

Cahier des Charges

Projet Final - Stack MERN

Examen Pratique

Abdelweheb GUEDDDES & Mohamed Ben Jazia
Ecole Polytechnique Sousse

Année Universitaire 2025-2026

INFORMATIONS IMPORTANTES

- **Date limite de soumission :** La veille de votre soutenance à 23h59
- **Format de livraison :** Code source + Mini rapport PDF
- **Soutenance obligatoire :** Questions techniques pour validation de la note
- **Thème libre :** Non imposé, laissez parler votre créativité!

Table des matières

1	Objectif du Projet	3
2	Spécifications Techniques	3
2.1	Choix du Thème	3
2.2	Architecture Backend (Node.js + Express + MongoDB)	3
2.2.1	Exigences Minimales	3
2.2.2	API REST	4
2.2.3	Sécurité et Middleware	4
2.2.4	Fonctionnalités Avancées	4
3	Livrables Attendus	5
3.1	Code Source	5
3.2	Validation de la Note	5
4	Modalités Pratiques	6
4.1	Date et Heure de Soumission	6

1 Objectif du Projet

Développer une application web complète utilisant le stack **MERN** (MongoDB, Express.js, React, Node.js) en appliquant l'ensemble des concepts et technologies étudiés durant le semestre.

Ce projet final a pour but de :

- Démontrer votre maîtrise du développement full-stack moderne
- Mettre en pratique les architectures REST API et SPA
- Appliquer les bonnes pratiques de sécurité et de structuration
- Intégrer des fonctionnalités avancées (authentification, IA, etc.)
- Produire un code propre, maintenable et documenté

2 Spécifications Techniques

2.1 Choix du Thème

Le thème de votre application est **entièrement libre**. Quelques suggestions :

- **E-Commerce** : Boutique en ligne, marketplace
- **Réseau Social** : Mini réseau, forum, blog collaboratif
- **Gestion** : CRM, gestion de projet, système de réservation
- **Éducation** : Plateforme de cours, quiz en ligne
- **Santé** : Suivi médical, gestion de rendez-vous
- **Finance** : Gestion de budget, tracker de dépenses
- **Divertissement** : Application de streaming, bibliothèque
- **Autre** : Toute idée originale validée par vos enseignants

Conseil : Choisissez un thème qui vous passionne ! Cela facilitera votre motivation et la qualité de votre travail.

2.2 Architecture Backend (Node.js + Express + MongoDB)

2.2.1 Exigences Minimales

1. Modèle de Données : 5 Entités Minimum

Votre base de données doit contenir **au minimum 5 entités** (modèles **Mon-goose**) avec des relations variées.

Types de relations obligatoires :

- **Relation 1-to-1 (Un à Un)** : Au moins 1 exemple
- **Relation 1-to-Many (Un à Plusieurs)** : Au moins 2 exemples
- **Relation Many-to-Many (Plusieurs à Plusieurs)** : Au moins 1 exemple

Exemple de Modélisation (E-Commerce) :

Entité	Relations
User	1-to-1 avec Profile 1-to-Many avec Order Many-to-Many avec Product (via Wishlist)
Profile	1-to-1 avec User
Product	1-to-Many avec Review Many-to-Many avec Category Many-to-Many avec User (Wishlist)
Order	1-to-Many avec OrderItem Appartient à 1 User
Category	Many-to-Many avec Product

Astuce : Documentez vos relations avec un schéma de base de données (diagramme ER) dans votre rapport.

2.2.2 API REST

2. Routes et Contrôleurs

Pour chaque entité, implémentez les opérations CRUD de base :

- **CREATE** : POST pour créer une ressource
- **READ** : GET pour récupérer (liste et détails)
- **UPDATE** : PUT/PATCH pour modifier
- **DELETE** : DELETE pour supprimer

Routes supplémentaires attendues :

- Routes d'authentification (Register, Login, Logout)
- Routes protégées avec middleware JWT

2.2.3 Sécurité et Middleware

3. Sécurité Obligatoire

- **Authentification JWT** : Système complet (register, login, token refresh)
- **Hashage des mots de passe** : bcrypt.js
- **Variables d'environnement** : Fichier .env pour secrets
- **Validation des données** : express-validator ou Joi
- **Protection CORS** : Configuration appropriée

2.2.4 Fonctionnalités Avancées

4. Intégration IA (Optionnel mais fortement recommandé - Bonus)

Intégrez au moins une fonctionnalité IA avec Gemini API :

- Génération automatique de contenu
- Analyse de sentiments
- Recommandations personnalisées
- Chatbot d'assistance

3 Livrables Attendus

3.1 Code Source

1. Structure du Projet

Format de livraison : Archive ZIP ou lien GitHub privé

Structure attendue :

```
projet-mern-[nom]/
|-- backend/
|   |-- config/
|   |-- controllers/
|   |-- models/
|   |-- routes/
|   |-- middleware/
|   |-- server.js
|   |-- package.json
|   +-- .env.example      # Exemple de variables
|-- frontend/
|   |-- src/
|   |-- public/
|   |-- package.json
|   +-- vite.config.js
|-- README.md             # Instructions d'installation
+-- rapport.pdf           # Mini rapport
```

3.2 Validation de la Note

Attention : Soutenance Obligatoire

La note finale n'est validée qu'après la soutenance. Vous devrez :

1. Présenter votre projet

- Démo en direct de l'application
- Explication de l'architecture
- Présentation des fonctionnalités clés

2. Répondre aux questions techniques

- Questions sur le code (backend et frontend)
- Explication des choix techniques
- Compréhension des concepts MERN
- Justification des relations de base de données

Important : Une excellente application sans capacité à expliquer le code peut entraîner une note réduite. L'objectif est de vérifier votre compréhension personnelle.

4 Modalités Pratiques

4.1 Date et Heure de Soumission

Deadline Stricte

- **Date limite** : La veille de votre soutenance programmée
- **Heure limite** : 23h59 précises
- **Format** :
 - Fichier ZIP nommé : `NOM_Prenom_ProjetMERN.zip`
 - OU Lien GitHub privé avec accès professeurs

Pénalités : Retard = -2 points par heure de retard (maximum -10 points)