Spécifications techniques

MENU MAKER by QWENTA

Version	Auteur	Date	Approbation
1.0	Hugo Pradier	27/08/2024	QWENTA

I. Choix technologiques 2

II. Liens avec le back-end 3

III. Préconisations concernant le domaine et l'hébergement 3

IV. Accessibilité 3

V. Recommandations en termes de sécurité 3

VI. Maintenance du site et futures mises à jour 4

I. Choix technologiques

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

Besoin	Contraintes	Solution	Description de la solution	Justification
UX / UI	L'integration de nouvelles bibliothèques doit être landreete ae existante	**SASS / TAILWIND CSS**	Utiliser SASS pour des styles modulaires et variables, et Tailsind CSS pour l'application rapide de style utilitaires	SASS : Permet une gestion efficace des styles via des variables, mixios et nesting facilitant les modifications globales Tailwind: Offre une approche utilitaire-first pour des styles rapides et uniformes.
Création d'une catégorie de menu	L'ajout d'une catégorie doit pouvoir 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) Cohérente avec le choix de React 2) Très utilisée et maintenue.
Inscription par e-mail (frontend)	Formulaire de saisie d'e-mail validable avec un lien "Besoin d'aide" fonctionnel.	**Formik** & **Yup**	**Formik** gère l'état et la soumission des formulaires de manière efficace, et **Yup** permet la validation des formulaires.	1) Efficace pour la gestion de formulaires. 2) Bonne intégration avec React.
Inscription par e-mail (backend)	Mail envoyé pour l'authentification et la confirmation.	**Nodemailer** & **JWT**	**Nodemailer** envoie des mails facilement, et **JWT** assure l'authentification sécurisée.	1) **Nodemailer** est simple et efficace. 2) **JWT** est sécurisé et standardisé pour l'authentification.
Ajout de plats au menu (frontend)	L'utilisateur doit pouvoir ajouter, nommer, décrire, tarifier et ajouter des photos de plats.	**Formik** & **React Dropzone**	**Formik** pour la gestion du formulaire et **React Dropzone** pour les uploads de photos.	1) **Formik** est performant pour les formulaires. 2) **React Dropzone** gère bien les uploads.
Ajout de plats au menu (backend)	Les plats doivent être enregistrés et récupérés depuis la base de données.	**Express**, **Multer** & **Mongoose**	**Express** pour l'API, **Multer** pour le upload des photos, et **Mongoose** pour la gestion des données dans MongoDB.	1) **Express** est léger et rapide. 2) **Mongoose** offre une intégration parfaite avec MongoDB.
Partage sur Deliveroo	Le partage du menu sur Deliveroo doit être sécurisé et automatique.	**Deliveroo API**	Intégration de l'API Deliveroo pour le partage des menus, gérant l'authentification et les requêtes de partage via endpoints sécurisés.	1) API officielle et maintenue par Deliveroo. 2) Officieusement recommandé pour une parfaite compatibilité.
Téléchargement du fichier PDF	Le PDF du menu doit pouvoir être téléchargé en un clic.	**React PDF**	Génération et téléchargement des fichiers PDF du menu de façon rapide et efficace en un clic.	1) Librairie React spécialisée. 2) Grand niveau de personnalisation et de performance.
Typographie et couleurs personnalisées du menu	L'utilisateur doit pouvoir choisir la typographie et les couleurs de son menu.	**Google Fonts API** & Palette	Utilisation de l'API Google Fonts pour les typographies et une palette de couleurs pour la sélection des couleurs du menu.	1) **Google Fonts API** offre une grande variété de typographies. 2) Palette de couleurs simples à implémenter.
Visualisation en temps réel du menu créé	Les utilisateurs doivent voir un aperçu en temps réel de leur menu.	**Preview Component**	Un composant React d'aperçu qui écoute les changements d'état de Redux ou Context API pour un retour immédiat sur les modifications du menu.	1) Interaction fluide et immédiate. 2) Technologie déjà utilisée dans le projet.
Connexion et Déconnexion	Les utilisateurs/restaurateurs peuvent se connecter et se déconnecter depuis n'importe quelle page.	**JWT** (JSON Web Tokens)	**JWT** pour gérer les sessions utilisateur de manière sécurisée, facilement déployable avec des middlewares d'Express.js pour les routes de connexion et déconnexion.	1) Sécurité renforcée. 2) Reconnue et adoptée par la communauté.

Authentification tierce partie	Permettre aux utilisateurs de se connecter via Google ou Facebook.	OAuth2 (**Google API**, **Facebook API**)	Intégration des APIs OAuth2 pour Google et Facebook, permettant une connexion simple et sécurisée via des comptes existants.	1) Simplifie le processus de connexion pour les utilisateurs. 2) Réduit la friction lors de l'inscription.
Notifications par SMS	Envoyer des SMS de confirmation et notifications aux utilisateurs.	**Twilio API**	Utilisation de l'API Twilio pour l'envoi d'SMS rapide et fiable, permettant de notifier les utilisateurs de manière proactive.	1) Communication rapide et efficace

II. Liens avec le back-end

- **Quel langage pour le backend ?**

Pour le serveur de votre projet Menu Maker, Node.js est un excellent choix pour plusieurs raisons. Voici un tableau résumé de cette recommandation :

Langage	Avantages	Description de la solution	Justification
Node.js (JavaScript côté serveur)	- Non-blocking I/O pour une haute performance - Utilise le même langage pour le frontend et le backend	Node.js est une plateforme qui permet de faire fonctionner JavaScript sur le côté serveur, optimisée pour des applications réseau rapides et scalables.	1) **Cohérence du Stack Technologique** : Utiliser JavaScript à la fois pour le frontend et le backend simplifie le développement et la maintenance. 2) **Performance & Scalabilité** : Node.js est reconnu pour ses performances élevées et sa capacité à gérer de nombreuses connexions simultanément de manière efficace.

Détails et Justifications

1. **Cohérence du Stack Technologique :**
   - **Un seul langage** : Utiliser JavaScript à la fois sur le frontend (React) et sur le backend (Node.js) simplifie le développement. Il n'y a pas de besoin de changer de paradigme ou de contexte, ce qui peut améliorer la productivité des développeurs.
   - **Écosystème commun** : Vous bénéficiez d'un écosystème riche avec des milliers de librairies et modules disponibles via npm (Node Package Manager).
2. **Performance & Scalabilité :**
   - **Non-blocking I/O** : Node.js utilise un modèle d'I/O non bloquant, ce qui permet de traiter de nombreuses requêtes simultanément sans bloquer le serveur, idéal pour les applications web nécessitant une haute performance. ○ **Événements et Callbacks**: Node.js est conçu à partir d'un modèle événementiel, ce qui est efficace pour les applications réactives en temps réel comme les chats, les notifications, et, bien sûr, les menus interactifs.

- **Quels API ?**

Pour votre projet Menu Maker, une API est essentielle. L'API permettra une communication standardisée entre le frontend (interface utilisateur) et le backend (serveur). Voici des exemples d'APIs internes que vous pourriez créer, ainsi que l'utilisation potentielle d'APIs externes pour enrichir certaines fonctionnalités de votre application.

API Interne

Dans le cadre de votre application, vous devrez développer une API RESTful interne pour gérer les différentes opérations CRUD (Create, Read, Update, Delete) liées aux menus, catégories, plats, utilisateurs, etc.

Exemples de Endpoints API Internes

Endpoint	Méthode	Description	Exemple de requête	Justification
/api/users	POST	Créer un nouvel utilisateur	**{ "email": "user@example.com", "password": "12345" }**	Pour gérer l'inscription des utilisateurs.
/api/users/login	POST	Authentifier un utilisateur	**{ "email": "user@example.com", "password": "12345" }**	Pour gérer la connexion des utilisateurs.
/api/menus	GET	Récupérer tous les menus	-	Pour afficher tous les menus disponibles.
/api/menus	POST	Créer un nouveau menu	**{ "title": "Menu de la semaine" }**	Pour permettre aux utilisateurs de créer un menu.
/api/menus/:id	PUT	Mettre à jour un menu existant	**{ "title": "Nouveau titre du menu" }**	Pour permettre la modification d'un menu.
/api/menus/:id	DELETE	Supprimer un menu	-	Pour permettre la suppression d'un menu.
/api/plats	POST	Ajouter un plat à un menu	**{ "menuId": "123456", "plat": {...} }**	Pour ajouter des plats dans un menu spécifique.

API Externe

En complément de votre API interne, vous pourriez intégrer certaines APIs externes pour enrichir les fonctionnalités de votre application.

Besoin	API Externe	Exemple d'utilisation	Justification
Authentification tierce partie	**OAuth2 Google API**, **OAuth2 Facebook API**	Permettre aux utilisateurs de se connecter à l'application via leur compte Google/Facebook.	Simplifie le processus de connexion en utilisant des comptes existants des utilisateurs.
Gestion des SMS	**Twilio API**	Envoyer des SMS de confirmation de commande ou notification aux utilisateurs.	Assure une communication efficace et rapide avec les utilisateurs.
Géolocalisation	**Google Maps API**	Montrer l'emplacement des restaurants sur une carte interactive.	Améliore l'expérience utilisateur avec des fonctionnalités de géolocalisation.

Base de données choisie : MongoDB

Introduction

L'application "Menu Maker" utilise MongoDB comme système de gestion de base de données (SGBD). MongoDB est une base de données NoSQL réputée pour sa flexibilité, sa scalabilité et ses performances. Son adoption permet d'assurer la qualité et la sécurité des données tout en offrant une structure dynamique adaptée aux besoins évolutifs de l'application.

Sécurité et Intégrité

Authentification et Cryptage

- **Stockage des mots de passe** : Les mots de passe des utilisateurs sont hachés à l'aide de l'algorithme bcrypt avant d'être stockés dans la base de données, garantissant leur protection même en cas de fuite.
- **SSL/TLS** : Toutes les communications entre l'application web et la base de données sont chiffrées via SSL/TLS pour protéger les données en transit.

Permissions et Accès

- **Contrôle d'Accès** : MongoDB offre une gestion fine des permissions grâce à son système de rôles basé sur l'accès (RBAC), permettant de restreindre l'accès aux données en fonction des rôles des utilisateurs.
- **Isolation des utilisateurs** : Chaque utilisateur de la base de données dispose des privilèges minimaux nécessaires, suivant le principe du moindre privilège.

Performance et Scalabilité

Optimisations de Performance

- **Indexation** : Les index sont créés sur les champs fréquemment utilisés dans les requêtes (par exemple user_id, menu_id) pour améliorer les performances de recherche.
- **Caches** : Utilisation de caches en mémoire pour accélérer les lectures fréquentes de données.

Scalabilité

- **Sharding** : MongoDB partitionne les collections de données en fragments, supportant une gestion efficace des grands volumes de données et assurant une scalabilité horizontale.
- **Réplication** : MongoDB utilise la réplication pour garantir la disponibilité des données en cas de panne de serveur et pour améliorer la lecture via des nœuds secondaires.

Maintenance et Sauvegarde

- **Sauvegardes Régulières** : Des sauvegardes régulières de la base de données sont programmées pour garantir la récupérabilité des données en cas de panne.
- **Surveillance et Logs** : MongoDB fournit des outils de surveillance et de journalisation pour suivre les performances, les erreurs et autres métriques clés. Ces logs sont régulièrement analysés pour identifier et résoudre les problèmes potentiels avant qu'ils n'affectent l'application.

Conclusion

En choisissant MongoDB comme base de données pour "Menu Maker", nous combinons flexibilité, performance et scalabilité. MongoDB est particulièrement adapté aux applications nécessitant une structure de données dynamique et évolutive. Cette décision s'aligne avec les meilleures pratiques du secteur et garantit que l'application peut croître et s'adapter aux exigences futures tout en assurant une sécurité et une gestion des données optimales. MongoDB surpasse les bases de données SQL traditionnelles dans ces domaines, offrant des avantages cruciaux pour le projet Menu Maker by QWENTA.

III. Préconisations concernant le domaine et l'hébergement

Préconisations concernant le domaine et l'hébergement

1. **Domaine**: Namecheap **Pourquoi Namecheap ?** ○ **Fiabilité et Popularité**: Namecheap est l'un des fournisseurs de domaines les plus populaires et les plus dignes de confiance.
    - **Coût**: Offre des prix compétitifs et des promotions régulières.
    - **Facilité d'utilisation**: L'interface utilisateur est conviviale, rendant l'achat et la gestion du domaine simples. ○ **Services supplémentaires**: Fournit des services complémentaires utiles comme le WHOISGuard pour protéger la confidentialité.
2. **Hébergement Frontend**: Vercel **Pourquoi Vercel ?** ○ **Performance**: Vercel est optimisé pour les déploiements rapides et les sites performants. Utilise un CDN global pour la distribution rapide du contenu.
    - **Focus sur React**: Très bien intégré avec Next.js, ce qui facilite le travail avec des applications React. ○ **Déploiement Facile**: Très simplifié avec des push Git. Permet des prévisualisations de chaque branche.
    - **Scalabilité**: S'adapte instantanément aux pics de trafic, ce qui est essentiel pour une application en production. 3. **Hébergement Backend**: Heroku **Pourquoi Heroku ?** ○ **Facilité de configuration**: Permet un déploiement rapide avec peu de configuration initiale, idéal pour le développement rapide.
    - **Support pour Node.js**: Excellente compatibilité avec les applications Node.js, le choix backend pour ce projet. ○ **Add-ons**: Large marché d'add-ons pour divers besoins comme bases de données, monitoring, etc.
    - **Scalabilité et Flexibilité**: Permet facilement le changement d'échelle selon les besoins de l'application sans interruption de service.
    - **Intégration CI/CD**: Intégration continue simplifiée avec des outils comme GitHub Actions pour automatiser les déploiements.

Considérations Techniques

1. **Domaine**:
    - **Redirection HTTPS**: Namecheap permet la gestion des DNS et de la redirection HTTPS via Let's Encrypt, garantissant une connexion sécurisée.
    - **Configuration DNS**: Simple à configurer, ce qui minimise les erreurs humaines et simplifie la gestion multi-domaines si nécessaire.
2. **Frontend (Vercel)**:
    - **Actions Automatisées**: Avec un push sur GitHub, Vercel déploie automatiquement les changements, améliorant ainsi l'efficacité et réduisant le risque d'erreurs manuelles. ○ **CDN Intégré**: Le CDN intégré de Vercel assure un chargement rapide des ressources, crucial pour une bonne expérience utilisateur.
3. **Backend (Heroku)**:
    - **Logs et Monitoring**: Heroku offre des outils intégrés de logging et de surveillance, facilitant la détection et la résolution des problèmes.
    - **Environnements de Prévisualisation**: Support pour les environnements de stagging et pré-production pour tester les fonctionnalités avant déploiement en prod.

Avantages Généraux

- **Sécurité**: Utiliser HTTPS par défaut pour toutes les communications.
- **SEO**: La performance et la sécurité améliorent le SEO, boostant ainsi la visibilité du site.

- **Scalabilité**: Les services choisis permettent une montée en charge facile, important pour le succès et la croissance future du site.
- **Support et Documentation**: Les services Vercel et Heroku sont bien documentés, et bénéficient d'un support communautaire et commercial de qualité.

IV. Accessibilité

L'accessibilité est un aspect crucial du développement web moderne, visant à rendre les sites et applications utilisables par tous, y compris les personnes souffrant de handicaps. Voici une explication détaillée des spécificités de l'accessibilité pour le projet "Menu Maker", couvrant les meilleures pratiques et les normes à respecter.

Spécificités de l'accessibilité

Normes et Méthodes

1. **WCAG (Web Content Accessibility Guidelines)**:
    - **WCAG 2.1**: Les WCAG 2.1 sont les lignes directrices les plus récentes définies par le W3C pour rendre le contenu web plus accessible.
    - **Niveaux de Conformité**:
        - **A**: Niveau de base, répond aux besoins essentiels d'accessibilité.
        - **AA**: Niveau recommandé, répond à la plupart des exigences en matière d'accessibilité.
        - **AAA**: Niveau avancé, pour une accessibilité optimale.

Principes Fondamentaux

1. **Perceivable**: L'information et les composants de l'interface utilisateur doivent être présentés de manière à ce que les utilisateurs puissent les percevoir.
    - **Textes Alternatifs (Alt Text)**: Utilisation de descriptions textuelles pour les images.
    - **Contraste Couleurs**: Assurer un contraste suffisant entre le texte et les arrière-plans (ratio de contraste minimum recommandé par WCAG AA est 4,5:1).
2. **Operable**: Les composants de l'interface et la navigation doivent être utilisables.
    - **Navigation par Clavier**: Toutes les fonctionnalités doivent être accessibles via un clavier. ○ **Temps Suffisant**: Donner aux utilisateurs suffisamment de temps pour lire et utiliser le contenu.
3. **Understandable**: Les informations et l'interface utilisateur doivent être compréhensibles.
    - **Prévisibilité**: Assurez-vous que toutes les interactions utilisateur sont prévisibles et cohérentes. ○ **Instructions Claires**: Fournir des instructions et des commentaires clairs pour les erreurs.
4. **Robust**: Le contenu doit être suffisamment robuste pour pouvoir être interprété de manière fiable par une large variété d'utilisateurs et technologies d'assistance.
    - **Compatibilité avec AT** (Assistive Technologies): Assurez-vous que le site fonctionne bien avec les technologies d'assistance comme les lecteurs d'écran.

Meilleures Pratiques

1. **Utilisation de l'HTML Sémantique**:
    - **Balises ARIA (Accessible Rich Internet Applications)**: Utiliser des rôles et états ARIA pour améliorer l'accessibilité des composants interactifs.
    - **Balises Sémantiques**: Utiliser **header**, **nav**, **main**, **article**, **section**, **footer**, etc., pour une meilleure structure de l'information.
2. **Design Inclusif**:

- **Responsive Design**: Assurez-vous que le site est réactif et fonctionne bien sur différents appareils et tailles d'écran.
- **Taille de Police**: Utiliser des unités relatives (**em**, **rem**) pour permettre aux utilisateurs d'ajuster facilement la taille du texte.

3. **Validations et Messages d'Erreur**:
   - **Formulaires Accessibles**: Étiqueter correctement les champs de formulaire avec des balises **<label>** et fournir des messages d'erreur clairs et utilisables.
   - **Feedback Visuel et Sonore**: Fournir des retours d'information visuels et auditifs appropriés pour les actions utilisateur.

Outils et Ressources

1. **Outils d'Audit d'Accessibilité**:
   - **Google Lighthouse**: Outil intégré dans Chrome DevTools pour vérifier l'accessibilité et proposer des améliorations.
   - **Wave**: Extension Chrome pour analyser et fournir des rapports détaillés sur l'accessibilité de votre site. ○ **axe DevTools**: Extension pour les navigateurs avec des fonctionnalités d'audit d'accessibilité puissantes. 2. **Ressources et Documentation**:
   - **MDN Web Docs**: Documentation complète sur les pratiques d'accessibilité.
   - **WebAIM**: Ressources éducatives et techniques sur l'accessibilité web.
   - **WCAG 2.1**: Texte officiel et explications des normes d'accessibilité.

Actions Concrètes

1. **Audit Initial**: Effectuer un audit initial de l'accessibilité du site avec les outils mentionnés.
2. **Plan d'Amélioration**: Établir un plan pour corriger les problèmes identifiés et améliorer l'accessibilité en continu.
3. **Tests Utilisateur**: Engager des utilisateurs ayant des besoins particuliers (par exemple, personnes aveugles ou malvoyantes) pour tester le site et fournir des retours.
4. **Formation de l'Équipe**: Former l'équipe de développement pour qu'elle intègre les meilleures pratiques d'accessibilité dans son workflow quotidien.

Conclusion

En respectant ces spécificités et en intégrant des pratiques d'accessibilité solides dès la phase de conception, vous vous assurez que le site "Menu Maker" est non seulement conforme aux normes actuelles, mais aussi utilisable par tous, ce qui améliorera l'expérience utilisateur globale et élargira votre audience.

v.Recommandations en termes de sécurité

La sécurité est une priorité majeure pour toute application web, et garantir la protection des données utilisateur et la prévention contre les vulnérabilités est essentiel pour maintenir la confiance des utilisateurs. Voici des recommandations détaillées en termes de sécurité pour l'application web "Menu Maker".

Recommandations de Sécurité

Gestion des Comptes et Authentification

1. **Authentification Forte**:
    - **Mot de Passe**: Utiliser des politiques de mot de passe robustes, exigeant une complexité minimale (longueur, caractères spéciaux, etc.). ○ **Authentification à Deux Facteurs (2FA)**: Implémenter 2FA pour ajouter une couche supplémentaire de sécurité. Utiliser des applications d'authentification comme Google Authenticator ou des SMS OTP.
2. **Stockage sécurisé des mots de passe**:
    - **Hashing**: Utiliser des algorithmes de hashage sécurisé comme bcrypt pour stocker les mots de passe. Ne jamais stocker les mots de passe en clair. ○ **Salage**: Ajouter un **sel** unique pour chaque utilisateur avant de hasher leur mot de passe afin de renforcer la sécurité contre les attaques par rainbow table.
3. **Contrôle d'Accès**:
    - **RBAC (Role-Based Access Control)**: Implémenter la gestion des accès basée sur les rôles pour restreindre l'accès aux fonctionnalités et données en fonction du rôle de l'utilisateur (administrateur, utilisateur standard, etc.).
    - **Principle of Least Privilege**: Limiter les permissions des comptes utilisateurs et des services tiers pour qu'ils n'aient accès qu'aux informations et fonctions nécessaires. **Sécurisation des Communications**

1. **HTTPS**:
    - **TLS/SSL**: Utiliser HTTPS pour toutes les communications entre le client et le serveur. Obtenez des certificats SSL auprès de fournisseurs comme Let's Encrypt. ○ **HSTS (HTTP Strict Transport Security)**: Configurer HSTS pour forcer les navigateurs à interagir uniquement via des connexions HTTPS.
2. **Chiffrement de Données**:
    - **Data at Rest**: Chiffrer les données sensibles stockées sur le serveur en utilisant des standards de chiffrement comme AES.
    - **Data in Transit**: Utiliser TLS pour chiffrer les données en transit entre le serveur et le navigateur.

Sécurisation des Plugins et Bibliothèques

1. **Sélection et Gestion des Plugins**:
    - **Réputation**: Choisir des plugins et des bibliothèques ayant une bonne réputation et soutenus par une communauté active.
    - **Mises à jour régulières**: S'assurer que tous les plugins et bibliothèques sont régulièrement mis à jour pour patcher les vulnérabilités connues.
2. **Isolation et Contrôle**:
    - **Least Privilege**: Donner les permissions minimales nécessaires aux plugins pour fonctionner.

- **Scan des Vulnérabilités**: Utiliser des outils d'analyse de vulnérabilités comme Snyk ou OWASP Dependency-Check pour vérifier les dépendances.

Prévention des Attaques

1. **Validation des Entrées**:
   - **Sanitization**: Valider et nettoyer toutes les données d'entrée pour prévenir les injections de script et attaques par manipulations de données. ○ **CSP (Content Security Policy)**: Implémenter CSP pour réduire le risque de XSS (Cross-Site Scripting).
2. **Protection contre les attaques de session**:
   - **CSRF (Cross-Site Request Forgery)**: Utiliser des tokens CSRF pour protéger les points de terminaison sensibles.
   - **Cookie Security**: Marquer les cookies comme **HttpOnly** et **Secure** pour les protéger contre les accès JavaScript et les attaques en clair.

Surveillance et Gestion des Incidents

1. **Journalisation et Monitoring**:
   - **Logs**: Conserver des journaux d'activité détaillés mais ne contenant pas d'informations sensibles. ○ **Surveillance en Temps Réel**: Utiliser des services de monitoring en temps réel pour détecter et répondre rapidement aux comportements suspects (intrusions, anomalies, etc.).
2. **Plan de Réponse aux Incidents**:
   - **Préparation**: Définir un plan de réponse aux incidents incluant les rôles et responsabilités en cas de faille de sécurité.
   - **Backups**: Maintenir des sauvegardes régulières des données critiques et tester régulièrement les procédures de restauration.

Outils Recommandés

1. **Frameworks et Bibliothèques Sécurisées**:
   - **Express.js**: Pour le backend Node.js, utiliser Express avec des middlewares de sécurité comme helmet.
   - **React**: Pour le frontend, suivre les meilleures pratiques de React pour éviter les injections XSS.
2. **Outils de Scanning et Surveillance**:
   - **OWASP ZAP**: Outil de test de sécurité des applications pour identifier les vulnérabilités.
   - **Nessus**: Scanner de vulnérabilités pour détecter les faiblesses de l'infrastructure.
   - **Sentry**: Pour le monitoring des erreurs et performances d'application.

Conclusion

La sécurité de l'application web "Menu Maker" doit englober une approche multi-couches, traitant chaque aspect de l'application, de l'authentification utilisateur à la gestion des dépendances, en passant par la surveillance continue. En mettant en œuvre ces recommandations, vous pouvez rendre l'application nettement plus sécurisée et résistante aux attaques courantes.

VI. Maintenance du site et futures mises à jour

Contrat de Maintenance pour Menu Maker

1. **Objet du Contrat**
   - Ce contrat de maintenance a pour objet de définir les responsabilités et les services fournis par Webgencia concernant la maintenance et les mises à jour de l'application Menu Maker pour le client QWENTA.
2. **Portée des Services**
   - **Correctifs** : Correction des bugs et problèmes identifiés dans l'application.
   - **Mises à Jour de Sécurité** : Application des patchs de sécurité et des mises à jour logicielles pour maintenir la sécurité de l'application.
   - **Support Technique** : Assistance technique pour les utilisateurs de l'application, disponible via email, chat ou téléphone.
   - **Améliorations Fonctionnelles** : Développement et déploiement de nouvelles fonctionnalités et améliorations basées sur le feedback des utilisateurs.
   - **Surveillance et Reporting** : Surveillance continue des performances de l'application et génération de rapports réguliers.
3. **Niveaux de Service (SLA)** • **Temps de Réponse** :
   - **Urgent** : Réponse en moins de 2 heures. ○ **Élevé** : Réponse en moins de 1 jour ouvrable. ○ **Moyen** : Réponse en moins de 3 jours ouvrables.
   - **Bas** : Réponse en moins d'une semaine.
   - **Résolution des Incidents** :
     - **Critique** : Résolution en moins de 4 heures.
     - **Élevé** : Résolution en moins de 2 jours ouvrables. ○ **Moyen** : Résolution en moins de 5 jours ouvrables.
     - **Mineur** : Résolution dans les 10 jours ouvrables.
4. **Responsabilités de QWENTA**
   - **Accès** : Fournir un accès approprié aux systèmes et aux données nécessaires pour effectuer la maintenance.
   - **Documentation** : Informer rapidement de tout problème ou incident observé et fournir une documentation détaillée si nécessaire.
   - **Environnement** : Maintenir un environnement compatible et sécurisé pour l'exécution de l'application.
5. **Responsabilités de Webgencia**
   - **Qualité du Service** : Assurer que les services sont fournis en respectant les standards de qualité et dans les délais définis.
   - **Confidentialité** : Protéger les informations confidentielles et les données sensibles de QWENTA.
   - **Communication** : Informer régulièrement QWENTA sur l'état de la maintenance et des mises à jour.
6. **Exclusions**
   - **Dommages Causés par le Client** : Les réparations nécessaires à cause de modifications non autorisées ou d'une utilisation incorrecte par le client ne sont pas couvertes.
   - **Propagation de Virus** : Les frais de réparation dus à des virus ou des logiciels malveillants

installés par le client.

- **Matériel ou Logiciels Tiers** : La maintenance de tout matériel ou logiciel tiers non fourni par Webgencia.

7. Tarification et Modalités de Paiement • **Frais de Maintenance** :

- Mensuels : 500 € ○ Trimestriels : 1400 € (réduction de 100 €) ○ Annuels : 5500 € (réduction de 1000 €)

- **Modalités de Paiement** : Le paiement doit être effectué dans les 30 jours suivant la réception de la facture. Un intérêt de 1,5% par mois sera appliqué pour tout paiement en retard.
- **Révisions de Prix** : Les frais peuvent être révisés annuellement avec un préavis de 30 jours.

8. Durée et Renouvellement

- **Durée Initiale** : Ce contrat a une durée initiale de 1 an à partir de la date de signature.
- **Renouvellement** : Le contrat sera automatiquement renouvelé pour des périodes successives de 1 an sauf avis contraire donné par écrit par l'une ou l'autre des parties avec un préavis de 30 jours.

9. Résiliation

- **À l'Initiative de QWENTA** : QWENTA peut résilier le contrat avec un préavis écrit de 30 jours, avec ou sans motif.
- **À l'Initiative de Webgencia** : Webgencia peut résilier le contrat en cas de non-paiement ou de non-respect des termes, avec un préavis de 30 jours.
- **Effets de la Résiliation** : Aucune des parties ne sera responsable de tout dommage indirect ou consécutif à la résiliation du contrat.

10. Dispositions Générales

- **Modification du Contrat** : Toute modification doit être convenue par écrit et signée par les deux parties.
- **Loi Applicable** : Ce contrat est régi par les lois de France.
- **Règlement des Litiges** : Tout litige sera résolu par arbitrage ou médiation, ou devant le tribunal compétent de Paris, France.

Signatures

- **Hugo Pradier, Représentant de Webgencia** : _____
- **[Nom et Titre du Représentant de QWENTA]** : _____ Date : **/**/_____

Ce contrat énonce clairement les termes et les responsabilités mutuelles entre Webgencia et QWENTA, assurant un cadre de coopération pour la maintenance et l'amélioration continue de l'application Menu Maker. N'oubliez pas de faire examiner ce contrat par des conseillers légaux pour en assurer la conformité avec les obligations légales et les besoins spécifiques.