Documentation Technique de la Plateforme Nouboudem Construction BTP SARL

Auteur : KUITO OUANDJI ANGE LUGRESSE
Date : 16/07/2025

Table des matières

1. Introduction

Ce document présente la documentation technique complète de la plateforme web développée pour Nouboudem Construction BTP SARL. Il détaille l'architecture, les choix de conception, l'implémentation, le déploiement, la sécurité, les tests et les modèles utilisés.

2. Architecture Générale

2.1. Vue d'ensemble

La plateforme repose sur une architecture multi-tiers :

- **Frontend** : Application web développée en **Nuxt.js** (Vue.js) pour l'espace public, l'espace client et l'administration.

- **Backend** : API REST Spring Boot, centralisant la logique métier et l'accès aux données.
- **Base de données** : MySQL, stockant toutes les entités métier.
- **Services externes** : SMTP (email), stockage de fichiers (local ou cloud).

2.2. Diagrammes d'architecture

Diagramme de composants

>  Diagramme de composants

Diagramme de déploiement

>  Diagramme de déploiement

3. Structure et Organisation du Code

3.1. Organisation des packages

```
com.nouboudem.api
├── user          // Gestion des utilisateurs, rôles, authentification
├── projet        // Gestion des projets et phases
├── devis         // Gestion des devis, lots, lignes de devis
├── formation     // Gestion des formations et inscriptions
├── media         // Gestion des fichiers et médias
├── temoignage    // Gestion des témoignages clients
├── service       // Gestion des services proposés
├── config        // Configuration Spring, sécurité, CORS, etc.
├── logs          // Gestion des logs applicatifs
```

3.2. Principales entités et modèles de données

- **User** : Utilisateur de la plateforme (admin, client)
- **Projet** : Projet de construction, phases, paiements
- **Devis** : Devis, lots de travaux, lignes de devis
- **Formation** : Formations proposées, inscriptions
- **Media** : Fichiers associés aux entités
- **Temoignage** : Avis et retours clients
- **Service** : Services proposés par l'entreprise

3.3. Diagramme de classes

>  Diagramme de classes

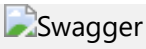
3.4. MLDR (Modèle Logique de Données Relationnel)

>  MLDR

4. API REST et Sécurité

4.1. Documentation Swagger/OpenAPI

L'API est documentée via Swagger/OpenAPI. Accès : `/swagger-ui.html` ou `/api-docs`

> 

4.2. Principaux endpoints

Ressource	Méthode	URL	Description	Authentification
Utilisateurs	POST	`/api/auth/register`	Inscription	Non
Utilisateurs	POST	`/api/auth/login`	Connexion	Non
Projets	GET	`/api/projets`	Liste des projets	Oui/Non*
Projets	POST	`/api/projets`	Création d'un projet	Oui (admin)
Devis	GET	`/api/devis`	Liste des devis	Oui/Non*
Formations	GET	`/api/formations`	Liste des formations	Non
Inscriptions	POST	`/api/inscriptions`	Inscription à une formation	Non
Médias	POST	`/api/media/upload`	Upload d'un fichier	Oui
Témoignages	GET	`/api/temoignages/publics`	Liste des témoignages validés	Non

* Selon configuration (projets publics/privés)

4.3. Sécurité et gestion des accès

- Authentification par email/mot de passe (Spring Security, JWT)
- Rôles : ADMIN, CLIENT, USER
- Routes publiques : espace public, inscription, consultation des services/formations/témoignages
- Routes protégées : gestion des projets, devis, formations, médias, etc.

5. Déploiement et Exploitation

5.1. Prérequis

- Docker & Docker Compose installés
- Variables d'environnement (fichier `.env`) :
  - `SPRING_DATASOURCE_URL`
  - `SPRING_DATASOURCE_USERNAME`
  - `SPRING_DATASOURCE_PASSWORD`
  - `JWT_SECRET`

- etc.

5.2. Déploiement Docker

```
# 1. Cloner le dépôt
git clone <url-du-repo>
cd <dossier>

# 2. Lancer les conteneurs
docker-compose up --build -d

# 3. Accéder à l'application
# Frontend : http://localhost:8080
# Backend  : http://localhost:8081
# Swagger  : http://localhost:8081/swagger-ui.html
```

Structure docker-compose (exemple)

```
version: '3.8'
services:
  backend:
    build: ./api
    ports:
      - "8081:8081"
    env_file:
      - .env
    depends_on:
      - db
  frontend:
    build: ./frontend
    ports:
      - "8080:80"
  db:
    image: mysql:8
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: btp
    ports:
      - "3306:3306"
    volumes:
      - db_data:/var/lib/mysql
volumes:
  db_data:
```

5.3. Supervision et maintenance

- **Logs**: Fichiers de logs accessibles dans `api/logs/`
- **Sauvegarde**: Dump régulier de la base MySQL

- **Mises à jour** : Déploiement continu via Docker
- **Documentation** : Ce document, Swagger, commentaires dans le code

6. Tests et Validation

- **Tests unitaires** : JUnit, Mockito (backend)
- **Tests d'intégration** : Postman, Swagger
- **Validation manuelle** : Parcours utilisateur, gestion des erreurs, sécurité

7. Annexes

- Diagrammes UML (classe, séquence, activité…)
- Scripts SQL d'initialisation
- Exemples de requêtes API
- Captures d'écran de l'application

Fin de la documentation technique.