Documentation de Déploiement - Alexia Énergies

Table des Matières

Vue d'ensemble du projet

Nom du projet : Alexia Énergies
Type : Application web de bien-être (Site vitrine + Administration)
Framework : Next.js 15.5.3
Base de données : PostgreSQL (Neon)
Hébergement actuel : Vercel
URL de production : alexia-energies.vercel.app

Architecture technique

- **Frontend :** Next.js avec React 19
- **Backend :** API Routes Next.js
- **Base de données :** PostgreSQL via Neon

- **ORM :** Prisma 6.13.0
- **Authentification :** JWT + BcryptJS
- **Upload d'images :** Cloudinary
- **Emails :** Resend + EmailJS
- **Rendez-vous :** Calendly (intégration)

Prérequis

Avant de commencer le déploiement, assurez-vous d'avoir :

Outils nécessaires

- **Node.js** (version 18.17 ou supérieure)
- **npm**, **yarn**, **pnpm** ou **bun**
- **Git** pour le versioning
- Un compte sur la plateforme de déploiement choisie

Comptes de services externes

- **Neon** (base de données PostgreSQL)
- **Cloudinary** (gestion des images)
- **Resend** (envoi d'emails)
- **Calendly** (prise de rendez-vous)
- **Vercel** (recommandé pour l'hébergement)

Configuration de l'environnement

Option A : Cloner le projet existant

1. Cloner le repository

```
git clone https://github.com/fannysaez/alexia-energies.git
cd alexia-energies
```

Option B : Créer une nouvelle application Next.js

1. Création d'un nouveau projet Next.js

Si vous souhaitez créer un nouveau projet Next.js à partir de zéro, utilisez la commande suivante :

```
npx create-next-app@latest nom-du-projet
```

Lors de l'exécution, vous serez invité à configurer les options suivantes :

- **TypeScript :** `No` (ou `Yes` selon vos préférences)
- **ESLint :** `Yes` (recommandé pour la qualité du code)
- **Tailwind CSS :** `No` (le projet utilise du CSS modules)
- **App Router :** `Yes` (recommandé - architecture moderne de Next.js)
- **Turbopack :** `No` (optionnel pour le développement)
- **Import alias :** `Yes` (facilite les imports avec `@/`)

Exemple de sortie console :

```
Creating a new Next.js app in C:\Users\fanny\Desktop\Technos\Next.js\projet-stage-freelance-dwwm\alexia-energies.

Using npm.

Initializing project with template: app

Installing dependencies:
- react
- react-dom
- next

Installing devDependencies:
- eslint
- eslint-config-next
- @eslint/eslintrc

Success! Created alexia-energies at C:\Users\fanny\Desktop\Technos\Next.js\projet-stage-freelance-dwwm\alexia-energies
```

2. Accéder au projet et démarrer le serveur

```
cd nom-du-projet
npm run dev
```

Le serveur de développement sera accessible sur :

- **Local :** http://localhost:3000
- **Network :** http://192.168.95.1:3000

Vérification du build de production :

```
npm run build
```

Cette commande génère une version optimisée pour la production avec :

- Compilation réussie en ~13.0s
- Vérification des types
- Collection des données de pages
- Génération des pages statiques
- Optimisation finale

3. Structure du projet Next.js généré

Après la création, votre projet aura cette structure de base :

```
nom-du-projet/
├── public/              # Fichiers statiques (images, favicon, etc.)
├── src/
│   └── app/            # App Router (Next.js 13+)
│       ├── globals.css # Styles globaux
│       ├── layout.js   # Layout principal
│       └── page.js     # Page d'accueil
├── .eslintrc.json      # Configuration ESLint
```

```
├── .gitignore          # Fichiers à ignorer par Git
├── jsconfig.json       # Configuration JavaScript/TypeScript
├── next.config.mjs     # Configuration Next.js
├── package.json        # Dépendances et scripts
└── README.md           # Documentation du projet
```

Fichiers clés à comprendre :

- `package.json` : Contient les dépendances et scripts npm
- `next.config.mjs` : Configuration avancée de Next.js
- `src/app/layout.js` : Template principal de l'application
- `src/app/page.js` : Page d'accueil par défaut
- `public/` : Dossier pour les ressources statiques

3. Installation des dépendances (pour les deux options)

```
npm install
# ou
yarn install
# ou
pnpm install
```

4. Configuration des variables d'environnement

Créez un fichier `.env.local` à la racine du projet :

```
# Base de données
DATABASE_URL="postgresql://username:password@host:port/database?sslmode=require"

# JWT Secret
JWT_SECRET="votre-clé-secrète-très-sécurisée"

# Cloudinary
CLOUDINARY_CLOUD_NAME="votre-cloud-name"
CLOUDINARY_API_KEY="votre-api-key"
CLOUDINARY_API_SECRET="votre-api-secret"

# Resend (emails)
RESEND_API_KEY="votre-clé-resend"

# EmailJS
NEXT_PUBLIC_EMAILJS_SERVICE_ID="votre-service-id"
NEXT_PUBLIC_EMAILJS_TEMPLATE_ID="votre-template-id"
NEXT_PUBLIC_EMAILJS_PUBLIC_KEY="votre-clé-publique"

# Calendly
NEXT_PUBLIC_CALENDLY_URL="votre-url-calendly"

# URL de base (production)
NEXTAUTH_URL="https://votre-domaine.com"
```

5. Configuration de la base de données

```
# Générer le client Prisma
npx prisma generate

# Appliquer les migrations
npx prisma migrate deploy
```

Déploiement sur Vercel

Méthode 1 : Via l'interface web Vercel

1. **Connectez-vous à Vercel**
2. **Cliquez sur "New Project"**
3. **Importez votre repository GitHub**
4. **Configuration du projet :**
   - Framework Preset : Next.js
   - Root Directory : ./
   - Build Command : `npm run build`
   - Output Directory : `.next`
   - Install Command : `npm install`
5. **Variables d'environnement :**
   - Ajoutez toutes les variables du fichier `.env.local`
   - **Important :** Ne pas inclure les variables commençant par `NEXT_PUBLIC_` si elles contiennent des informations sensibles
6. **Déployez le projet**

Méthode 2 : Via CLI Vercel

```
# Installation de Vercel CLI
npm i -g vercel

# Connexion à votre compte
vercel login

# Déploiement
vercel

# Pour un déploiement en production
vercel --prod
```

Configuration de la base de données

Configuration Neon (PostgreSQL)

1. **Créez un compte sur Neon**
2. **Créez une nouvelle database**
3. **Récupérez l'URL de connexion :**

```
postgresql://username:password@ep-xxx.us-east-1.aws.neon.tech/neondb?sslmode=require
```

4. **Configuration des migrations automatiques :**

Dans votre pipeline de déploiement, ajoutez :

```
# Avant le build
npx prisma migrate deploy
npx prisma generate
```

Variables d'environnement

Variables obligatoires

Variable	Description	Exemple
`DATABASE_URL`	URL de connexion PostgreSQL	`postgresql://user:pass@host/db`
`JWT_SECRET`	Clé secrète pour JWT	`your-super-secret-key`
`CLOUDINARY_CLOUD_NAME`	Nom du cloud Cloudinary	`your-cloud-name`
`CLOUDINARY_API_KEY`	Clé API Cloudinary	`123456789012345`
`CLOUDINARY_API_SECRET`	Secret API Cloudinary	`your-api-secret`
`RESEND_API_KEY`	Clé API Resend	`re_xxxxxxxxx`

Variables publiques (frontend)

Variable	Description
`NEXT_PUBLIC_EMAILJS_SERVICE_ID`	ID service EmailJS
`NEXT_PUBLIC_EMAILJS_TEMPLATE_ID`	ID template EmailJS
`NEXT_PUBLIC_EMAILJS_PUBLIC_KEY`	Clé publique EmailJS
`NEXT_PUBLIC_CALENDLY_URL`	URL Calendly

Sécurité des variables

 Attention :

- Les variables `NEXT_PUBLIC_*` sont exposées côté client
- Ne jamais y mettre d'informations sensibles
- Utilisez des clés API avec des permissions limitées

Services externes

Cloudinary (Gestion des images)

1. **Configuration :**
    - Créez un compte sur Cloudinary
    - Récupérez vos credentials dans le Dashboard

- Configurez les presets d'upload

2. **Configuration des presets :**
   - Preset name : `alexia-energies`
   - Mode : `Unsigned`
   - Folder : `alexia-energies/`

Resend (Emails transactionnels)

1. **Configuration :**
   - Créez un compte sur Resend
   - Générez une clé API
   - Configurez votre domaine d'envoi
2. **Templates d'emails :**
   - Email de bienvenue
   - Reset de mot de passe
   - Notifications admin

Calendly (Prise de rendez-vous)

1. **Configuration :**
   - Créez un compte Calendly
   - Configurez vos créneaux disponibles
   - Récupérez l'URL d'intégration

Tests avant déploiement

Tests unitaires

```
# Lancer les tests
npm run test
```

Tests d'intégration

```
# Build de production local
npm run build

# Lancer en mode production
npm start
```

Contacts et support

- **Documentation Next.js :** nextjs.org/docs
- **Support Vercel :** vercel.com/support
- **Documentation Prisma :** prisma.io/docs

Annexes

Commandes utiles

```
# Déploiement complet
npm run build && vercel --prod

# Reset base de données (DEV UNIQUEMENT)
npx prisma migrate reset

# Génération du client Prisma
npx prisma generate

# Visualisation de la base de données
npx prisma studio

# Audit de sécurité
npm audit

# Optimisation des images
next/image optimization activée par défaut
```

URLs importantes

- **Production :** https://alexia-energies.vercel.app/
- **Vercel Dashboard :** https://vercel.com/fannysaez/alexia-energies
- **Neon Dashboard :** https://neon.tech/dashboard

Date de création : 11 octobre 2025
Version : 1.0
Auteur : Fanny Saez (fannysaez)
Projet : Alexia Énergies - App bien-être & formation

Cette documentation sera mise à jour selon les évolutions du projet et les retours d'expérience du déploiement.