

Neurona - Apprends, Partage, Progresse
Développeur Web & Web Mobile
Thomas PENA-BERMOND
09/01/2025

Table des matières avec pagination

Introduction.....	3
Présentation générale.....	3
Contexte.....	3
Objectifs.....	3
Cahier des charges.....	5
Besoins fonctionnels.....	5
Besoin techniques.....	5
Utilisateurs cibles.....	6
Contraintes.....	6
Gestion de projet.....	6
Méthodologie.....	6
Outils.....	6
Planning & Répartition des tâches.....	7
Analyse et conception.....	7
Diagrammes et architecture.....	7
Schéma de base de données.....	7
Maquettes / wireframes.....	8
Choix techniques et justification.....	8
Développement.....	8
Fonctionnalité principale.....	8
Exemple de code & API.....	8
Sécurité.....	9
Accessibilité & Responsive design.....	9
Tests et Validation.....	9
Statégie de tests.....	9
Exemples de cas.....	9
Difficultés rencontrés.....	10
Problèmes techniques et organisationnels.....	10
Résolutions des problèmes.....	11
Bilan et perspective.....	11
Bilan.....	11
Perspective.....	12
Axes d'amélioration.....	12
Evolutions possibles.....	12
Conclusion.....	13
Annexes.....	14

Introduction

Ce dossier présente la réalisation du projet Neurona, une plateforme collaborative augmentée par l'intelligence artificielle. Ce projet a été développé dans le cadre de la validation du titre professionnel Dwwm (Développeur Web & Web Mobile). Il synthétise les compétences acquises en conception du développement front-end et back-end.

Présentation générale

Contexte

Le développement de l'information évolue à grande vitesse. Les développeurs font face à une documentation souvent trop technique, ou des forums souvent obsolètes. La recherche d'information devient fastidieuse. C'est dans ce contexte qu'est né Neurona, un "hub" de connaissances centralisé maintenu par la communauté et assisté par l'IA pour garantir sa pertinence et sa clarté.

Objectifs

L'objectif principal est de fournir un outil qui simplifie l'apprentissage continu :

- Pour les juniors : Offrir un parcours d'apprentissage progressifs et explicite.
- Pour les seniors : Proposer un espace pour partager l'expertise et rester informer.

Résumé du projet

Neurona est une application web (SaaS) pensée pour les développeurs qui cherchent à apprendre, progresser et partager leurs connaissances dans un environnement collaboratif. Son rôle est de centraliser en un seul endroit ce qui est souvent dispersé : la documentation, les discussions communautaires et les actualités.

Cahier des charges

Besoins fonctionnels

- Authentification & Gestion de compte : Inscription, Connexion, Modification, Suppression.
- Documentation : Création assistée par IA, Recherche, Consultation structurée.
- Communauté : Espace d'entraide permettant de poster des sujets.
- Veille : Intégration d'articles récents via api pour permettre aux utilisateurs de suivre l'actualité.

Besoin techniques

- Front-end : Interface simple, composant réutilisables.
- Back-end : API sécurisé, gestion de base de données.
- Performance : Chargement rapide, optimisation des requêtes IA.

Utilisateurs cibles

- Développeurs en formation ou juniors.
- Développeurs expérimentés : Formateurs ou Mentors Techniques.

Contraintes

- Respect du RGPD : Protection des données personnelles.
- Accessibilité : Respect des normes WCAG.
- Budget : Utilisation de technologies open-source et gratuites.

Gestion de projet

Méthodologie

Le projet a suivi une méthodologie Agile (Scrum) :

- Utilisation de Notion pour la gestion du projet.
- Utilisation de Trello pour le suivi des tâches.
- Utilisation de Clockify pour pour le suivi.
- Utilisation de Canva pour la présentation du projet.

Outils

- Notion : Centralisation des ressources (Note d'intention, cahier des charges, etc.).
- Trello : Gestion des sprints avec tableau (A faire, En cours, En attente, Terminer).
- Clockify : Mesure du temps passé par tâche.
- Canva : Création des supports de présentations

Planning & Répartition des tâches

- Semaine 1 (27/10 - 05/11): Initialisation du projet, mise en place de l'authentification back-end, débogage des fournisseurs OAuth, et début de l'implémentation front-end.
- Semaine 2 (05/11 - 12/11): Développement des fonctionnalités "Membres", intégration de l'IA pour la génération de documents.
- Semaine 3 (12/11 - 19/11): Développement des fonctionnalités communautaire (CRUD), intégration du système veille technologique, et tests sur les fonctionnalités de l'ia.
- Semaine 4 (19/11 - 21/11): Graphique (scss, responsive mobile, dark mode), mise en place de Google Analytic, déploiement sur vercel/render et préparation de l'oral.

Analyse et conception

Diagrammes et architecture

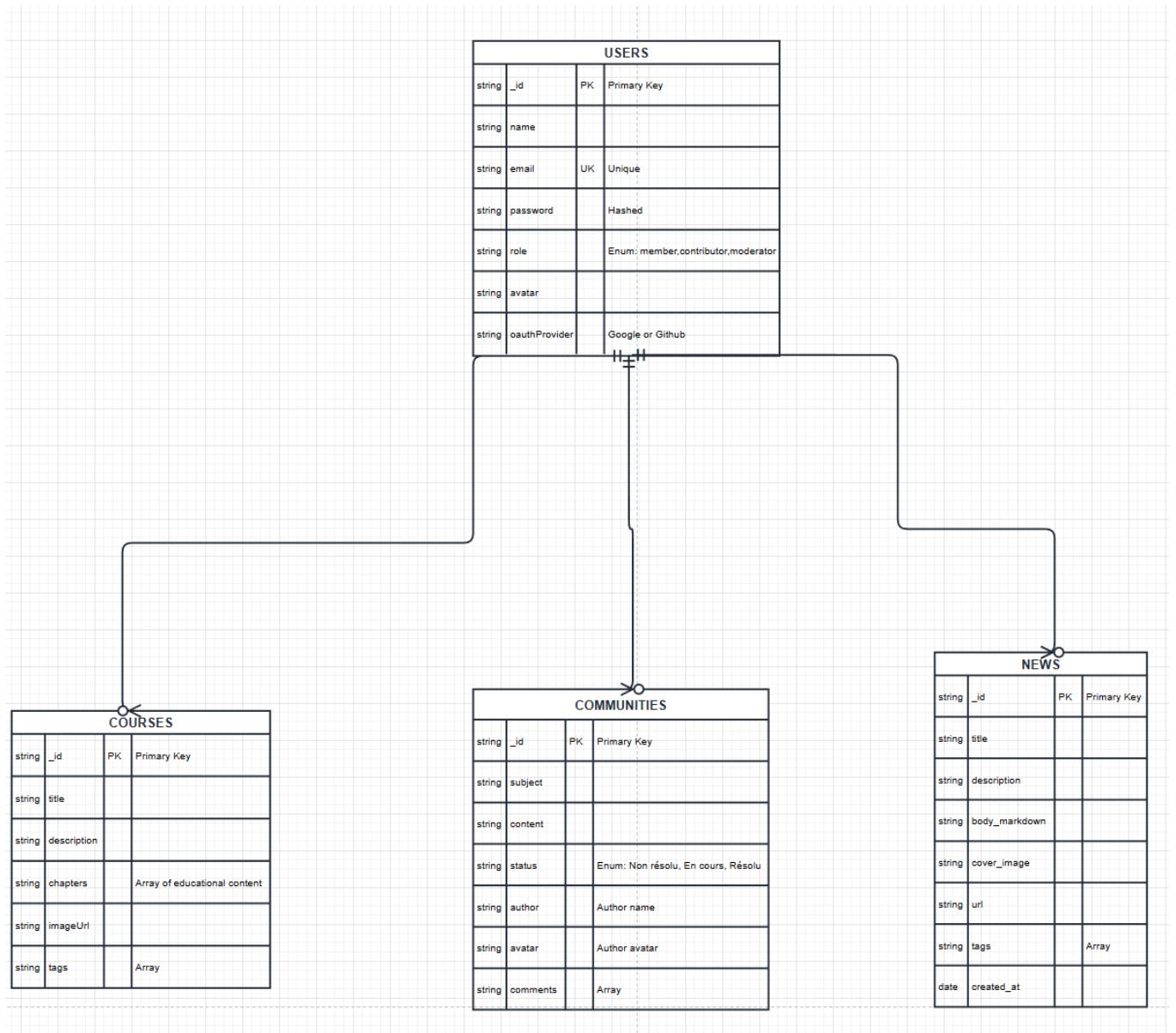
L'architecture retenue est de type MVC (Modèle-Vue-Contrôleur) coté back-end, utilisée via une API REST par le front-end.

- Client : React + Vite
- Serveur : Node + Express
- Base de données : MongoDB

Schéma de base de données

Utilisation de Mongoose pour modéliser les données.

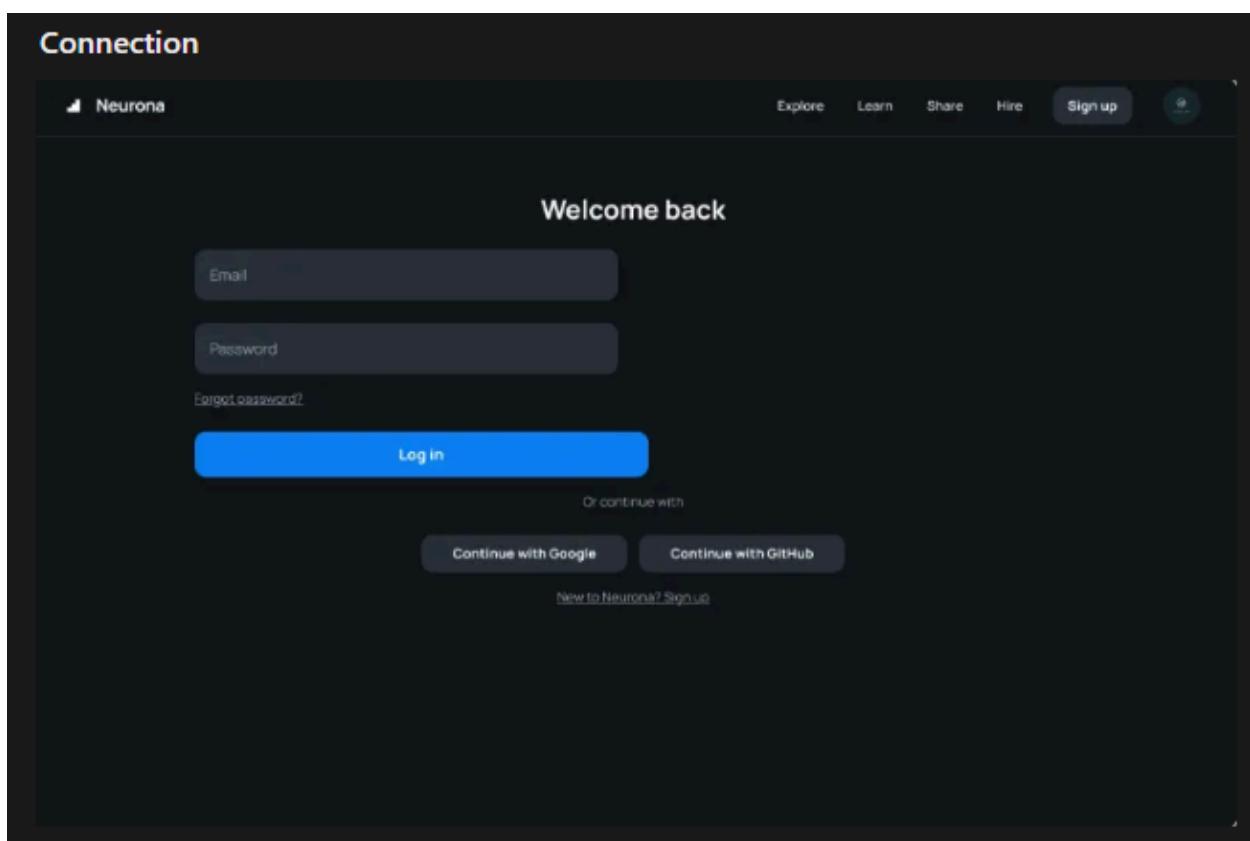
- User : avatar, name, email, password, role.
- Course : title, descriptions, chapters, imgURL, tags.
- Community : author, subject, content, status, comments.
- Veille : title, description, body_markdown, cover_image, tags, url.



Maquettes / wireframes

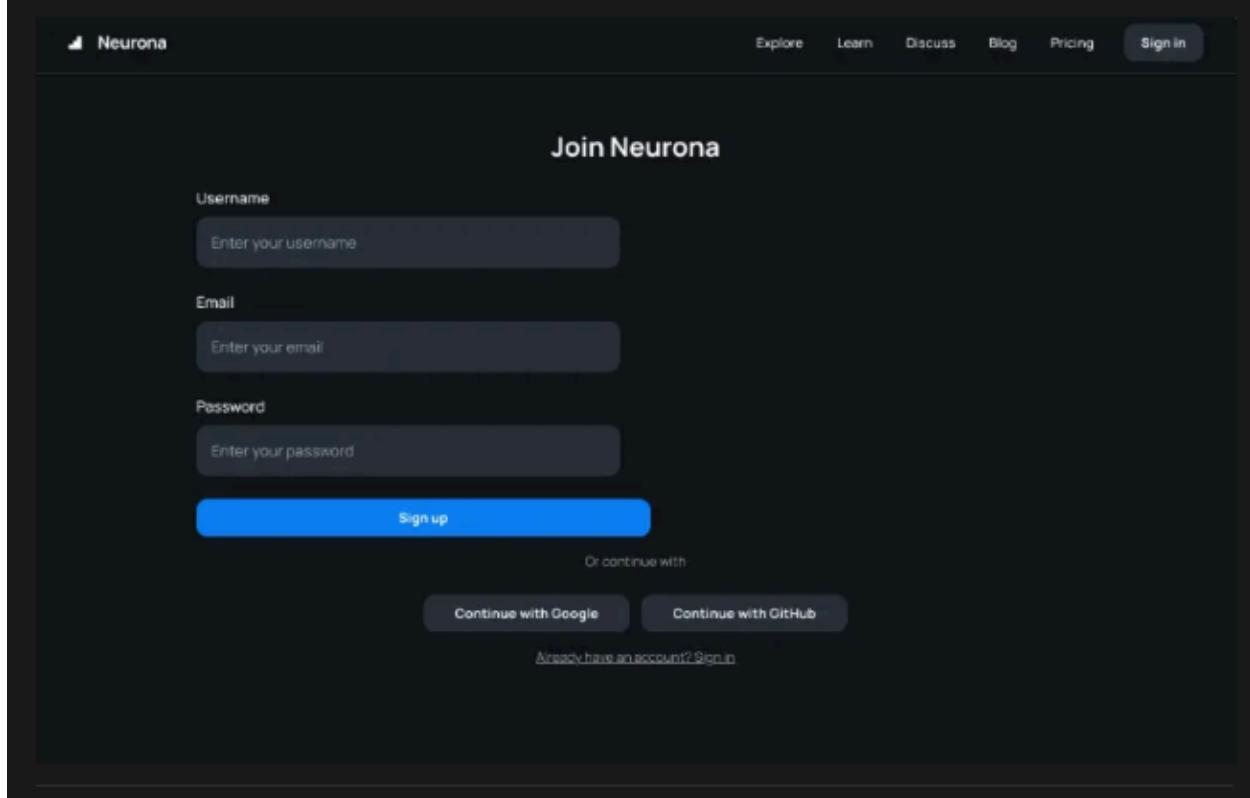
Les maquettes ont été réalisées avant le développement pour valider l'UX/UI. L'accent a été mis sur la lisibilité qui est essentielle pour une plateforme de lecture longue.

Connection



The Connection page features a dark header with the Neurona logo and navigation links for Explore, Learn, Share, Hire, Sign up, and a user icon. The main area has a dark background with light-colored input fields. It displays a "Welcome back" message above two input fields labeled "Email" and "Password". Below the password field is a "Forgot password?" link. A large blue "Log in" button is centered below the fields. Below the button, a "Or continue with" section offers "Continue with Google" and "Continue with GitHub" options. At the bottom, a link reads "New to Neurona? Sign up".

Inscription



The Inscription page features a dark header with the Neurona logo and navigation links for Explore, Learn, Discuss, Blog, Pricing, Sign in, and a user icon. The main area has a dark background with light-colored input fields. It displays a "Join Neurona" message above three input fields labeled "Username", "Email", and "Password". Each field has a placeholder text: "Enter your username", "Enter your email", and "Enter your password". Below the password field is a "Sign up" button. Below the button, a "Or continue with" section offers "Continue with Google" and "Continue with GitHub" options. At the bottom, a link reads "Already have an account? Sign in".

Page d'accueil

More than a forum. Better than a blog.

Neurona is a platform for developers to learn, share, and collaborate, explore tutorials, documentation, and community discussions to enhance your skills and connect with fellow developers.

Recherche

Documentation

Search documentation

Language ▾ Level ▾

Featured

Featured

Getting Started with Neurona
Learn the basics of Neurona and start building your first project.

All Documentation

- Environment Setup**
Learn how to set up your development environment for Neurona projects.
- Core Concepts**
Understand the core concepts and architecture of Neurona.
- API Reference**
Explore the Neurona API and learn how to interact with it.
- Deployment Guide**
Learn how to deploy your Neurona applications to production.
- Troubleshooting**
Troubleshoot common issues and find solutions to problems.

Communauté

Neurona Home Learn Projects Community Events Jobs Search

Community

All Posts Following My Posts Saved

Top Posts

 Need help with Django backend setup
I'm working on a new project and need some help with the backend. I'm using Python and Django, but I'm having trouble with the database setup. Any advice would be greatly appreciated!
120 votes · 20 comments · 2 days ago

 Best resources for learning machine learning
I'm looking for a good resource to learn about machine learning. I'm a beginner, so I need something that's easy to understand. Any recommendations?
95 votes · 15 comments · 3 days ago

 Debugging JavaScript applications
I'm trying to debug a JavaScript application, but I'm having trouble finding the source of the error. Any tips for debugging JavaScript?
80 votes · 10 comments · 4 days ago

New Posts

 Need help with React frontend setup
I'm working on a new project and need some help with the frontend. I'm using React, but I'm having trouble with the component structure. Any advice would be greatly appreciated!
50 votes · 5 comments · 1 day ago

 Best resources for learning data science
I'm looking for a good resource to learn about data science. I'm a beginner, so I need something that's easy to understand. Any recommendations?
30 votes · 3 comments · 2 days ago

 Debugging Python applications
I'm trying to debug a Python application, but I'm having trouble finding the source of the error. Any tips for debugging Python?
10 votes · 1 comment · 3 days ago

Documentation

Neurona Home Explore Notifications Create Search

News & Updates

The State of JavaScript 2023
A comprehensive survey of the JavaScript ecosystem, covering frameworks, tools, and trends.



React Conf 2023 Highlights
Key takeaways and announcements from the annual React conference, including new features and updates.



Node.js v20 Released
The latest version of Node.js introduces performance improvements, new APIs, and enhanced security features.



The Future of WebAssembly
Exploring the potential of WebAssembly for building high-performance web applications.

Communauté

Neurona Home Learn Projects Community Events Jobs

Search

Community

All Posts Following My Posts Saved

Top Posts

 Need help with Django backend setup
I'm working on a new project and need some help with the backend. I'm using Python and Django, but I'm having trouble with the database setup. Any advice would be greatly appreciated!
120 votes · 20 comments · 2 days ago

 Best resources for learning machine learning
I'm looking for a good resource to learn about machine learning. I'm a beginner, so I need something that's easy to understand. Any recommendations?
95 votes · 15 comments · 3 days ago

 Debugging JavaScript applications
I'm trying to debug a JavaScript application, but I'm having trouble finding the source of the error. Any tips for debugging JavaScript?
80 votes · 10 comments · 4 days ago

New Posts

 Need help with React frontend setup
I'm working on a new project and need some help with the frontend. I'm using React, but I'm having trouble with the component structure. Any advice would be greatly appreciated!
50 votes · 5 comments · 1 day ago

 Best resources for learning data science
I'm looking for a good resource to learn about data science. I'm a beginner, so I need something that's easy to understand. Any recommendations?
30 votes · 3 comments · 2 days ago

 Debugging Python applications
I'm trying to debug a Python application, but I'm having trouble finding the source of the error. Any tips for debugging Python?
10 votes · 1 comment · 3 days ago

Documentation / veille

Neurona Home Explore Notifications Create

Search

News & Updates

The State of JavaScript 2023
A comprehensive survey of the JavaScript ecosystem, covering frameworks, tools, and trends.



React Conf 2023 Highlights
Key takeaways and announcements from the annual React conference, including new features and updates.



Node.js v20 Released
The latest version of Node.js introduces performance improvements, new APIs, and enhanced security features.



The Future of WebAssembly
Exploring the potential of WebAssembly for building high-performance web applications.



Profil

Neurona

Profile



Alex Turner
Software Engineer
Joined in 2021

[Edit profile](#)

About

Location

Website

Settings

- [Account](#)
Manage your account settings
- [Notifications](#)
Manage your email notifications
- [Privacy](#)
Manage your privacy settings
- [Security](#)
Manage your security settings
- [Billing](#)
Manage your billing information
- [Help](#)
Get help and support
- [About](#)
Learn more about Neurona

More

Choix techniques et justification

Le projet a suivi une technique moderne et cohérente : MERN (MongoDB, Express, React, Node). Ce choix se justifie par l'utilisation du langage JavaScript (ou TypeScript) à la fois côté client et serveur, ce qui facilite le développement et la maintenance. MongoDB est idéal pour stocker des documents JSON flexibles comme ceux générés par l'IA. Tailwind est utilisé spécifiquement pour l'intégration des composants shadcn/ui, tandis que le scss a été retenu pour le styling personnalisé afin de garantir une structure propre et modulaire.

Développement

Fonctionnalité principale

- Authentification & Gestion de compte : Système sécurisé par Token JWT & hachage du mot de passe.
- Documentation : Générateur de documentation via Google Gemini.
- Communauté : CRUD (Create, Read, Update, Delete).
- Veille : Système d'affichage d'articles récupérés via api.

Exemple de code & API

```

const { GoogleGenAI } = require("@google/genai");
const { z } = require("zod");
const { zodToJsonSchema } = require("zod-to-json-schema");
const Course = require("../models/docModel");

// 1. Définition de la structure exacte que l'IA doit renvoyer (Validation Zod)
const docSchema = z.object({
  imageKeywords: z
    .string()
    .describe("Mots-clés pour trouver une image d'illustration"),
  title: z
    .string()
    .describe("Titre principal du cours"),
  // ... (autres champs définis comme difficulté, chapitres, etc.)
  chapters: z.array(z.object({
    title: z.string(),
    content: z.string().describe("Contenu éducatif du chapitre")
  })).describe("Liste des chapitres du cours")
});

exports.getDocumentation = async (req, res) => {
  try {
    const subject = req.body.subject; // Sujet demandé par l'utilisateur
    // 2. Création du prompt pour l'IA
    const prompt = `

Génère une documentation technique complète sur le sujet suivant : "${subject}".
Retourne un JSON valide respectant ce schéma.
`;

    // 3. Connexion à l'IA (Google Gemini)
    const ai = new GoogleGenAI({ apiKey: process.env.GEMINI_API_KEY });
    // 4. Génération du contenu
    const request = await ai.models.generateContent({
      model: "gemini-2.0-flash-lite",
      contents: prompt,
      config: {
        responseMimeType: "application/json", // On force le format JSON
        responseJsonSchema: zodToJsonSchema(docSchema), // On impose notre structure
      },
    });

    // 5. Validation et nettoyage des données reçues
    const doc = docSchema.parse(JSON.parse(request.text));

    // 6. Récupération d'une image gratuite via l'API Pexels basée sur les mots-clés de l'IA
    const imgRes = await fetch(
      `https://api.pexels.com/v1/search?query=${doc.imageKeywords}&per_page=1`,
      { headers: { Authorization: process.env.PEXELS_API_KEY } }
    );
    const imgData = await imgRes.json();
    const imageUrl = imgData.photos?.[0]?.src?.large || "url_par_defaut.jpg";

    // 7. Sauvegarde du cours complet en base de données
    const course = await Course.create({ ...doc, imageUrl });
    // 8. Réponse au client (Frontend) avec le cours créé
    res.status(201).json(course);
  } catch (e) {
    res.status(500).json({ message: e.message });
  }
};

```

Sécurité

- Mots de passe : Hachage avec bcrypt avant stockage.
- Authentication : Token JWT pour sécuriser les routes.
- Protection HTTP : Helmet pour les headers de sécurité et Cors pour contrôler l'accès aux ressources.
- Validation : Zod pour valider les données entrantes.

Accessibilité & Responsive design

- Utilisation de balises sémantiques HTML5.
- Contraste des couleurs pour le confort de lecture.

Tests et Validation

Stratégie de tests

- Tests Unitaires : Vérification des fonctions isolées.
- Tests d'Intégration : Test des endpoints avec Thunder Client.
- Tests End-To-End : Test du parcours utilisateurs.

Exemples de cas

- Tentative d'inscription avec email existant : L'API doit retourner une erreur 401 "Email déjà utilisé".
- Accès à une route protégée sans token : Le middleware d'auth doit bloquer la requête et renvoyer une 401.
- Génération de doc avec prompt vide : Le contrôleur renvoie une 400.

Difficultés rencontrées

Problèmes techniques et organisationnels

- Organisationnel : Sous-estimation du temps nécessaire pour finaliser l'application.
- CORS : La gestion des cookies entre le front-end et le back-end à demandé une configuration spéciale.
- IA : Le format de réponse de l'IA n'était pas toujours un json valide ce qui faisait planter le front-end.

Résolutions des problèmes

- Organisationnel : Priorisation des fonctionnalités MVP (Minimum Viable Product) en reportant les fonctionnalités complexes.
- CORS : Configuration avec origin et credentials.
- IA : Utilisation de la librairie zod pour une schématisation du json avant de l'envoyer au front-end.

Bilan et perspective

Bilan

Bien que l'application ne soit pas entièrement finalisée, l'objectif pédagogique était avant tout de montrer ce dont je suis capable. La contrainte de travailler en totale autonomie, due à une situation personnelle complexe, m'a poussé à dépasser mes limites techniques. Ce qui m'a permis de consolider mes compétences sur la stack MERN et de gérer seul les imprévues d'un projet réel. Aujourd'hui, je suis capable de concevoir, développer, et déployer une application sécurisée en toute indépendance. Actuellement en stage sur un nouveau projet, de nouveau en autonomie, je continue de mettre à profit cette expérience pour m'améliorer de jour en jour.

Perspective

- Finaliser l'application : Terminer l'intégration des pages et les fonctionnalités.

Axes d'amélioration

- SEO : Améliorer le référencement naturel via du Server Side Rendering (SSR).

Evolutions possibles

- Premium : Intégration d'un mode premium permettant à l'utilisateur de générer ses propres documentations.

Annexes

- Lien application : <https://neurona-xi.vercel.app/>