

# Spécifications techniques – Menu Maker by Qwenta

Version : 1.2

Auteur : Mr Laurent

Date : 25/09/2025

Approbation : Soufiane / Qwenta

## I. Choix technologiques

Élément	Détails
Création d'un menu avec catégories	<ul style="list-style-type: none"><li>- Ajouter plats par catégories : entrée, plat, dessert</li><li>- Solution : React + State local + modales (react-modal)</li><li>- Gestion dynamique via hooks React</li><li>- Justification : React facilite le rendu dynamique et les modales sont accessibles</li></ul>
Saisie nom, prix et description	<ul style="list-style-type: none"><li>- Interface claire pour restaurateur</li><li>- Solution : Composants React Form + validation front-end</li><li>- Champs contrôlés avec messages d'erreur</li><li>- Justification : UX claire et facilement extensible</li></ul>
Personnalisation branding	<ul style="list-style-type: none"><li>- Sauvegarde préférences pour chaque restaurateur</li><li>- Solution : React + API back-end /preferences</li><li>- Justification : données liées à l'utilisateur, aperçu temps réel</li></ul>

Export PDF	<ul style="list-style-type: none"><li>- Générer un PDF fidèle au design</li><li>- Solution : jspdf ou pdfkit</li><li>- Justification : Compatible React/Node, téléchargement ou impression possible</li></ul>
Diffusion Deliveroo / Instagram	<ul style="list-style-type: none"><li>- Partage via lien ou API</li><li>- Solution : API REST back-end / Webhooks</li><li>- Justification : automatisation sécurisée et standardisée</li></ul>
Impression	<ul style="list-style-type: none"><li>- Impression depuis navigateur</li><li>- Solution : CSS Print + React, bouton window.print()</li><li>- Justification : simple, compatible navigateurs</li></ul>
Gestion erreurs utilisateurs	<ul style="list-style-type: none"><li>- Pages 404 / 500 personnalisées, messages d'erreur front et back</li><li>- Justification : UX améliorée et informations claires en cas de problème</li></ul>
Performance front-end	<ul style="list-style-type: none"><li>- Lazy loading images, code splitting, compression gzip/brotli</li><li>- Justification : accélère le rendu et diminue la latence initiale</li></ul>



## II. Liens avec le back-end

Élément	Détails
---------	---------

Langage serveur	Node.js avec Express
API	<ol style="list-style-type: none"> <li>1. Menus <ul style="list-style-type: none"> <li>• <code>GET /menus</code> → lister tous les menus d'un restaurateur (ou filtrer par catégorie, date...)</li> <li>• <code>GET /menus/:id</code> → récupérer un menu précis</li> <li>• <code>POST /menus</code> → créer un menu</li> <li>• <code>PUT /menus/:id</code> → modifier un menu</li> <li>• <code>DELETE /menus/:id</code> → supprimer un menu</li> </ul> </li> <li>2. Dishes (plats individuels) <ul style="list-style-type: none"> <li>• <code>POST /menus/:menuId/dishes</code> → ajouter un plat à un menu</li> <li>• <code>PUT /menus/:menuId/dishes/:dishId</code> → modifier un plat</li> <li>• <code>DELETE /menus/:menuId/dishes/:dishId</code> → supprimer un plat</li> </ul> </li> <li>3. Preferences / Branding <ul style="list-style-type: none"> <li>• <code>GET /preferences</code> → récupérer préférences de l'utilisateur</li> <li>• <code>PUT /preferences</code> → mettre à jour le logo, couleurs, police</li> </ul> </li> <li>4. Export / PDF <ul style="list-style-type: none"> <li>• <code>GET /export/pdf/:menuId</code> → générer et télécharger le PDF</li> </ul> </li> <li>5. Diffusion / Webhooks <ul style="list-style-type: none"> <li>• <code>POST /deliveroo/:menuId</code> → envoyer un menu à Deliveroo</li> <li>• <code>POST /instagram/:menuId</code> → envoyer un menu sur Instagram via webhook</li> </ul> </li> </ol>

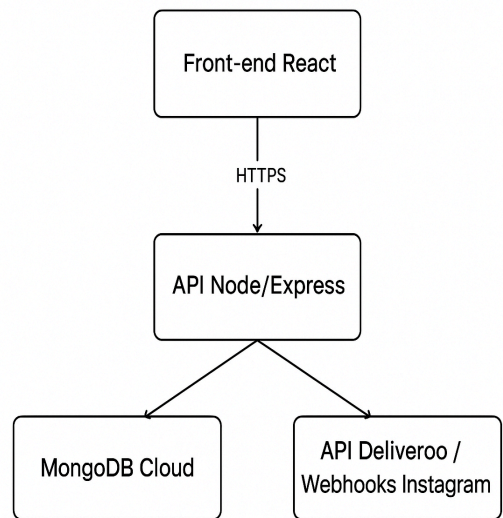
	<p>6. Sécurité / Auth</p> <ul style="list-style-type: none"> <li>• <b>POST /auth/refresh-token</b> → renouveler JWT si expiration</li> <li>• Middleware pour vérifier rôle utilisateur sur tous les endpoints sensibles</li> </ul> <p>7. Monitoring / Logs</p> <ul style="list-style-type: none"> <li>• <b>GET /logs/front</b> → logs front (Sentry / LogRocket)</li> <li>• <b>GET /logs/back</b> → logs back (Winston / Morgan)</li> </ul>
Version API	/api/v1/...
Documentation	Swagger / Postman prévue
Base de données	MongoDB (NoSQL)
Collections principales	<ul style="list-style-type: none"> <li>- <b>users</b> : { _id, email, passwordHash, role, createdAt }</li> <li>- <b>menus</b> : { _id, userId, title, sections:[{name, dishes:[{name, price, description}]}], createdAt }</li> <li>- <b>preferences</b> : {userId, branding:{logo, colors, font}, lastUpdated }</li> </ul>
Indexation	_id, userId pour requêtes rapides
Sécurité back-end	<ul style="list-style-type: none"> <li>- JWT 24h + refresh token</li> <li>- bcrypt pour mots de passe</li> <li>- Rate limiting</li> <li>- Validation côté serveur</li> </ul>

	- HTTPS obligatoire, CORS configuré
Tests	<ul style="list-style-type: none"> <li>- Unitaires : Jest / Supertest</li> <li>- Intégration : API / Base de données</li> <li>- End-to-end : Cypress</li> </ul>



### III. Architecture et hébergement

Élément	Détails
Front-end	React (hébergement : Vercel ou Netlify)
Back-end	Node.js / Express (Heroku, Render ou AWS)
Base de données	MongoDB cloud (Mongo Atlas)
Communication	API REST sécurisée front ↔ back ↔ plateformes externes
Schéma technique	<b>Schéma technique – Architecture Menu Maker</b>



**Légende :**

- **Front-end React** : Interface utilisateur sur navigateur (desktop).
- **API Node/Express** : Back-end sécurisé, gestion des menus et préférences.
- **MongoDB Cloud** : Base de données NoSQL pour utilisateurs, menus et préférences.
- **API Deliveroo / Webhooks Instagram** : Intégration pour diffusion externe des menus.
- **HTTPS** : Communication sécurisée entre front et back-end.

Nom de domaine

sous-domaine Qwenta (ex. menu-maker.qwenta.com)

Emails

contact@menu-maker-qwenta.com,  
support@menu-maker-qwenta.com

## IV. Accessibilité

Élément	Détails
Navigateurs compatibles	Chrome, Safari, Firefox (dernières versions)
Types d'appareils	Desktop uniquement (mobile non prioritaire)
Préconisations	<ul style="list-style-type: none"><li>- Navigation clavier</li><li>- Support lecteur d'écran</li><li>- Contraste suffisant (WCAG 2.1)</li><li>- Aria-labels sur boutons et champs</li></ul>

---

## V. Sécurité front-end

Élément	Détails
Validation inputs	Éviter XSS / injections
Librairies utilisées	react-modal, jspdf (librairies maintenues)
Protection routes	Composants sensibles sécurisés et protégés

---

## VI. Maintenance et monitoring

Élément	Détails
Vérification dépendances	npm audit régulier
Sauvegarde	Automatique de la base de données
Mise à jour	Composants React et librairies régulièrement
Futur	Nouveaux templates, options de partage
Monitoring	<ul style="list-style-type: none"><li>- Logs serveur : winston, morgan</li><li>- Erreurs front : Sentry / LogRocket</li><li>- Alertes en cas de panne ou erreur critique</li></ul>
RGPD / Données personnelles	Gestion des droits utilisateurs : accès, suppression, conservation des données

---

## VII. CI/CD et déploiement

Élément	Détails
Déploiement front	Automatisé sur Vercel / Netlify



Déploiement back	Automatisé sur Heroku / Render / AWS
Environnements	Gestion dev / staging / prod via variables d'environnement
Pipeline CI/CD	Lint → Tests → Build → Deploy