# Name :- Sarthak Pagar

# Roll No. :- 40

# Class :- TE(IT)

# Practical 3b :- Create four API using Node.JS, ExpressJS and MongoDB for CURD Operations on assignment 2.C.

**client/app.js**

const app = angular.module('crudApp', []);

app.controller('UserController', function($scope, $http) {

$scope.name = '';

$scope.email = '';

$scope.users = [];

$scope.editRow = null;

$scope.message = '';

$scope.addUser = function() {

if (!$scope.name || !$scope.email) return;

$http.post('/api/add', {

name: $scope.name,

email: $scope.email

}).then(() => {

$scope.message = "✅ User added successfully!";

$scope.name = '';

$scope.email = '';

setTimeout(() => $scope.message = '', 2000); // auto-clear message

});

};

$scope.showUsers = function() {

$http.get('/api/show').then(res => {

$scope.users = res.data;

});

};

$scope.deleteUser = function(id) {

$http.delete('/api/delete/' + id).then(() => {

$scope.users = $scope.users.filter(user => user.\_id !== id);

});

};

$scope.enableEdit = function(user) {

$scope.editRow = user.\_id;

};

$scope.saveUpdate = function(user) {

$http.put('/api/update/' + user.\_id, {

name: user.name,

email: user.email

}).then(() => {

$scope.editRow = null;

$scope.message = "✅ User updated!";

setTimeout(() => $scope.message = '', 2000);

});

};

});

**client/index.html**

<!DOCTYPE html>

<html ng-app="crudApp">

<head>

<title>AngularJS MongoDB CRUD</title>

<meta charset="UTF-8" />

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

<script src="app.js"></script>

<link rel="stylesheet" href="styles.css">

</head>

<body ng-controller="UserController">

<div class="container">

<h1>User Management</h1>

<div class="form-section">

<input type="text" ng-model="name" placeholder="Enter Name" required>

<input type="email" ng-model="email" placeholder="Enter Email" required>

<button ng-click="addUser()">Add</button>

<button ng-click="showUsers()">Show Users</button>

</div>

<p class="message" ng-if="message">{{ message }}</p>

<div class="user-list" ng-if="users.length > 0">

<h2>All Users</h2>

<table>

<tr>

<th>Name</th>

<th>Email</th>

<th>Actions</th>

</tr>

<tr ng-repeat="u in users">

<td ng-if="editRow !== u.\_id">{{ u.name }}</td>

<td ng-if="editRow !== u.\_id">{{ u.email }}</td>

<td ng-if="editRow === u.\_id">

<input type="text" ng-model="u.name">

</td>

<td ng-if="editRow === u.\_id">

<input type="email" ng-model="u.email">

</td>

<td>

<button class="update-btn" ng-if="editRow !== u.\_id" ng-click="enableEdit(u)">Update</button>

<button class="save-btn" ng-if="editRow === u.\_id" ng-click="saveUpdate(u)">Save</button>

<button class="delete-btn" ng-click="deleteUser(u.\_id)">Delete</button>

</td>

</tr>

</table>

</div>

</div>

</body>

</html>

**client/styles.css**

body {

font-family: 'Segoe UI', sans-serif;

background-color: #f4f6f8;

padding: 20px;

}

.container {

max-width: 700px;

margin: auto;

background: white;

padding: 30px;

border-radius: 8px;

box-shadow: 0 0 15px rgba(0,0,0,0.1);

}

h1, h2 {

text-align: center;

color: #333;

}

.form-section {

display: flex;

flex-direction: column;

gap: 10px;

margin-bottom: 20px;

}

input[type="text"],

input[type="email"] {

padding: 10px;

font-size: 16px;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

padding: 10px;

font-size: 15px;

border: none;

border-radius: 4px;

cursor: pointer;

margin: 5px 0;

}

button:hover {

opacity: 0.9;

}

button[ng-click="addUser()"] {

background-color: #4CAF50;

color: white;

}

button[ng-click="showUsers()"] {

background-color: #2196F3;

color: white;

}

.update-btn {

background-color: #FFC107;

color: black;

margin-right: 5px;

}

.save-btn {

background-color: #8BC34A;

color: white;

margin-right: 5px;

}

.delete-btn {

background-color: #F44336;

color: white;

}

.message {

color: green;

text-align: center;

font-weight: bold;

margin-top: 10px;

}

table {

width: 100%;

border-collapse: collapse;

margin-top: 15px;

}

table th, table td {

padding: 12px;

border-bottom: 1px solid #ddd;

text-align: left;

}

**models/User.js**

const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({

name: String,

email: String

});

module.exports = mongoose.model('User', userSchema);

**routes/userRoutes.js**

const express = require('express');

const router = express.Router();

const User = require('../models/User');

// CREATE

router.post('/add', async (req, res) => {

try {

const user = new User(req.body);

const saved = await user.save();

res.json(saved);

} catch (err) {

res.status(400).json({ error: err.message });

}

});

// READ

router.get('/show', async (req, res) => {

const users = await User.find();

res.json(users);

});

// UPDATE

router.put('/update/:id', async (req, res) => {

const updated = await User.findByIdAndUpdate(req.params.id, req.body, { new: true });

res.json(updated);

});

// DELETE

router.delete('/delete/:id', async (req, res) => {

await User.findByIdAndDelete(req.params.id);

res.json({ message: "User deleted" });

});

module.exports = router;

**server.js**

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const bodyParser = require('body-parser');

const path = require('path');

const userRoutes = require('./routes/userRoutes');

const app = express();

const PORT = 3000;

// MongoDB Compass connection

mongoose.connect('mongodb://localhost:27017/Angular', {

useNewUrlParser: true,

useUnifiedTopology: true

}).then(() => console.log("Connected to MongoDB Compass"))

.catch(err => console.error("MongoDB error:", err));

// Middleware

app.use(cors());

app.use(bodyParser.json());

// Routes

app.use('/api', userRoutes);

// Serve static files

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

// Serve AngularJS frontend

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

res.sendFile(path.join(\_\_dirname, 'client/index.html'));

});

app.listen(PORT, () => {

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

});

**How to start**

open cmd in your project folder

type - > node server.js

you will get local host site

open in your browser

**Command Prompt**

Microsoft Windows [Version 10.0.19045.5608]

(c) Microsoft Corporation. All rights reserved.

C:\Users\sspag>cd C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 3B\crud-app

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 3B\crud-app>npm init -y

Wrote to C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 3B\crud-app\package.json:

{

"name": "crud-app",

"version": "1.0.0",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"keywords": [],

"author": "",

"license": "ISC",

"description": ""

}

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 3B\crud-app>npm install express mongoose cors body-parser

added 85 packages, and audited 86 packages in 24s

15 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 3B\crud-app>node server.js

(node:16100) [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

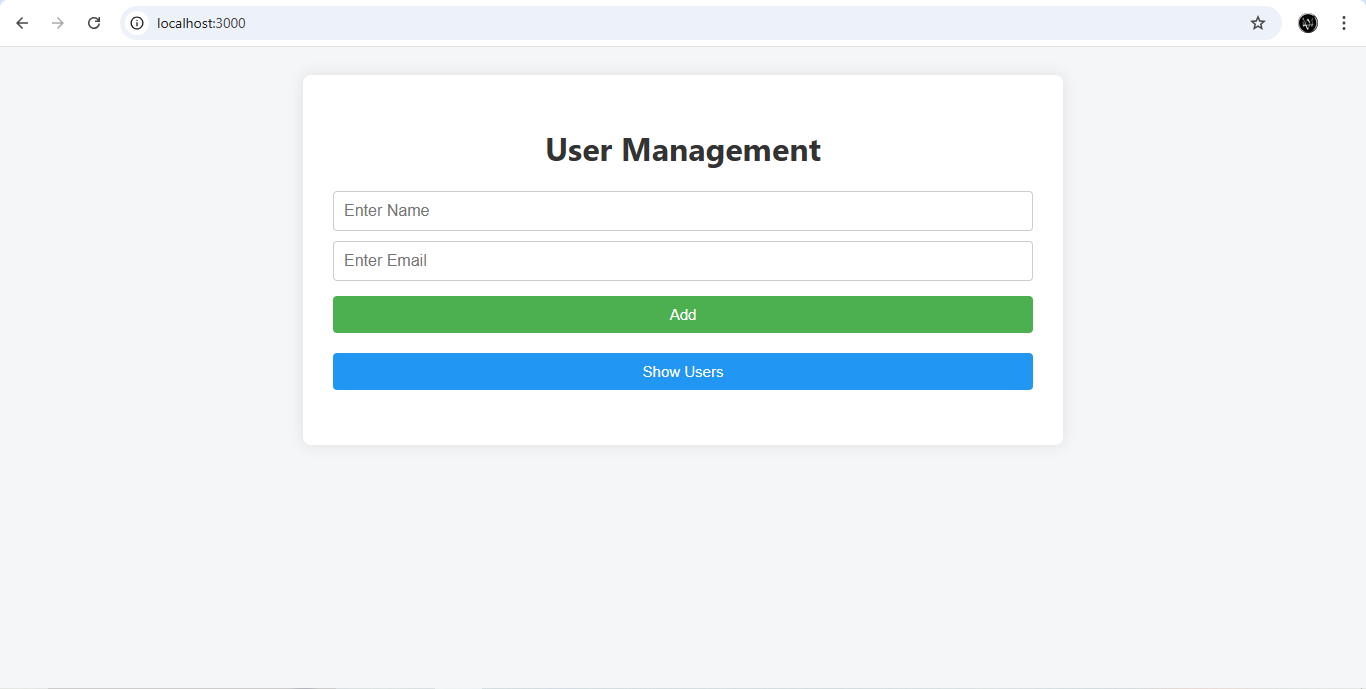
(Use `node --trace-warnings ...` to show where the warning was created)

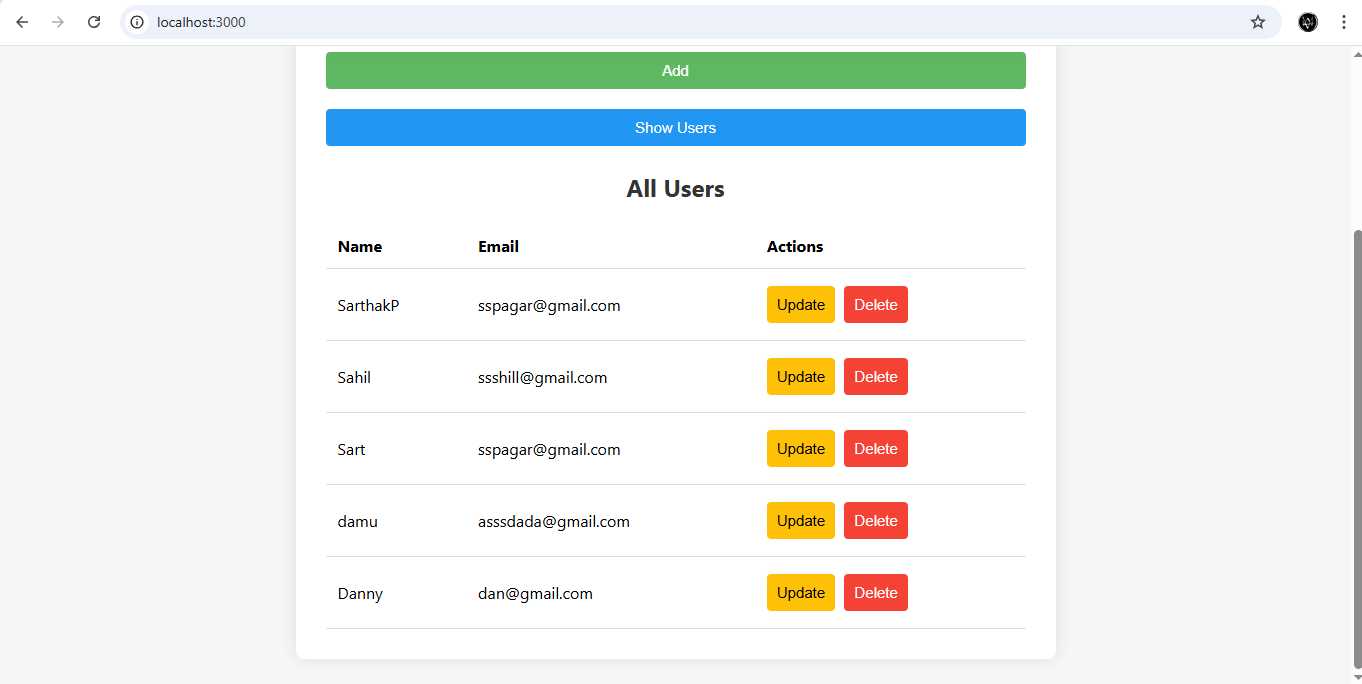
(node:16100) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version

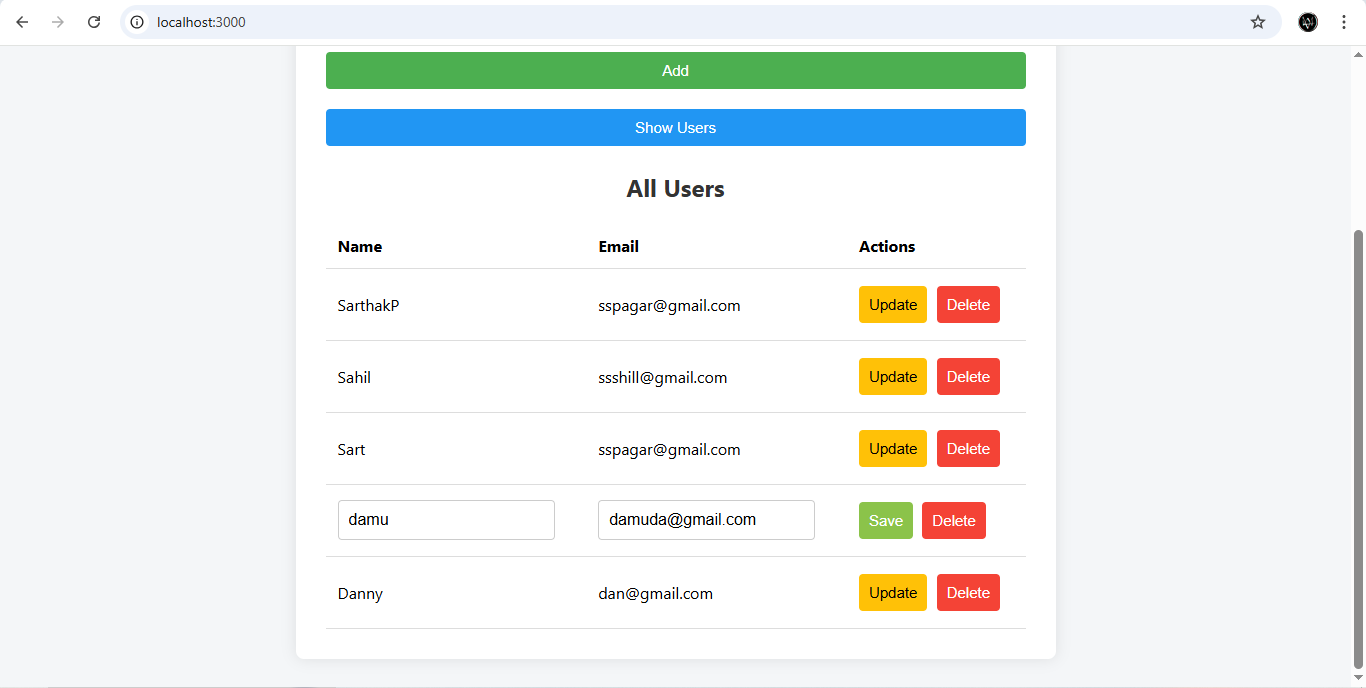
Server running at http://localhost:3000

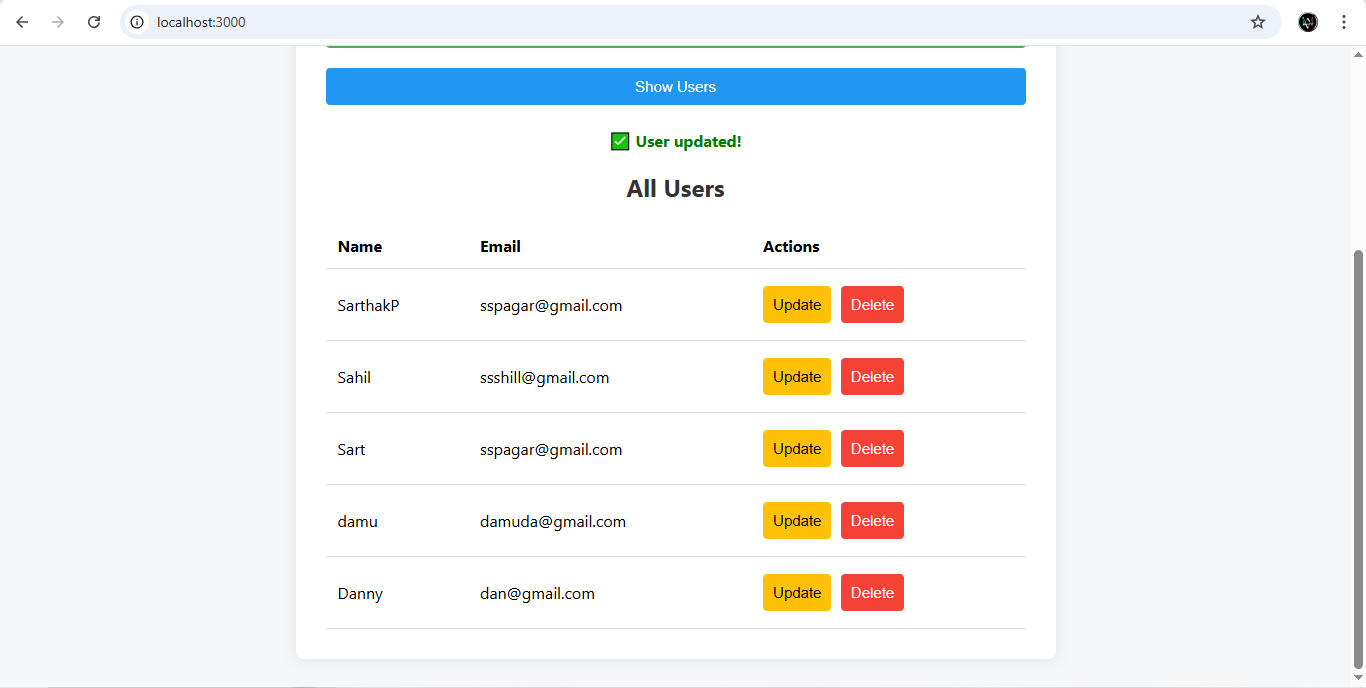
Connected to MongoDB Compass

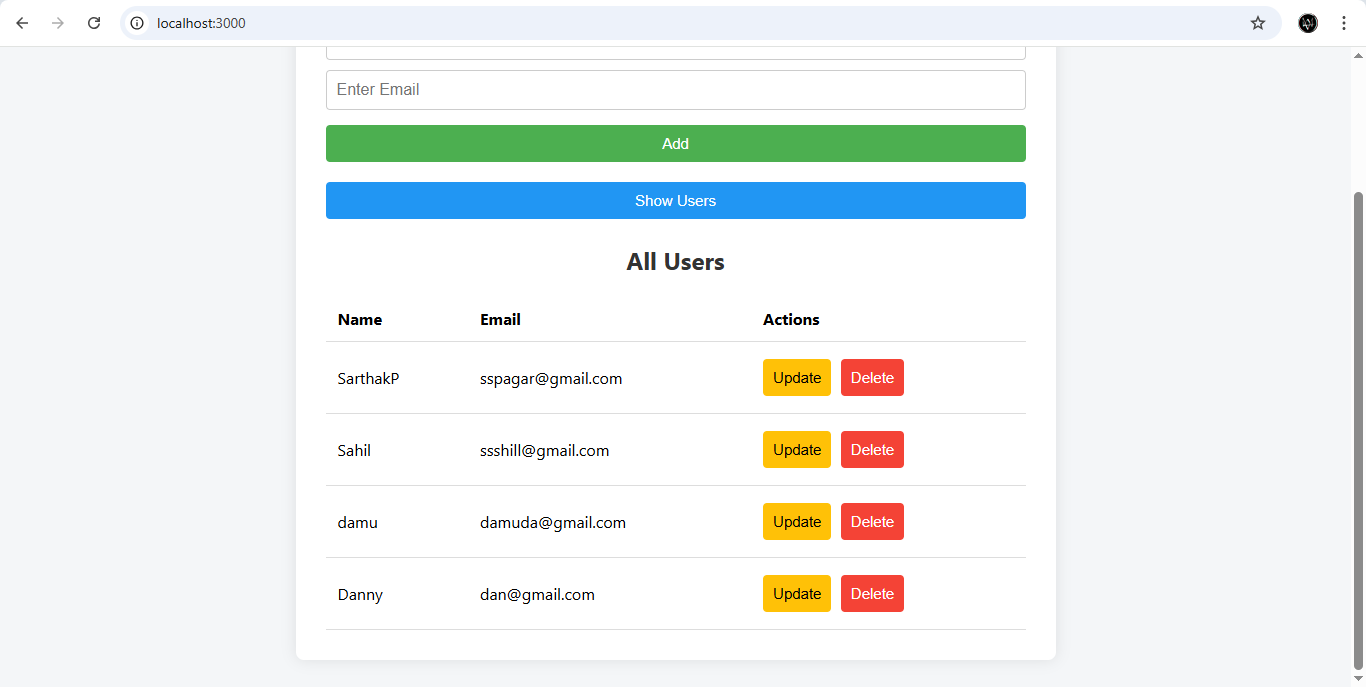
**Output :-**

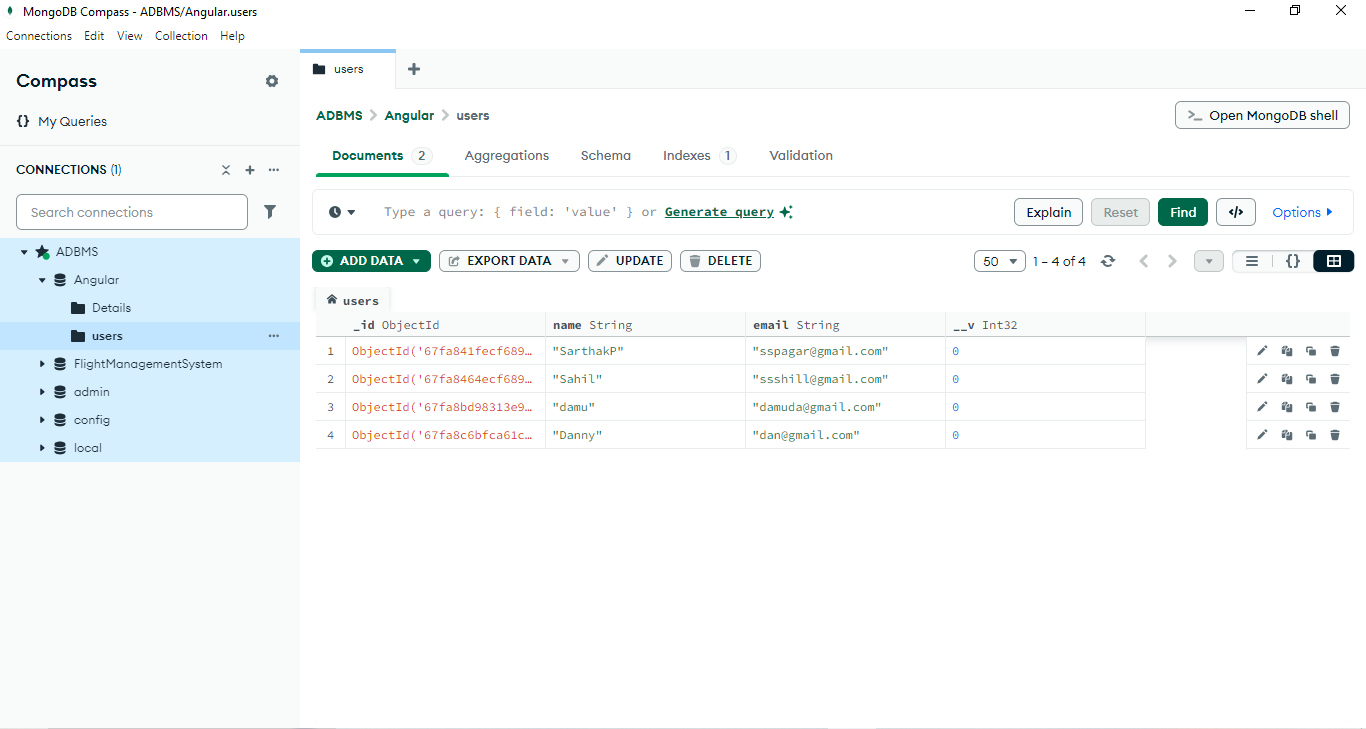
****

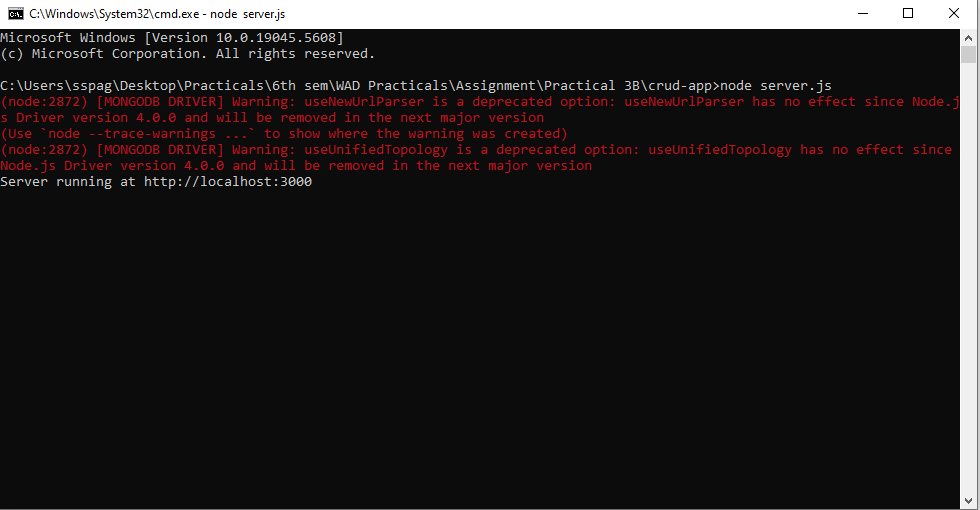
****

****

****

****

****

****