

Name: Bitra Nandini

Roll number: 2022MCA16010

Assignment:3: A RESTful API using express.js and create a database and index in MongoDB

Source code:

```
const express = require('express');
const mongoose = require('mongoose');
const app = express();
const port = 3001;

// MongoDB connection
mongoose.connect('mongodb://localhost:27017/myDatabase', { useNewUrlParser:
true, useUnifiedTopology: true });
const db = mongoose.connection;
db.on('error', console.error.bind(console, 'MongoDB connection error:'));
db.once('open', function() {
  console.log('Connected to MongoDB...');
});

// Define a schema for your data
const Schema = mongoose.Schema;
const exampleSchema = new Schema({
  name: String,
  age: Number,
  salary: Number,
  role: String
});

// Create a model based on the schema
const Example = mongoose.model('Example', exampleSchema);

// Express middleware for parsing JSON bodies
app.use(express.json());

// Route to create a new example
app.post('/examples', async (req, res) => {
  try {
    const example = await Example.create(req.body);
    res.status(201).json(example);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});

// Route to retrieve all examples where salary and role have values greater
than 2
app.get('/examples', async (req, res) => {
```

```

    try {
      const examples = await Example.find({
        $and: [
          { salary: { $gt: 2 } }, // Salary greater than 2
          { role: { $gt: 2 } }    // Role greater than 2
        ]
      });
      res.json(examples);
    } catch (err) {
      res.status(500).json({ message: err.message });
    }
  });

// Route to retrieve a specific example by ID
app.get('/examples/:id', async (req, res) => {
  try {
    const example = await Example.findById(req.params.id);
    if (!example) {
      return res.status(404).json({ message: 'Example not found' });
    }
    res.json(example);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

// Route to update an example by ID
app.put('/examples/:id', async (req, res) => {
  try {
    const example = await Example.findByIdAndUpdate(req.params.id,
req.body, { new: true });
    if (!example) {
      return res.status(404).json({ message: 'Example not found' });
    }
    res.json(example);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});

// Route to delete an example by ID
app.delete('/examples/:id', async (req, res) => {
  try {
    const example = await Example.findByIdAndDelete(req.params.id);
    if (!example) {
      return res.status(404).json({ message: 'Example not found' });
    }
    res.json({ message: 'Example deleted successfully' });
  }
});

```

```

    } catch (err) {
      res.status(500).json({ message: err.message });
    }
  });

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

```

OutPut:

Post method:

The screenshot shows the VS Code interface with a REST client tab open. The request is a POST to `http://localhost:3001/examples` with a JSON body. The response is a 201 Created status with a JSON body containing user details.

Request:

```

POST http://localhost:3001/examples
{
  "name": "srilatha",
  "age": 25,
  "salary": 50000,
  "role": "analyst"
}

```

Response:

```

{
  "name": "srilatha",
  "age": 25,
  "salary": 50000,
  "role": "analyst",
  "_id": "65f061f77845176c14cc7323",
  "__v": 0
}

```

Terminal Output:

```

sion 4.0.0 and will be removed in the next major version
Server is running on port 3000
Connected to MongoDB...
PS C:\Users\nandini\Desktop\nodeapi> node server.js
(node:18268) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4
.0.0 and will be removed in the next major version
(Use 'node --trace-warnings ...' to show where the warning was created)
(node:18268) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver ver
sion 4.0.0 and will be removed in the next major version
Server is running on port 3001
Connected to MongoDB...

```

Get Method:

The screenshot shows the NodeAPI interface with a GET request to `http://localhost:3001/examples`. The response is a JSON array with two objects. The terminal shows the server running on port 3000.

Request: GET `http://localhost:3001/examples`

Response: Status: 200 OK, Size: 206 Bytes, Time: 23 ms

JSON Content:

```
1 {
2   "name": "srilatha",
3   "age": 25,
4   "salary": 50000,
5   "role": "analyst"
6 }
```

Response:

```
3 {
4   "_id": "65f061dd7845176c14cc7321",
5   "name": "nandini",
6   "age": 25,
7   "salary": 50000,
8   "role": "developer",
9   "__v": 0
10 },
11 {
12   "_id": "65f061f77845176c14cc7323",
13   "name": "srilatha",
14   "age": 25,
```

Terminal:

```
sion 4.0.0 and will be removed in the next major version
Server is running on port 3000
Connected to MongoDB...
PS C:\Users\nandini\Desktop\nodeapi> node server.js
(node:18268) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4
.0.0 and will be removed in the next major version
(Use 'node --trace-warnings ...' to show where the warning was created)
(node:18268) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver ver
sion 4.0.0 and will be removed in the next major version
Server is running on port 3001
Connected to MongoDB...
```

Put method:

The screenshot shows the NodeAPI interface with a PUT request to `http://localhost:3001/examples/65f061f77845176c14cc7323`. The response is a JSON object. The terminal shows the server running on port 3000.

Request: PUT `http://localhost:3001/examples/65f061f77845176c14cc7323`

Response: Status: 200 OK, Size: 96 Bytes, Time: 12 ms

JSON Content:

```
1 {
2   "name": "sri",
3   "age": 25,
4   "salary": 50000,
5   "role": "analyst"
6 }
```

Response:

```
1 {
2   "_id": "65f061f77845176c14cc7323",
3   "name": "sri",
4   "age": 25,
5   "salary": 50000,
6   "role": "analyst",
7   "__v": 0
8 }
```

Terminal:

```
sion 4.0.0 and will be removed in the next major version
Server is running on port 3000
Connected to MongoDB...
PS C:\Users\nandini\Desktop\nodeapi> node server.js
(node:18268) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4
.0.0 and will be removed in the next major version
(Use 'node --trace-warnings ...' to show where the warning was created)
(node:18268) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver ver
sion 4.0.0 and will be removed in the next major version
Server is running on port 3001
Connected to MongoDB...
```

Delete method:

The screenshot displays the Thunder Client interface with a DELETE request configured. The request URL is `http://localhost:3001/examples/65f061f77845176c14cc73`. The request body is a JSON object: `{ "name": "sri", "age": 25, "salary": 50000, "role": "analyst" }`. The response status is 200 OK, with a size of 42 Bytes and a time of 19 ms. The response body is a JSON object: `{ "message": "Example deleted successfully" }`.

The terminal output shows the following commands and warnings:

```
sion 4.0.0 and will be removed in the next major version
Server is running on port 3000
Connected to MongoDB...
PS C:\Users\nandini\Desktop\nodeapi> node server.js
(node:18268) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4
.0.0 and will be removed in the next major version
(Use 'node --trace-warnings ...' to show where the warning was created)
(node:18268) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver ver
sion 4.0.0 and will be removed in the next major version
Server is running on port 3001
Connected to MongoDB...
```