



University of Colombo School of Computing

SCS 2208 - Rapid Application Development

Lab Sheet 11

Basic Commands in MongoDB

1. How to show the databases present
 - *show dbs*
2. How to create a database
 - *use [name of database]*
Example: *use users*
3. How to check the current database
 - *db*
4. How to create a collection
 - *db.createCollection("[name of collection]")*
Example: *db.createCollection("customer")*
5. How to show collections
 - *show collections*
6. How to Delete a collection
 - *db.<collection name>.drop()*
Example: *db.customer.drop()*
7. How to insert a document into a collection
 - *db.<collection>.insertOne(document, [writeConcern])*
Example: *db.users.insertOne({ name: "Lakshitha", gender: "M"})*
8. How to show all documents in a collection
 - *db.[collection Name].find()* or *db.[collection name].find().pretty()*
for a well indented list.
Example: *db.customer.find().pretty()*

How to create a simple form using MERN?

1. Set up your Express.js project:

If you haven't already, create a new directory for your project and initialize it with npm:

```
mkdir express-mongodb-form
cd express-mongodb-form
npm init -y
```

Install the necessary dependencies:

```
npm install express mongoose body-parser
```

2. Create your Express.js application:

Create an **app.js** file and set up your Express application with middleware for handling requests and connecting to the MongoDB database:

```
// app.js
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');

const app = express();
const port = process.env.PORT || 3000;

// MongoDB connection
mongoose.connect('mongodb://localhost/your_database_name', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});
const db = mongoose.connection;

db.on('error', (error) => console.error('MongoDB connection error:', error))
db.once('open', () => console.log('Connected to MongoDB'));
```

```
// Middleware
app.use(bodyParser.urlencoded({ extended: true }));

// Define a MongoDB schema and model for your form data
const FormData = mongoose.model('FormData', {
  message: String,
});

// Routes
app.get('/', (req, res) => {
  // Render an HTML form
  res.send(`
    <form action="/submit" method="post">
      <input type="text" name="message" placeholder="Enter your message">
      <button type="submit">Submit</button>
    </form>
  `);
});

app.post('/submit', (req, res) => {
  const { message } = req.body;
```

```
// Create a new instance of the FormData model
const formData = new FormData({ message });

// Save the submitted data to MongoDB
formData.save((err) => {
  if (err) {
    console.error('Error saving form data:', err);
    res.send('Error saving form data');
  } else {
    console.log('Form data saved successfully');
    res.send('Form data saved successfully');
  }
});

// Start the server
app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});
```

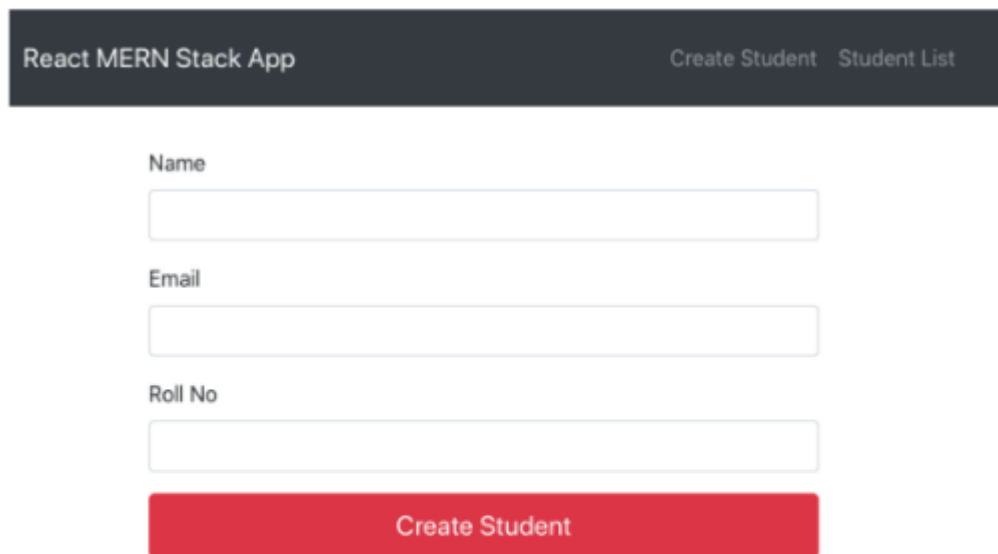
3. Replace 'mongodb://localhost/your_database_name' with your actual MongoDB connection string and customize the form according to your needs.
4. Start your Express.js application:


```
node app.js
```
5. Access your application in a web browser at **http://localhost:3000/**. You should see a simple form with a text input field. You can enter text and submit it to save the data in your MongoDB database

Activity

1. Develop a React MERN Stack App to store the details of your Second Year Group Project team members. You are encouraged to develop the design according to your creativity.

Example UI:

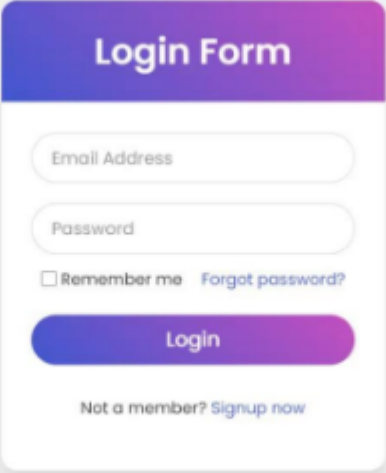


The image shows a web application interface for a React MERN Stack App. At the top, there is a dark grey header bar with the text "React MERN Stack App" on the left and "Create Student" and "Student List" on the right. Below the header, there are three input fields stacked vertically, each with a label above it: "Name", "Email", and "Roll No". Each input field is a simple white rectangle with a thin grey border. Below the input fields, there is a prominent red button with the text "Create Student" in white.

Reference : <https://www.positronx.io/react-mern-stack-crud-app-tutorial/>

<https://www.scaler.com/topics/nodejs/connect-mongodb-with-node-js/>

2.



The image shows a login form with a purple header containing the text "Login Form". Below the header are two input fields: "Email Address" and "Password". Under the "Password" field is a checkbox labeled "Remember me" and a link labeled "Forgot password?". Below these is a purple button labeled "Login". At the bottom of the form is a link labeled "Not a member? Signup now".

- a) Create a login form for a Node application using **Passport.js**.
- b) Use the session authentication strategy with **Passport.js**.
- c) Connect **Passport** to a **MongoDB** database to store user data.
- d) Authorize only logged-in users to access a page.