St. Francis Institute of Technology Borivli (West), Mumbai-400103

(Autonomous Institute)
Department of Information Technology

Academic Year: 2024-25

Class: TE-ITA/B Semester: VI

Subject: Web Lab

Experiment – 7: Node.js and MongoDB connectivity and using Mongoose for Structured Schema and Validation.

1. **Aim:** To connect MongoDB with NodeJS and use mongoose for structured schema.
2. **Objectives:** Aim of this experiment is that, the students will be able
   - To connect MongoDB with NodeJS
   - Create a structured schema and collection
3. **Outcomes:** After study of this experiment, the students will be able
   - To perform CRUD operations
   - to handle coding and syntax error
4. **Prerequisite:** Basic understanding of database, MongoDB data types and commands, NodeJS etc.
5. **Requirements:** Personal Computer, Windows operating system, VSCode, TypeScript, browser, Internet Connection, google doc, Node JS, MongoDB.
6. **Pre-Experiment Exercise:**
   Brief Theory: Refer shared material
7. **Laboratory Exercise**
   A. **Procedure:**
      a. **Answer the following:**
         - What is the use of mongoose library?
         - How to create a schema using mongoose?
         - Why to use await inside the async () method?
      b. **Attach screenshots:**
         - MongoDB code and output with your own comments.

8. **Post-Experiments Exercise**
   A. **Extended Theory:**
      Nil
   B. **Questions:**

- What are the benefits of mongoose in MongoDB?
- How to create a model?

   C. **Conclusion:**
      - Write what was performed in the experiment.
      - Write the significance of the topic studied in the experiment.
   D. **References:**

1. MongoDB in Action, Second Edition, by Kyle Banker, Peter Bakkum
2. https://docs.mongodb.com/manual/

Name: Vishal Rajesh Mahajan Web Lab EXP07

Class: TE IT A Roll No: 56

Step 1: Initialize a Node.js Project

1. Open a terminal and navigate to the project directory.
2. Run the following command to initialize a Node.js project:

```
Windows PowerShell
PS C:\VishalRMahajan\exp07> npm init -y
Wrote to C:\VishalRMahajan\exp07\package.json:

{
  "name": "exp07",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

Step 2: Install Required Packages

3. Install Express and Mongoose using:

```
Windows PowerShell
PS C:\VishalRMahajan\exp07> npm install express mongoose

added 83 packages, and audited 84 packages in 3s

15 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\VishalRMahajan\exp07>
```

Step 3: Setup MongoDB Server

4. Start the MongoDB server:

```
Windows PowerShell
the spelling of the name, or if a path was included, verify
PS C:\VishalRMahajan\exp07> mongod
{"t":{"$date":"2025-04-01T15:38:34.362+05:30"},"s":"I",  "c":"CO
NTROL",  "id":23285,   "ctx":"thread1","msg":"Automatically disa
bling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProt
ocols 'none'"}
{"t":{"$date":"2025-04-01T15:38:35.681+05:30"},"s":"I",  "c":"CO
NTROL",  "id":5945603, "ctx":"thread1","msg":"Multi threading in
itialized"}
{"t":{"$date":"2025-04-01T15:38:35.681+05:30"},"s":"I",  "c":"NE
TWORK",  "id":4648601, "ctx":"thread1","msg":"Implicit TCP FastO
pen unavailable. If TCP FastOpen is required, set at least one o
f the related parameters","attr":{"relatedParameters":["tcpFastO
penServer","tcpFastOpenClient","tcpFastOpenQueueSize"]}}
{"t":{"$date":"2025-04-01T15:38:35.682+05:30"},"s":"I",  "c":"NE
TWORK",  "id":4915701, "ctx":"thread1","msg":"Initialized wire s
```

Name: Vishal Rajesh Mahajan Web Lab EXP07

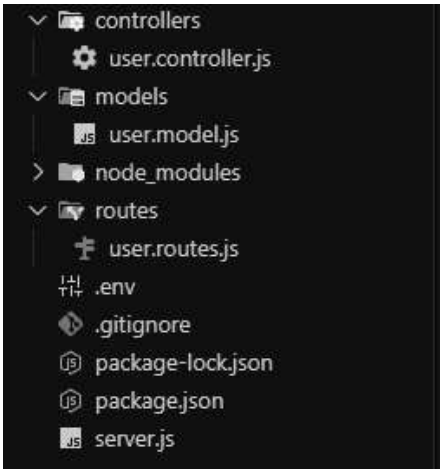
Class: TE IT A Roll No: 56

5. Open the MongoDB shell and create a database:

```
mongosh mongodb://127.0.0.
PS C:\Users\Student> mongosh
PS C:\Users\Student> mongosh
Current Mongosh Log ID: 67ebbb6fdad2732664b71235
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTi
meoutMS=2000&appName=mongosh+2.4.2
Using MongoDB:          8.0.5
Using Mongosh:          2.4.2

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-04-01T12:59:57.981+05:30: Access control is not enabled for the database. Read and
 write access to data and configuration is unrestricted
------

test> use exp07
switched to db exp07
exp07>
```

Step 4: Create Project Structure

6. Organize the project with the following structure:



Step 5: Establish MongoDB Connection

7. Add the following in server.js:

```
const express = require('express');
const mongoose = require('mongoose');
require('dotenv').config();

const app = express();
app.use(express.json());

// MongoDB Connection
mongoose.connect(process.env.MONGO_URI, {
  useNewUrlParser: true,
  useUnifiedTopology: true,
```

```
})
.then(() => console.log('MongoDB Connected'))
.catch(err => console.error('MongoDB connection error:', err));

// Routes
const userRoutes = require('./routes/user.routes');
app.use('/users', userRoutes);

const PORT = process.env.PORT || 3000;
app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
```

Step 6: Create Mongoose Schema

8.In models/user.model.js, define a User schema:

```
const mongoose = require('mongoose');
const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, 'Name is required'],
    minlength: [3, 'Name should have at least 3 characters'],
    maxlength: [50, 'Name should not exceed 50 characters']
  },
  email: {
    type: String,
    required: [true, 'Email is required'],
    unique: true,
    match: [/^\S+@\S+\.\S+$/, 'Please provide a valid email address']
  },
  password: {
    type: String,
    required: true,
    minlength: [6, 'Password must be at least 6 characters']
  },
  age: {
    type: Number,
    min: [18, 'You must be at least 18 years old'],
    max: [100, 'Age must be below 100']
  },
}, { timestamps: true });
const User = mongoose.model('User', userSchema);
module.exports = User;
```

Step 7: Define Routes

9. In routes/user.routes.js, set up API endpoints:

```
const express = require('express');
const router = express.Router();
const { createUser, getUsers } = require('../controllers/user.controller');

router.post('/', createUser);
router.get('/', getUsers);

module.exports = router;
```

Step 8: Implement Controller Logic

10. In controllers/user.controller.js, write database operations:

```
const User = require('../models/user.model');

exports.createUser = async (req, res) => {
  try {
    const user = new User(req.body);
    await user.save();
    res.status(201).json({ message: 'User created successfully', user });
  } catch (error) {
    res.status(400).json({ error: error.message });
  }
};

exports.getUsers = async (req, res) => {
  try {
    const users = await User.find();
    res.json(users);
  } catch (error) {
    res.status(500).json({ error: error.message });
  }
};
```

Step 9: Configure Environment Variables

11. Create a .env file in the root directory and add:

```
MONGO_URI=mongodb://localhost:27017/exp07
PORT=3000
```

Name: Vishal Rajesh Mahajan Web Lab EXP07

Class: TE IT A Roll No: 56

Step 10: Run the Application

12. Start the Node.js server:

```
Windows PowerShell
PS C:\VishalRMahajan\exp07> node server.js
(node:14468) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: u
seNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be remo
ved in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:14468) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option
: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will b
e removed in the next major version
Server running on port 3000
MongoDB Connected
```

13. Open Thunder Client in VS Code and test the API:

Create a user (POST request):

- URL: http://localhost:3000/users
- Body (JSON):

```
POST http://localhost:3000/users  Send
Status: 201 Created  Size: 253 Bytes  Time: 27 ms
Body JSON
JSON Content
{
  "name": "Vishal Mahajan",
  "email": "vism06@gmail.com",
  "password": "password@123",
  "age": 20
}
Response
{
  "message": "User created successfully",
  "user": {
    "name": "Vishal Mahajan",
    "email": "vism06@gmail.com",
    "password": "password@123",
    "age": 20,
    "_id": "67ebbe06eb50325244ec72c6",
    "createdAt": "2025-04-01T10:20:54.505Z",
    "updatedAt": "2025-04-01T10:20:54.505Z",
    "__v": 0
  }
}
```

Get all users (GET request):

- URL: http://localhost:3000/users

```
GET http://localhost:3000/users/  Send
Status: 200 OK  Size: 208 Bytes  Time: 13 ms
Body JSON
JSON Content
Response
[
  {
    "_id": "67ebbe06eb50325244ec72c6",
    "name": "Vishal Mahajan",
    "email": "vism06@gmail.com",
    "password": "password@123",
    "age": 20,
    "createdAt": "2025-04-01T10:20:54.505Z",
    "updatedAt": "2025-04-01T10:20:54.505Z",
    "__v": 0
  }
]
```