Webgencia

# Spécifications techniques

[Menu_Maker + Qwenta]

| Version | Auteur | Date | Approbation |
|---|---|---|---|
| 1.0 | Pryvalov, Oleksandr | 16/08/2024 | Qwenta, John |



## I. Choix technologiques

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

| Besoin | Contraintes | Solution | Description de la solution | Justification (2 arguments) |
|---|---|---|---|---|
| Comprendre l'utilité de l'application. | *L'internaute doit avoir accès aux sections informatives de la landing page non connectée.* | **React**<br><br>**react-router-dom** | *- Sections dynamiques : La page est divisée en sections claires (Bannière,* Personnalisez votre menu, *Explications étape par étape ) crées avec React.js pour une navigation fluide.* | 1) Permet une expérience utilisateur fluide et réactive.<br>2) Compatible avec l'architecture globale de l'application. |
| • Connexion pour les restaurateurs.<br>• Création d'un compte. | La fenêtre de connexion/création doit s'afficher sous forme de modale et permettre l'authentification/création par e-mail. | **-React-modal**<br><br>-**Firebase Authentication** | - Intégrer une fenêtre via React modal pour une connexion fluide.<br>- Utiliser Firebase Authentication pour gérer les connexions et créations de compte avec adresse e-mail.<br>- Implémenter l'envoi d'e-mails de confirmation avec Firebase Authentication et vérifier les adresses e-mail des utilisateurs.<br>- Ajouter un lien permettant d'envoyer directement un e-mail à l'équipe de support. | 1) React Modal permet une expérience utilisateur fluide et cohérente.<br>2) Firebase simplifie la gestion des utilisateurs et des confirmations par e-mail. |

| | | | | |
|---|---|---|---|---|
| Créer une catégorie de plats. | L'ajout d'une catégorie doit pouvoir se faire directement sur l'écran de création de menu depuis une modale. | **-React-modal**<br><br>**-Firebase Firestore** | Utilisation de **react-modal** pour créer des modales interactives permettant la saisie de catégories, et **Firebase Firestore** pour stocker et gérer ces catégories. | 1) **react-modal** est compatible avec React et aligné avec les technologies choisies.<br>2) **Firebase Firestore** est scalable et permet une gestion efficace des données structurées. |
| Ajouter un plat dans le menu. | Permettre l'ajout de plusieurs plats avec des champs personnalisables. | **-React Modal**<br><br>**-Firebase Storage**<br><br>**-Firebase Firestore** | Utilisation de **react-modal** pour afficher une modale de saisie des informations des plats, **Firebase Firestore** pour stocker les détails des plats, et **Firebase Storage** pour gérer les images associées aux plats. | 1) **Firebase Firestore** et **Storage** permettent une gestion centralisée des données et images.<br>2) La solution permet de modifier et d'accéder aux plats et menus de manière fluide. |
| Personnaliser le style du menu. | Permettre la sélection de typographies, couleurs de texte et visualiser le menu créé. | **-React Modal**<br><br>**-react-color** .<br><br>**-Google Fonts API** | Une modale s'ouvre pour permettre au restaurateur de visualiser le menu en temps réel et de personnaliser les typographies et couleurs. React-color permet de choisir une couleur, et Google Fonts API offre une sélection de typographies. | **1. react-color** et **Google Fonts API** sont faciles à intégrer avec React, assurant une expérience utilisateur fluide.<br>2. **Personnalisation en temps réel** : Le restaurateur peut voir les modifications en direct, améliorant ainsi l'interactivité. |

| | | | | |
|---|---|---|---|---|
| Exporter le menu en PDF. | Exportation en un clic avec un format de fichier standard et lisible sur tous les appareils. | **-react pdf** | Intégration de **-react pdf** pour générer un fichier PDF directement à partir des données du menu. | 1. Intégration facile avec React.<br>2. Permet créer et télécharger le PDF avec les styles choisis. |
| Commander l'impression du menu. | Le lien d'impression doit rediriger vers une page spécifique et s'ouvrir dans un nouvel onglet. | **-react-pdf** | 1. **Création du Lien**: Ajouter un bouton "Imprimer un menu" sur la page d'accueil.<br>2.Utiliser `react-pdf` pour générer le PDF du menu et intégrer une option pour imprimer directement depuis le navigateur dans un nouvel onglet. | 1. **Intégration Efficace**: `react-pdf` facilite la génération et l'impression de PDFs depuis une application React.<br>2. **Expérience Utilisateur Simplifiée**: Permet une gestion fluide de l'impression directement depuis l'application, améliorant l'efficacité pour le restaurateur. |

| | | | | |
|---|---|---|---|---|
| Accéder à une vue regroupant les menus créés précédemment. | - Au clic sur "Mes menus", avoir accès aux menus créés précédemment.<br>- Affichage de la date de création.<br>- Options pour modifier, supprimer ou créer un menu. | **-React**<br><br>**-Firebase Firestore** | - Affichage des Menus : Créer une vue dans React pour lister tous les menus créés précédemment en récupérant les données depuis Firebase Firestore.<br>- Gestion des Menus : Permettre la modification, suppression, et création de nouveaux menus directement dans Firestore, avec mise à jour en temps réel sur l'interface React. | **1. Gestion en temps réel** : Firestore assure une synchronisation instantanée des données, pour une interface toujours à jour.<br>2. **Expérience Utilisateur Complète** : React avec Firestore offre une vue intégrée efficace pour gérer les menus (création, modification, suppression). |
| Accéder aux mentions légales | - Accès depuis toutes les pages.<br>- Affichage de "Tous droits réservés" sur toutes les pages. | **-react-modal** | Utilisation de **react-modal** pour afficher les mentions légales dans une modale accessible depuis chaque page. | 1) Cohérent avec l'implémentation d'autres modales dans l'application.<br>2) Maintient une expérience utilisateur cohérente. |
| Accéder aux tarifs de Menu Maker. | - Accès via un lien ouvrant un nouvel onglet.<br>- URL à créer côté Qwenta. | **lien "Tarifs"** sur MenuMaker qui redirige vers la page des tarifs de Qwenta. | 1. **Création du Lien**: Ajouter un onglet ou bouton "Tarifs" sur MenuMaker.<br>2. **Redirection**: Le lien ouvre la page des tarifs dans un nouvel onglet en suivre la logique (`https://URL_DE_QWENTA/tarifs/menumaker`). | 1. **Simplicité**: Permet un accès direct aux tarifs via un lien simple.<br>2. **Accessibilité**: Ouvre la page des tarifs dans un nouvel onglet pour ne pas interrompre l'expérience sur MenuMaker. |

| | | | | |
|---|---|---|---|---|
| Exporter le menu vers Deliveroo. | Le restaurateur doit être redirigé vers l'application Deliveroo. | **-API Deliveroo** | 1. Ajouter un bouton "Diffuser sur Deliveroo" dans la section "Exportez et diffusez".<br>2. Lors du clic, pouvoir exporter le menu et être redirigé sur l'application Deliveroo. | 1. **Automatisation**: L'intégration de l'API simplifie le processus d'exportation et réduit les erreurs manuelles.<br>2. **Expérience Utilisateur Optimisée**: Permet une exportation fluide et directe du menu vers Deliveroo, améliorant l'efficacité du restaurateur. |
| Partager le menu sur Instagram. | - Affichage du bouton dans la catégorie "Exportez et diffusez".<br>- Générer des images du menu au format carré pour Instagram.<br>- Rediriger l'utilisateur vers Instagram avec les images prêtes. | **-API Instagram Graph**<br><br>**-Firebase Storage** | 1. **Génération d'images**: L'utilisateur clique sur "Partager sur Instagram", ce qui déclenche l'appel à l'API pour générer et préparer les images au format carré.<br>2. **Publication**: Les images sont ensuite publiées directement sur le compte Instagram du restaurateur via l'API. | 1. **Intégration directe**: L'API Instagram permet une publication directe, simplifiant le processus de partage pour l'utilisateur.<br>2. **Automatisation**: L'utilisation de l'API permet d'automatiser la génération et le partage des images, réduisant les étapes manuelles. |
| Déconnexion du compte. | - Déconnexion disponible depuis toutes les pages connectées. | **- Firebase Authentication** | 1. **Bouton de Déconnexion**: Ajouter un bouton "Se déconnecter" visible sur toutes les pages.<br><br>2. Utiliser Firebase pour gérer la déconnexion sécurisée et React pour l'interface. | 1. **Sécurité** : Firebase Authentication assure une gestion sécurisée des sessions utilisateur. 2. **Intégration fluide** : Firebase s'intègre parfaitement avec React pour une expérience utilisateur fluide. |

| | | | | |
|---|---|---|---|---|
| Modification des informations utilisateur. | - Lier plusieurs adresses e-mail au compte.<br>- Modifier l'adresse e-mail de base. | **-Firebase Authentication** | Ajouter des fonctionnalités dans l'interface utilisateur pour lier plusieurs adresses e-mail et modifier l'adresse e-mail de base. Gérer ces actions via Firebase Authentication. | 1. **Gestion flexible des utilisateurs** : Firebase permet de gérer facilement plusieurs adresses e-mail par utilisateur.<br>2. **Sécurité** : Firebase Authentication garantit une gestion sécurisée des informations sensibles. |
| Accès à un dashboard regroupant les fonctionnalités principales. | - Affichage des options de création, diffusion, impression de menu.<br>- Affichage des derniers articles du blog. | **Firebase Firestore + React** | - Stockage et récupération des données du dashboard en temps réel via Firestore, gestion de l'interface avec React. | 1. Performance en temps réel : Firestore synchronise et affiche instantanément les données.<br>2. Simplicité de gestion des données : Firestore facilite le stockage structuré des éléments du dashboard. |
| Création du branding du restaurant. | -Ajouter/Modifier/ Supprimer le logo.<br><br>-Ajouter/Modifier/Supprimer les couleurs de base. | **-Firebase Storage**<br><br>**-Firestore.** | Utiliser Firebase Storage pour stocker les fichiers de logo et Firestore pour gérer les informations de branding (comme les couleurs de base). L'interface utilisateur est gérée par React pour faciliter ces actions. | 1. **Gestion centralisée** : Firebase facilite le stockage et la modification des éléments de branding.<br>2. **Mise à jour instantanée** : Les changements sont immédiatement visibles dans l'application. |

## II. Liens avec le back-end

- Quel langage pour le serveur ?

  ***Node.js** pour le langage côté serveur. Node.js offre plusieurs avantages dans ce contexte :*

- **Cohérence JavaScript :** Il permet d'utiliser le même langage (JavaScript) aussi bien en frontend qu'en backend, ce qui peut améliorer l'efficacité de développement.
- **Excellente intégration avec Firebase :** Node.js dispose de robustes SDKs et bibliothèques Firebase.
- **Performance :** Node.js est bien adapté pour gérer les applications en temps réel, ce qui correspond à l'utilisation de Firebase Firestore.
- **Écosystème étendu :** Accès à un vaste nombre de packages npm pour des fonctionnalités supplémentaires.

- A-t-on besoin d'une API ? Si oui laquelle ?

  Une API serait bénéfique pour ce projet. Créer une API RESTful en utilisant Express.js, qui fonctionne parfaitement avec Node.js servirait plusieurs objectifs :

- Agir comme intermédiaire entre le frontend et les services Firebase, ajoutant une couche de sécurité supplémentaire.
- Gérer les opérations côté serveur qui ne devraient pas être exposées au client.
- S'intégrer avec des services externes comme les API de Deliveroo et Instagram.
- Gérer toute logique métier complexe qui ne devrait pas être traitée côté client.

- Base de données choisie :

  NoSQL. Ce choix s'aligne parfaitement avec les exigences du projet :

- **NoSQL (Firestore) :** Offre une flexibilité dans la structure des données, ce qui est bénéfique pour un projet où les éléments de menu et les données de restaurant peuvent varier considérablement.
- **Capacités en temps réel :** Firestore fournit une synchronisation de données en temps réel, utile pour des fonctionnalités comme les mises à jour de menu en direct.
- **Scalabilité :** Firestore peut gérer efficacement la croissance des données et des utilisateurs.
- **Intégration :** S'intègre parfaitement avec les autres services Firebase utilisés dans le projet.

## III. Préconisations concernant le domaine et l’hébergement

- Nom du domaine : qwentamenumaker.com
- Nom de l’hébergement : Hostinger
- Adresses e-mail :
  info@menumaker.com
  support@menumaker.com

## IV. Accessibilité

**ARIA labels :** Elles fournissent des informations essentielles aux utilisateurs de lecteurs d'écran et autres technologies d'assistance, leur permettant de comprendre et d'interagir efficacement avec le contenu web.

**HTML sémantique :** L'utilisation d'éléments HTML appropriés (titres, listes, paragraphes, etc.) apporte structure et signification aux technologies d'assistance.

**Navigation au clavier :** Permet de naviguer et d'interagir avec tous les éléments à l'aide du clavier

**Contraste des couleurs :** Maintenez un contraste de couleur suffisant entre le texte et l'arrière-plan pour une bonne lisibilité.

- Compatibilité navigateur :
  Google Chrome (**Version 129.0.6668.71 (Official Build) (64-bit)** )
  Mozilla Firefox
  Apple Safari

- Types d’appareils.
  Desktop

## V. Recommandations en termes de sécurité

- Exiger des mots de passe complexes (au moins 12 caractères, incluant majuscules, minuscules, chiffres et caractères spéciaux).
- Limiter les tentatives de connexion infructueuses et implémenter un système de verrouillage temporaire du compte après plusieurs échecs.
- Utilisez les jetons d'authentification Firebase pour la gestion des sessions.
- Configurez les temps d'expiration des jetons d'authentification Firebase.
- Utiliser HTTPS pour toutes les communications.
- Maintenir un inventaire à jour de tous les plugins et bibliothèques utilisés.
- Effectuer des audits de sécurité réguliers des dépendances (par exemple, avec npm audit pour les packages Node.js).
- Mettre à jour régulièrement les plugins et les dépendances vers les dernières versions stables.
- N'utiliser que des plugins de sources fiables et bien maintenues.
- Mettre en place un système de surveillance en temps réel pour détecter les activités suspectes.

## VI. Maintenance du site et futures mises à jour

**Mises à jour régulières:**

- Mise à jour régulière des bibliothèques et frameworks utilisés (Node.js, React, Firebase, etc.) pour bénéficier des dernières fonctionnalités et améliorations de sécurité.
- Correctifs de bugs**:** Identification et correction rapide des anomalies signalées par les utilisateurs ou détectées lors de tests.

**Documentation:**

- Maintenance d'une documentation technique complète et à jour (code, architecture, procédures).
- Documentation des processus de déploiement et de maintenance.

**Performance** :

- Optimisation du code pour réduire les temps de chargement et améliorer la réactivité de l'application.
- Suivi des évolutions technologiques et adaptation de l'application en conséquence (nouvelles versions de frameworks, nouvelles API).
- Développement de nouvelles fonctionnalités répondant aux besoins utilisateurs.