

# Rapport de Projet : Application Full stack de Gestion des Utilisateurs

**Réalisé par : Mohamed Mbarkiou, Benhamout Mohammed**

**Lien GITHUB :** [Potatoman987654321/fullstackapp \(github.com\)](https://github.com/Potatoman987654321/fullstackapp)

**Lien DEMO :** [https://www.mediafire.com/file/6lk7qbjhty9jeel/DEMO\\_MohamedMbarkiou\\_BenhamouMohammed/file](https://www.mediafire.com/file/6lk7qbjhty9jeel/DEMO_MohamedMbarkiou_BenhamouMohammed/file)

## Introduction :

Pour notre projet pour la matière de développement web ASEDS, je vais créer une application FULLSTACK avec une base donnée qui peut faire les opérations simple CRUD.

## 1. Modélisation :

Modèle de Données : Le modèle de données représente les informations essentielles stockées pour chaque utilisateur dans la base de données. Il comprend les champs suivants :

- name : Le nom de l'utilisateur.
- email : L'adresse e-mail unique de l'utilisateur.
- password : Le mot de passe haché de l'utilisateur pour des raisons de sécurité.

Dans le cadre de ce projet, j'ai développé une application web full stack de gestion des utilisateurs en utilisant plusieurs technologies clés. La modélisation a été initiée par la création d'un modèle de données clair décrivant les champs nécessaires pour chaque utilisateur, notamment le nom, l'e-mail, et le mot de passe. Cette représentation a été complétée par un schéma visuel de la base de données, mettant en lumière les relations entre les entités et la manière dont les données sont stockées dans MongoDB.

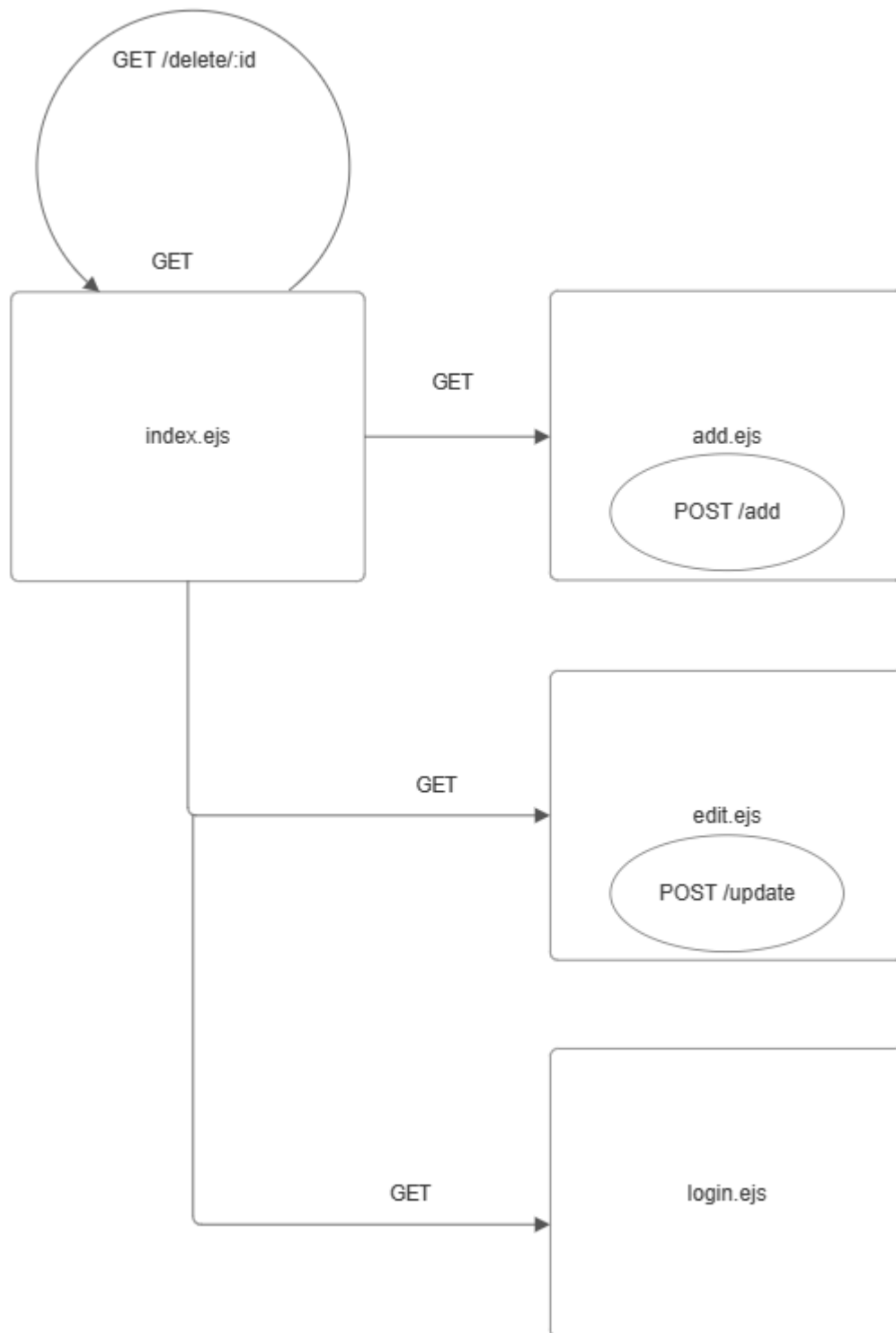
L'implémentation a été réalisée en suivant une structure de projet organisée avec des dossiers distincts pour les modèles, les routes, les vues, et les fichiers de mise en page commune. Node.js et Express.js ont été choisis comme plateforme côté serveur, offrant une exécution rapide et une gestion efficace des routes. La base de données MongoDB a été intégrée avec l'aide de Mongoose, facilitant ainsi l'interaction avec les données.

La sécurité a été priorisée en utilisant Bcrypt pour le hachage sécurisé des mots de passe. En ce qui concerne la gestion de session utilisateur, j'ai implémenté Express-session pour maintenir l'état de l'utilisateur entre les requêtes, notamment lors de l'authentification. Par ailleurs, l'utilisation d'EJS (Embedded JavaScript) comme moteur de template a simplifié la génération dynamique des pages HTML, assurant une présentation fluide et réactive.

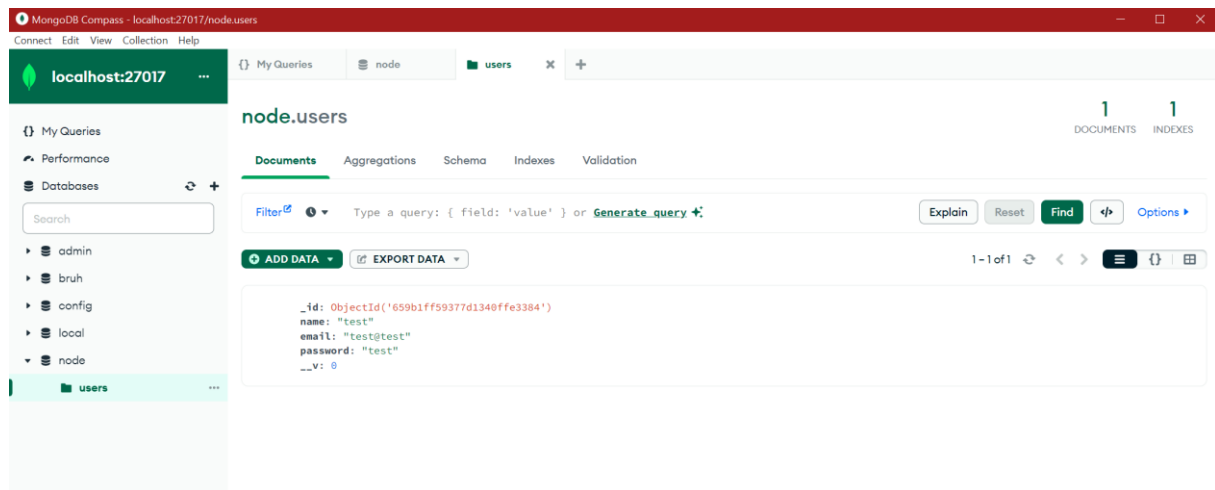
Le fichier routes.js définit les réponses du serveur à chaque requête par exemple la requête initial get « / » on répond par afficher la page index.ejs. Ces actions incluent l'affichage, l'ajout, la modification, et la suppression d'utilisateurs, ainsi que la gestion de l'authentification. Le projet a été une expérience enrichissante, me permettant d'appliquer mes connaissances acquises à l'INPT dans un contexte pratique de développement web fullstack.

## **2. Implémentation :**



Le schéma suivant représente les différentes routes qu'on a mise en place dans l'application.



Screenshot de mongoDB :



Equivalent dans index :

</> Currently logged in as test					Home	Add user	Login	Contact
ID	Name	Email	Password	Action				
0	test	test@test	test	 				

### 3-Conclusion :

Comme Conclusion, le projet est très très simple mais il contient les fonctionnalités de bases qu'on a besoin pour communiquer avec la DB est authentifier les utilisateurs, et on peut toujours incorporer de nouvelles fonctionnalités comme l'upload des fichiers et le cryptage etc...