**Full Stack Development with MERN**

**Database Design and Development Report**

|  |  |
| --- | --- |
| Date | 12 July 2024 |
| Team ID | SWTID1720010842 |
| Project Name | Book a Doctor |
| Maximum Marks | 5 Marks |

**Project Title**: Book a Doctor

**Date**: 12-07-24

**Prepared by**: V Kavya Sahithi, Maturi Nikhitha

**Objective**

The objective of this report is to outline the database design and implementation details for the Book a Doctor project, including schema design and database management system (DBMS) integration.

**Technologies Used**

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

**Database Schema Design**

The database schema is designed to accommodate the following entities and relationships:

**1. Users**

- Attributes:

* \_id: ObjectId
* name: String
* email: String
* password: String
* createdAt: Date
* updatedAt: Date

**2. Posts**

- Attributes:

* \_id: ObjectId
* title: String
* content: String
* author: ObjectId (references User)
* createdAt: Date
* updatedAt: Date

**3. Comments**

- Attributes:

* \_id: ObjectId
* text: String
* post: ObjectId (references Post)
* author: ObjectId (references User)
* createdAt: Date
* updatedAt: Date

**Implement the Database using MongoDB**

The MongoDB database is implemented with the following collections and structures:

Database Name: book\_a\_doctor

1. Collection: users

- Schema:

```

{

\_id: ObjectId,

name: String,

email: String,

password: String,

createdAt: Date,

updatedAt: Date

}

```

2. Collection: posts

- Schema:

```

{

\_id: ObjectId,

title: String,

content: String,

author: ObjectId (references users),

createdAt: Date,

updatedAt: Date

}

```

3. Collection: comments

- Schema:

```

{

\_id: ObjectId,

text: String,

post: ObjectId (references posts),

author: ObjectId (references users),

createdAt: Date,

updatedAt: Date

}

```

**Integration with Backend**

* The backend APIs interact with MongoDB using Mongoose ODM Key interactions include:
  + User Management: CRUD operations for users.

const User = mongoose.model('User', userSchema);

* + Post Management: CRUD operations for posts, with user authentication.

const Post = mongoose.model('Post', postSchema);

* + Comment Management: CRUD operations for comments associated with posts.

const Comment = mongoose.model('Comment', commentSchema);