

JEAN Jean-louis
16/06/25

PRÉSENTATION

**Menu Maker by
Qwenta**

Sommaire

1. Contexte du projet
2. Aperçu de la maquette
3. Méthodologie utilisée
4. Tableau Kanban
5. Spécifications techniques
6. Veille technologique
7. Conclusion
8. Questions

Contexte du Projet

Menu Maker by Qwenta est un site permettant aux restaurateurs d'afficher, de mettre en page, de personnaliser et de partager leurs menus facilement, en quelques clics.

Fonctionnalités :

- Créer un menu
- Personnaliser un menu
- Diffuser un menu
- Imprimer un menu
- Assistance dans les démarches via articles et conseils ou support par mail

Aperçu de la maquette

C

Méthodologie utilisée

Méthode Agile :

C'est une philosophie pour gérer des projets.

Elle repose sur 4 valeurs et 12 principes définis dans le Manifeste Agile (2001).

Objectif : → Livrer un produit de qualité progressivement, en s'adaptant rapidement aux changements.

Caractéristiques :

- Travail en cycles courts.
- Collaboration constante avec le client / Product-owner.
- Équipe autonome et collaborative.
- Réactivité face aux imprévus
-

Agile apporte la vision et la culture : valeurs, principes, état d'esprit.

Méthodologie utilisée

Méthode Scrum :

C'est un cadre de travail qui applique les principes Agile de manière concrète et structurée.

Objectif : → Organiser le travail en sprints pour livrer des fonctionnalités utilisables à chaque itération.

Composants clés :

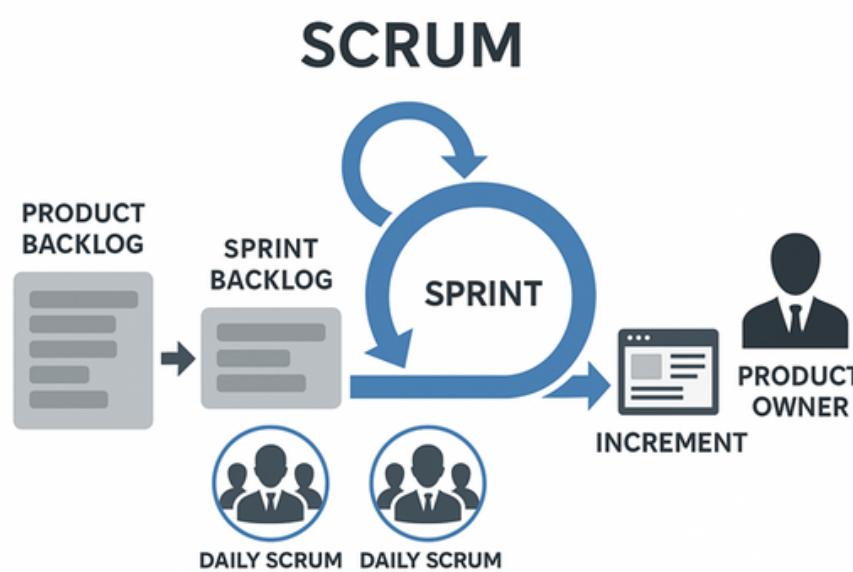
Rôles bien définis : Product Owner (vision produit) ,Scrum Master (facilitateur), Équipe de développement (réalisation)

Événements :

- Sprint (itération de 1 à 4 semaines)
- Daily Scrum (réunion quotidienne)
- Sprint Planning, Review, Rétrospective

Artefacts :

Product Backlog (rédigé sous forme de User Story, Sprint Backlog, Incrément (C'est une version utilisable et testable du produit à la fin du sprint)



Kanban enrichi de pratiques Scrum

Avantages de cette approche pour le projet Menu Maker.

1. Flexibilité et adaptation rapide

- Les retours fréquents (chaque Sprint) permettent d'ajuster les fonctionnalités de génération et personnalisation de menus selon les besoins réels des utilisateurs.

2. Livraison incrémentale de valeur

- Dès les premiers Sprints, on peut proposer un prototype de création de menus, puis enrichir progressivement (ajout de thèmes, export PDF, intégrations).

3. Visibilité et transparence

- Les réunions de revue associent les parties prenantes (designers UX, restaurateurs, pilotes, développeurs), garantissant que le produit reste aligné sur les objectifs métier.

4. Gestion maîtrisée des risques

- Les problèmes techniques (intégration d'API d'images ou préférences mobiles) sont identifiés dès les premiers Sprints, limitant les surprises en fin de projet.

5. Amélioration continue

- Grâce aux rétrospectives, l'équipe affine ses pratiques (testing, revue de code, automatisation CI/CD), accélérant le développement du Menu Maker.

6. Engagement de l'équipe

- Le cadre Scrum responsabilise chacun (Product Owner, Scrum Master, équipe de développement), favorisant la motivation et la qualité du travail.

En résumé, adopter Agile/Scrum pour Menu Maker garantit une progression itérative, un ajustement constant aux besoins utilisateurs et une livraison de fonctionnalités exploitables dès la fin de chaque Sprint.

Critère	Agile	Scrum
Type	Philosophie / approche globale	Framework (cadre de travail) Agile
Structure	Flexible, principes directeurs	Structuré: rôles, événements, artefacts
Guide de référence	Manifeste Agile (2001)	Scrum Guide (Schwaber & Sutherland)
Itérations	Itératives, durée variable	Sprints fixes (1 à 4 semaines)
Rôles définis	Non	Oui: PO, SM, Équipe Dev
Événements rituels	Non imposés	Oui: Sprint Planning, Daily, Review, Retro
Adaptabilité	Très élevée	Élevée, mais dans le cadre des sprints
Outils recommandés	N'importe quel outil ou méthode (Kanban, XP...)	Aucun outil imposé, mais Trello, Jira... souvent utilisés
Idéal pour	Projets complexes, évolutifs	Équipes cherchant un cadre Agile pragmatique

Suivi du projet avec le Kanban

P7 Menu Maker by Qwenta

Product Owner Soufiane

- Sprint
- Page d'accueil (0/1)
- Intégration API Deliveroo ?
- Intégration de l'API Instagram ?
- + Ajouter une carte

Frontend : Jean-louis

- Connexion et diffusion de votre menu personnalisé
- Personnalisez votre menu
- Élément dans votre menu ?
- Intégration Page d'Accueil (9 juin - 13 juin, 1/1)
- Connexion / Inscription + modal logi (9 juin - 13 juin, 1/4)
- Création de menu avec modale (9 juin - 13 juin, 3/3)
- + Ajouter une carte

Backend dev : à définir

- Connexion à la base de données (MongoDB Atlas)
- Implémentation Auth via Firebase Admin SDK (9 juin - 13 juin, 0/5)
- Gestion des rôles et autorisations (RBAC)
- Endpoints CRUD pour les menus (16 juin - 20 juin, 0/5)
- Endpoints CRUD pour les plats (16 juin - 20 juin, 0/5)
- Upload logo restaurant (Firebase Storage / Multer) (16 juin - 20 juin, 0/4)
- Gestion du thème du menu (16 juin - 20 juin, 0/4)
- Export PDF serveur avec PDFKit (23 juin - 27 juin, 0/4)
- Protection CSRF sur endpoints sensibles (23 juin - 27 juin, 0/4)
- Logs et monitoring (Winston) (23 juin - 27 juin, 0/4)
- Déploiement de l'API sur le
- + Ajouter une carte

À faire

- Ajouter une carte

En cours

- Ajouter une carte

À tester : Équipe QA

- Recette fonctionnelle
- Tests multi-navigateurs et responsive
- Tests d'accessibilité
- Détection de bugs / régressions
- Tests automatisés (si applicable)
- + Ajouter une carte

À corriger : Équipe QA

- Rapports de bugs + suivi
- Ajouter une carte

Déploiement : DevOps

- CI/CD – Intégration et déploiement continu
- Déploiement
- Sécurité & certificats
- Monitoring & alerting
- Dockerisation & conteneurs
- Sauvegardes et restauration
- + Ajouter une carte

Terminer

- Menu Maker by Qwenta (1/1)
- Ajouter une carte

Une User Story est une description concise d'une fonctionnalité vue du point de vue de l'utilisateur final. Elle sert à cadrer ce que doit apporter chaque élément de travail, en se centrant sur la valeur métier et l'expérience utilisateur.

Structure :

En tant que <rôle>,
je souhaite <fonctionnalité>,
afin de <bénéfice / valeur>.

- Rôle : qui va utiliser la fonctionnalité (ex. “restaurateur”, “visiteur”).
- Fonctionnalité : ce que l'on veut faire (ex. “accéder aux fonctionnalités admin”).
- Bénéfice : pourquoi c'est utile (ex. “se connecter rapidement”, “attirer des clients”).

The screenshot shows a Trello card with the following details:

- Title:** Connexion / Inscription + modale logi
- Members:** JL
- Labels:** Priorité Absolut, Développement Front-end
- Dates:** 9 juin - 13 juin, 17:00 (En retard)
- Description:** User stories
 - En tant que restaurateur ayant un compte, je veux pouvoir me connecter.
 - En tant qu'internaute, je veux pouvoir créer un compte automatiquement avec mon adresse mail. [Maquette desktop - Menu Maker by Qwenta](#)
 - [] Cette fenêtre s'ouvre sous forme de modale
 - [] L'utilisateur peut entrer son adresse mail
 - [] Qu'il se soit déjà connecté ou non, un mail lui est envoyé pour lui permettre de s'authentifier, ou au contraire de confirmer son mail pour accéder pour la première fois à l'application
 - [] Un lien "Besoin d'aide" permet d'envoyer directement un e-mail à nos équipes
- Integration:** Implémenter le formulaire avec React Hook Form, Intégrer Firebase Authentication pour la gestion des utilisateurs, Appliquer Tailwind CSS pour le design responsive.
- Pieces jointes:** Capture d'écran 2022-05-02 à 11.29.35.png
- To-dos:** Creation composant LoginForm.jsx, Creation composant RegisterForm.jsx



Spécifications techniques

Liste des principales spécifications techniques

Back-end

- Node.js permet d'exécuter du JavaScript et créer des applications rapides et performantes.
- Express pour la création de routes
- API REST pour gérer les menus, catégories et plats.
- Opérations CRUD sécurisées avec vérification de l'authentification.
- Utilisation de Mongoose pour les schémas de données.
- Middleware de validation (express-validator).

Front-end

- React : Bibliothèque JavaScript.
- SPA avec React Router.
- React Hook Form : formulaires dynamiques et validation en front.
- Affichage des menus en temps réel (si synchronisation via Firebase ou fetch).
- Design responsive : mobile/tablette/desktop.

Authentification

- Firebase : service de Google permettant d'ajouter facilement un système de connexion
- Sécurité des sessions utilisateurs.
- Rôles utilisateur : Attribution de permissions spécifiques à certains utilisateurs (ex. : un restaurateur peut créer des menus, mais un visiteur ne peut que les consulter).

Base de données MongoDB

- Modèles : Utilisateur, Menu, Plat, Catégorie.
- Indexes pour les performances (ex. : menu par utilisateur).
- Hébergement sur MongoDB Atlas.

CRUD sécurisé d'un menu via l'API REST (Express + MongoDB)

C'est un pilier du projet, car le restaurateur doit pouvoir créer, modifier, publier et supprimer un menu, en toute sécurité, et de manière fluide.



CRUD des menus – Menu Maker

Create :

Créer un nouveau menu et l'ajouter en base de donnée.

- Le restaurateur saisit les informations (titre, description, plats, catégories).
- Envoi d'une requête POST /menus vers l'API.
- Le menu est stocké dans MongoDB avec une liaison à l'utilisateur connecté.

Read :

- Hydratation des composants React (template Admin, Preview client).
- Requête GET /menus ou GET /menus/:id selon le contexte.
- Utilisé pour :
 - Visualiser les menus sur l'espace admin.
 - Prévisualiser un menu avant publication.
 - Afficher le menu sur le site public.

Update :

Modifier un menu existant (ou en brouillon)

- Édition possible via l'interface React.
- Requête PUT /menus/:id avec vérification d'identité.
- Mongoose met à jour uniquement les champs modifiés dans MongoDB.

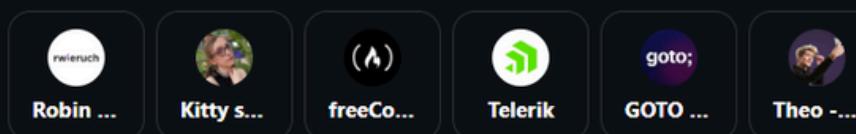
Destroy :

Supprimer un menu ou une information contenue en base de donnée :

- Requête DELETE /menus/:id pour un menu entier ou
- Requête PATCH ou PUT pour supprimer un plat ou une catégorie.
- Vérification que l'utilisateur est propriétaire du menu.

Veille Technologique

- Captures d'écran de la veille
- Méthode de classification des sources d'information.
- Exemple et explication du choix d'une source pour chacun des 2 axes de veille
- Explication de la contribution de la veille à l'élaboration des spécifications techniques.

Top sources covering it**All posts about react-router**

This Week In React #238: React Router, RSC, shadcn/ui, React Aria, TanStack,...

#react #nodejs #nextjs +2

Jun 11 • 7m read time

Up 8 Down 2 Comment 2 Share 2

Why URL state matters: A guide to useSearchParams in React

#webdev #react #ux +1

May 21 • 8m read time

Up 9 Down 2 Comment 2 Share 2

Cloudflare Enhances Developer Experience with Vite Plugin and React Route...

#webdev #cloudflare #vite +1

May 18 • 3m read time

Up 2 Down 1 Comment 1 Share 1

Tutorials by sergiodoxa

#react #rails #react-router

Mar 06 • 5m read time

Up 1 Down 1 Comment 1 Share 1

How React Router works under the hood

#webdev #javascript #react +2

Jan 26 • 7m read time

Up 14 Down 2 Comment 2 Share 2

How to handle multiple concurrent submissions in Remix

#webdev #javascript #react +1

Jan 22 • 7m read time

Up 2 Down 1 Comment 1 Share 1

Server-side rendering with React Router v7

#webdev #react #react-router

Jan 13 • 25m read time

Up 4 Down 1 Comment 1 Share 1

Cookie-based Authentication in Remix

#security #webdev #nodejs +2

Jan 12 • 16m read time

Up 5 Down 1 Comment 1 Share 1

React Status Issue 412: January 8, 2025

#webdev #javascript #react +2

Jan 08 • 2m read time

Up 12 Down 1 Comment 1 Share 1

React Router 7 With Strapi 5 is Awesome

#webdev #javascript #react +2

Jan 07 • 1m read time

Up 15 Down 1 Comment 1 Share 1

The bridge to modern React

#webdev #javascript #react +2

Dec 28, 2024 • 6m read time

Up 4 Down 1 Comment 1 Share 1

Can React Router v7 Save React?

Oct 11, 2024 • 1m read time

Hi everyone!! I'm new to this group I have

React Router Official Documentation

Oct 11, 2024 • 1m read time

Understanding Layout Components and React Router Outlet in React

Oct 11, 2024 • 1m read time

Big with React Router code

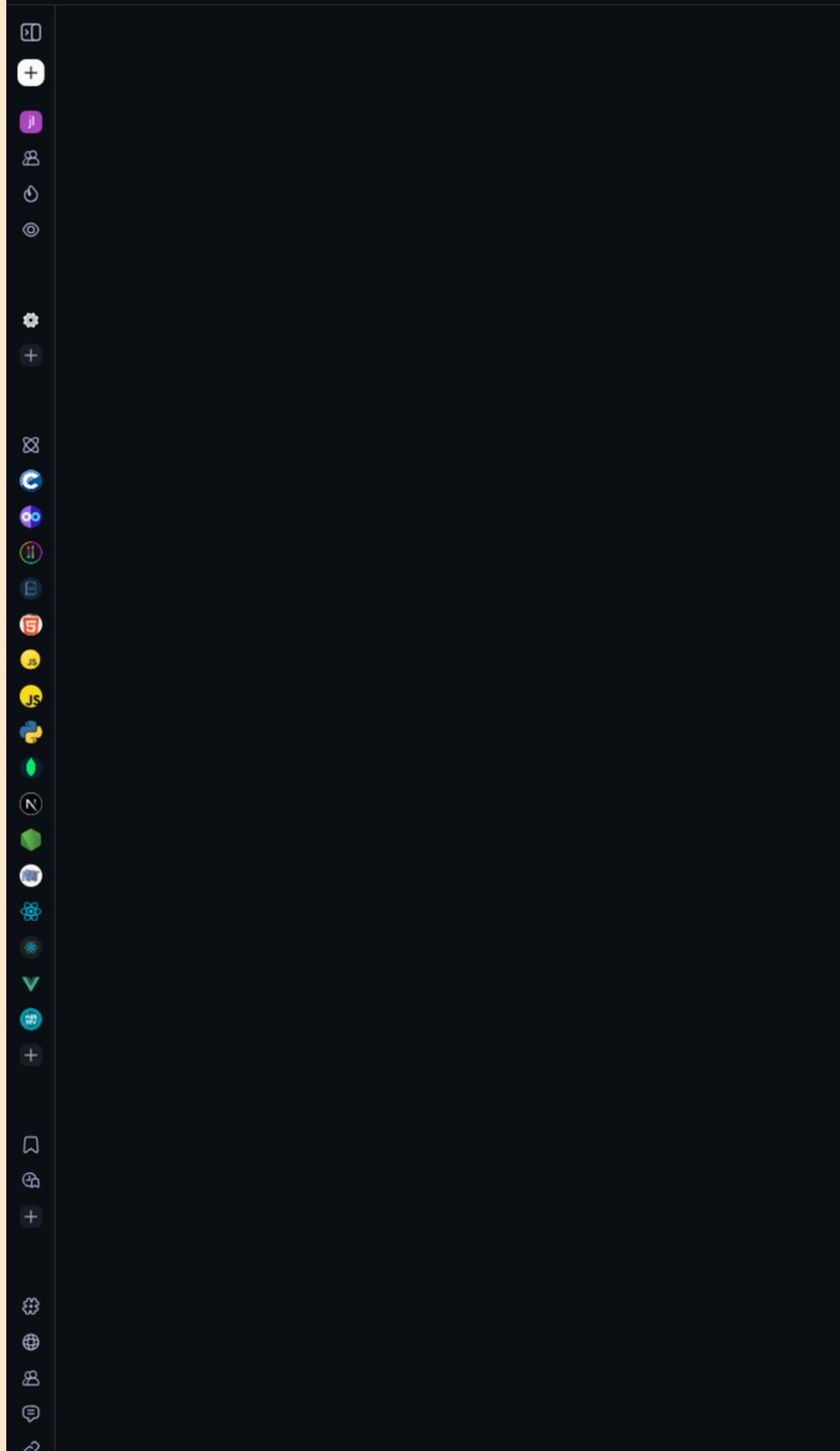
Oct 11, 2024 • 1m read time

Hi everyone!! I'm new to this group I have

Fog of War

Oct 11, 2024 • 1m read time

Mastering React Router Hooks: A Comprehensive Guide



Express.js developers can now add authorization in minutes with Amazon Verified Permissions

TLDR Simplify it Remove fluff Challenge this 5+ More

AWS released an open source JavaScript package that enables Express.js developers to implement authorization using Amazon Verified Permissions and Cedar policies. The package moves authorization logic outside application code into externally managed policies, allowing developers to define role-based access controls without Show more

#security #javascript #aws #express + #authorization +

Today • 2m read time • From aws.amazon.com



1 Upvote

Upvote Downvote Comment Bookmark Copy

Sort: Oldest first

jl Share your thoughts Post



Promoted by Carbon

Transform downtime into uptime triumphs. Drive productivity with CloudBees CI's HA tools.

Be the first to comment.

Read post

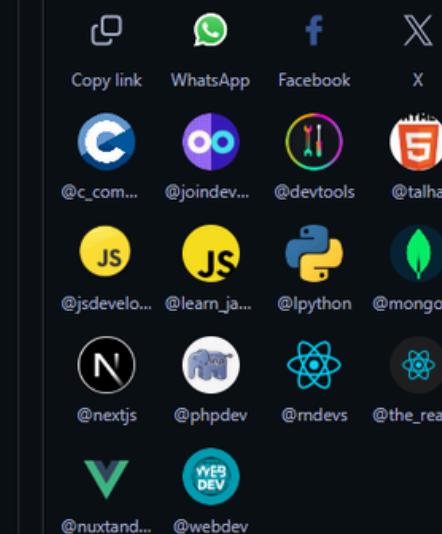


AWS
@aws

Follow

⋮

Would you recommend this post?



Best discussions

How to Harden Your Node.js APIs – Security Best Practices

11 Comments

Hyper: Simple React alternative

16 Comments

React library for LLMs

18 Comments

I'm feeling lucky

© 2025 Daily Dev Ltd. Guidelines Explore Tags

Sources Squads Leaderboard

Express@5.1.0: Now the Default on npm with LTS Timeline

Blog posts

How Express.js Rebuilt Its

Vulnerability Reporting Process

May 2025 Security Releases

Spring Cleaning in Express.js: Deprecations and the Path Ahead

Express@5.1.0: Now the Default on npm with LTS Timeline

A New Chapter for Express.js: Triumphs of 2024 and an ambitious 2025

Express.js Security Audit: A Milestone Achievement

Introducing Express v5: A New Era for the Node.js Framework

Express Never Ending Support Launched by HeroDevs and Express.js

September 2024 Security Releases

Welcome to The Express Blog!

By [Express Technical Committee](#)

31 Mar 2025

Express v5.0.0 was released on September 9th last year, but we didn't make it the `latest` release on npm. Many asked us why and when it would be, and frankly we were not ready at the time to take that jump. If you have not followed the news from the project this past year, we have been [hard at work reviving the project](#) and when we pushed the initial v5 release there were many loose ends still hanging. So first lets quickly go over some of those loose ends.

Documentation updates

We had not updated the docs, provided migration guides, or even fully reviewed some of the stagnated v4/5 docs in a long time. Since then we have had tons of great contributors help get things into better shape. As with any volunteer based Open Source project, we love contributions to help us improve so as you upgrade please continue to open PRs to fix anything we missed.

You can find our [v5 docs](#) and our [migration guide](#) on the website.

Migration Support

We know that migrating between versions can be challenging, especially when it involves significant changes in a widely used framework like Express. That's why we have worked on a solution to simplify part of the process and reduce the impact on developers.

Thanks to the incredible efforts of [Sebastian](#) and [Filip](#), we have developed a new [codemod package](#) specifically designed to facilitate the transition from Express v4 to v5, as well as future major versions. This package automates many of the necessary code changes, minimizing manual effort and making the upgrade as smooth and efficient as possible.

However, we understand that not all changes can be automated. Some breaking changes, such as the [new Path Route Matching syntax](#), require manual modifications by developers. You can read more about all of the breaking changes which came with v5 in our [original release announcement](#).

For more details on the migration process and how to use the codemod package, check the [repository's README](#) and the [migration guide](#).

Ecosystem compatibility

The Express ecosystem is one of its strongest assets. It goes back to the early days of Node.js and is the backbone that keeps express popular. When it goes [10 years without a major release](#) everything from middleware to documentation needed updates. We wanted to make sure folks had some time to get all of that updated before we had everyone moving over. Particularly we care about our very large beginner user base who may not know the blog post they are reading is not compatible with what they get from `npm i express`.

We recognize that some friction is inevitable during major upgrades, but thanks to work from ecosystem partners like [Kamil](#) from NestJS [working to update express before we went latest](#) we will hopefully be ahead of the curve. And as I said above, we always welcome help to make this transition easier for those who follow after you, PRs are the best support you can give.

Long Term Support

We had been discussing how to support v4 now that v5 was out, but we had not defined a clear guideline or expectation, and we had (still don't have) end user docs on our plans here. While we still have progress to make here, we have a [proposed LTS strategy](#) which will be the basis for our forthcoming docs. Input is very welcome on this so we can make sure it is clearly communicated and acceptable to the community.

Additionally since then [we have announced a partnership with HeroDevs](#) to help companies who are less capable of updating. More information on how this will work when v4 EOL will come when we get closer to that time.

Support Phases and Going latest

What does it mean to "go latest"? If you are unfamiliar with how npm `dist-tags` work, the `latest` tag is what users will get when they run `npm install express`. This is important because it means it is the "default installed version" and will trigger the transition of nearly 17 million weekly downloads from our current latest v4.21.2 to v5. As we start this transition we



Top chrono, voilà un tour d'horizon des **meilleurs outils, méthodes, accessoires et bonnes pratiques** pour faire une veille techno efficace en développement web :

1. Outils de veille technologique

Agrégateurs & lecteurs RSS

- [Feedly](#) : centralise les flux RSS de blogs, sites d'actu, et dev newsletters.
- [Inoreader](#) : similaire à Feedly, très personnalisable.
- [Flipboard](#) : style magazine, bonne découverte visuelle.

Plateformes sociales & communautés

- [Twitter](#) : suivre les dev influents, hashtags comme #webdev, #javascript, #css...
- [Reddit \(r/webdev, r/javascript, r/frontend\)](#) : discussions, news, projets open source.
- [Dev.to](#) : articles et tutoriels écrits par la communauté dev.
- [Hacker News](#) : tech, startups, innovation.

Newsletters spécialisées

- [JavaScript Weekly](#), [Frontend Focus](#), [CSS-Tricks Newsletter](#), [Smashing Newsletter...](#)
Un bon mix à recevoir chaque semaine pour ne rien rater.

Outils de curation

- [Pocket](#) : sauvegarder des articles pour plus tard, lire offline.
- [Raindrop.io](#) : bookmarker et organiser liens avec tags et dossiers.

Plateformes de découverte de projets

- [GitHub Trending](#) : projets populaires du moment.
- [Product Hunt](#) : nouveautés tech & outils.



Poser une question

+ Outils



Méthode de classification des sources d'information.

Pour organiser les évolutions technologiques dans le monde du dev web :

- Je sélectionne les thématiques ex : Framework, Front-end, Backend, Sécurité, déploiement, etc.
- Je vérifie les sources lorsqu'elles sont partagées.
- Je vérifie la fiabilité et la pertinence.
- La date de publication.
- Ma capacité a apprécié un outil, une technologie.
- La place d'une technologie et de son impacte sur le marché/entreprises.

En définitive, je préfère me rendre directement sur le site web d'une technologie afin être certain de son authenticité. Les plateformes comme YouTube, DailyDev, ChatGPT me permettent une interaction variée afin d'apprendre et de rester en marge et à l'affût des dernières technologies ou leurs évolutions.

Pourquoi faire de la veille?

POURQUOI FAIRE DE DE LA VEILLE?



La veille permet de :

- Choisir des librairies fiables (React Router, React Hook Form, Firebase).
- Spécifications précises sur les dépendances à installer.
- Identifier les meilleures pratiques (ex : sécurité, structuration BDD).
- Exigences techniques : utilisation de JWT, nommage des collections Mongo.
- Adapter l'interface aux standards UX/UI modernes.
- Spécifications fonctionnelles liées au responsive et à l'accessibilité.
- Anticiper les limitations techniques (ex : quotas Firebase, limites de stockage).
- Contraintes techniques documentées dans les specs.
- Être pertinent en toute situation et être force de proposition au sein d'une équipe.

Conclusion

Objectif du projet

Le projet Menu Maker consiste à développer une application web responsive destinée aux restaurateurs. Elle leur permet de créer, gérer et publier leurs menus en ligne via une interface simple, sécurisée et intuitive.

Stack technique utilisée

L'application repose sur le stack MERN (MongoDB, Express, React, Node.js), avec Firebase pour l'authentification.

- Le front-end est développé en React, avec React Router pour la navigation et React Hook Form pour la gestion des formulaires.
- Le back-end est construit avec Express.js et permet la création d'une API REST sécurisée.
- Les données sont stockées dans une base MongoDB, avec des modèles définis via Mongoose.
- L'application est hébergée sur des plateformes comme Vercel pour le front et Render ou Railway pour l'API.

Fonctionnalités principales

L'utilisateur peut :

Créer un nouveau menu, y ajouter des plats et catégories.

Prévisualiser le menu en temps réel.

Modifier ou supprimer un menu existant.

Gérer tous ses menus via une interface d'administration.

Toutes les actions sont sécurisées par une authentication Firebase.

QUESTIONS ?