Centre Européen de Formation

Dossier de Projet

Présentation de l'entreprise fictive

Digital Symbiosis

Join the digital revolution

Tiago Machado

Nº d'élève F471277A

Mai 2024

Index

1. Présentation et sommaire de l'entreprise fictive « Digital Symbiosis »

Ce portfolio est une démonstration pratique des compétences acquises au cours du parcours académique au CEF. Il comprend une variété de travaux qui utilisent les technologies *front-end* pour créer des solutions web complètes et fonctionnelles.

Pour ce projet, qui avait comme objectif la création d'un portfolio avec *VUE.JS*, l'entreprise fictive Digital Symbiosis a été imaginée.

Digital Symbiosis est une société fictive de développement web fondée par Tiago Machado. Notre objectif est de créer des solutions numériques innovantes, efficaces et personnalisées en utilisant les dernières technologies web. Notre mission est de transformer les idées en réalités numériques qui non seulement répondent aux attentes de nos clients, mais les dépassent.

2. Résumé du Projet

Ce projet consiste à créer un portfolio interactif développé à l'aide de *Vue.js, HTML, CSS et JavaScript*.

Un portfolio développé en Vue.js est un outil essentiel pour tout développeur Full Stack qui souhaite présenter ses compétences et ses projets de manière professionnelle et interactive. Le choix de Vue.js, ainsi que de HTML, CSS et JavaScript, permet la création d'une application moderne, réactive et facile à entretenir, soulignant la capacité du développeur à utiliser des technologies de pointe pour créer des solutions web efficaces.

L'objectif du portfolio est de démontrer les compétences et les projets d'un développeur Full Stack, en fournissant une plateforme visuellement attrayante et fonctionnelle pour afficher le travail effectué, les compétences techniques et l'expérience professionnelle.

3. Conception et codage des composant *front-end*

Digital Symbiosis est fier de son approche méticuleuse de la conception et du codage des composants front-end, garantissant une expérience utilisateur fluide et interactive. Nous détaillons ci-dessous notre processus et les technologies utilisées.

Conception des composants :

1. Analyse des besoins :
   - Identification des besoins et des fonctionnalités souhaitées.
   - Création de wireframes et de prototypes pour une première visualisation de l'interface.
   - Définition de composants réutilisables pour optimiser le développement.

2. Conception UI/UX :
   - Utilisation des principes de conception centrée sur l'utilisateur pour assurer une navigation intuitive.
   - Mise en œuvre d'un design réactif pour adapter l'interface à différents appareils.
   - Création d'une palette de couleurs et d'une typographie cohérente avec l'identité visuelle du projet.

Codage des composants :

1. Structuration avec *Vue.js* :

   - Componentisation : développement de composants Vue.js réutilisables, modularisation de l'interface pour faciliter la maintenance et l'évolutivité.
   - Réactivité : utilisation du système de réactivité de *Vue.js* pour mettre à jour automatiquement l'interface en réponse aux changements de données.

2. Stylisation avec CSS :

   - CSS modulaire : création de styles modulaires, permettant un code CSS organisé et réutilisable.
   - Responsive Design : utilisation de media queries et d'unités flexibles pour s'assurer que l'interface s'adapte à différentes tailles d'écran.
   - Animations et transitions : Ajout d'animations et de transitions douces pour améliorer l'expérience de l'utilisateur et donner une impression de fluidité.

3. Interactivité avec JavaScript :

   - Événements DOM : mise en œuvre de gestionnaires d'événements pour les interactions de l'utilisateur telles que les clics, les tapotements et les mouvements de la souris.

- Validations et retour d'information : codage des validations de formulaires en temps réel et retour d'information visuel pour garantir une expérience utilisateur solide.
- Intégrations API : appels à des API externes pour récupérer et envoyer des données, ce qui rend l'application dynamique et interactive.

4. Présentation des éléments les plus significatifs du projet

1. Utilisation du serveur JSON

- Description : Mise en œuvre d'un serveur JSON pour simuler une API RESTful.
- Fonctionnalité : Permet de stocker des données dans un fichier JSON et d'effectuer des opérations de manière dynamique.
- Avantages :
    I. Facilite le développement et le test des fonctionnalités front-end sans qu'il soit nécessaire de disposer d'un backend complet.
    II. Offre une solution légère et rapide pour la gestion des données pendant la phase de développement.

3. Importation dynamique de données

- Description : les données sont stockées dans un fichier JSON et importées dynamiquement en fonction des besoins.
- Fonctionnalité : permet à l'application de charger et de mettre à jour les données en temps réel, en reflétant instantanément les changements.

- Avantages :
    I. Améliore l'efficacité et la réactivité de l'application.
    II. Permet une plus grande flexibilité dans la manipulation et l'affichage des données.

3. Envoi d'e-mails avec SMTP JS et Elastic Email
    - Description : Intégration de la bibliothèque SMTP JS pour l'envoi d'emails via le service Elastic Email.
    - Fonctionnalité :
        I. SMTP JS : Fournit une interface simple pour envoyer des courriels directement depuis le navigateur.
        II. Elastic Email : Service de messagerie qui facilite l'envoi de courriels en masse et offre des analyses détaillées.
    - Avantages :
        I. Simplifie la configuration et l'envoi de courriels sans avoir besoin d'un serveur de messagerie interne.
        II. Offre une solution évolutive et fiable pour l'envoi de communications par courrier électronique.

5. Présentation de recherche effectuée

Recherche sur Vue.js à l'aide du tutoriel officiel

Pour comprendre et maîtriser Vue.js, j'ai effectué des recherches approfondies à l'aide du tutoriel officiel disponible sur le site Web vuejs.org. Ce tutoriel est une ressource complète qui fournit une introduction étape par étape au framework, des concepts de base aux fonctionnalités avancées.

https://vuejs.org/tutorial/

Recherche sur l'envoi d'e-mails avec SMTP.js

Lors du développement de mon projet de portfolio en Vue.js, le besoin s'est fait sentir d'implémenter une fonctionnalité permettant d'envoyer des courriels directement depuis l'application web. Après avoir soigneusement analysé les options disponibles, j'ai choisi d'utiliser la bibliothèque SMTP.js en raison de sa simplicité et de son efficacité. Cette décision s'est appuyée sur des recherches approfondies couvrant divers aspects techniques et pratiques.

https://smtpjs.com/

6. Repository GitHub et validateur W3C

L'intégralité de ce projet est disponible dans le dépôt *GitHub*, ainsi que ce document et les validations du *W3C*. Pour configurer le projet, suivez les étapes du fichier README à la racine du projet.

GitHub:

https://github.com/masterxamss/portfolio.git

Validation W3C:

https://github.com/masterxamss/portfolio/tree/main/public/validator%20w3c

7. Conclusion

Le développement de mon portefeuille en Vue.js m'a apporté une expérience pratique et enrichissante dans l'utilisation des technologies frontales modernes. En utilisant JSON Server, j'ai pu créer une API simulée permettant une manipulation dynamique des données stockées dans un fichier JSON. L'intégration avec la bibliothèque SMTP.js et le service Elastic Email pour l'envoi de courriels a démontré la capacité d'ajouter des fonctionnalités interactives et utiles à l'application. Ce projet a non seulement élargi mes compétences techniques, mais a également amélioré ma capacité à créer des applications web évolutives et réactives.