17 NOVEMBRE 2024

TP - DÉVELOPPEUR WEB ET WEB MOBILE

ARCADIA

Belhadi

MARWANE

ZOO ARCADIA

HTTPS://ARCADIA-PAIMPONT-AD54852A65E5.HEROKUAPP.COM/ACCEUIL.PHP

DÉBUT DU PROJET ARCADIA

Au début du projet Arcadia j'ai commencer par prendre connaissance du projet dans sa globalité.

COMPRENDRE LES DEMANDE DU CLIENT:

José veut montrer une certaine image de sont zoo via différentes demandes:

1. **Thème écologique** : l'application doit refléter les «écologique du zoo avec un design qui inspire la nature et la durabilité. Les couleurs et le thème doivent évoquer une forte conscience environnementale.

2. **Visualisation des animaux et habitats** : José veux que les visiteurs puissent consulter les différents animaux du zoo ainsi que leurs états de santé, en lien avec les habitats (savane, jungle, marais).

3. **Présentation des services et horaires** : l'application doit permettre aux utilisateurs de découvrir les services proposés par le zoo (restauration, visites guidées, petit train), ainsi que les horaire d'ouverture.

4. **Navigation intuitive** : l'interface utilisateur doit être simple et fluide, avec un menu d'accès facile pour naviguer entre les différentes sections.

Le projet et divisé en plusieurs User Stories:

1. **Page d'accueil (US 1) :** Présentation du zoo avec des images, description des habitats et des animaux, ainsi qu'une section d'avis des visiteurs.

2. **Menu de l'application (US 2) :** Un menu permettant de naviguer vers la page d'accueil, les services, les habitats, la connexion (pour le personnel du zoo), et la page de contact.

3. **Vue globale des services (US 3) :** Une page récapitulative de tous les services proposés, modifiable par l'administrateur via un espace dédié.

4. **Vue globale des habitats (US 4) :** Une page listant les différents habitats (savane, jungle, marais) avec un lien vers une description plus détaillée incluant les animaux présents dans chaque habitat. Chaque animal possède une fiche avec son prénom, sa race, son état de santé (mis à jour par les vétérinaires) et des informations complémentaires.

5. **Avis des visiteurs (US 5) :** Les visiteurs peuvent laisser des avis après validation par un employé via son espace dédié.

6. **Espace administrateur (US 6) :** L'administrateur peut créer des comptes pour les employés et vétérinaires, ainsi que gérer les services, horaires, habitats et animaux. Un tableau de bord permet de visualiser les consultations par animal.

7. **Espace employé (US 7) :** Les employés peuvent valider des avis, modifier les services, et renseigner la quantité de nourriture donnée aux animaux.

8. **Espace vétérinaire (US 8) :** Les vétérinaires saisissent des comptes rendus sur l'état de santé des animaux et peuvent laisser des commentaires sur les habitats.

9. **Connexion (US 9) :** Seuls les administrateurs, vétérinaires et employés peuvent se connecter. Aucun visiteur ne peut créer de compte.

10. **Contact (US 10) :** Un formulaire de contact est disponible pour les visiteurs, avec un système de réponse par e-mail.

11. **Statistiques des consultations (US 11) :** Les consultations d'animaux par les visiteurs sont comptabilisées et affichées dans un tableau de bord pour l'administrateur.

ORGANISATION VIA UN OUTIL KANBAN

En me basant sur les demandes du client et les User Stories (US) du projet Arcadia, j'ai mis en place un tableau sur Trello pour organiser et gérer les différentes étapes du développement. Ce tableau me permet de structurer le projet en plusieurs phases, tout en suivant une méthodologie Kanban. Chaque tâche ou fonctionnalité est représentée par une carte, et les colonnes indiquent les différentes étapes, telles que "À faire", "En cours", "En test", et "Terminé". De plus, j'ai défini des deadlines pour chaque tâche afin de respecter la date butoir du 21 novembre 2024, en fonction des priorités que je me fixerai au fur et à mesure. J'ai commencé le projet le 9 octobre 2024, ce qui me laisse une

période de travail suffisamment longue pour atteindre les objectifs en respectant les exigences du client.

N'ayant pas eu l'habitude de gérer un Kanban je ne l'ai pas réellement tenu a jours. Vu que je travaillais seul sur le projet je savais toujours ou j'en étais.

CHARTE GRAPHIQUE

Le client a des demande spécifique sur ce à quoi son site doit ressembler avec une charte graphique claire : **l'écologie et le respect de l'environnement.**

J'ai donc d'abord commencer par me faire un moodbord pour avoir de l'inspiration et trouver des code couleur et un police d'écriture qui respecterais le souhait du client.

Moodbord

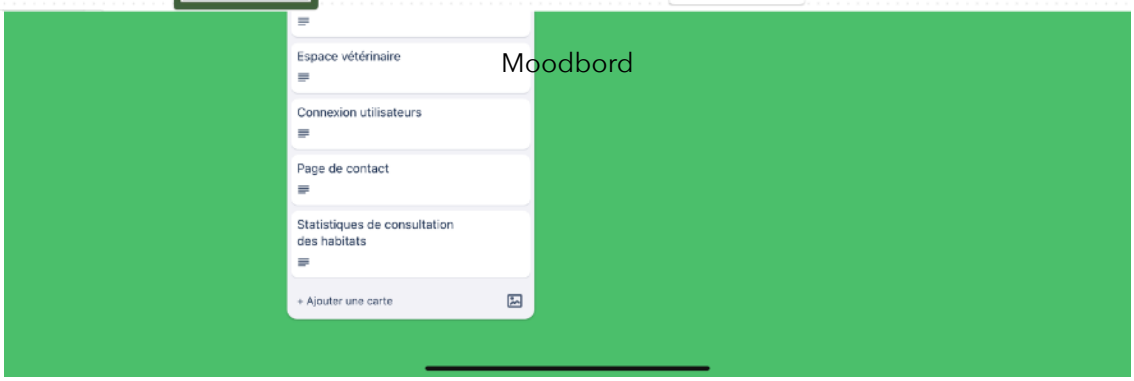

Tableau Trello

Le moodbord me donne une bonne direction pour la charte graphique avec un choix de couleur plutôt évident. J'essaye pour le projet et pour ne pas faire une surcharge cognitive de prendre une couleur primaire, secondaire et tertiaire qui donneront le thème général de l'application.

PREMIÈRE MAQUETTE FIGMA

Premier rendu avec les couleur choisies

CHOIX DES COULEURS ET POLICE

Pour les couleur j'ai décidé de prendre des couleurs qui rappelle le Zoo et l'écologie:

Couleur primaire : #1A4D2E

Couleur secondaire: #E8DFCA

Couleur tertiaire: #4F6F52

Couleur bouton: #E9A300

Pour la police d'écriture j'ai pris une font google la « Signika Negative » car je la trouve jolie.

LIEN FIGMA ET MAQUETTE

https://www.figma.com/design/7LIz0lRLY1mSyZX61CjtDX/Untitled?t=8bhAAgTd5orzuToE-1

Ce site est le premier site que je réalise. Donc le rendu final et très different de la maquette.

J'ai voulu un style épuré et minimaliste sans surcharger les page du site en respectant la demande du client.
Présenté les information de manière claire tout en gardant une bonne lisibilité et intuitivité du site.

DÉVELOPPEMENT

- PHP: Pour ce projet, j'ai choisi PHP parce que c'est un langage qui permet de gérer facilement des sites avec beaucoup de pages. J'ai préféré me passer de Symfony, car même si c'est puissant, ça demande pas mal de maîtrise. Avec PHP, j'ai pu rendre les pages plus interconnectées grâce au require, ce qui simplifie la gestion des différentes sections et rend le développement plus fluide. PHP reste un langage

simple et efficace pour gérer le back-end, surtout dans un contexte de projet web classique.

- JavaScript: JavaScript a été essentiel pour tout ce qui est dynamique sur le site. Que ce soit pour les petites animations ou pour faire fonctionner certains éléments de Bootstrap, JavaScript m'a permis de donner un côté plus interactif à l'interface. Ça a été un vrai plus pour améliorer l'expérience utilisateur et rendre le site plus agréable à utiliser.

- Bootstrap: J'ai choisi Bootstrap pour gagner du temps et assurer une cohérence visuelle. Le framework permet d'utiliser des composants tout faits, ce qui accélère le développement, surtout quand on veut un site responsive qui s'adapte bien aux différents écrans. Ça m'a permis de me concentrer sur les fonctionnalités sans avoir à tout créer de zéro côté design.

- MySQL:Étant donné que je travaille en local avec MAMP, MySQL était le choix logique pour gérer la base de données. Il est facile à configurer et parfaitement adapté pour stocker toutes les informations du zoo, que ce soit les animaux, les habitats ou les utilisateurs. MySQL est un système de base de données fiable et bien documenté, donc c'était une bonne option pour ce projet.

LIEN VERS LE GITHUB

https://github.com/NoSoussaille/Arcadia

Sur le git je n'ai pas assez suivie les consigne car j'ai développé avec une seul branche je maitrise a peine git et j'ai beaucoup codé page par page avec des vérification en local constante.
Je pense que j'aurais utilisé des branches différentes si le sites avait été en ligne pour ne pas impacter le site.

LE DÉVELOPPEMENT

Le projet Arcadia a marqué ma première expérience de développement d'un site web complet. Ce projet m'a permis de découvrir la puissance de PHP, en particulier sa capacité à simplifier la gestion de fonctionnalités dynamiques et complexes. Cependant,

ce premier développement a également présenté de nombreux défis, notamment liés à l'arborescence des dossiers, à l'organisation générale du code, et au déploiement en ligne. Si j'avais utilisé un framework comme Symfony ou Laravel, ces aspects auraient probablement été plus simples à gérer.

A. DÉVELOPPEMENT DES PAGES UTILISATEUR

J'ai commencé par suivre ma maquette initiale pour concevoir la partie dédiée aux visiteurs du site.

Certaines pages n'étaient pas définies dans ma maquette initiale, car je ne savais pas encore comment les implémenter ou quelles fonctionnalités précises elles devraient intégrer. J'ai donc choisi de me concentrer d'abord sur les pages que je comprenais bien, en reportant les sections complexes pour plus tard.

Les pages utilisateur ont été développées en priorité, à l'exception des pages animaux, services et habitats, qui nécessitaient une meilleure compréhension des bases de données.

B. CONCEPTION ET DÉVELOPPEMENT DU TABLEAU DE BORD

Le développement du Dashboard a constitué une étape clé du projet. Contrairement à la partie utilisateur, je n'ai pas suivi de maquette pour le Dashboard, mais je me suis assuré de respecter la charte graphique générale du site.

Cette étape a été un véritable apprentissage de l'utilisation des bases de données MySQL. J'ai progressivement ajouté de nouvelles tables et relations pour répondre aux besoins émergents des fonctionnalités du tableau de bord.

Gestion des rôles :

Le Dashboard prend en charge trois types d'utilisateurs : l'administrateur, le vétérinaire et l'employé.

Chaque rôle dispose d'un accès spécifique et de fonctionnalités adaptées :

- L'administrateur peut gérer les animaux, habitats, services et employés.
- Le vétérinaire peut effectuer le suivi des animaux et consulter leurs antécédents.
- L'employé peut gérer les avis clients.

C. FINALISATION DES FONCTIONNALITÉS POUR LES VISITEURS

Après avoir terminé le Dashboard, je suis revenu à la partie utilisateur pour implémenter les pages manquantes (animaux, services et habitats).

J'ai intégré une gestion des horaires configurable par l'administrateur via le tableau de bord.

Chaque section a été conçue pour refléter la structure et les données définies dans la base de données.

DÉFIS RENCONTRÉS ET SOLUTIONS

Pendant le développement, j'ai rencontré plusieurs défis, notamment en ce qui concerne l'arborescence des fichiers. La gestion des chemins relatifs a été un casse-tête constant. Pour résoudre ce problème, j'ai décidé d'introduire une constante BASE_URL qui m'a permis d'uniformiser toutes les redirections et de mieux structurer les liens entre les fichiers.

En ce qui concerne la base de données, c'était une première pour moi, et je ne savais pas du tout par où commencer. J'ai dû apprendre progressivement, en ajoutant des tables

une par une en fonction des besoins spécifiques de mon projet. À mesure que j'avançais, j'ai établi des relations claires entre les tables, ce qui m'a permis de simplifier mes requêtes et d'améliorer la gestion des données.

Enfin, la sécurisation des sessions a été un autre défi important. Je voulais m'assurer que les connexions des utilisateurs étaient sécurisées. Pour cela, j'ai mis en place des vérifications de l'adresse IP et du user agent de chaque utilisateur afin de détecter d'éventuelles anomalies. J'ai également implémenté un système de régénération des sessions pour limiter les risques de vol de session et garantir une meilleure sécurité.

DÉPLOIEMENT

Le déploiement du projet a été une étape particulièrement difficile, surtout parce que c'était la première fois que je m'y confrontais. Pour commencer, le choix de l'hébergeur a été un vrai casse-tête. J'ai exploré plusieurs options, comme Fly.io ou encore l'utilisation d'un serveur local avec Ngrok, avant de finalement opter pour Heroku, qui m'a semblé être la solution la plus adaptée pour ce projet.

Ensuite, j'ai dû apprendre à configurer les variables d'environnement, ce qui n'était pas évident. Par exemple, il a fallu définir les informations de connexion à la base de données tout en m'assurant qu'elles étaient sécurisées. Pour cela, j'ai utilisé les commandes heroku config:set pour configurer les variables nécessaires et éviter les erreurs.

La migration de la base de données a également été un vrai défi. Exporter ma base locale pour l'importer dans un environnement distant comme JawsDB MySQL n'était pas intuitif. J'ai fini par utiliser mysqldump pour exporter les données et mysql pour les importer sur la base distante, ce qui a permis de résoudre le problème.

Un autre obstacle concernait les chemins relatifs, notamment dans le tableau de bord. De nombreuses redirections et accès à des fichiers ne fonctionnaient pas correctement après le déploiement. J'ai dû revoir tous les liens pour les rendre cohérents avec la structure imposée par Heroku.

Enfin, j'ai également rencontré des erreurs comme des 404 Not Found ou des 500 Internal Server Error. Pour les résoudre, j'ai passé beaucoup de temps à analyser les logs Heroku (heroku logs --tail) afin d'identifier les problèmes et d'ajuster les fichiers concernés. Cela m'a permis d'affiner mon projet et de le rendre fonctionnel.