**Spécifications techniques**

Menu Maker by Qwenta

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Auteur** | **Date** | **Approbation** |
| 1.0 | Camille, Webgencia | 03/11/2024 | John, Qwenta |

[I. Choix technologiques 1](#_Toc180644942)

[II. Liens avec le back-end 5](#_Toc180644943)

[III. Préconisations concernant le domaine et l’hébergement 6](#_Toc180644944)

[IV. Accessibilité 8](#_Toc180644945)

[V. Recommandations en termes de sécurité 9](#_Toc180644946)

[VI. Maintenance du site et futures mises à jour 10](#_Toc180644947)

## 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 non connectée | L'internaute doit pouvoir avoir accès aux différentes sections de la landing page non connectée :  - Bannière  - Personnalisez votre menu  - Explications étape par étape | React.js | React permet la création de composants réutilisables qui peuvent être mis à jour facilement sur la page de landing non connectée | 1. React est très populaire, notamment grâce à son interactivité et la possibilité de faire des mises à jour instantanées de l'interface utilisateur sans rechargement de la page.  2. Sa popularité garantit un accès à un écosystème riche et à une communauté de développeurs active. |
| Page login | - S'ouvre sous forme de modale  - L'utilisateur peut entrer son adresse mail  - Qu'il se soit déjà connecté ou non, un mail lui est envoyé pour lui permettre de s'authentifier, ou au contraire de confirmer son mail pour accéder pour la première fois à l'application  - Un lien "Besoin d'aide” permet d'envoyer un e-mail à nos équipes. | React-modal et MongoDB | Création de compte et authentification grâce à MongoDB en utilisant une adresse e-mail.  La fenêtre d’authentification s’ouvre sous forme de modale grâce à la bibliothèque react-modal. | 1. MongoDB offre des fonctionnalités d’authentification sécurisées pour protéger les données des utilisateurs et restaurateurs.  2. React-modal accorde une facilité pour la mise en place de la modale pour garantir une interactivité fluide et agréable avec les utilisateurs. |
| Catégorie de plat | L'utilisateur doit pouvoir :  - Créer une nouvelle catégorie  - La création de catégorie s'ouvre dans une modale spécifique, et doit pouvoir être validée  - Sélectionner une catégorie créée précédemment | Bibliothèque React-modal  et MongoDB + API Express de Node.js | La création de catégorie est gérée dans une modale React-modal, et grâce aux routes CRUD de Express par Node.js.  Chaque catégorie créée sera stockée dans la base de données de MongoDB. | 1. Express de Node permet la mise en place d’une API pour gérer les routes CRUD pour la création des nouvelles catégories et leur affichage une fois créées.  2. L’enregistrement des informations sur la base de données MongoDB permet une sécurisation des données et une réutilisation de ces données pour la création de futurs plats ou menus. |
| Création de plat | Le restaurateur doit pouvoir :  - Sélectionner une catégorie  - Modifier la catégorie sélectionnée  - Entrer les informations de son plat  - Une modale s'ouvre pour saisir les différents plats de la catégorie  - Chaque plat peut avoir une photo associée, un nom, un prix, une description  - Il est possible de créer autant de plats que l'on souhaite | Bibliothèque React-modal et MongoDB + API Express de Node.js | Le restaurateur pourra choisir une catégorie parmi celles crées précédemment et stockées dans la base de données de MongoDB, et les modifier grâce aux routes GET et PUT de l’API.  La saisie des plats s’affiche grâce à une modale générée via react-modal | 1. Express de Node permet grâce à son API de gérer facilement les routes CRUD déjà en places pour l'affichage et la modification des catégories déjà créées.  2. MongoDB permet une réutilisation des informations crées précédemment par l’utilisateur, permettant ainsi une approche dynamique et personnalisée. |
| Style de menu | Le restaurateur doit pouvoir :  - Visualiser le menu actuellement créé  - Sélectionner une typographie  - Choisir une couleur de texte | Bibliothèque styled-components | Styled-components va permettre d’appliquer les styles et de modifier dynamiquement l'apparence du menu.  On pourra aussi modifier les styles en fonction des choix de typographie de l'utilisateur pour générer du CSS adapté sans recharger la page. | 1. On peut facilement passer des props (propriétés) pour changer la typographie et les couleurs à la volée, rendant le style du menu interactif.  2. Les styles sont encapsulés dans les composants, évitant les conflits avec d'autres parties du projet.  3. styled-components permet de gérer des thèmes facilement, ce qui est utile pour des fonctionnalités comme le choix de typographies et de couleurs. |
| Exportation PDF | Le restaurateur doit pouvoir en un clic télécharger le fichier PDF correspondant à son menu. | Bibliothèque **jsPDF** en combinaison avec Node.js | En cliquant sur le bouton "exporter en .pdf", une requête est envoyée au serveur pour générer le PDF à partir des données du menu stockées dans la BDD, puis le fichier créé est renvoyé au client pour téléchargement. | 1. L'utilisation de bibliothèques pour la génération de PDF permet une personnalisation avancée et un rendu de qualité.  2. L'intégration avec le backend assure que les données du menu sont toujours à jour et reflètent les dernières modifications. |
| Commander l’impression d’un menu | - L'encart "Imprimer un menu” doit être visible depuis la page d'accueil  - Il s'agit d'un lien qui s'ouvre dans un nouvel onglet  - Le lien doit être fait vers le back-office de Qwenta | Lien vers le back-office de Qwenta avec attribut target=”\_blank” | L’ajout de l’attribut target=”\_blank” permet l’ouverture d’un nouvel onglet en cliquant sur “Imprimer un menu”. | 1. Amélioration de l'expérience utilisateur en ouvrant l'impression dans un nouvel onglet. L’utilisateur peut continuer de naviguer sur la page d'accueil sans perdre l'accès à ses informations actuelles, tout en accédant facilement à une nouvelle interface.  2. Le processus d’impression devient intuitif et facile d’accès pour le restaurateur. |
| Menus précédents | - Au clic sur "Mes menus”, le restaurateur doit avoir accès aux menus créés précédemment  - La date de création s'affiche  - Il est possible de modifier un menu précédent  - Il est possible de supprimer un menu précédent  - Sur la même vue, le restaurateur doit pouvoir créer un nouveau menu | React router avec Express de Node et ses routes CRUD, et MongoDB | React router côté front-end et Node avec Express côté back-end vont gérer les routes CRUD pour afficher, modifier, supprimer ou créer les éléments souhaités. Ces modifications seront enregistrées dans la base de données de MongoDB | 1. L'utilisation de React Router pour créer une vue dédiée aux menus précédents permettra au restaurateur d'accéder facilement à ses créations antérieures, de les modifier ou de les supprimer en toute simplicité.  2. La base de données stockera les informations des menus précédents, y compris la date de création, offrant ainsi une référence centrale pour le restaurateur. |
| Informations légales | - L'internaute doit pouvoir cliquer sur "Mentions légales” depuis les pages connectées et déconnectées  - La modale s'ouvre alors  - La mention "Tous droits réservés” doit figurer sur la page d'accueil et sur toutes les autres pages. | React-modal | Les mentions sont affichées grâce à un composant React.  La modale est créée grâce à la bibliothèque react-modal | 1. React-modal accorde une facilité pour la mise en place de la modale pour garantir une interactivité fluide et agréable avec les utilisateurs.  2. React nous permet d’ajouter facilement des composants réutilisables sur chaque page souhaitée. Ces composants peuvent être mis à jour facilement. |
| Tarifs | L'internaute doit pouvoir cliquer sur l'onglet Tarifs, et ouvrir un nouvel onglet. | Attribut target=”\_blank” | L’ajout de l’attribut target=”\_blank” permet l’ouverture d’un nouvel onglet en cliquant sur l’onglet “Tarifs”. | 1. Amélioration de l'expérience utilisateur en ouvrant l'impression dans un nouvel onglet. L’utilisateur peut continuer de naviguer sur la page d'accueil sans perdre l'accès à ses informations actuelles, tout en accédant facilement à une nouvelle interface.  2. L’accès aux tarifs devient intuitif et facile d’accès pour le restaurateur. |
| Exportation Deliveroo | - L'encart "Diffuser sur Deliveroo” doit s'afficher dans la catégorie "Exportez et diffusez”  - Au clic sur l'encart, l'utilisateur doit être redirigé sur l'application Deliveroo | Bouton de redirection vers Deliveroo et API Deliveroo | En cliquant sur l’encart, l’utilisateur est redirigé vers Deliveroo grâce au lien de redirection.  L’API de Deliveroo permet à l’utilisateur de gérer l’exportation sur la plateforme | 1. Deliveroo propose des APIs pour permettre aux restaurants de gérer directement leurs menus sur la plateforme.  2. L'API Menu de Deliveroo permet aux restaurants d'intégrer leurs systèmes internes avec Deliveroo pour synchroniser les données de menus. |
| Partage sur Instagram | - L'encart "Partager sur Instagram” doit s'afficher dans la catégorie "Exportez et diffusez”  - Au clic sur l'encart, des images du menu au format carré (pour les mettre sur Instagram) sont générées  - Le restaurateur est redirigé vers son compte Instagram avec les photos carrées des menus. | React-to-image et API officielle d’Instagram | Utilisation de la bibliothèque react-to-image pour transformer un élément en image, et de l’API d’Instagram pour exporter facilement ces images sur la plateforme | 1. react-to-image est une solution simple pour capturer des composants React et les transformer en image, ce qui s'avère pratique pour des cas comme l’exportation d’un menu à partager sur des réseaux sociaux comme Instagram.  2. L’API d’Instagram nous permet d’utiliser une source sécurisée pour exporter nos menus sur Instagram. |
| Déconnexion | Le restaurateur doit pouvoir se déconnecter depuis n'importe quelle page connectée. | React, composant bouton de déconnexion | Création d’un composant React pour le bouton de déconnexion rajouté sur chaque page concernée. | 1. Cela facilite la déconnexion du restaurateur, sans avoir à se rendre sur une page spécifique.  2. Le restaurateur profite d’une expérience fluide et conviviale avec le site. |
| Infos utilisateur | - Lier plusieurs adresses e-mail à son compte  - Modifier son adresse e-mail de base | MongoDB et bibliothèque Mongoose | Nous utilisons MongoDB avec la bibliothèque Mongoose pour modifier l’adresse e-mail ou stocker plusieurs adresses e-mail pour le même utilisateur | 1. MongoDB est une base de données NoSQL qui permet de stocker plusieurs adresses e-mail liées à un même utilisateur en utilisant un tableau d'objets dans le document utilisateur.  2. Les modifications et mises à jour se font via des routes CRUD dans Node.js avec Mongoose pour modifier l'adresse e-mail de base ou ajouter/supprimer des e-mails secondaires. |
| Dashboard | L'internaute doit pouvoir avoir accès :  - À l'encart "Créer un menu”  - À l'encart "Diffuser un menu”  - À l'encart "Imprimer un menu”  - À la section “Pour aller plus loin”  - Aux 3 derniers articles de blog qui parlent de Menu-Maker :  - Titre  - Photo de couverture  - Lien | Back Office Qwenta et React | Les encarts et sections sont affichés grâce à des composants React.  Les articles de blog sont générés via le back-office de Qwenta. | 1. L'utilisation de React permet de créer une interface utilisateur dynamique où les encarts s'actualisent sans recharger la page, offrant ainsi une expérience utilisateur fluide et moderne.  2. Grâce au back-office de Qwenta, tous les contenus sont facilement gérables et mis à jour par les administrateurs, ce qui facilite la maintenance et les mises à jour. |
| Branding restaurateur | Le restaurateur doit pouvoir ajouter / modifier / supprimer les éléments suivants :  - Logo  - Couleurs de base | MongoDB, multer et styled-components | Le logo sera géré par multer et enregistré dans la base de données de MongoDB.  La modification des couleurs sera gérée par la bibliothèque styled-components. | 1. L’utilisation de multer avec MongoDB permet une simplicité et une modularité, notamment pour faciliter toute modification future.  2. Styled-Components permet aux restaurateurs de voir en temps réel la modification de couleurs ou de logo, ce qui améliore l'expérience utilisateur et rend le processus de personnalisation fluide et engageant. |

## Liens avec le back-end

* **Quel langage pour le serveur ?**

Nous avons opté pour Node.js associé à Express pour notre serveur en raison des avantages suivants :

* **Rapidité de mise en œuvre :** Node.js permet un développement fluide et rapide. Le framework Express simplifie la création de serveurs web performants, en accélérant le processus de développement tout en assurant robustesse et flexibilité.
* **Évolutivité et gestion des performances :** Node.js est reconnu pour sa capacité à gérer un grand volume de connexions simultanées. Cela le rend idéal pour des applications à forte demande en temps réel. Couplé à Express, cela permet de concevoir des solutions web hautement performantes et évolutives.
* **A-t-on besoin d’une API ?**

Nous allons également développer une API Node (API REST) pour accéder aux données via des requêtes HTTP.

Nous aurons besoin de deux API externes :

* L'API officielle **d’Instagram ;**
* Menu API de **Deliveroo.**
* **Base de données choisie :**

**MongoDB** sera préféré pour ce projet, notamment pour deux principales raisons :

* **Flexibilité avec les données non structurées :** MongoDB, étant une base de données NoSQL, permet de stocker des documents JSON flexibles, ce qui est idéal pour gérer les menus de restaurants qui peuvent avoir des structures de données variées (plats avec ou sans options, descriptions, prix, etc.).
* **Scalabilité et performances** **:** Grâce à sa capacité à gérer des quantités importantes de données et à répartir la charge sur plusieurs serveurs (sharding), il est idéal pour des projets qui pourraient évoluer avec le temps et nécessiter une gestion de nombreuses données utilisateurs et menus.

## Préconisations concernant le domaine et l’hébergement

* **Nom du domaine.**

Nous suggérons un sous-domaine Qwenta, pour profiter de l’autorité de domaine de Qwenta et renforcer l'identité de la marque tout en associant clairement le projet à la société mère. Par exemple :

* menu.qwenta.com
* Menumaker.qwenta.com

Dans le cas contraire, un nom de domaine autonome peut accorder plus de visibilité et de spécificité. Par exemple :

* www.menu-maker.com.
* **Nom de l’hébergement.**

Nous suggérons Vercel pour héberger le projet, pour plusieurs raisons :

* **Facilité de déploiement :** Vercel est particulièrement adapté pour des projets utilisant React et Node.js.
* **CDN intégré :** Vercel offre un CDN performant pour une rapidité de chargement optimale, ce qui est essentiel pour offrir une expérience fluide aux utilisateurs.
* **Déploiements automatiques :** Il permet de déployer des modifications instantanément à chaque push sur le repository GitHub, facilitant la gestion des mises à jour.
* **Support Node.js :** Il gère très bien les applications Node.js et les routes d'API, ce qui est idéal pour un projet full-stack avec Express.
* **Adresses e-mail.**

Des adresses comme contact@menu-maker.com ou support@menu-maker.com fournissent une apparence professionnelle et facilitent la communication avec les utilisateurs et les clients.

## Accessibilité

* **Compatibilité navigateur**

Une comptabilité avec Google Chrome, Safari et Firefox est nécessaire pour les premiers temps de ce projet.

* **Types d’appareils**

Menu-Maker nécessite une version Desktop, mais pas de version mobile ou tablette à développer pour le moment.

* **Accessibilité pour les personnes handicapées**

Le respect des standards WCAG pourra rendre le site accessible aux utilisateurs ayant des handicaps visuels, auditifs ou moteurs : compatibilité des lecteurs d'écran, descriptions des images (attributs alt), et navigation au clavier.

* **Contrastes et choix de couleurs**

Il faudra assurer un contraste suffisant entre le texte et l'arrière-plan pour les personnes à déficiences visuelles.

* **Taille de texte ajustable**

La taille de texte ajustée (en em ou rem plutôt qu’en pixels) va permettre aux utilisateurs d'ajuster la taille du texte sans perdre en qualité de mise en page.

* **Structure sémantique**

La sémantique des balises HTML sera appliquée pour garantir que la structure du contenu est logique et compréhensible par les technologies d'assistance (ex : titres hiérarchisés : h1, h2, etc..).

## Recommandations en termes de sécurité

* **Audit et surveillance :** Mettre en place une journalisation et une surveillance des requêtes vers MongoDB pour détecter toute activité suspecte. Des outils comme le suivi des opérations (auditing) permettent de suivre les tentatives d'accès non autorisées et les anomalies dans l'utilisation des données.
* **Règles d'accès et d'autorisation :** Configurer des rôles et permissions d'accès stricts via **MongoDB Atlas** ou la gestion des utilisateurs dans MongoDB. Cela permet de s'assurer que seuls les utilisateurs et services autorisés peuvent lire ou modifier les données.
* **Mises à jour régulières :** MongoDB ainsi que les bibliothèques utilisées avec Node.js doivent régulièrement être mises à jour pour corriger les failles de sécurité. Utiliser des versions obsolètes expose l'application à des vulnérabilités connues.
* **Chiffrement des données et HTTPS :** Activer le chiffrement des données dans MongoDB pour garantir la sécurité des échanges de données et du stockage. MongoDB Atlas, par exemple, propose des options de chiffrement transparent.
* **Protection contre les injections :** La validation des données d'entrée est cruciale pour prévenir les attaques d'injection. Avec MongoDB, l'utilisation des bibliothèques de validation (comme **Joi** pour Node.js) permet de s’assurer que toutes les requêtes vers la base de données sont filtrées et sécurisées.
* **Durée de vie des tokens d'authentification :** Les tokens d'authentification (ex : JWT) doivent être utilisés pour sécuriser les sessions utilisateur. Ils doivent avoir une durée de vie appropriée. En cas de vol de token, cela limite les risques d'accès non autorisé.
* **Journalisation et surveillance des erreurs :** Un système de **logging** détaillé doit être en place pour surveiller les erreurs et les événements critiques dans MongoDB et Node.js. Les erreurs liées aux requêtes, aux connexions à la base de données et aux tentatives d'accès doivent être surveillées et documentées pour détecter rapidement des tentatives d'attaques.
* **Sauvegarde régulière des données :** Il faudra mettre en place un plan de sauvegarde régulier pour MongoDB afin de garantir la restauration des données en cas de problème. MongoDB Atlas propose des options de sauvegarde automatisée, mais il est également possible de configurer manuellement des scripts de sauvegarde journalière. Des tests réguliers des restaurations pourront assurer la fiabilité des sauvegardes.
* **Conformité RGPD et bonnes pratiques OWASP :** Le projet doit respecter les règles du **RGPD**, notamment en ce qui concerne la collecte, le traitement et la sécurisation des données personnelles. Il faudra également suivre les recommandations de l'OWASP pour protéger l'application contre les vulnérabilités courantes, telles que les attaques **XSS**, **CSRF**, ou encore les **failles d'injection**.
* **Authentification sécurisée :** Utilise des systèmes d'authentification robustes comme **OAuth,** en combinaison avec MongoDB, pour assurer que seuls les utilisateurs authentifiés peuvent accéder à des sections sensibles de l'application. L'authentification multi-facteurs (MFA) est également recommandée pour renforcer la sécurité des utilisateurs.

## Maintenance du site et futures mises à jour

* **Mises à jour régulières du code :** Il est crucial de maintenir le site à jour en ce qui concerne les dépendances, les bibliothèques et les frameworks utilisés, afin de corriger les bugs, d'améliorer les performances et de sécuriser les nouvelles failles découvertes. Des vérifications trimestrielles seront planifiées.
* **Correction des bugs et support technique :** Un système de suivi des bugs sera mis en place pour identifier et résoudre rapidement les problèmes signalés par les utilisateurs ou détectés via la surveillance du système. Nous privilégions une solution comme **Notion** pour organiser et gérer les tâches liées à la maintenance.
* **Surveillance des performances et de la disponibilité :** La surveillance continue des performances est essentielle pour identifier d’éventuelles lenteurs ou interruptions de service. Nous utiliserons des outils comme **Uptime Robot** pour suivre les performances du site et recevoir des alertes en cas de problème.
* **Améliorations fonctionnelles :** Nous prévoyons des mises à jour fonctionnelles basées sur les retours utilisateurs et l’accord des responsables de Menu-Maker. Chaque fonctionnalité majeure passera par une phase de test avant d’être intégrée à la version en production.
* **Plan de sécurité et gestion des vulnérabilités :** Nous mettrons en place des audits de sécurité réguliers pour identifier et résoudre toute vulnérabilité. Cela inclut la mise à jour des politiques de sécurité en fonction des nouvelles normes du secteur, l'amélioration des pratiques d'authentification et d'autorisation, ainsi que la vérification continue des accès non autorisés via les journaux d’audit.
* **Sauvegardes et récupération de données :** Des sauvegardes automatiques des bases de données et des fichiers critiques seront effectuées quotidiennement. Cela garantit la sécurité des données et permet une récupération rapide en cas de défaillance du système. Nous réviserons et testerons ces systèmes régulièrement pour assurer leur efficacité.
* **Test et déploiement continus :** Avant chaque mise à jour, nous suivrons une procédure de tests approfondis pour éviter l'introduction de bugs. Un environnement de pré-production sera mis en place pour valider les changements avant déploiement. Les nouvelles versions seront déployées en dehors des heures de pointe pour éviter de perturber les utilisateurs.
* **Documentation des mises à jour :** Chaque mise à jour du site sera documentée pour assurer la transparence. Cela inclut des **release notes** avec les changements effectués, ainsi que des informations sur les mises à jour majeures et les corrections de bugs.