Spécifications techniques

[Planifiez le développement du site de votre client,SUNTING]

Version	Auteur	Date	Approbation
1.0	SUN TING	08/10/2025	

I.Choix technologiques

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

Besoin	Contraintes	Solution	Descripion de la solution	Justification (2 arguments)
Création et modifiation de menu	L'utilisateur doit pouvoir ajouter,modification ou supprimer des plats directement depuis l'interface sans recharger la page.	React+Redux Toolkit	Cette technologie permet de créer une interface dynamique et réactive.Redux Toolkit assure la gestion centralisée des données du menu	1) La solution est cohérente avec une architecture SPA moderne. 2) Elle est recommandée dans la formation OpenClassrooms
Ajout d'une catégorie de plat	L'ajout 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,accesibles avec un minimum de code	1) La librairie s'intègre parfaitement à l'environnement React. 2) Il s'agit de la solution la plus utilisée pour ce type de fonctionnalité.
Téléversement d'image de plat	Les fichiers doivent être stockés sur un sevice externe pour ne pas sur charger le server.	Service cloud externe(type Cloudinary/Fiebase Storage)	Ce service permet d'hébergier les images et de générer un URL pubique sécurisée pour leur affichage	1) Cette solution améliore les performances et la rapidité du site. 2) Elle évite la saturation du serveur back-end.
Exportation du menu en PDF	L'utilisateur doit pouvoir exporter son menu au format PDF en un seul clic.	jsPDF	Cette librairie JavaScript permet de générer des fichiers PDF directement depuis le navigateur,sans intervention du serveur.	1) Elle s'intègre facilement dans application React. 2) Il s'agit d'une librairie open so fiable et largement utilisée.
Authentification utilisateur	L'accès à l'espace restaurateur doit être sécurisé et rester actif pendant la session.	JSON WEB TOKEN(JWT)	Ce système d'authentification repose sur des tokens stockés localement et vérifiés àchaque requête.	1) JWT garantit un communication sécurisée entre le front et le back-end. 2) Il s'agit d'un stantdard moderne pour les application web.

Communication entre le front-end et le back-end	Les échanges entre le client et l'API doivent être stable et protégé.	Axios	Cette librairie permet de gérer facilement les requêtes HTTP et	1) Elle simplifie la gestion des appels API. 2) Elle offre une syntaxe
			d'ajouter les en-têtes d'authentification	Plus claire que fetch().
Base de données	Le site doit stocker des données structurées(menus,utilis ateurs,catégories).	MongoDB Atlas+Mongoose	Cette base de données NoSQL permet de gérer des données sous format JSON et d'assurer la filexibilité des modèles.	1) Elle s'intègre naturellement à Node.js et Express. 2) Elle est adaptée à la structure dynamique du projet.
Hébergement	Le site doit être en ligne,accessible et automatiquement mis à jour à chaque commit.	IONOS	La plateforme IONOS offre un hébergement sécurisé avec certificat SSL, un domaine personnalisé et la gestion intégrée des e-mails professionnels	1)IONOS garantit un hébergement fiable, sécurisé et conforme aux standards professionnels. 2)L'intégration continue avec GitHub permet un déploiement automatique (CI/CD) à chaque mise à jour du code.
Design et responsivité	L'affichage doit s'adapter à tous les écrans conformément à la maquette Figma.	SCSS+Flexbox+Med ia Queries	Cette combinaison permet de structurer le code Css et de gérer la mise en page responsive de manière claire.	1) SCSS facilite la maintenance et la modularité du style. 2) L'approche mobile-frist garantiti une expérience utilisateur optimale.
Diffuser un menu	Le restaurateur souhaite partager facilement son menu en ligne avec ses clients.	Fonction de diffusion publique.	Génération d'un lien public ou d'un QR code accessible aux clients, hébergé sur le sous-domaine menumaker.qwenta.fr.	1) Améliore la visibilité du restaurant. 2) Simplifie l'accès client via smartphone.

II Liens avec le back-end

- **Quel langage pour le serveur ?**

  NodeJS,est un environnement JavaScript côté serveur.Ce choix permet d'utilier le même langage(JavaScript) sur l'ensemble du projet,
  Autant pour le front-end(React) que pour le back-end(Express).
  Node.js est particulièrement qdapté aux applications web modernes nécessitant des échanges rapides et asynchrones.

  Justification:

1. L'Unification du langage simplifie la commnunication entre le front*end et le back-end.
2. Node.js offre de bonnes performances pour la gestion des requêtes API et le traitement en temps réel.

- **A-t-on besoins d' une API ?**

Oui, le site a besoin d'une **API REST** afin d'assurer la communication entre le **front-end** et le **back-end**.

Un back-end est indispensable pour la persistance des données, c'est-à-dire pour enregistrer, modifier et conserver les informations de manière durable.

Cette API permettra de gérer les différentes opérations liées aux menus (création, modification, suppression, exportation, etc.), ainsi que la gestion des utilisateurs.
Elle servira d'intermédiaire entre l'interface utilisateur et la base de données, garantissant la cohérence et la mise à jour en temps réel des informations.

Endpoints principaux(exemples):

- POST /api/login → Authentification et génération d'un token JWT
- GET /api/menus → Récupération des menus d'un utilisateur
- POST /api/menus → Création d'un nouveau menu
- PUT /api/menus/:id → Modification d'un menu existant
- DELETE /api/menus/:id → Suppression d'un menu

Justification :

1. Une API REST assure la modularité et la réutilisabilité du code.
2. Elle permet une séparation claire entre les responsabilités du front et du back-end.

- **A-t-on utilisé SQL?**

Oui,le projet utilisera une base de données NoSQL, plus précisément MongoDB Atlas,hébergée sur le cloud pour simplifier lagestion et les sauvegardes automatiques.Ce type de base est idéal pour stocker des

objets structurés sous forme JSON comme les utilisateurs,les menues, les catégories et les plates.

Justification:

1. MongoDB est parfaitement compatible avec l'environnement Node.js et Express.
2. Sa flexibilité permet de faire évoluer la structure des données sans contrainte stricte de schéma.

III. Préconisations concernant le domaine et l'hébergement

- **Nom du domaine**

  www.menumaker.qwenta.fr

  Ce choix permet d'intégrer l'application directement dans l'écosystème numérique de Qwenta, tout en conservant une identité propre au service Menu Maker.
  L'utilisation d'un sous-domaine facilite également la gestion centralisée du nom de domaine, des certificats SSL et du référencement SEO sous le domaine principal.

- **Nom de l'hébergement.**

  Le site sera hébergé chez IONOS, un hébergeur professionnel reconnu pour sa fiabilité et ses solutions complètes.
  IONOS offre à la fois un hébergement web sécurisé (HTTP inclus), un nom de domaine personnalisé et la gestion des adresses e-mail professionnelles.

Ce choix permet de centraliser l'ensemble des serveces(hébergement,domaine,messagerie)au même endroit, ce qui simplifie la maintenance et la gestion du projet.

Justification:

1. IONOS garantit une excellente disponibilité du site (99,9%) et une sécurité renforcée.
2. Il propose un support technique en français et des outils d'administration faciles à utiliser.

Front-end : hébergé sur IONOS (hébergement mutualisé avec certificat SSL).
Back-end : déployé sur Render pour gérer l'API Node.js et la base de données.

- **Adresses e-mail**

Afin d'assurer la communication officielle du service, plusieurs adresses e-mail seront configurées via le domaine principal :

1. **contact@qwenta-menumaker.fr** → pour les demandes générales et les nouveaux clients ;
2. **support@qwenta-menumaker.fr** → pour l'assistance technique ;
3. **admin@qwenta-menumaker.fr** → pour la gestion interne du site.

Ces adresses seront créées depuis le tableau de bord d'IONOS, en lien direct avec le domaine enregistré.

Justification :

1.L'utilisation d'adresses personnalisées renforce la crédibilité du service.
2.Elle permet une meilleure organisation et un suivi efficace des échanges.

IV. Accessibilité

- **Compatibilité navigateur**

Le site sera compatible avec les principaux navigateurs du marché:
Google Chrome,Mozilla Firefox,Safari,Microsoft Edge et Opera.
Des tests d'affichage et de performance seront effectués sur chacun d'eux afin de garantir une expérience utilisateur cohérente.

Justification:

1. Ces navigateurs représentent plus de 95% des parts de marché et couvrent la majorité des utilisateurs.
2. Le respect des standards du **W3C (HTML5, CSS3, ECMAScript)** assure un compatibilité optimale entre navigateurs.

- **Types d'appareils.**

Le site a été conçu principalement pour une utilisation sur ordinateur de bureau (desktop).
L'interface est optimisée pour un affichage clair et ergonomique sur des écrans de grande taille.
La mise en page repose sur l'utilisation de Flexbox et de Media Queries dans le code SCSS, garantissantune
présentation stable et cohérente sur les résolutions d'écran standard de bureau.

.

Justification:

1. Le projet cible une utilisation **professionnelle**, principalement sur poste fixe.
2. L'optimisation pour le format **desktop** permet de privilégier la lisibilité et la clarté de l'interface

V. Recommandations en termes de sécurité

● Accès aux comptes utilisateurs

L'accès aux comptes restaurateurs est sécurisé par un système d'authentification basé sur les **JSON Web Token(JWT).**
Lors de la connexion, un token chiffré est généré et stocké dans le navigateur.
Ce token est vérifié à chaque requête vers l'API afin d'assurer que seules les personnes autorisées puissent accéder à leurs données.

Justification:

1. Le système JWT garantit la confidentialité et l'intégrité des sessions utilisateur.
2. Il évite le stockage des informations sensibles sur le serveur et réduit les risques de piratage.

● Protection des données

Toutes les communications entre le front-end et le back-end passent par le protocole sécurisé **HTTPS**.
Les mots de passe sont **hachés avec bcrypt** avant d'être enregistrés dans la base de données.
Les données personnelles des utilisateurs (noms, e-mails, menus) sont stockées sur **MongoDB Atlas**, un service cloud sécurisé respectant le RGPD (Règlement Général sur la Protection des Données).

Justification :

1 .Le chiffrement HTTPS protège contre les attaques de type "man-in-the-middle".
2 Le hachage des mots de passe empêche toute exploitation en cas de fuite de données.

• Gestion des permissions et plugins

L'accès aux fonctions d'administration (ex. : modification, suppression de menu) est restreint aux utilisateurs authentifiés et autorisés.
Aucun plugin externe non vérifié n'est intégré au projet afin de limiter les failles potentielles.
Les dépendances (npm packages) sont régulièrement mises à jour via npm audit pour prévenir les vulnérabilités connues.

Justification :

1. La séparation claire des rôles empêche les accès non autorisés.
2. La maintenance régulière des dépendances renforce la sécurité globale du site.

Justification :

1.Les sauvegardes garantissent la continuité du service.
2.Elles permettent de rétablir les menus et comptes sans perte de données.

Bonnes pratiques issues d'OWASP

Afin de renforcer la sécurité de l'application, plusieurs recommandations issues du guide OWASP Top 10 sont appliquées :
Validation systématique des entrées utilisateurs pour éviter les attaques de type Injection (SQL, NoSQL, XSS).

Mise en place de headers HTTP sécurisés (Content-Security-Policy, X-Frame-Options, X-XSS-Protection, etc.).
Configuration stricte du CORS (Cross-Origin Resource Sharing) pour limiter les requêtes provenant de domaines non autorisés.
Journalisation et suivi des tentatives de connexion pour détecter toute activité suspecte.
Expiration automatique des tokens JWT afin de réduire les risques liés aux sessions prolongées
Utilisation du principe du moindre privilège (least privilege) : chaque utilisateur n'a accès qu'aux ressources nécessaires.
.

VI. Maintenance du site et futures mises à jour

● Contrat de maintenance – grandes lignes

Une maintenance régulière sera mise en place afin d'assurer la stabilité, la sécurité et la pérennité du site Menu Maker by Qwenta.
Le contrat de maintenance comprend trois volets principaux :

1. Maintenance corrective

Correction rapide des bugs signalés par les utilisateurs (affichage, connexion, export PDF, etc.).

➜ *Durée de garantie : 3 mois après la mise en ligne*, pendant laquelle les anomalies seront corrigées **sans frais supplémentaires**.

2. Maintenance préventive

Vérification mensuelle des dépendances, mises à jour des librairies JavaScript (React, Node, Redux Toolkit, etc.) et contrôle de la sécurité (npm audit, correctifs de vulnérabilités).

➜ *Durée d'engagement : 6 mois* à compter de la mise en ligne.

3. Maintenance évolutive

Intégration de nouvelles fonctionnalités selon les besoins des restaurateurs, comme :

1) ajout de statistiques d'utilisation,
2) intégration avec les plateformes de livraison (Deliveroo, UberEats),
3) amélioration du design ou du module d'export.

➜ *Durée indicative : jusqu'à 12 mois après livraison*, renouvelable selon l'évolution du projet.

Justification :

1. Ces trois types de maintenance garantissent la continuité et la qualité du service.
2. Ils permettent d'anticiper les problèmes techniques tout en

accompagnant la croissance du projet.

- **Suivi et support technique**

Un système de support sera mis en place via l'adresse **support@qwenta-menumaker.fr,** permettant aux utilisateurs de signaler tout incident ou demande d'amélioration.
Les demandes seront traitées selon un délai moyen de 48 heures ouvrées.

Justification :

1. La centralisation du support améliore la communication entre les utilisateurs et l'équipe technique.
2. Elle permet de prioriser les interventions selon la gravité des incidents.

- **Sauvegardes, récupération et mises à jour**

Les données hébergées sur **MongoDB Atlas** font l'objet de **sauvegardes automatiques quotidiennes**, garantissant la sécurité et la disponibilité des informations.
En cas de défaillance du serveur, une **procédure de récupération** permet de restaurer rapidement les données critiques.

Chaque mise à jour du code source sur **GitHub** déclenche un **déploiement automatique (CI/CD)** sur la plateforme **IONOS**, assurant la continuité du service et la stabilité des nouvelles versions.

Justification :

1. Les sauvegardes quotidiennes réduisent le risque de perte de données.
2. L'intégration continue facilite les mises à jour sans interruption de service.
3. La récupération automatisée garantit la continuité du service.

VII. Résumé final / Conclusion du document

Le présent document de spécifications techniques définit l'ensemble des choix technologiques, des méthodes et des outils retenus pour le développement du projet Menu Maker by Qwenta.
Ce projet repose sur une architecture moderne et cohérente, combinant React pour le front-end, Node.js pour le back-end et MongoDB Atlas pour la base de données. L'hébergement sur IONOS permet de garantir la stabilité, la sécurité et la maintenance continue du site.

Les différentes sections ont présenté :

- les solutions techniques choisies pour répondre aux besoins fonctionnels (création de menus, exportation PDF, gestion d'images, etc.) ;
- la structure de communication entre le front-end et le back-end ;
- les recommandations en matière d'hébergement, d'accessibilité et de sécurité ;
- ainsi qu'un plan de maintenance pour assurer la pérennité du service.

Ce document constitue une base solide pour le développement et le suivi du projet,
en garantissant une vision claire, structurée et alignée avec les bonnes pratiques du développement web moderne.