Spécifications techniques
Menu Maker Qwenta

Version	Auteur	Date	Approbation
1.0	Tiphanie, Webgencia	25/09/2024	John, Qwenta

SOMMAIRE

I. CHOIX TECHNOLOGIQUES

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Accès utilisateur : la landing page doit être accessible aux utilisateurs non connectés.	La landing page doit être accessible sans connexion, tout en étant attrayante pour inciter à utiliser le Menu Maker.	**React router**	Une route publique est créée pour la landing page via React Router, permettant à tous les utilisateurs, connectés ou non, d'y accéder. Des sections visuellement engageantes encouragent à découvrir le Menu Maker.	1. **Accessibilité** : Utiliser React Router permet de gérer facilement les routes publiques, garantissant un accès sans authentification. 2. **Attractivité** : Une page attrayante encourage l'utilisateur à s'engager.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Structuration du contenu de la landing page	La landing page doit contenir les sections : bannière, personnalisation du menu, et explications étape par étape. Ces sections doivent être visibles sans authentification.	Structure avec des composants **React** pour chaque section	Utilisation de composants React indépendants pour chaque section : - **Bannière** en haut de la page - **Personnalisation du menu** avec des options visibles - **Explications étape par étape** guidant l'utilisateur à travers le processus. Ces composants sont organisés de manière logique et sont accessibles sans besoin de connexion.	1. **Modularité** : Les composants React permettent de réutiliser et de maintenir facilement chaque section individuellement. 2. **Visibilité** : Les sections sont accessibles à tous les utilisateurs, garantissant une navigation fluide et intuitive.
Authentification	L'utilisateur doit pouvoir entrer son adresse mail dans la modale, qu'il soit déjà inscrit ou non.	**Firebase authentification**	La modale d'authentification permet à l'utilisateur d'entrer son adresse e-mail. Si l'utilisateur est déjà inscrit, un e-mail est envoyé pour l'authentifier. Sinon, un e-mail de confirmation est envoyé pour créer un compte.	1. **Simplicité** : La gestion des comptes utilisateurs est centralisée grâce à Firebase, simplifiant le processus d'inscription et d'authentification. 2. **Flexibilité** : Permet de gérer les utilisateurs déjà inscrits ainsi que les nouveaux utilisateurs.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Fenêtre de connexion sous forme de modale	La fenêtre de connexion doit s'ouvrir sous forme de modale d'authentification intégrée à la landing page Un lien besoin d'aide doit apparaitre en bas de la modale	**React Modal** pour la modal et **Firebase authentification** pour la connexion	La modale d'authentification s'ouvre lorsqu'un utilisateur clique sur le bouton de connexion. Elle inclut un champ pour l'adresse e-mail et un lien "Besoin d'aide" permettant d'envoyer un e-mail au support. Firebase est une base de donnée sur le cloud qui nous permet de nous abstraire de toute configuration	1) une connexion rapide et facile 2) Une connexion sécurisée
Assistance utilisateur avec le lien besoin d'aide	Un lien « Besoin d'aide » doit être disponible dans la modale, permettant d'envoyer un email directement aux équipes de support Lien de contact dans la modale pour le support	**React-email**	Un lien est intégré dans la modale d'authentification qui, lorsqu'il est cliqué, ouvre le client de messagerie de l'utilisateur avec un e-mail prérempli adressé à l'équipe de support Qwenta, grâce à la bibliothèque React Email.	1. **Accessibilité** : Permet aux utilisateurs d'obtenir rapidement de l'aide en un clic, améliorant l'expérience utilisateur. 2. **Efficacité** : Facilite la communication entre les utilisateurs et l'équipe de support, réduisant le temps nécessaire pour résoudre les problèmes.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Gestion des comptes utilisateurs	Le système doit permettre la création de comptes multiples. Chaque utilisateur doit avoir accès à son propre compte et à ses données.	**Firebase authentification**	Firebase gère les comptes utilisateurs via authentification avec email/mot de passe ou via OAuth (Google, Facebook). Chaque utilisateur a un identifiant unique pour accéder à ses données et gérer ses comptes (créer, modifier, supprimer).	1. **Sécurité** : Firebase Authentication assure une gestion sécurisée des utilisateurs avec des options de connexion fiables et reconnues (OAuth). 2. **Scalabilité** : La gestion des comptes multiples est facilitée par Firebase, permettant d'agrandir l'application sans complexité supplémentaire.
Stockage sécurisé des données	Le système doit permettre de stocker les menus et les informations clients (comptes restaurateurs) dans un environnement sécurisé.	**Firebase Firestore** et **Firebase Security Rules**	Utilisation de Firebase Firestore pour stocker les données sensibles (comptes utilisateurs, menus) avec des Firebase Security Rules pour restreindre l'accès selon l'authentification et les permissions des utilisateurs.	1. **Sécurité** : Les Firebase Security Rules assurent une protection des données en limitant l'accès en fonction de l'état d'authentification et des permissions. 2. **Conformité** : Le stockage sécurisé des données aide à respecter les normes de protection des données comme le RGPD, garantissant la confidentialité des utilisateurs.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Modale de confirmation et envoi d'un email	Une modale de confirmation doit s’ouvrir et un email automatique doit être envoyé à l'utilisateur pour l'authentification ou la confirmation de son adresse mail.	**React Modal** et **Firebase**	Utilisation de **React Modal** pour afficher une modale de confirmation. **Firebase Authentication** envoie automatiquement un email de vérification ou de réinitialisation, selon le cas (authentification ou première connexion).	1. **Expérience utilisateur** : La modale améliore l'interaction en confirmant visuellement l'action et l'envoi de l'email, renforçant la transparence du processus. 2. **Automatisation** : Firebase gère automatiquement l'envoi des emails, réduisant la charge de développement et assurant une communication efficace avec l'utilisateur.
Dashboard pour restaurateurs	Accès à un dashboard regroupant la création, diffusion, et impression de menu.	**React js, redux**	Création d'un tableau de bord avec les fonctionnalités clés et les derniers articles de blogs en rapport à leur activité.	1. **Centralisation** : Regroupe toutes les fonctionnalités essentielles en un seul endroit pour une gestion simplifiée. **Facilité d'utilisation** : Améliore l'expérience utilisateur avec un accès intuitif aux outils du dashboard.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Encart "Imprimer un menu"	L'encart "Imprimer un menu” doit être visible sur le dashboard	Création d'un composant **React** pour l'encart	Un composant React sera placé sur le **dashboard** pour permettre un accès rapide à la fonction d'impression.	1. **Accès rapide à la fonctionnalité** : Un encart visible sur le dashboard permet de retrouver facilement l'option d'impression. 2. **Amélioration de l'efficacité** : Cela centralise les actions importantes dans le dashboard, facilitant la gestion pour le restaurateur.
Création de catégorie	Le restaurateur doit pouvoir créer une nouvelle catégorie. S'assurer que le formulaire de création inclut des validations appropriées (ex: nom unique).	Utilisation de **React-modal** pour la création de catégories avec validations	Création d'une modale qui contient un formulaire avec validation (par exemple, vérification de l'unicité du nom, pour chaque restaurateur).	1. **Prévention des erreurs** : La validation des champs réduit le risque de doublons ou d'erreurs dans les catégories créées, pour chaque restaurateur. 2. **Fluidité de l'interface** : La modale permet de ne pas interrompre la navigation tout en assurant une interface intuitive.
Création de plats	Le restaurateur doit pouvoir créer autant de plats qu'il le souhaite dans la catégorie sélectionnée. La liste des plats doit être visible et modifiable.	Utilisation de **Redux** pour gérer l’état de l'application (catégories, plats) et permettre une gestion fluide et centralisée.	Mise en place d'une fonctionnalité permettant d'ajouter plusieurs plats dans une catégorie via une liste dynamique et réactive.	1. **Extensibilité** : Permet de gérer un nombre illimité de plats par catégorie sans limitations techniques. 2. **Optimisation de l'état global** : Redux assure une gestion efficace des données des plats au sein de l'application.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Détails du plat	Chaque plat doit pouvoir inclure : - Une photo associée - Un nom - Un prix - Une description	Intégration d'un **formulaire avec champ pour uploader une photo.** Ajout d'une barre avec **react-custom-scroller** pour défiler et voir toutes les informations	Grâce à la modale le restaurateur pourra ajouter la photo, le nom, le prix ainsi qu'une description du plat. La barre de défilement permettra de voir toutes les informations dans un formulaire long.	1. **Facilité d'utilisation** : Permet au restaurateur d'ajouter toutes les informations nécessaires dans un formulaire intuitif avec une expérience fluide. 2. **Gestion des médias** : Le champ d'upload de photo permet de gérer les visuels directement dans le formulaire, sans quitter la modale.
Sauvegarde continue et en temps réel	Sauvegarde continue et en temps réel des modifications effectuées Historique des changements pour récupérer les versions précédentes si nécessaire.	Utilisation de **Firestore** pour le stockage en temps réel	Chaque modification apportée au menu est sauvegardée automatiquement dans Firestore, permettant une récupération facile des versions antérieures.	1. Assure la sécurité des données en évitant la perte d'informations. 2. Permet de restaurer facilement une version précédente en cas d'erreur.
Personnalisation de la typographie	Le restaurateur doit pouvoir choisir et changer directement la typographie et la couleur.	Utilisation de **react-select** pour la typographie et **react-color** pour la couleur.	Intégration d'un composant `react-select` pour permettre la sélection de la typographie et d'un sélecteur de couleur (`react-color`) dans la même modale.	1. Améliore l'expérience utilisateur en offrant plus de flexibilité. 2. Permet aux restaurateurs de personnaliser rapidement l'apparence de leur menu sans difficulté.
Exportation des menus en pdf	Le système doit permettre au restaurateur de télécharger le fichier PDF correspondant à son menu en un clic via un bouton de téléchargement	Utilisation de **react-pdf**	Intégration de la bibliothèque `react-pdf` pour générer et télécharger le fichier PDF correspondant au menu directement depuis l'application.	. Simplifie le processus d'exportation pour les restaurateurs. 2. Améliore l'efficacité en permettant un téléchargement en un seul clic.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Commande d'impression en un clic	Le système doit permettre au restaurateur d'imprimer son menu en un seul clic, sans passer par une interface de configuration complexe. Le PDF doit être formaté pour l'impression.	Développer une route dans **Node.js** qui prépare et renvoyer le PDF à imprimer, intégration d'un bouton d'impression dans **React.**	Intégration d'une route dans Node.js qui génère le PDF formaté pour l'impression. Utilisation d'un bouton React qui appelle cette route et active la fonctionnalité d'impression du navigateur.	1. Simplifie le processus d'impression pour le restaurateur. 2. Assure que le menu soit imprimé avec un format et une mise en page appropriés.
Diffusion sur Deliveroo et Instagram	Le restaurateur doit pouvoir partager ses menus directement sur Deliveroo et Instagram via des API.	Utiliser **l'API de Deliveroo** pour soumettre les menus, et **l'API Instagram Graph** pour publier des photos et descriptions de menus.	Intégration de l'API Deliveroo pour envoyer les menus et infos du restaurant. Utilisation de l'API Instagram Graph pour publier du contenu (menus, images).	1. Permet aux restaurateurs d'atteindre plus facilement leurs clients en ligne. 2. Automatise la diffusion du menu sur plusieurs plateformes.
Déconnexion de l'utilisateur	Le restaurateur doit pouvoir se déconnecter depuis n'importe quelle page connectée.	Utilisation de **local storage** pour gérer la déconnexion	Utilisation du local storage pour stocker l'état de connexion de l'utilisateur, permettant de se déconnecter facilement depuis n'importe quelle page.	. Améliore la sécurité des comptes utilisateurs en permettant une déconnexion simple. 2. Facilite l'expérience utilisateur en offrant une déconnexion rapide et intuitive.
Modification des informations utilisateur	Le restaurateur doit pouvoir modifier son adresse e-mail de base et lier plusieurs adresses e-mail à son compte.	Utilisation de **Firebase Authentication** et **Firestore.**	- Créer un formulaire pour la modification de l'adresse e-mail principale. - Ajouter la fonctionnalité pour lier plusieurs adresses e-mail via un autre formulaire.	1. **Firebase Authentication** gère les comptes utilisateurs de manière sécurisée, permettant des mises à jour simples des informations. 2. L'utilisation de **Firestore** pour stocker les adresses e-mail supplémentaires permet une gestion facile et dynamique des données.

Besoin technique	Contraintes	Solution	Description de la solution	Justification
Intégration des tarifs	L'internaute doit pouvoir accéder aux tarifs via un lien situé en haut à droite de la landing page.	Ajouter un **lien vers une page** dédiée aux tarifs dans la navigation.	ntégrer un lien visible à côté du bouton "Se connecter", menant directement à une page qui affiche les tarifs actuels et détaillés.	1. Offre un accès rapide et clair aux informations tarifaires. 2. Permet de gérer facilement la mise à jour des prix à travers une page dédiée.
Intégration des informations légales	Le système doit afficher les mentions légales et indiquer les droits d'auteur sur toutes les pages. - Lien "Mentions légales" visible sur toutes les pages. - Affichage des informations légales dans une modale. - Mention "Tous droits réservés" visible en bas de page sur toutes les pages.	Utilisation de **React Modal** pour afficher une modale avec les informations légales. Ajout d'un composant **React Footer** incluant le lien "Mentions légales" et le texte "Tous droits réservés".	- Créer une modale React en utilisant **React Modal** qui s'affiche lorsqu'on clique sur le lien "Mentions légales". - Le footer doit être un composant global, réutilisable sur toutes les pages. - Le texte "Tous droits réservés" est intégré en permanence au bas de chaque page.	1. Utilisation d'une modale pour un accès rapide aux informations légales, sans redirection. 2. Le composant Footer centralisé permet une gestion cohérente du lien et du texte sur toutes les pages.

II. LIENS AVEC LE BACK END

• QUEL LANGAGE POUR LE SERVEUR ?

Le serveur utilisera **Node.js** avec **Express.js** pour créer les routes API nécessaires à la gestion des utilisateurs et des sessions. **Firebase Authentication** sera utilisé pour gérer l'authentification des utilisateurs, en restant cohérent avec la bibliothèque front-end **React** (basée sur **JavaScript**). Cela assure une communication fluide entre le front-end et le back-end, facilitée par l'utilisation de JavaScript des deux côtés.

• A-T-ON BESOIN D'UNE API ? SI OUI LAQUELLE ?

Oui, nous avons besoin de plusieurs API pour notre projet :

1. **Firebase** : Nous utiliserons Firebase pour gérer l'authentification des utilisateurs et le stockage des données, y compris les fichiers PDF. Firebase nous offre des fonctionnalités robustes pour la gestion des utilisateurs et le stockage sécurisé des informations.
2. **Instagram Graph API** : Cette API sera nécessaire pour permettre aux restaurateurs de diffuser rapidement leur menu sur leurs comptes Instagram.
3. **Deliveroo API** : Nous devrons également contacter Deliveroo pour accéder à leur API publique, ce qui permettra d'intégrer la fonctionnalité de diffusion des menus sur leur plateforme.

• BASE DE DONNÉES CHOISIE :

Nous avons choisi Firestore, un service de Firebase, comme base de données NoSQL. Ce choix est motivé par sa flexibilité dans la gestion des données, sa capacité à évoluer avec un grand volume d'utilisateurs, et ses fonctionnalités de mise à jour en temps réel. Firestore est idéal pour gérer les informations des utilisateurs, les catégories de menus et les plats, tout en offrant une expérience utilisateur fluide.

III. PRÉCONISATIONS CONCERNANT LE DOMAINE ET L'HÉBERGEMENT

- Nom du domaine : menumaker.com

Nous avons choisi un nom de domaine clair et mémorable pour renforcer l'identité de marque et faciliter l'accès des utilisateurs aux services de Menu Maker.

- Nom de l'hébergement : Pour l'hébergement, on peut opter pour **Firebase Hosting**. Ce service est idéal pour les applications web, offrant un déploiement rapide, une intégration facile avec les autres services Firebase, et une gestion simplifiée des ressources. Il garantit également une sécurité optimale et une mise à l'échelle automatique en fonction de la demande.

- Adresses e-mail : contact@menumaker.com ou support@menumaker.com pour entrer en contact direct avec l'équipe de Qwenta et info@menumaker.com pour demander des informations plus générales

IV. ACCESSIBILITÉ

- Compatibilité navigateur.

Le site doit être compatible avec les dernières versions de Chrome, Safari et Firefox. L'application devra également être accessible au minimum : elle doit être navigable depuis le clavier et lisible par un lecteur d'écran, garantissant ainsi une expérience utilisateur inclusive pour toutes les personnes, y compris celles ayant des besoins spécifiques en matière d'accessibilité.

- Types d'appareils.

Écran d'ordinateur, la version mobile n'est pas demandé.

V. RECOMMANDATIONS EN TERMES DE SÉCURITÉ

Gestion des accès : Utiliser l'authentification par Firebase pour sécuriser l'accès aux comptes utilisateurs. Il est essentiel d'appliquer des rôles et des permissions appropriés pour chaque type d'utilisateur afin de limiter l'accès aux informations sensibles.

Mettre en place les recommandations OWASP

S’assurer du respect du RGPD

Mises à jour régulières : S'assurer que tous les plugins et dépendances utilisés sont régulièrement mis à jour pour éviter les vulnérabilités de sécurité connues.

HTTPS : Utiliser HTTPS pour chiffrer les données échangées entre le client et le serveur, protégeant ainsi les informations personnelles des utilisateurs.

Validation des entrées : Mettre en place une validation rigoureuse des données côté serveur pour éviter les attaques par injection et garantir que seules les données valides soient acceptées.

Sauvegardes régulières : Effectuer des sauvegardes fréquentes des données afin de pouvoir restaurer rapidement le système en cas d'incident de sécurité.

Tests de sécurité : Réaliser régulièrement des tests de sécurité, tels que des tests d'intrusion et des analyses de vulnérabilité, pour identifier et corriger les failles potentielles de l'application, assurant ainsi la protection des données des utilisateurs et la conformité aux normes de sécurité.

VI. MAINTENANCE DU SITE ET FUTURES MISES À JOUR

La maintenance du site sera essentielle pour assurer sa performance et sa sécurité continues. Voici les grandes lignes du contrat de maintenance :

1. **Mises à jour régulières** : Effectuer des mises à jour fréquentes du système, y compris des mises à jour de sécurité, des plugins et des dépendances, pour garantir que le site reste protégé contre les menaces connues.

2. **Support technique** : Fournir un support technique pour résoudre rapidement les problèmes techniques ou les bugs signalés par les utilisateurs, garantissant ainsi une expérience utilisateur optimale.

3. **Sauvegardes des données** : Mettre en place un système de sauvegarde régulier des données pour minimiser la perte d'informations en cas de problème technique ou d'incident de sécurité.

4. **Surveillance de la performance** : Analyser régulièrement la performance du site pour identifier les opportunités d'amélioration et optimiser la vitesse de chargement et la réactivité.

5. **Planification des mises à jour fonctionnelles** : Prévoir des mises à jour fonctionnelles pour intégrer de nouvelles fonctionnalités et répondre aux évolutions des besoins des utilisateurs et des tendances du marché.

6. **Documentation à jour** : Maintenir la documentation technique à jour pour garantir que toutes les modifications et mises à jour sont correctement enregistrées, facilitant ainsi la compréhension et l'entretien futur du site.

7. **Rapports réguliers** : Fournir des rapports réguliers sur l'état du site, y compris les mises à jour effectuées, les problèmes résolus et les recommandations pour l'amélioration continue.