : Informations sur les billets disponibles et les conditions de réservation.

Ajout de tickets dans le panier en vue d'une réservation future.

* **Utilisateur connecté** : Un utilisateur connecté est un utilisateur authentifié qui dispose de fonctionnalités supplémentaires lui permettant d'interagir avec le site et de personnaliser son expérience.
* Effectuer une réservation :
* Finalisation d'un achat de tickets : Possibilité d’ajouter, modifier ou de supprimer des tickets dans son panier avant de finaliser sa commande. Confirmation effectuée par email incluant le détail de la commande.
* Paramétrer son compte :
* Consulter son historique de commande : Accéder à la liste des réservations passées et aux détails de chaque commande.
* Modifier ses informations : Mettre à jour ses coordonnées ou son mot de passe.
* Supprimer son compte : Le membre peut demander la suppression de son compte et de ses données personnelles.
* Participer aux discussions : Partage d'opinions et échanges avec la communauté.

Les champs d’action de l’utilisateur en tant que :

* **Root** : Il dispose des droits les plus élevés. Ses actions se limitent à la gestion des autres administrateurs et à la configuration globale du système.

Ses responsabilités incluent :

* Attribution et modification des rôles.
* Suppression de comptes administrateurs.
* Gestion des permissions d'accès.
* Surveillance de l'activité des administrateurs (logs d'actions).
* **Administrateur** :
* Utilisateur :
* Profil utilisateur : Consultation, modification et suppression (à la demande ou si nécessaire).
* Historique des commandes : Consulter les réservations effectuées et envoie de facture sur demande.
* Artistes :
* Renseignement des données des artistes en vue de la création de la page d’exposition.
* Expositions :
* Création des pages complètes des nouvelles expositions (titre, description, prix, artistes, salle, upload des images)
* Tickets :
* Gestion des tickets : Ajouter, modifier ou supprimer des tickets (type, prix, nombre).
* Gestion des stocks : Mise en place d’un stock d’alerte.

* Blog :
* Expositions : Vérifier que les expositions passées soient transférées dans l’espace blog.
* Commentaires : Modérer les commentaires postés par les utilisateurs sur les articles ou les expositions (modification ou suppression si nécessaire)

MVP

L'ensemble des fonctionnalités présentées constitue le Minimum Viable Product (MVP), visant à démontrer la faisabilité d'une plateforme de gestion d'expositions et de réservations de tickets. Ce projet répond aux besoins essentiels des utilisateurs tout en garantissant une expérience utilisateur fluide et une gestion optimisée des utilisateurs, des expositions et des communications. Ce MVP pose les bases d'une plateforme évolutive, permettant d'ajouter de nouvelles fonctionnalités en fonction des besoins futurs des utilisateurs et des exigences du marché.

**Liste des compétences couvertes par REAC1**

* Développer la partie front-end d’une application web ou web mobile sécurisée
* Installer et configurer son environnement de travail en fonction du projet web ou web mobile
* Maquetter des interfaces utilisateur web ou web mobile
* Réaliser des interfaces utilisateur statiques web ou web mobile
* Développer la partie dynamique des interfaces utilisateur web ou web mobile
* Développer la partie back-end d’une application web ou web mobile sécurisée
* Mettre en place une base de données relationnelle
* Développer des composants d’accès aux données SQL et NoSQL
* Développer des composants métier coté serveur
* Documenter le déploiement d’une application dynamique web ou web mobile

Gestion de projet

La gestion de projet moderne exige des outils flexibles et des méthodologies agiles. Dans ce contexte, Trello, la méthode MoSCoW et le Kanban, au sein d'une approche agile, constituent une combinaison puissante pour organiser, prioriser et suivre l'avancement du projet de manière efficace.

Méthode MoSCoW : Priorisation des exigences

La méthode MoSCoW est utilisée pour prioriser les exigences du projet en fonction de leur importance. Elle classe les exigences en quatre catégories :

Must have (Doit avoir) : Les exigences indispensables pour que le projet soit considéré comme un succès.

Should have (Devrait avoir) : Les exigences importantes, mais qui peuvent être reportées si nécessaire.

Could have (Pourrait avoir) : Les exigences souhaitables, mais qui ont une priorité plus faible.

Won't have (N'aura pas) : Les exigences qui ne seront pas incluses dans cette version du projet.

Cette méthode permet de se concentrer sur les fonctionnalités essentielles pour la réalisation d'un MVP viable.

Kanban : Un flux de travail continu

Le Kanban est une méthode de gestion de flux de travail qui vise à optimiser la productivité en limitant le travail en cours. Il repose sur un tableau visuel qui représente les différentes étapes du processus de travail ("À faire", "En cours", "Terminé"). Les tâches sont déplacées d'une étape à l'autre, ce qui permet de visualiser l'avancement du projet et d'identifier les obstacles. Le Kanban encourage l'amélioration continue en permettant à l'équipe de s'adapter aux changements et d'optimiser son flux de travail.

Trello : Un tableau de bord visuel

L'intégration de ces méthodes est facilitée par Trello. un tableau de bord centralisé offrant une vue d'ensemble claire de l'état d'avancement. Son interface intuitive, basée sur des cartes et des listes, permet de visualiser les tâches et les échéances. Chaque carte représente une tâche, classée selon la méthode MoSCoW, et peut être déplacée entre les listes représentant les étapes du projet (Kanban).

Méthodologie Agile

L'approche agile a permis de structurer mon travail et de maintenir une organisation rigoureuse, même en travaillant seul. Elle a favorisé l'adaptation aux changements et aux imprévus, la concentration sur les tâches prioritaires et la limitation de la dispersion. Les itérations courtes et les objectifs réalisables ont contribué à maintenir la motivation et à visualiser l'avancement du projet. Au quotidien, une liste de tâches était établie, réajustée en fin de journée ou servant de base à la planification du lendemain.

Intégration et avantages :

Cette combinaison d'outils et de méthodes a permis de :

Visualiser l'avancement du projet en temps réel.

Identifier et résoudre les problèmes rapidement.

S'adapter aux changements et aux imprévus.

Réaliser le MVP dans les délais impartis.

Liste des compétences couvertes par REAC1

* Développer la partie front-end d’une application web ou web mobile sécurisée

Installer et configurer son environnement de travail en fonction du projet web ou web mobile

Maquetter des interfaces utilisateur web ou web mobile

Réaliser des interfaces utilisateur statiques web ou web mobile

Développer la partie dynamique des interfaces utilisateur web ou web mobile

* Développer la partie back-end d’une application web ou web mobile sécurisée

Mettre en place une base de données relationnelle

Développer des composants d’accès aux données SQL et NoSQL

Développer des composants métier coté serveur

Documenter le déploiement d’une application dynamique web ou web mobile

**Langages**

Côté client :

* HTML (HyperText Markup Language) : Langage de balisage utilisé pour structurer et organiser le contenu des pages web. Il définit la signification et la structure des éléments.
* CSS (Cascading Style Sheets) : Langage de style utilisé pour définir la présentation et l'apparence des pages web. Il permet de contrôler la mise en page, les couleurs, les polices, etc.
* JS (JavaScript) : Langage de programmation permettant d'ajouter de l'interactivité et des fonctionnalités dynamiques aux pages web (carrousel, menu burger).

Côté serveur :

* PHP (Hypertext Preprocessor) : Langage de programmation principal qui gère la logique métier, la communication avec la base de données, traiter des formulaires, personnaliser les pages web et la génération de contenu dynamique.
* SQL (Structured Query Language) : Langage de requête utilisé pour interagir avec des bases de données relationnelles. Il permet de récupérer, de modifier et de gérer des données stockées dans des tables.

**Technologies :**

Symfony :

Framework PHP doté d'une architecture Modèle-Vue-Contrôleur (MVC) qui favorise une organisation claire du code, facilitant ainsi la séparation des responsabilités et la maintenance. Ses composants réutilisables (formulaires, routes, sécurité) et sa haute configurabilité permettent de l'adapter précisément aux besoins de chaque projet.

Doctrine :

Object Relationnal Mapping (ORM) qui établit une passerelle entre PHP et la base de données relationnelle MySQL. Son approche orientée objet permet de manipuler les données avec DQL, un langage de requête intuitif, évitant ainsi l'écriture de SQL complexe. Les migrations facilitent la gestion des schémas de base de données, assurant la cohérence et la traçabilité des changements.

Twig :

Moteur de templates de Symfony favorise la séparation entre la logique de présentation et la logique métier, améliorant ainsi la lisibilité et la maintenabilité du code. L'héritage de templates permet de créer des mises en page réutilisables, réduisant la duplication de code et assurant une cohérence visuelle. De plus, il intègre des fonctionnalités de sécurité robustes pour protéger les applications web contre les attaques courantes.

**Environnement**

VS Code (Visual Studio Code) :

VS Code est l'éditeur de code de choix pour ce projet. Sa légèreté, sa performance et sa vaste gamme d'extensions en font un outil indispensable.

Exemple d’extensions utilisées :

* Prettier : automatise le formattage du code
* PHP intelephense : autocomplétion et verification de refactoring
* Twig Pack : Templates réutilisables, Fonctionnalités avancées (héritage de templates, fonctions (ex : path())

Son terminal intégré permet d'exécuter les commandes Symfony et Composer directement depuis l'éditeur.

Laragon :

C’est un environnement de développement local portable, isolé, rapide et facile à utiliser pour Windows. Il simplifie considérablement la configuration d'un serveur web, d'une base de données.

HeidiSQL :

C’est un outil de gestion de base de données qui, grâce à son interface graphique intuitive et à ses fonctionnalités complètes, facilite grandement la manipulation et la gestion des données. Il permet de visualiser et de modifier les données, de créer et de modifier les tables, d'exécuter et de tester des requêtes SQL avant de les adapter pour Doctrine Query Language (DQL).

Github Desktop et GitHub :

Github desktop, grâce à son interface intuitive, facilite le suivi des modifications de code, la gestion des branches et la résolution des conflits.

GitHub, quant à lui, est la plateforme de contrôle de version et de collaboration utilisée pour ce projet et permet à mes formateurs de suivre l’avancement de mon projet en temps réel.

Composer :

Gestionnaire de dépendances qui simplifie l'installation et la mise à jour des bibliothèques et des composants nécessaires à une application PHP, assurant la cohérence des versions et évitant les conflits. Il utilise un fichier « composer.json » pour définir les dépendances du projet (doctrine, bundles (ex : la sécurité), packages (ex : les formulaires), bibliothèques (ex : Symfony mailer).

Bundle :

Ensemble structuré de fichiers PHP, de configurations, de routes, de contrôleurs, de templates et d'autres ressources qui fournissent une fonctionnalité spécifique. (exs : SecurityBundle, SymfonyCastsResetPasswordBundle, domPdf).