**Spécifications techniques**

MENU MAKER - QWENTA

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Auteur** | **Date** | **Approbation** |
| 1.0 | Maxence DELVINQUIER | 3 décembre 2024 | Soufiane |

[I. Choix technologiques 1](#_Toc184143904)

[II. Liens avec le back-end 5](#_Toc184143905)

[III. Préconisations concernant le domaine et l’hébergement 5](#_Toc184143906)

[IV. Accessibilité 7](#_Toc184143907)

[V. Recommandations en termes de sécurité 7](#_Toc184143908)

[VI. Maintenance du site et futures mises à jour 8](#_Toc184143909)

## Choix technologiques

* État des lieux des besoins fonctionnels et de leurs solutions techniques :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Besoin** | **Contraintes** | **Solution** | **Description de la solution** | **Justification (2 arguments)** |
| Landing Page | L’utilisateur doit être capable de comprendre l’utilité de l’application rapidement. | HTML – CSS (SASS) - REACT | HTML : structure sémantique pour optimiser l’accessibilité et le SEO.  CSS (SASS) : Mise en forme esthétique et réutilisation des styles grâce à des variables.  REACT : Composant réutilisables pour une expérience utilisateur fluide. | 1 – Performance élevées : React permet de construire des interfaces rapides et modulaires grâce à son DOM virtuel.  2 – Flexibilité : HTML, CSS (SASS) et React sont des standards largement adoptés, ce qui facilite le développement et la maintenance future. |
| Gestion des utilisateurs (connexion) | Authentification sécurisée via email/mot de passe et google | Firebase Auth | Fournit un service d’authentification prêt à l’emploi pour gérer les utilisateurs | 1 – Réduit le temps de développement.  2 – Facilité et rapidité pour l’utilisateur final de création de compte. |
| Connexion, ajout de catégorie de plat, ajout de menu, mentions légales. | Afficher des fenêtres modales pour des actions spécifiques sans recharger la page.  Garantir une bonne accessibilité et ergonomie. | React modal | React Modal affiche des fenêtre dynamiques et accessibles. Chaque modal s’ouvre et se ferme dynamiquement, offrant une expérience utilisateur fluide. | 1 – Accessibilité : Respect des normes ARIA pour une navigation fluide et accessible.  2 – Flexibilité : Intégration rapide et styles personnalisables. |
| Base de données pour les menus | Gérer une structure flexible pour les données des menus (catégories, plats, images, prix) | MongoDB | MongoDB est une base de données NoSQL qui stocke les informations sous forme de documents JSON. Cela facilite l’ajout ou la modification de champs sans altérer la base existante. | 1 – Flexibilité : Adaptée aux structures évolutives et non fixe, MongoDB permet une gestion fluide des menus personnalisables.  2 – Scalabilité : Conçue pour gérer des grandes quantités de données et des charges croissantes, MongoDB est idéal pour des applications modernes. |
| Modification du style des menus utilisateurs | Permettre aux utilisateurs de personnaliser leurs menus, couleurs, polices, ou autre styles du menu des plats.  Affichage en temps réel des modifications | React – styled-components | React gère la mise à jour en temps réel avec un système d’état.  Styled-components permet d’appliquer des styles dynamiques basés sur les choix utilisateurs afin de refléter les changements dans l’aperçu. | 1 – Aperçu en temps réel : React met à jour la vue directement grâce a son DOM virtuel  2 – Flexibilité des styles : styled-components génère des styles dynamiques en fonction des entrées utilisateurs. |
| Exportation du menu finalisé en PDF ou impression | Permettre aux utilisateurs de télécharger une copie PDF fidèle à l’affichage React. | React-pdf | React-pdf génère un PDF à partir des composants React eux-mêmes. Les données textes, styles, images, sont traduites en document PDF avec une précision élevée. | 1 – Fidélité du rendu : React-pdf permet une gestion précise des styles CSS  2 – Composabilité : Chaque partie du document PDF peut être construite comme un composant React indépendant (usage future branding) |
| Exportation pour instagram | Permettre aux utilisateurs d’exporter leur menu en format image en respectant les dimensions, ratio, format. | React – html-to-image | Html-to-image capture le menu visible en un fichier image. Le fichier est optimisé pour les dimensions et formats requis par instagram. | 1 – Optimisation pour instagram : le format d’export respecte les dimensions et exigences techniques d’Instagram. 2 – Simplicité d’intégration : html-to-image s’intègre rapidement dans un projet React. |

Firebase Auth : <https://firebase.google.com/docs/auth?hl=fr>

React Modal : <https://www.npmjs.com/package/react-modal>

MongoDB : <https://www.mongodb.com/products/platform/atlas-database>

Styled-components : <https://www.npmjs.com/package/styled-components>

React-pdf : <https://www.npmjs.com/package/react-pdf>

HTML-to-image : <https://www.npmjs.com/package/html-to-image>

## Liens avec le back-end

* + - 1. Langage pour le serveur

Node.js : Utilisé pour sa compatibilité avec JavaScript, simplifiant le partage entre le front-end et le back-end. Node.js est conçu pour des applications où la rapidité est clé, comme l’affichage en temps réel ou les interactions utilisateur fréquente, tel que les modifications de menus.

* + - 1. API nécessaire

API REST : développé avec Express.js peut être utilisé pour ajouter, mettre à jour, supprimer des menus et catégories.

Authentifier les utilisateurs via Firebase Auth,

Exporter les données utilisateurs comme les menus vers la base de données MongoDB.

* + - 1. Base de données choisie

MongoDB (NoSQL) : adapté pour les applications modernes nécessitant une grande souplesse dans la gestion des données, comme les menus personnalisables. La structure en JSON s’aligne parfaitement avec le choix des technologies du reste de l’application.

## Préconisations concernant le domaine et l’hébergement

* Nom du domaine.

Le nom de domaine sera très probablement un sous-domaine de Qwenta (en cours de validation) de type :

<https://menu-maker.qwenta.com>

Le TLD peut changer en fonction de l’ambition et du plan de développement de l’entreprise. Un .fr si la cible est principalement les restaurateurs francais. Cela serait bénéfique pour le SEO.

* Nom de l’hébergement.

L’hébergement est à réfléchir tout comme le nom de domaine pour cette application.

Coté back-end : AWS (grande scalabilité) ou Vercel (simple et efficace pour des projet front-end et full-stack légers).

Alternatif : O2Switch si le budget est une contrainte.

Coté front-end : Firebase Hostin gen cas d’utilisation de Firebase Auth.

Vercel, AWS ou O2switch fonctionneront très bien pour le front-end.

Base de données : Utilisation de MongoDB Atlas.

* Adresses e-mail.

Pour une communication professionnelle l’adresse email, il est recommandé d’utiliser un adresse email du même domaine, soit un exemple : [menu-maker@qwenta.com](mailto:menu-maker@qwenta.com), [contact@qwenta.com](mailto:contact@qwenta.com) , [support@qwenta.com](mailto:support@qwenta.com)

D’autre services comme Google Workspace peut être un plus pour intégrer une messagerie et de collaboration.

Pour des emails transactionnels ou type de newsletter, Brevo. Cela permet une bonne délivrabilité.

## Accessibilité

* Compatibilité navigateur.

L’application doit être dans un premier temps être compatible avec les dernières versions de Chrome, Safari et Firefox.

* Types d’appareils

Le site sera développé uniquement pour un affichage de type desktop (ordinateur) sans version responsive.

## Recommandations en termes de sécurité

* + - 1. Authentification et communication sécurisée :

Activer la vérification par email pour valider les comptes utilisateurs.

Utilisation de HTTPS (SSL) pour toutes les communications.

Implémentation de JWT pour sécuriser les appels API (requêtes utilisateurs)

* + - 1. Base de données  :

Configurer l’IP Whitelisting pour autoriser uniquement les serveurs de confiance.

Protéger l’API en configurant les CORS pour n’autoriser que des origines spécifiques à accéder à

* + - 1. Fichiers :

N’autoriser que certain type de fichier (images) et limiter leur taille.

* + - 1. Surveillance :

Analyser régulièrement les bibliothèques utilisées (npm audit) et les garder à jour.

Valider les données saisies par les utilisateurs (formulaires, token).

## Maintenance du site et futures mises à jour

Mettre à jour les bibliothèques obsolètes ou vulnérables (npm-check).

Surveiller les performances

Sauvegarder les bases de données

Modification de l’application selon les feed-back.