**Spécifications techniques**

Menu Maker by Qwenta

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Auteur** | **Date** | **Approbation** |
| 1.0 | SALHI Hanane | 26/09/2024 | Soufian |

[I. Choix technologiques](#_409jm82qlgst) 2

[II. Liens avec le back-end](#_ex0md8td8283) 3

[III. Préconisations concernant le domaine et l’hébergement](#_a61v3uh5vs19) 3

[IV. Accessibilité](#_n56najuvb5ys) 3

[V. Recommandations en termes de sécurité](#_l80x4g9stizp) 3

[VI. Maintenance du site et futures mises à jour](#_vbbknpfyus5o) 4

## 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*** | *Avoir accès aux sections « Bannière » , «Personnalisez votre menu» et «Explications étape par étape»* | *React,js avec React-router pour la navigation fluide entre les sections* | *React.js pour le développement de la landing page. Nous intégrerons également la bibliothèque react-scroll pour permettre une navigation fluide et agréable entre les différentes sections* | *1)React.js permet de créer des composants modulaires et réutilisables, ce qui facilitera la création et la gestion des différentes sections de la landing page.*  *2) L'utilisation de reactscroll garantit une expérience utilisateur fluide en permettant aux visiteurs de faire défiler la page de manière intuitive vers les différentes sections, ce qui améliore l'accessibilité et l’engagement.* |
| ***Connection restaurateur***  ***Création de compte internaute*** | *Cette fenêtre s'ouvre sous forme de modale .*  *L'utilisateur peut entrer son adresse mail .*  *Mail d’authentification ou confirmation d’email .*  *Un lien "Besoin d'aide” permet d'envoyer directement un e-mail à nos équipes* | *react-modal*  *Node.js avec Express pour le serveur*  *MongoDB : pour stocké les donnees et en utilisant mongoose*  *JWT (JSON Web Tokens)* | *Utilisation* ***Nodemailer*** *une bibliothèque Node.js qui permet d'envoyer des emails facilement.*  *La fenêtre d'authentification sera mise en œuvre en utilisant la bibliothèque react-modal.*  *MongoDB :* ***Base de Données*** *pour stocker les informations des utilisateurs (emails, jetons d’authentification)* | ***1 Nodemailer*** *facilite l'envoi d'emails transactionnels, comme les confirmations d'inscription ou les réinitialisations de mots de passe, qui sont essentiels dans tout système d'authentification*  *2 ) L'utilisation de Mongoose garantit une meilleure cohérence et structure des données dans MongoDB*  *JWT permet de créer des jetons d'authentification auto-contenus qui peuvent être utilisés pour vérifier l'identité de l'utilisateur* |
| ***Création d'une catégorie de menu*** | *L'ajout d'une catégorie doit se faire directement sur l'écran de création de menu depuis une modale* | *react-modal.* | *Cette librairie React permet de créer simplement des modales performantes, accessibles avec un minimum de code* | *1)* ***React Modal*** *s'intègre parfaitement dans une application React, en utilisant des composants réactifs et un état centralisé*  *2)* ***React Modal*** *permet à la fois une implémentation rapide et personnalisée des modales, tout en respectant les normes d'accessibilité nécessaires pour offrir une expérience inclusive.* |
| ***Création de plats restaurateur*** | *Le restaurateur doit pouvoir: .Sélectionner une catégorie, la modifié, .Entrer les infos de son plat .Saisir les différents plats de la catégorie via une modale .Créer autant de plats qu’il le souhaite . Chaque plat peut avoir : Une photo associée Un nom Un prix Une description* | *React avec Reactmodal pour la creation de plats* | *Nous développerons une interface utilisateur en utilisant React.*  *Une fois connecté, le restaurateur pourra sélectionner une catégorie et la modifier. Ensuite, en utilisant une modale (react-modal), le restaurateur pourra entrer les informations spécifiques de chaque plat, y compris une photo, un nom, un prix et une description. La possibilité de créer autant de plats que nécessaire sera également implémentée.* | *1) React offre la possibilité de créer une interface utilisateur conviviale et réactive. L'utilisation de modales avec react-modal simplifie le flux de création de plats en permettant au restaurateur de gérer les informations de chaque plat de manière organisée.*  *2) En permettant au restaurateur de saisir une photo, un nom, un prix et une description pour chaque plat, nous offrons une personnalisation complète, aidant ainsi à refléter la diversité et l'unicité des plats offerts par le restaurant.* |
| ***Personnalisation menu restaurateur*** | *Le restaurateur doit pouvoir : Visualiser le menu créé Sélectionner une typographie Choisir une couleur de texte* | *Css personnalisé et bibliothèques de typographie* | *Pour la personnalisation, nous utiliserons des technologies frontales telles que CSS personnalisé pour gérer la couleur du texte, ainsi que des bibliothèques de typographie pour permettre au restaurateur de choisir parmi une gamme de polices.* | *1)L'utilisation de CSS personnalisé offre une personnalisation précise de la couleur du texte*  *2) l'utilisation de bibliothèques de typographie élargit les options de choix de typographie, permettant au restaurateur de refléter l'identité visuelle de son restaurant.* |
| ***Exportation menu PDF restaurateur*** | *Le restaurateur doit pouvoir: En un clic télécharger le fichier PDF correspondant à son menu* | *Utilisation de la bibliothèque* ***jsPDF*** *pour générer des PDF* | *Nous utiliserons la bibliothèque JavaScript* ***jsPDF****, qui permet de générer des fichiers PDF à partir des données du menu personnalisé. Une fois que le restaurateur est satisfait de son menu, il pourra simplement cliquer sur un bouton pour déclencher la génération et le téléchargement du fichier PDF.* | *1)L'utilisation de la bibliothèque jsPDF simplifie le processus de génération de PDF, ce qui permet au restaurateur de créer rapidement un fichier PDF de son menu personnalisé.*  *2) En offrant la possibilité de télécharger le menu au format PDF en un clic, nous facilitons le processus pour le restaurateur et lui permettons d'avoir une copie imprimable de son menu avec tous les détails personnalisés.* |
| ***Impression menu restaurateur*** | *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* | *Ajout d’un lien vers le back office de Qwenta* | *Nous ajouterons un lien "Imprimer un menu” sur la page d'accueil du restaurateur. Ce lien redirigera le restaurateur vers le back office de Qwenta en utilisant l'attribut HTML target="\_blank", ce qui ouvrira le lien dans un nouvel onglet du navigateur.* | *1) En plaçant un lien visible sur la page d'accueil, nous rendons le processus d'impression du menu intuitif et facile d'accès pour le restaurateur.*  *2) L'ouverture du lien dans un nouvel onglet permet au restaurateur de conserver l'interface de gestion de son menu tout en accédant au back office de Qwenta pour l'impression* |
| ***Vue regroupant les menus précédents restaurateur*** | *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/supprimer un menu précédent .Sur la même vue, le restaurateur doit pouvoir créer un nouveau menu* | *Utilisation de React Router et gestion des menus dans une base de données* | *Nous créerons une vue dédiée dans l'application utilisant React Router. Cette vue affichera les menus précédemment créés par le restaurateur. Les informations de chaque menu, y compris la date de création, seront extraites d'une base de données. Pour la modification et la suppression des menus, des boutons correspondants seront fournis. Pour créer un nouveau menu, un bouton "Nouveau menu" sera également présent sur la même vue.* | *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 internaute*** | *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 pour les mentions légales et ajout de la mention sur les pages* | *Nous mettrons en place un*  *lien "Mentions légales”*  *sur les pages connectées*  *et déconnectées. Lorsque*  *ce lien est cliqué, une*  *modale (réalisée avec*  *react-modal) s'ouvrira*  *pour afficher les*  *informations légales. Par*  *ailleurs, la mention "Tous*  *droits réservés” sera*  *ajoutée au bas de la page*  *d'accueil et de toutes les*  *autres pages du site.* | *1) En permettant à l'internaute d'accéder aux mentions légales depuis n'importe quelle page, nous favorisons la transparence et le respect des réglementations.*  *2) L'ajout de la mention "Tous droits réservés” sur toutes les pages assure la conformité juridique du site et renforce la cohérence de l'identité légale de l'entreprise sur l'ensemble du site.* |
| ***Tarifs internaute*** | *L'internaute doit pouvoir cliquer sur l'onglet Tarifs, et ouvrir un nouvel onglet* | *Lien avec l’attribut target="\_blank"* | *Nous créerons un onglet Tarifs sur le site. Lorsque l'internaute clique sur cet onglet, un lien sera configuré avec l'attribut HTML target="\_blank", ce qui permettra d'ouvrir une nouvelle fenêtre ou un nouvel onglet du navigateur, affichant ainsi les informations tarifaires* | *1) L'utilisation de target="\_blank" garantit que l'internaute peut accéder aux informations tarifaires sans quitter complètement la page actuelle, améliorant ainsi l'expérience utilisateur.*  *2) Ouvrir les tarifs dans un nouvel onglet permet à l'internaute de naviguer entre les informations tarifaires et le site principal de manière plus fluide.* |
| ***Exportation Deliveroo restaurateur*** | *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 l’application Deliveroo* | *Nous ajouterons un bouton "Diffuser sur Deliveroo” dans la catégorie "Exportez et diffusez” de l'interface du restaurateur. Lorsque l'utilisateur clique sur ce bouton, il sera redirigé vers l'application Deliveroo, ce qui lui permettra de démarrer le processus d'exportation de son menu vers la plateforme Deliveroo.* | *1) Placer le bouton "Diffuser sur Deliveroo” dans la catégorie appropriée permet au restaurateur de trouver rapidement l'option d'exportation, simplifiant ainsi le processus.*  *2) La redirection vers l'application Deliveroo garantit une interaction transparente pour le restaurateur, l'aidant à commencer immédiatement le processus d'intégration de son menu sur la plateforme de livraison.* |
| ***Partage sur instagram restaurateur*** | *"Partager sur Instagram” doit s'afficher dans la catégorie "Exportez et diffusez”. Au clic, des images du menu au format carré sont générées. Le restaurateur est redirigé vers son compte Instagram avec les photos carrées des menus* | ***html2canvas*** | *Utilise une bibliothèque comme* ***html2canvas*** *pour capturer des sections du DOM correspondant au menu créé.*  *. Rediriger l’utilisateur vers l’application Instagram avec les images générées.* | *1 : En utilisant une bibliothèque comme* ***html2canvas****, tu peux capturer la section exacte du menu que le restaurateur souhaite partager. Cela permet une personnalisation flexible, tout en automatisant le processus d’extraction d’images.*  *2 :* ***Expérience utilisateur fluide*** *: En redirigeant directement l'utilisateur vers Instagram* |
| ***Déconnexion restaurateur*** | *Le restaurateur doit pouvoir se déconnecter depuis n'importe quelle page connectée* | *ReactRouter* | *Utilise le* ***React Router*** *pour la redirection après déconnexion.*  *Supprime le token de session dans le* ***localStorage*** | *1 :* ***Prévention des accès non autorisés*** *: En permettant une déconnexion facile et accessible depuis n'importe quelle page, on renforces la sécurité de l'application en empêchant qu'une session reste ouverte accidentellement*  *2 : Une fonctionnalité de déconnexion bien intégrée permet de gérer efficacement les sessions utilisateurs* |

## Liens avec le back-end

* + 1. **Quel langage pour le serveur ?**

Le langage recommandé pour le serveur serait **React** une bibliothèque de **javascript**, avec **Node.js** comme environnement d'exécution. Voici pourquoi :

**Node.js** est particulièrement performant pour les applications web grâce à son modèle asynchrone non-bloquant, ce qui permet de traiter plusieurs requêtes simultanément avec une grande efficacité.

En utilisant **JavaScript** à la fois côté client (**React**) et côté serveur, on réduis la complexité liée à l’apprentissage d’un nouveau langage, ce qui facilite la communication entre les différentes parties du projet.

**2. A-t-on besoin d’une API ? Si oui laquelle ?**

Oui, une **API RESTful** est nécessaire pour permettre la communication entre le **frontend** (React) et le **back-end** (Node.js). Cette API servira à gérer :

La gestion des utilisateurs (authentification, création de compte, déconnexion).

La création, la mise à jour et la récupération des données des menus des restaurateurs.

Le partage des données comme les images des menus pour Instagram.

**Express.js** serait un bon choix pour construire cette API en Node.js. Il est léger, flexible, et s'intègre bien avec d'autres middleware et services nécessaires pour l'application, notamment pour l'authentification (JWT) et les connexions à la base de données.

**3. Base de données choisie :**

La base de données choisie pour ce projet serait **MongoDB**, une base de données NoSQL, pour les raisons suivantes :

* **Flexibilité** : MongoDB stocke les données sous forme de documents JSON/BSON, ce qui s'intègre parfaitement avec les données manipulées en JavaScript côté back-end (Node.js) et front-end (React). Cette structure permet de gérer des données non structurées ou évolutives comme les informations des menus des restaurants.
* **Scalabilité** : MongoDB permet une scalabilité horizontale, ce qui est utile pour une application avec une croissance rapide des utilisateurs (restaurateurs) et des menus.
* **Performance** : Les performances en lecture/écriture de MongoDB sont optimisées pour des applications web qui nécessitent un accès rapide à de nombreuses données.

## Préconisations concernant le domaine et l’hébergement

* **Nom du domaine:**  www.menu-maker.com

Ce nom de domaine est simple, mémorable et directement lié à la fonctionnalité principale du service, ce qui facilite la recherche par les utilisateurs potentiels.

* **Nom de l’hebergement:**
* Vercel ou Netlify : Si des fonctionnalités front-end avancées sont nécessaires, ces plateformes sont également optimisées pour les applications React et offrent un déploiement facile.
* Heroku : Une plateforme populaire qui prend en charge des déploiements full-stack (front-end et back-end) et est idéale pour les projets qui nécessitent une API ou une base de données
* AWS Amplify pour une gestion full-stack à grande échelle.
* GitHub Pages pour des projets statiques,
* Render pour des déploiements full-stack simples et efficaces.
* **Adresses e-mail**:
* contact@menu-maker.com : Utilisé pour les demandes de renseignements généraux, le support technique ou la communication avec les clients.
* infos@menu-maker.com : Pour des communications informatives, comme des newsletters, des mises à jour produit ou des annonces.

## Accessibilité

* Compatibilité navigateur :

**Compatibilité cross-navigateur** : L'application doit être compatible avec les principaux navigateurs, y compris **Google Chrome**, **Mozilla Firefox**, **Microsoft Edge**, **Safari**, et **Opera**. Cela garantit que les utilisateurs peuvent accéder à la plateforme peu importe leur navigateur préféré.

**Tests et ajustements** : Il sera important de tester régulièrement l'application sur plusieurs versions de ces navigateurs pour identifier d'éventuelles incompatibilités liées à certaines fonctionnalités (comme les modales ou les animations).

* Types d’appareils.  
  \* **Ordinateurs de bureau et ordinateurs portables** : L'application doit être entièrement fonctionnelle et optimisée pour les écrans de grandes tailles, avec une interface fluide et responsive

\* **Tablettes** : L'interface doit être optimisée pour les tablettes, avec une mise en page adaptée pour les écrans tactiles (par exemple, des boutons plus larges et des éléments plus espacés pour faciliter la navigation tactile).

## Recommandations en termes de sécurité

#### 1. **Accès aux comptes** :

**Authentification sécurisée** :

* + **Utilisation de JWT (JSON Web Tokens)** : Pour sécuriser l'authentification des utilisateurs (restaurateurs et clients), il est recommandé d’utiliser des **tokens JWT** signés qui sont stockés côté client (dans le **localStorage** ou **sessionStorage**). Les tokens JWT permettent de s'assurer que seules les personnes authentifiées peuvent accéder aux fonctionnalités protégées.

**Vérification des dépendances tierces** :

* Les dépendances et plugins utilisés dans le projet (bibliothèques npm, par exemple) doivent être régulièrement vérifiés pour des vulnérabilités de sécurité connues. Outils comme **npm audit** ou **Snyk** peuvent être utilisés pour détecter et corriger ces vulnérabilités dans les versions des bibliothèques.
* S'assurer de toujours utiliser des versions à jour des bibliothèques et frameworks, car les mises à jour contiennent souvent des correctifs de sécurité.

2 : **Outils et bonnes pratiques supplémentaires :**

* **Gestion des sessions** : Implémenter un **timeout** pour les sessions utilisateur. Si une session reste inactive pendant une période définie (par exemple 15 minutes), il est conseillé de déconnecter automatiquement l'utilisateur pour minimiser les risques d'abus.

## Maintenance du site et futures mises à jour

**1. Grandes lignes du contrat de maintenance**

Pour assurer la maintenance du site web et anticiper les futures mises à jour, un **contrat de maintenance** doit couvrir plusieurs aspects essentiels. Voici les grandes lignes à inclure :

**a. Maintenance corrective :**

* **Correction des bugs** : Le contrat doit garantir la correction des **erreurs ou bugs** qui peuvent survenir après le déploiement de l'application. Cela inclut :
* Les erreurs **front-end** (problèmes d'affichage ou d'interaction sur React).
* Les erreurs **back-end** (erreurs liées à l'API ou à la base de données).
* La correction des vulnérabilités de sécurité qui pourraient être découvertes après la mise en ligne.

**b. Maintenance préventive :**

* **Surveillance régulière** : Mise en place de mécanismes de **monitoring** pour surveiller la performance et la disponibilité du site (ex. **uptime monitoring** avec des outils comme Pingdom, ou New Relic).
* **Vérifications des dépendances** : Mise à jour proactive des **bibliothèques** utilisées (React, Node.js, Express, MongoDB, etc.) pour éviter les problèmes de sécurité ou d'incompatibilité future.

**c. Maintenance évolutive :**

* **Ajout de nouvelles fonctionnalités** : Le contrat doit prévoir la possibilité d'ajouter de nouvelles fonctionnalités ou modules à l'application à la demande du client. Par exemple, la mise à jour des modales, l'intégration de nouveaux réseaux sociaux pour le partage, ou l'ajout de nouveaux types de contenu.
* **Optimisation des performances** : La maintenance doit inclure des efforts pour améliorer les performances de l'application (chargement des pages, vitesse de réponse de l'API) en fonction des retours d'utilisateurs et des besoins évolutifs.

**e. Support technique :**

* **Support utilisateur** : Le contrat doit inclure une option de **support technique** pour les utilisateurs du site (restaurateurs, clients). Ce support peut être fourni via email, téléphone, ou un système de ticket en ligne.
* **Temps de réponse** : Définir des **SLAs (Service Level Agreements)** pour les temps de réponse et de résolution des problèmes. Par exemple, un engagement à répondre aux tickets critiques dans les 24 heures, et à corriger les problèmes critiques dans les 48 heures.
* **f. Mises à jour de sécurité :**
* **Patching** : Le contrat doit garantir des **mises à jour régulières** de sécurité, que ce soit pour corriger des failles dans les bibliothèques utilisées (Node.js, MongoDB, etc.) ou pour intégrer les dernières pratiques de sécurité (ex. passage à une version de TLS plus sécurisée, gestion des authentifications).
* **Test des mises à jour** : Chaque mise à jour (fonctionnelle ou de sécurité) devra être testée en amont sur un environnement de **staging** avant d'être déployée en production, pour éviter tout impact négatif sur les utilisateurs finaux.

**g. Sauvegarde des données :**

* **Sauvegarde régulière de la base de données** : Il est essentiel de prévoir des **backups** réguliers de la base de données (MongoDB) pour garantir que les données des utilisateurs (menus, comptes, etc.) soient sauvegardées et restaurables en cas d'incident.
* **Plan de reprise après sinistre (Disaster Recovery)** : Le contrat doit inclure un **plan de reprise après sinistre** pour restaurer les données et remettre en ligne le site en cas de panne majeure.
* **Analyse des performances** : Ce rapport peut également inclure des **statistiques de performance** (temps de chargement des pages, taux d'erreur, etc.) et proposer des axes d'amélioration.