Full Stack Development with MERN

Database Design and Development Report

Date: 15/04/2025

Team ID: SWTID1742901502

Project Name: Nexus Social App

Maximum Marks: —

# Project Title: Nexus Social App

Date: 15/04/2025

Prepared by: SWTID1742901502

# Objective

The objective of this report is to outline the database design and implementation details for the NEXUS Social App project, including schema modeling and database integration using MongoDB Atlas.

# Technologies Used

• Database Management System (DBMS): MongoDB Atlas  
• Object-Document Mapper (ODM): Mongoose

# Design the Database Schema

**1. Users**

* **Attributes:**
  + **name**: { type: String, required: true }
  + **email**: { type: String, required: true, unique: true }
  + **password**: { type: String, required: true }
  + **role**: { type: String, enum: ['user', 'admin'], default: 'user' }
  + **profilePicture**: { type: String, default: 'default.jpg' }
  + **bio**: { type: String, default: '' } // Short user bio
  + **createdAt**: { type: Date, default: Date.now }
  + **updatedAt**: { type: Date, default: Date.now }

**2. Posts**

* **Attributes:**
  + **userId**: { type: ObjectId, ref: 'User ', required: true }
  + **content**: { type: String, required: true }
  + **image**: { type: String, default: '' }
  + **createdAt**: { type: Date, default: Date.now }
  + **updatedAt**: { type: Date, default: Date.now }

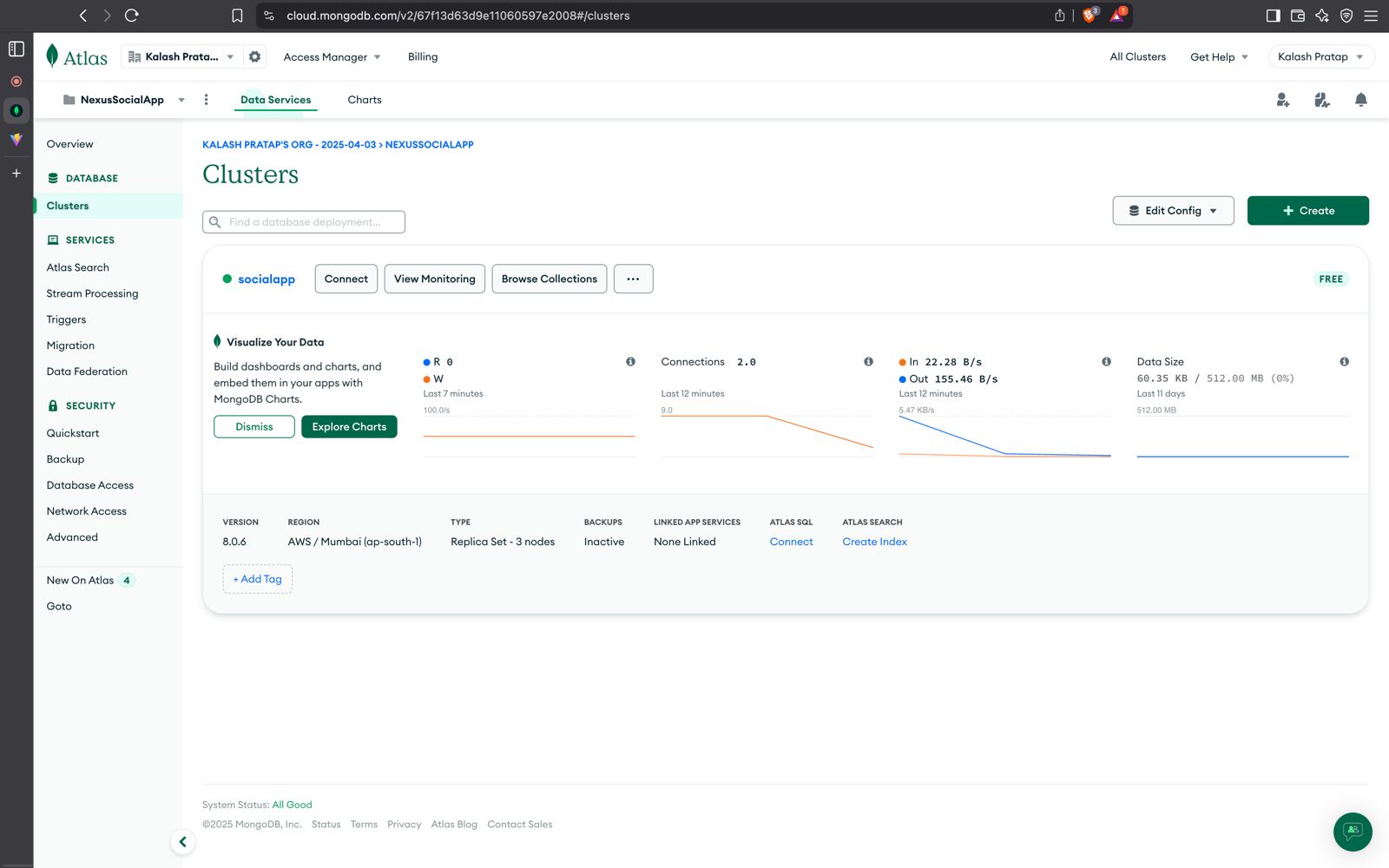
**3. Comments**

* **Attributes:**
  + **postId**: { type: ObjectId, ref: 'Post', required: true }
  + **userId**: { type: ObjectId, ref: 'User ', required: true }
  + **content**: { type: String, required: true }
  + **createdAt**: { type: Date, default: Date.now }
  + **updatedAt**: { type: Date, default: Date.now }

**4. Likes**

* **Attributes:**
  + **postId**: { type: ObjectId, ref: 'Post', required: true }
  + **userId**: { type: ObjectId, ref: 'User ', required: true }
  + **createdAt**: { type: Date, default: Date.now }

# Implemented Collections using MongoDB Atlas



Database Name: book-a-doctor

1. Collection: users  
 Schema:-  
{

\_id: ObjectId,

name: String,

email: String,

password: String,

role: String,

profilePicture: String,

bio: String,

createdAt: Date,

updatedAt: Date

}

2. Collection: posts  
Schema:-  
{

\_id: ObjectId,

userId: ObjectId,

content: String,

image: String,

createdAt: Date,

updatedAt: Date

}

3.Collection: comments

Schema:-

{

\_id: ObjectId,

postId: ObjectId,

userId: ObjectId,

content: String,

createdAt: Date,

updatedAt: Date

}

4. Collection: likes

Schema:-

{

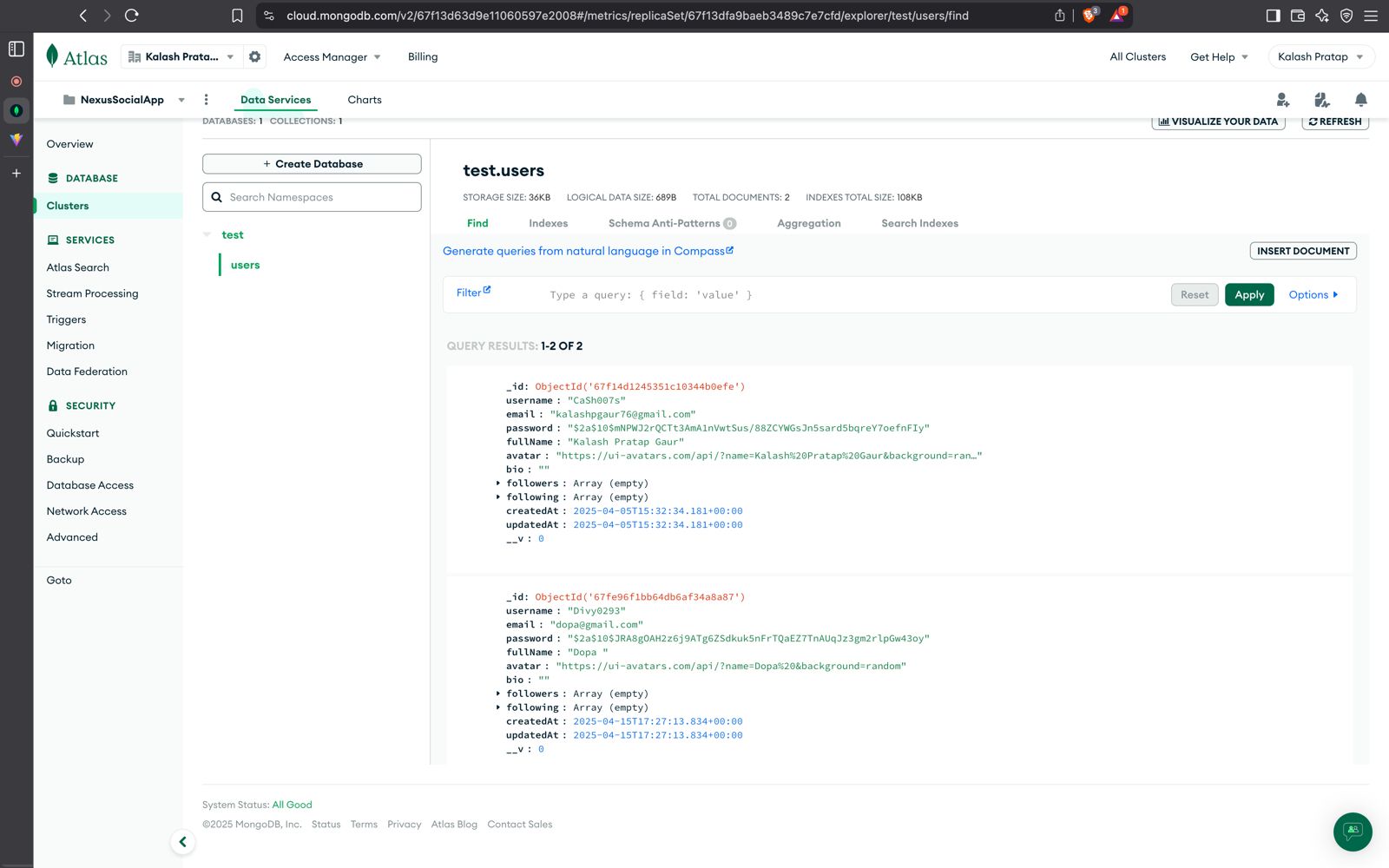
\_id: ObjectId,

postId: ObjectId,

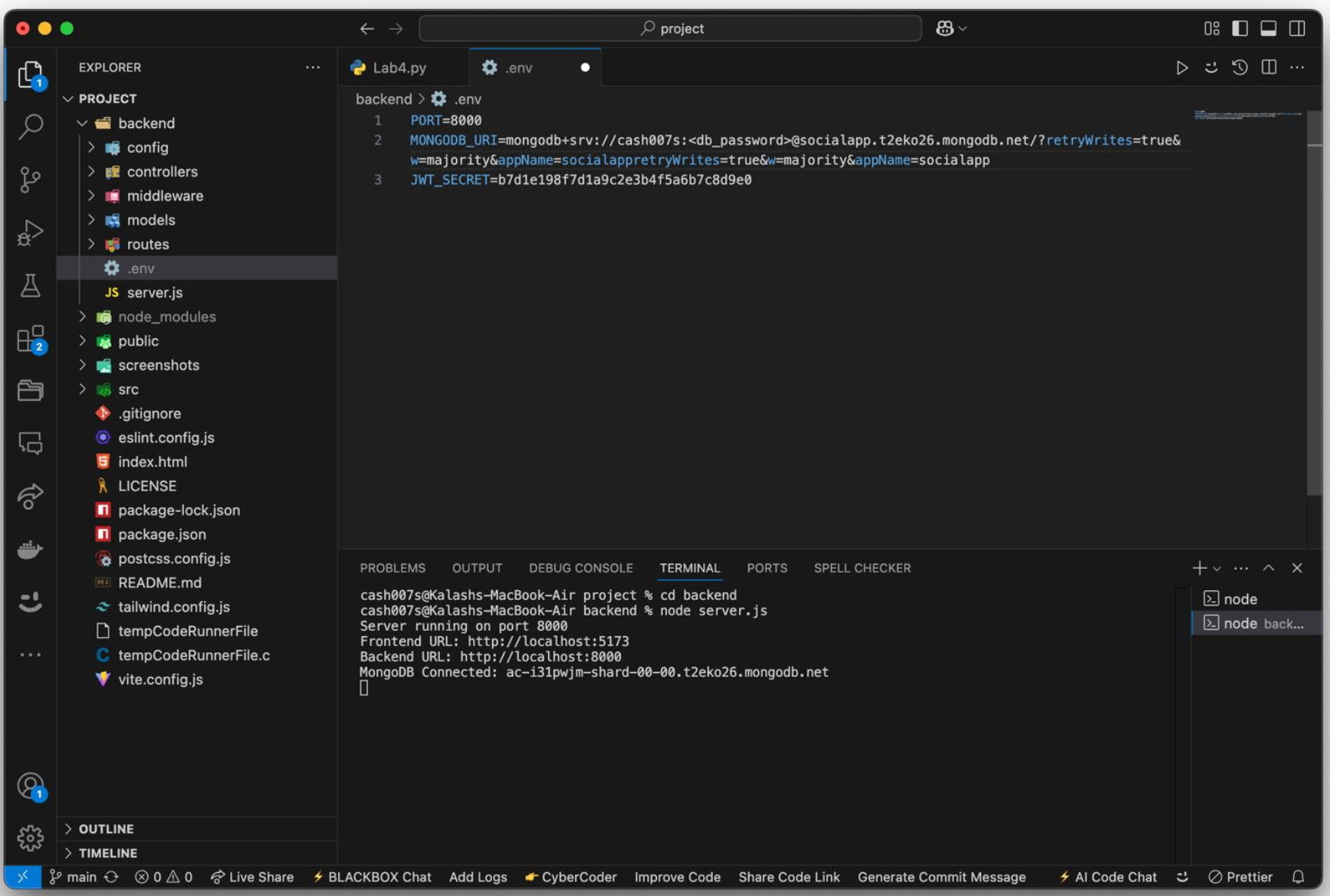
userId: ObjectId,

createdAt: Date

}



# Integration with Backend



The backend communicates with MongoDB Atlas using Mongoose. Key operations include CRUD operations for users and appointments.

* User Registration :

const user = await User.findOne({ email: req.body.email });

if (!user) {

const newUser = new User(req.body);

await newUser .save();

res.status(201).json({ success: 'User created successfully' });

} else {

res.status(409).json({ success: 'Email already exists' });

}

* Post Creation

const newPost = new Post(req.body);

await newPost.save();

res.status(201).json({ success: 'Post created successfully' });

* Comment Posting:

const newComment = new Comment(req.body);

await newComment.save();

res.status(201).json({ success: 'Comment added successfully' });