Run the index.js first

Then deploy the html page

index.js

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>MongoDB CRUD Example</title>

</head>

<body>

<h1>MongoDB CRUD Example</h1>

<form id="crudForm" action="/performOperation" method="POST">

<label for="username">Username:</label>

<input type="text" id="username" name="username" required><br>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required><br>

<label for="age">Age:</label>

<input type="number" id="age" name="age"><br>

<label for="operation">Choose Operation:</label>

<select id="operation" name="operation" required>

<option value="create">Create</option>

<option value="read">Read</option>

<option value="update">Update</option>

<option value="delete">Delete</option>

</select><br>

<button type="submit">Submit</button>

</form>

<script>

// You can add JavaScript logic here if needed

</script>

</body>

</html>

EXPRESS:

app.js

const express = require('express');

const mongoose = require('mongoose');

const path = require('path');

const app = express();

const port = 9000;

// MongoDB connection URL

const mongoURI = 'mongodb://localhost:27017/d15a';

// Connect to MongoDB

mongoose.connect(mongoURI, { useNewUrlParser: true, useUnifiedTopology: true });

const db = mongoose.connection;

db.on('error', console.error.bind(console, 'MongoDB connection error:'));

db.once('open', () => {

console.log('Connected to MongoDB');

});

// Create a User model and schema

const userSchema = new mongoose.Schema({

username: { type: String, required: true },

email: { type: String, required: true, unique: true },

age: { type: Number }

});

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

// Serve static files from the "public" directory

app.use(express.static(path.join(\_\_dirname, 'public')));

// Middleware to parse JSON in the request body

app.use(express.json());

app.get('/', (req, res) => {

res.send('GET request to the homepage')

})

// Express route to handle CRUD operations

// app.post('/performOperation', async (req, res) => {

// const { operation, data } = req.body;

// try {

// switch (operation) {

// case 'create':

// const newUser = new User(data);

// const savedUser = await newUser.save();

// res.json({ result: 'User created', user: savedUser });

// break;

// case 'read':

// const foundUser = await User.findOne({ username: data.username });

// res.json({ result: 'User found', user: foundUser });

// break;

// case 'update':

// const updatedUser = await User.findOneAndUpdate(

// { username: data.username },

// { $set: data },

// { new: true }

// );

// res.json({ result: 'User updated', user: updatedUser });

// break;

// case 'delete':

// const deletedResult = await User.deleteOne({ username: data.username });

// res.json({ result: 'User deleted', deletedCount: deletedResult.deletedCount });

// break;

// default:

// res.status(400).json({ error: 'Invalid operation' });

// }

// } catch (error) {

// console.error('Error:', error);

// res.status(500).json({ error: 'Server error' });

// }

// });

// Start the server

app.listen(port, () => {

console.log(`Server is running at http://localhost:${port}`);

});