PROGRAM:

Exercise 1.html

```
<html>
<head>
<script>
function VALIDATEDETAIL()
{
var name = document.forms["RegForm"]["Name"];
var email = document.forms["RegForm"]["EMail"];
var phone = document.forms["RegForm"]["Telephone"];
var what =  document.forms["RegForm"]["Subject"];
var password = document.forms["RegForm"]["Password"];
var address = document.forms["RegForm"]["Address"];

   if (name.value == "")
   {
      window.alert("Please enter your name.");
      name.focus();
      return false;
   }

   if (address.value == "")
   {
      window.alert("Please enter your address.");
      name.focus();
      return false;
   }

   if (email.value == "")
   {
      window.alert("Please enter a valid e-mail address.");
      email.focus();
      return false;
```

```
    }

    if (email.value.indexOf("@", 0) < 0)
{
        window.alert("Please enter a valid e-mail address.");
        email.focus();
        return false;
    }
    if (email.value.indexOf(".", 0) < 0)
    {
        window.alert("Please enter a valid e-mail address.");
        email.focus();
        return false;
    }
    if (phone.value == "")
    {
        window.alert("Please enter your telephone number.");
        phone.focus();
        return false;
    }
    if (password.value == "")
    {
        window.alert("Please enter your password");
        password.focus();
        return flase;
    }

    if (what.selectedIndex < 1)
    {
        alert("Please enter your course.");
        what.focus();
        return false;
    }
```

```
    return true;
}</script>

  <style>
VALIDATEDETAIL {
    font-weight: bold ;
    float: left;
    width: 100px;
    text-align: left;
    margin-right: 10px;
    font-size:14px;
}
 div {
box-sizing: border-box;
    width: 100%;
    border: 100px solid black;
    float: left;
    align-content: center;
    align-items: center;
}

form {
    margin: 0 auto;
    width: 600px;
}</style></head>

<body>
<h1 style="text-align: center"> REGISTRATION FORM </h1>
<form name="RegForm" action="submit.php" onsubmit="return VALIDATEDETAIL()"
method="post">

    <p>Name: <input type="text" size=65 name="Name"> </p><br>
    <p> Address: <input type="text" size=65 name="Address">  </p><br>
```

```
    <p>E-mail Address: <input type="text" size=65 name="EMail">  </p><br>
     <p>Password: <input type="text" size=65 name="Password"> </p><br>
    <p>Telephone: <input type="text" size=65 name="Telephone"> </p><br>

    <p>SELECT YOUR COURSE
        <select type="text" value="" name="Subject">
            <option>BTECH</option>
            <option>BBA</option>
            <option>BCA</option>
            <option>B.COM</option>
            <option>VALIDATEDETAIL</option>
        </select></p><br><br>
    <p>Comments:  <textarea cols="55" name="Comment">  </textarea></p>
    <p><input type="submit" value="send" name="Submit">
        <input type="reset" value="Reset" name="Reset">



</p>
</form>
</body>
</html>
```

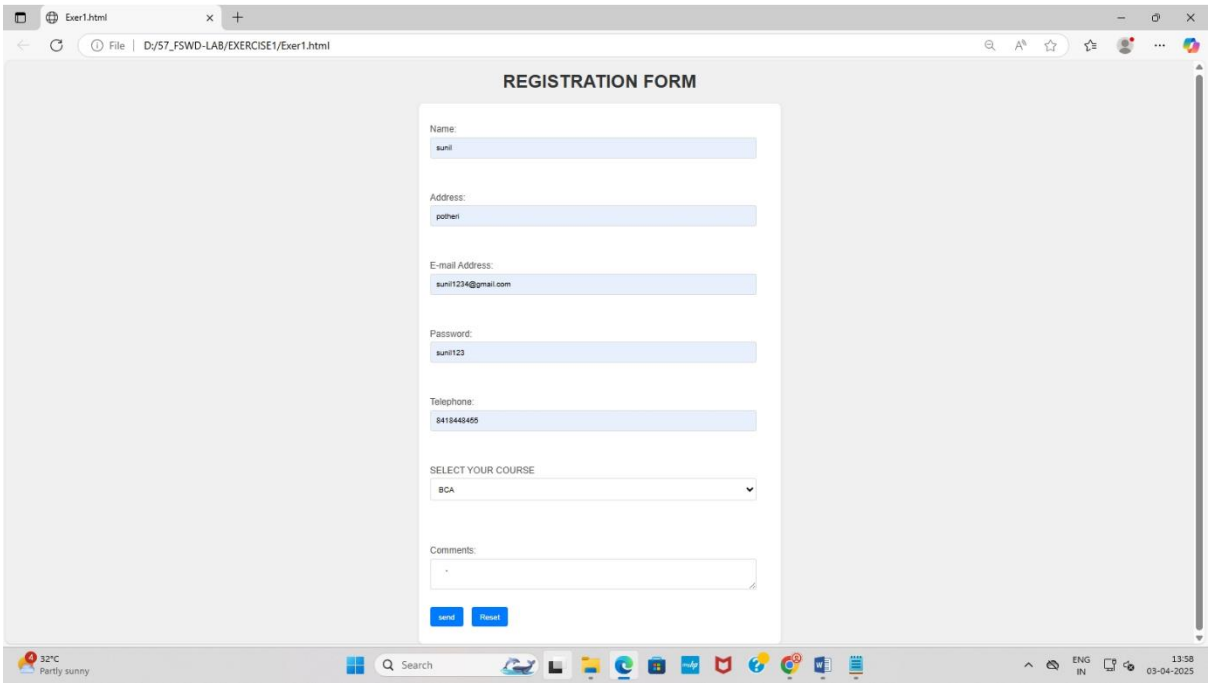
Style.css

```
body {
  font-family: sans-serif;
  background-color: #f0f0f0; /* Light gray background */
  margin: 0;
  padding: 0;
  display: flex;
  flex-direction: column; /* Stack elements vertically */
  align-items: center;
  min-height: 100vh;
}
```

```
h1 {
  text-align: center;
  color: #333; /* Dark gray title */
  margin-top: 20px;
  margin-bottom: 20px;
}

form {
  background-color: #fff; /* White form background */
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  width: 500px;
  max-width: 90%;
  margin-bottom: 20px; /* Space between form and buttons */
}

p {
  margin-bottom: 10px;
  color: #555;
}

input[type="text"],
select,
textarea {
  width: calc(100% - 22px);
  padding: 10px;
  margin-top: 5px;
  margin-bottom: 15px;
  border: 1px solid #ddd;
  border-radius: 4px;
  box-sizing: border-box;
}
```

```
select {
  height: 40px;
}

textarea {
  resize: vertical;
}

input[type="submit"],
input[type="reset"] {
  background-color: #007bff; /* Blue button */
  color: white;
  padding: 10px 15px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  margin-right: 10px;
}

input[type="submit"]:hover,
input[type="reset"]:hover {
  background-color: #0056b3; /* Darker blue on hover */
}

/* Optional: Style the button container for better alignment */
.button-container {
    display: flex;
    justify-content: center;
    width: 100%;
}
```

Output:


Exer1.html
File | D:/57_FSWD-LAB/EXERCISE1/Exer1.html
REGISTRATION FORM
Name:
sunil
Address:
potheri
E-mail Address:
sunil1234@gmail.com
Password:
sunil123
Telephone:
8418448466
SELECT YOUR COURSE
BCA
Comments:
send
Reset
32°C
Partly sunny
Search
ENG
IN
13:58
03-04-2025

PROGRAM :

Exercise 2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fetch API Example</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #08335c;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            flex-direction: column;
        }
        .card {
            background: rgb(210, 212, 223);
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
            margin: 10px;
            width: 300px;
            text-align: center;
        }
    </style>
</head>
<body>

    <h2>Fetch API Data Display</h2>
    <div id="card-container"></div>
```

```
    <script>
        async function fetchData() {
            try {
                //const response = await fetch('https://jsonplaceholder.typicode.com/users');
                const response = await fetch('http://127.0.0.1:8081/users.json');
                const data = await response.json();

                const container = document.getElementById('card-container');
                container.innerHTML = ''; // Clear previous content
                data.slice(0, 5).forEach(user => {
                    const card = document.createElement('div');
                    card.classList.add('card');
                    card.innerHTML = `
                        <h3>${user.name}</h3>
                        <p><strong>Email:</strong> ${user.email}</p>
                        <p><strong>City:</strong> ${user.address.city}</p>
                    `;
                    container.appendChild(card);
                });
            } catch (error) {
                console.error('Error fetching data:', error);
            }
        }
        fetchData(); // Call function to fetch and display data
</script>
</body>
</html>
```

Users.json:

```
[
  {
    "id": 1,
    "name": "Sunil Aldo S A",
    "username": "sunil007",
```

```
  "email": "sunilado2004@srmvec.ac.in",
  "address": {
    "street": "Kulas Light",
    "suite": "Apt. 556",
    "city": "Potheri",
    "zipcode": "627005",
    "geo": {
      "lat": "-37.3159",
      "lng": "81.1496"
    }
  },
  "phone": "91-80729 22340",
  "website": "cits.org",
  "company": {
    "name": "Creative-Crona",
    "catchPhrase": "Multi-layered client-server neural-net",
    "bs": "harness real-time e-markets"
  }
},
{
  "id": 2,
  "name": "Sibi Karthick",
  "username": "Antonette",
  "email": "Shanna@melissa.tv",
  "address": {
    "street": "Victor Plains",
    "suite": "Suite 879",
    "city": "Wisokyburgh",
    "zipcode": "90566-7771",
    "geo": {
      "lat": "-43.9509",
      "lng": "-34.4618"
    }
```

```
  },
  "phone": "010-692-6593 x09125",
  "website": "anastasia.net",
  "company": {
    "name": "Deckow-Crist",
    "catchPhrase": "Proactive didactic contingency",
    "bs": "synergize scalable supply-chains"
  }
},
{
  "id": 3,
  "name": "Antro Jeffrie",
  "username": "Samantha",
  "email": "Nathan@yesenia.net",
  "address": {
    "street": "Douglas Extension",
    "suite": "Suite 847",
    "city": "McKenziehaven",
    "zipcode": "59590-4157",
    "geo": {
      "lat": "-68.6102",
      "lng": "-47.0653"
    }
  },
  "phone": "2-653-123-3337",
  "website": "ramiro.info",
  "company": {
    "name": "Romaguera-Jacobson",
    "catchPhrase": "Face to face bifurcated interface",
    "bs": "e-enable strategic applications"
  }
},
{
```

```
  "id": 4,
  "name": "Fathima",
  "username": "Karianne",
  "email": "Julianne.OConner@kory.org",
  "address": {
    "street": "Hoeger Mall",
    "suite": "Apt. 692",
    "city": "South Elvis",
    "zipcode": "53919-4257",
    "geo": {
      "lat": "29.4572",
      "lng": "-164.2990"
    }
  },
  "phone": "493-170-9623 x156",
  "website": "kale.biz",
  "company": {
    "name": "Robel-Corkery",
    "catchPhrase": "Multi-tiered zero tolerance productivity",
    "bs": "transition cutting-edge web services"
  }
},
{
  "id": 5,
  "name": "Ragul Gandhi",
  "username": "Kamren",
  "email": "Lucio_Hettinger@annie.ca",
  "address": {
    "street": "Skiles Walks",
    "suite": "Suite 351",
    "city": "Roscoeview",
    "zipcode": "33263",
    "geo": {
```

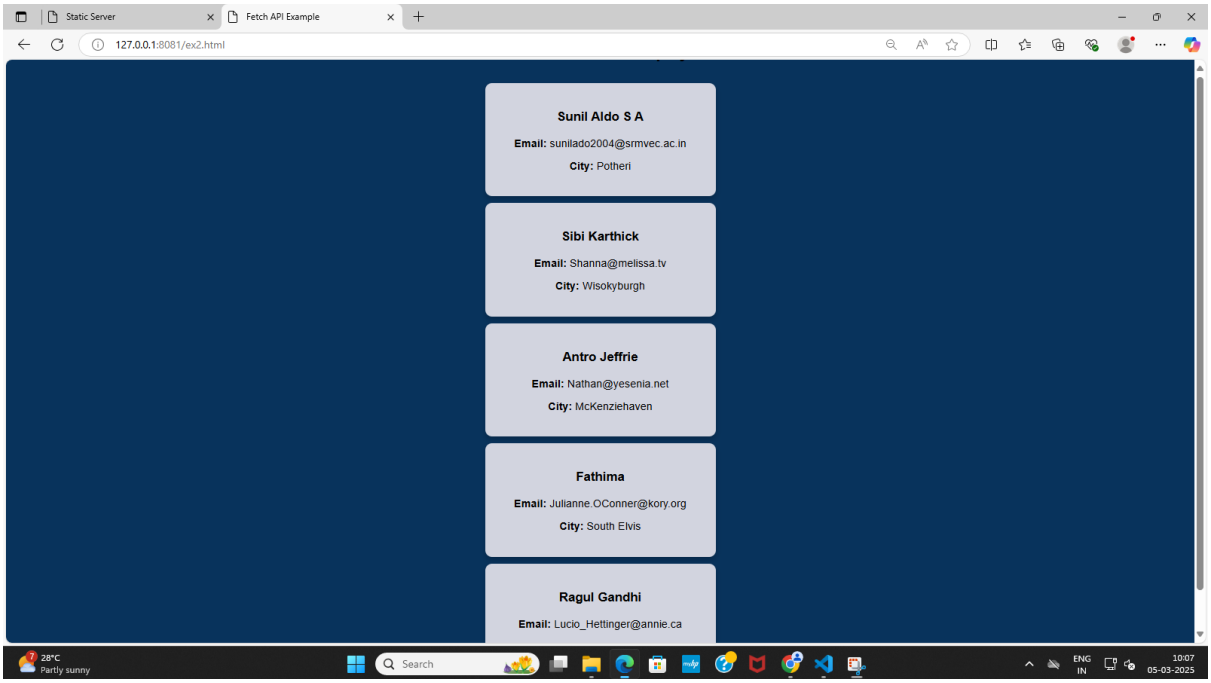
```
      "lat": "-31.8129",
      "lng": "62.5342"
    }
  },
  "phone": "(254)954-1289",
  "website": "demarco.info",
  "company": {
    "name": "Keebler LLC",
    "catchPhrase": "User-centric fault-tolerant solution",
    "bs": "revolutionize end-to-end systems"
  }
},
 {
  "id": 6,
  "name": "Mrs. Dennis Schulist",
  "username": "Leopoldo_Corkery",
  "email": "Karley_Dach@jasper.info",
  "address": {
    "street": "Norberto Crossing",
    "suite": "Apt. 950",
    "city": "South Christy",
    "zipcode": "23505-1337",
    "geo": {
      "lat": "-71.4197",
      "lng": "71.7478"
    }
  },
  "phone": "1-477-935-8478 x6430",
  "website": "ola.org",
  "company": {
    "name": "Considine-Lockman",
    "catchPhrase": "Synchronised bottom-line interface",
    "bs": "e-enable innovative applications"
```

```
    }
  },
  {
    "id": 7,
    "name": "Kurtis Weissnat",
    "username": "Elwyn.Skiles",
    "email": "Telly.Hoeger@billy.biz",
    "address": {
      "street": "Rex Trail",
      "suite": "Suite 280",
      "city": "Howemouth",
      "zipcode": "58804-1099",
      "geo": {
        "lat": "24.8918",
        "lng": "21.8984"
      }
    },
    "phone": "210.067.6132",
    "website": "elvis.io",
    "company": {
      "name": "Johns Group",
      "catchPhrase": "Configurable multimedia task-force",
      "bs": "generate enterprise e-tailers"
    }
  },
  {
    "id": 8,
    "name": "Nicholas Runolfsdottir V",
    "username": "Maxime_Nienow",
    "email": "Sherwood@rosamond.me",
    "address": {
      "street": "Ellsworth Summit",
      "suite": "Suite 729",
```

```
    "city": "Aliyaview",
    "zipcode": "45169",
    "geo": {
      "lat": "-14.3990",
      "lng": "-120.7677"
    }
  },
  "phone": "586.493.6943 x140",
  "website": "jacynthe.com",
  "company": {
    "name": "Abernathy Group",
    "catchPhrase": "Implemented secondary concept",
    "bs": "e-enable extensible e-tailers"
  }
},
{
  "id": 9,
  "name": "Glenna Reichert",
  "username": "Delphine",
  "email": "Chaim_McDermott@dana.io",
  "address": {
    "street": "Dayna Park",
    "suite": "Suite 449",
    "city": "Bartholomebury",
    "zipcode": "76495-3109",
    "geo": {
      "lat": "24.6463",
      "lng": "-168.8889"
    }
  },
  "phone": "(775)976-6794 x41206",
  "website": "conrad.com",
  "company": {
```

```
      "name": "Yost and Sons",
      "catchPhrase": "Switchable contextually-based project",
      "bs": "aggregate real-time technologies"
    }
  },
  {
    "id": 10,
    "name": "Clementina DuBuque",
    "username": "Moriah.Stanton",
    "email": "Rey.Padberg@karina.biz",
    "address": {
      "street": "Kattie Turnpike",
      "suite": "Suite 198",
      "city": "Lebsackbury",
      "zipcode": "31428-2261",
      "geo": {
        "lat": "-38.2386",
        "lng": "57.2232"
      }

    },
    "phone": "024-648-3804",
    "website": "ambrose.net",
    "company": {
      "name": "Hoeger LLC",
      "catchPhrase": "Centralized empowering task-force",
      "bs": "target end-to-end models"
    }
  }
]
```

Output:


Static Server
Fetch API Example
127.0.0.1:8081/ex2.html
Sunil Aldo S A
Email: sunilado2004@srmvec.ac.in
City: Potheri
Sibi Karthick
Email: Shanna@melissa.tv
City: Wisokyburgh
Antro Jeffrie
Email: Nathan@yesenia.net
City: McKenziehaven
Fathima
Email: Julianne.OConner@kory.org
City: South Elvis
Ragul Gandhi
Email: Lucio_Hettinger@annie.ca
28°C
Partly sunny
Search
ENG
IN
10:07
05-03-2025

PROGRAM:

Exercise 3.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Static Server</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <h1>Welcome to My Node.js Server</h1>
    <p>This is a static HTML page served without Express.</p>
</body>
</html>
```

Ex_3_server.js

```
const http = require('http');
const fs = require('fs');
const path = require('path');

const PORT = 3000;
const PUBLIC_DIR = path.join(__dirname, 'public');

// MIME types for various file extensions
const mimeTypes = {
    '.html': 'text/html',
    '.css': 'text/css',
    '.js': 'text/javascript',
    '.json': 'application/json',
    '.jpg': 'image/jpeg',
    '.jpeg': 'image/jpeg',
    '.png': 'image/png',
```

```
    '.gif': 'image/gif',
    '.svg': 'image/svg+xml',
    '.ico': 'image/x-icon',
    '.woff': 'font/woff',
    '.woff2': 'font/woff2',
    '.ttf': 'font/ttf',
    '.eot': 'application/vnd.ms-fontobject',
};

const server = http.createServer((req, res) => {
    let filePath = path.join(PUBLIC_DIR, req.url === '/' ? 'ex_3.html' : req.url);

    // Get file extension
    let extname = path.extname(filePath).toLowerCase();

    // Default to text/html if file extension not found
    let contentType = mimeTypes[extname] || 'text/html';

    // Read and serve the file
    fs.readFile(filePath, (err, content) => {
        if (err) {
            if (err.code === 'ENOENT') {
                res.writeHead(404, { 'Content-Type': 'text/html' });
                res.end('<h1>404 - Not Found</h1>');
            } else {
                res.writeHead(500);
                res.end(`<h1>500 - Server Error</h1><p>${err.code}</p>`);
            }
        } else {
            res.writeHead(200, { 'Content-Type': contentType });
            res.end(content, 'utf-8');
        }
    });
```

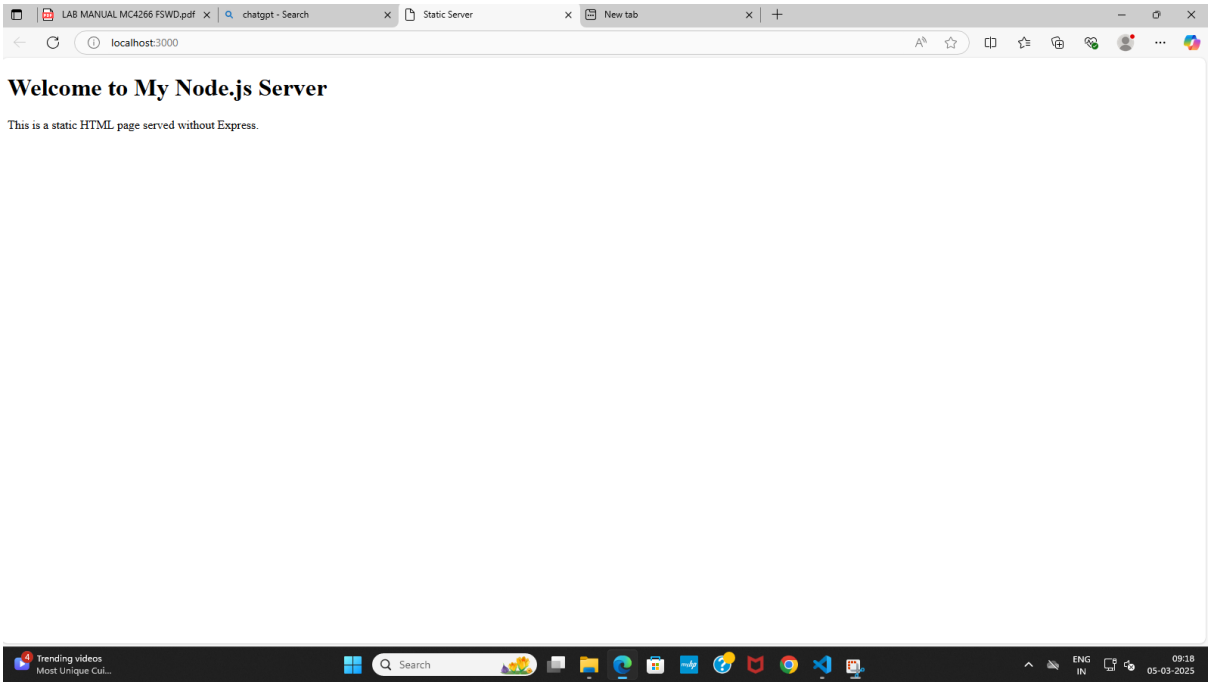
```
});

server.listen(PORT, () => {
    console.log(`Server running at http://localhost:${PORT}`);
});
```

Server.js

```
const http = require('http');
const fs = require('fs');
const path = require('path');

const PORT = 3000;
const PUBLIC_DIR = path.join(__dirname, 'public');

const server = http.createServer((req, res) => {
    let filePath = path.join(PUBLIC_DIR, req.url === '/' ? 'ex_3.html' : req.url);

    // Get file extension
    let extname = path.extname(filePath);

    // Set content type
    let contentType = 'text/html';
    if (extname === '.css') contentType = 'text/css';
    if (extname === '.js') contentType = 'text/javascript';

    // Read and serve the file
    fs.readFile(filePath, (err, content) => {
        if (err) {
            if (err.code === 'ENOENT') {
                res.writeHead(404, { 'Content-Type': 'text/html' });
                res.end('<h1>404 - Not Found</h1>');
            } else {
                res.writeHead(500);
```

```
            res.end(`Server Error: ${err.code}`);
        }
    } else {
        res.writeHead(200, { 'Content-Type': contentType });
        res.end(content, 'utf-8');
    }
  });
});

server.listen(PORT, () => {
  console.log(`Server running at http://localhost:${PORT}`);
});
```

Output:


LAB MANUAL MC4266 FSWD.pdf
chatgpt - Search
Static Server
New tab
localhost:3000
Welcome to My Node.js Server
This is a static HTML page served without Express.
Trending videos
Most Unique Cui...
Search
ENG
IN
09:18
05-03-2025

PROGRAM:

Exercise4.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Form Submission</title>
</head>
<body>
    <h2>Submit Your Details</h2>
    <form action="/submit" method="post">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required><br><br>

        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required><br><br>

        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

Ex_4_Server.js

```
const express = require('express');
const bodyParser = require('body-parser');
const fs = require('fs');

const app = express();
const PORT = 3000;

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

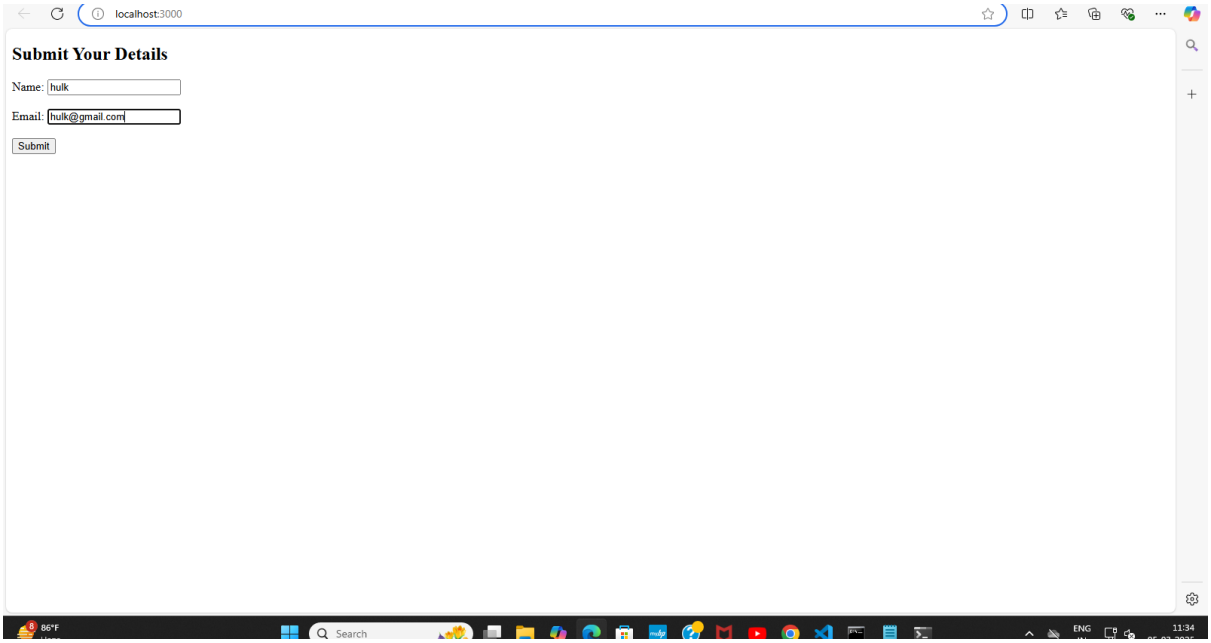
```
app.use(bodyParser.json());

// Serve the HTML form (optional)
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/index.html');
});

// Handle form submission
app.post('/submit', (req, res) => {
    const formData = req.body;

    // Read existing data
    let existingData = [];
    if (fs.existsSync('data.json')) {
        const fileContent = fs.readFileSync('data.json');
        existingData = JSON.parse(fileContent);
    }
    existingData.push(formData);
    fs.writeFileSync('data.json', JSON.stringify(existingData, null, 2));
    res.send('Form data saved successfully!');
});

app.listen(PORT, () => {
    console.log(`Server running at http://localhost:${PORT}`);
});
```

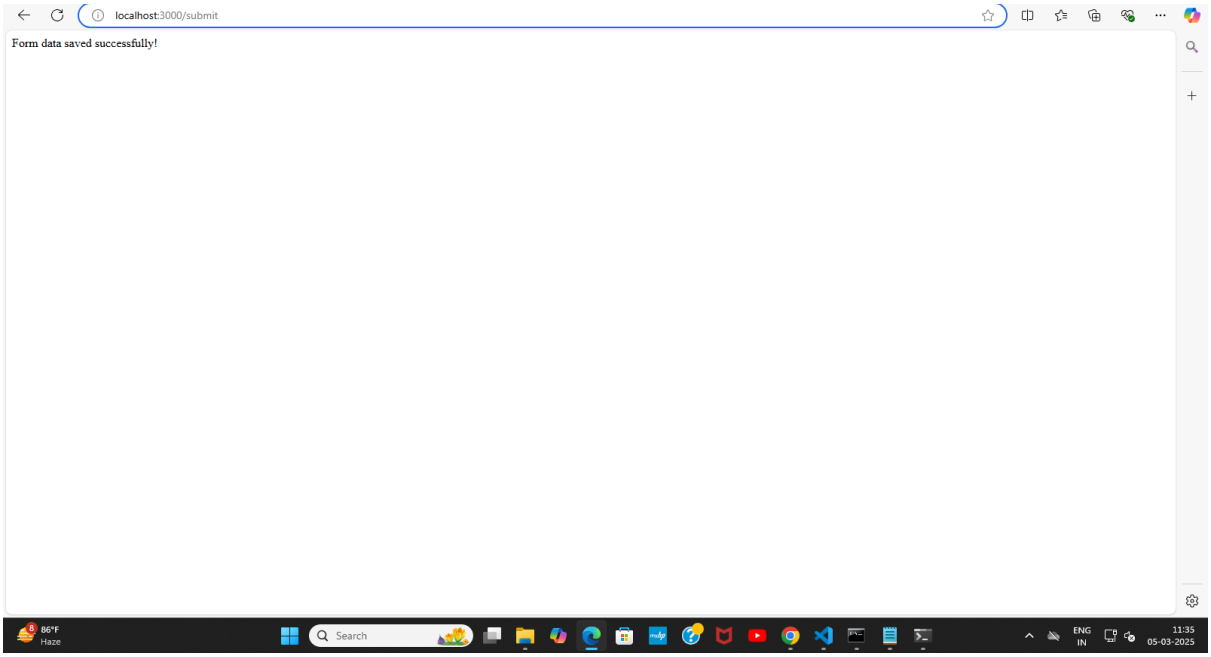
Data.json

```
[
  {
    "name": "Hulk",
    "email": "Hulk@gmail.com"
  },
  {
```

```
    "name": "Hulk",
    "email": "Hulk@gmail.com"
  }
]
```

Output:


localhost:3000
Submit Your Details
Name: hulk
Email: hulk@gmail.com
Submit


localhost:3000/submit
Form data saved successfully!

PROGRAM:

Exercise 5.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Customer CRUD App</title>

        <link rel="stylesheet" href="ex_5_style.css"> <!-- Link the CSS file -->
</head>
<body>

    <h2>Customer Management System</h2>

    <form id="customerForm">
        <input type="text" id="name" placeholder="Enter Name" required>
        <input type="text" id="city" placeholder="Enter City" required>
        <input type="text" id="mobile" placeholder="Enter Mobile No" required>
        <button type="submit">Add Customer</button>
    </form>

    <table>
        <thead>
            <tr>
                <th>Name</th>
                <th>City</th>
                <th>Mobile No</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody id="customersList"></tbody>
    </table>
```

```
<script>
    const API_URL = 'http://localhost:5000/customers';

    async function fetchCustomers() {
        const res = await fetch(API_URL);
        const customers = await res.json();
        document.getElementById('customersList').innerHTML = customers.map(customer
=> `
            <tr>
                <td><input type="text" value="${customer.name}" id="name-
${customer._id}"></td>
                <td><input type="text" value="${customer.city}" id="city-
${customer._id}"></td>
                <td><input type="text" value="${customer.mobile}" id="mobile-
${customer._id}"></td>
                <td>
                    <button onclick="updateCustomer('${customer._id}')">Update</button>
                    <button onclick="deleteCustomer('${customer._id}')">Delete</button>
                </td>
            </tr>
        `).join('');
    }

    async function addCustomer(event) {
        event.preventDefault();
        const name = document.getElementById('name').value;
        const city = document.getElementById('city').value;
        const mobile = document.getElementById('mobile').value;

        await fetch(API_URL, {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ name, city, mobile })
        });
```

```
        document.getElementById('customerForm').reset();
        fetchCustomers();
    }

    async function updateCustomer(id) {
        const name = document.getElementById(`name-${id}`).value;
        const city = document.getElementById(`city-${id}`).value;
        const mobile = document.getElementById(`mobile-${id}`).value;

        await fetch(`${API_URL}/${id}`, {
            method: 'PUT',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ name, city, mobile })
        });

        fetchCustomers();
    }

    async function deleteCustomer(id) {
        await fetch(`${API_URL}/${id}`, { method: 'DELETE' });
        fetchCustomers();
    }

    document.getElementById('customerForm').addEventListener('submit', addCustomer);
    fetchCustomers();
  </script>

</body>
</html>
```

Exercise5 style.css

```
/* General Page Styling */
body {
    font-family: Arial, sans-serif;
    text-align: center;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
}

/* Form Styling */
form {
    margin: 20px auto;
    width: 50%;
    padding: 15px;
    background: white;
    border-radius: 8px;
    box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
}

input {
    padding: 10px;
    margin: 5px;
    width: 80%;
    border: 1px solid #ccc;
    border-radius: 5px;
}

/* Table Styling */
table {
    width: 60%;
    margin: 20px auto;
    border-collapse: collapse;
```

```
    background: white;
    box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
}

th, td {
    padding: 10px;
    border: 1px solid #ddd;
    text-align: center;
}

th {
    background-color: #007bff;
    color: white;
}

/* Button Styling */
button {
    padding: 8px 12px;
    border: none;
    cursor: pointer;
    color: white;
    border-radius: 5px;
}

button:hover {
    opacity: 0.8;
}

button:nth-child(1) { background-color: #28a745; } /* Update button - Green */
button:nth-child(2) { background-color: #dc3545; } /* Delete button - Red */
```

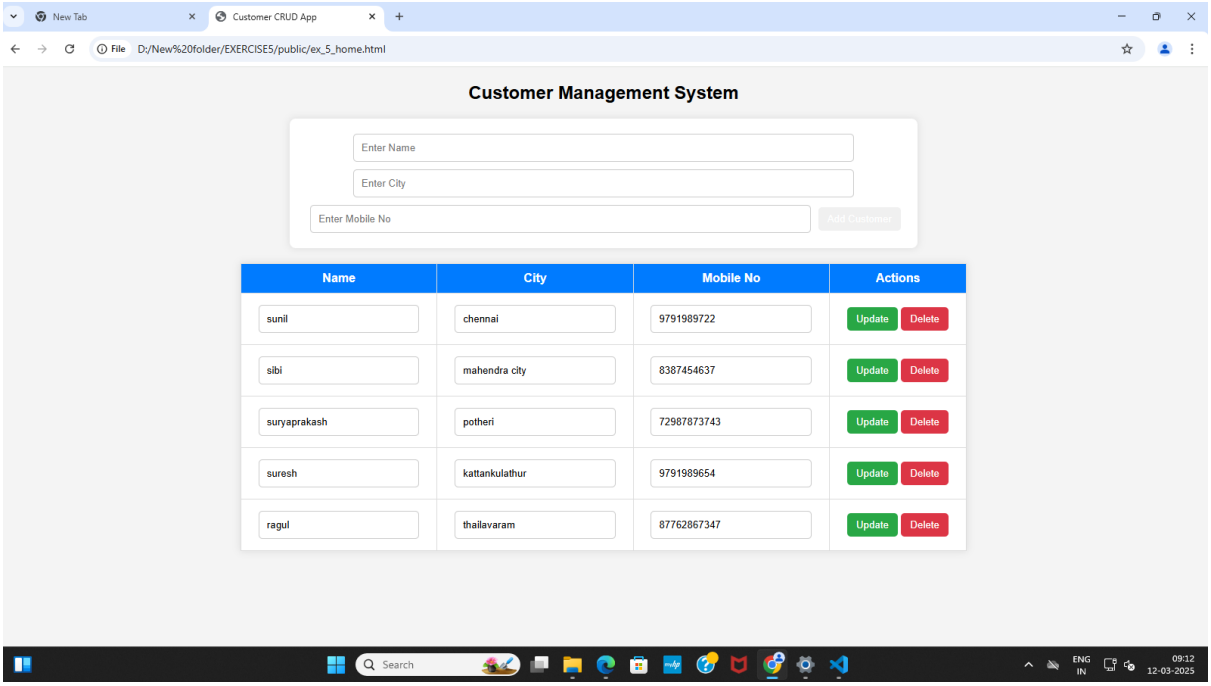
Ex_5_server.js

```
require('dotenv').config();
```

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');

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

const MONGO_URI = process.env.MONGO_URI;
if (!MONGO_URI) {
    console.error("❌ MONGO_URI is not set in .env file");
    process.exit(1);
}

mongoose.connect(MONGO_URI, { useNewUrlParser: true, useUnifiedTopology: true })
    .then(() => console.log('✅ MongoDB Connected'))
    .catch(err => console.error(err));

// Customer Schema
const CustomerSchema = new mongoose.Schema({
    name: { type: String, required: true },
    city: { type: String, required: true },
    mobile: { type: String, required: true }
});
const Customer = mongoose.model('Customer', CustomerSchema);

// Create
app.post('/customers', async (req, res) => {
    try {
        const newCustomer = new Customer(req.body);
        await newCustomer.save();
        res.json(newCustomer);
    } catch (err) {
```

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

// Read
app.get('/customers', async (req, res) => {
    const customers = await Customer.find();
    res.json(customers);
});

// Update
app.put('/customers/:id', async (req, res) => {
    try {
        const updatedCustomer = await Customer.findByIdAndUpdate(req.params.id, req.body,
{ new: true });
        res.json(updatedCustomer);
    } catch (err) {
        res.status(500).json({ error: err.message });
    }
});

// Delete
app.delete('/customers/:id', async (req, res) => {
    try {
        await Customer.findByIdAndDelete(req.params.id);
        res.json({ message: 'Customer deleted' });
    } catch (err) {
        res.status(500).json({ error: err.message });
    }
});

app.listen(5000, () => console.log('🚀 Server running on port 5000'));
```

Output:


New Tab
Customer CRUD App
File
D:/New%20folder/EXERCISE5/public/ex_5_home.html
Customer Management System
Enter Name
Enter City
Enter Mobile No
Add Customer
Name
City
Mobile No
Actions
sunil
chennai
9791989722
Update
Delete
sibi
mahendra city
8387454637
Update
Delete
suryaprakash
potheri
72987873743
Update
Delete
suresh
kattankulathur
9791989654
Update
Delete
ragul
thaillavaram
87762867347
Update
Delete
Search
ENG
IN
09:12
12-03-2025

PROGRAM:

Exercise 6.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Customer CRUD App</title>
    <link rel="stylesheet" href="ex_6_style.css">
</head>
<body>
    <h2>Customer Management System</h2>
    <form id="customerForm">
        <input type="text" id="name" placeholder="Enter Name" required>
        <input type="text" id="city" placeholder="Enter City" required>
        <input type="text" id="mobile" placeholder="Enter Mobile No" required>
        <button type="submit">Add Customer</button>
    </form>
    <table>
        <thead>
            <tr>
                <th>Name</th>
                <th>City</th>
                <th>Mobile No</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody id="customersList"></tbody>
    </table>
    <script>
      const API_URL = 'http://localhost:5000/customers';
       async function fetchCustomers() {
           const res = await fetch(API_URL);
```

```
        const customers = await res.json();
        document.getElementById('customersList').innerHTML = customers.map(customer
=> `
            <tr>
                <td><input type="text" value="${customer.name}" id="name-
${customer.id}"></td>
                <td><input type="text" value="${customer.city}" id="city-
${customer.id}"></td>
                <td><input type="text" value="${customer.mobile}" id="mobile-
${customer.id}"></td>
                <td>
                    <button onclick="updateCustomer('${customer.id}')">Update</button>
                    <button onclick="deleteCustomer('${customer.id}')">Delete</button>
                </td>
            </tr>
        `).join('');
    }
    async function addCustomer(event) {
        event.preventDefault();
        const name = document.getElementById('name').value;
        const city = document.getElementById('city').value;
        const mobile = document.getElementById('mobile').value;
        await fetch(API_URL, {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ name, city, mobile })
        });
        document.getElementById('customerForm').reset();
        fetchCustomers();
    }
    async function updateCustomer(id) {
        const name = document.getElementById(`name-${id}`).value;
        const city = document.getElementById(`city-${id}`).value;
        const mobile = document.getElementById(`mobile-${id}`).value;
        await fetch(`${API_URL}/${id}`, {
```

```
            method: 'PUT',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ name, city, mobile })
        });
        fetchCustomers();
    }
    async function deleteCustomer(id) {
        await fetch(`${API_URL}/${id}`, { method: 'DELETE' });
        fetchCustomers();
    }
    document.getElementById('customerForm').addEventListener('submit', addCustomer);
    fetchCustomers();
  </script>
</body>
</html>
```

Exercise 6 style.css

```
body {
  font-family: Arial, sans-serif;
  text-align: center;
  background-color: #f4f4f4;
}
form {
  margin: 20px auto;
  width: 50%;
  padding: 15px;
  background: white;
  border-radius: 8px;
}
input {
  padding: 10px;
  margin: 5px;
  width: 80%;
  border: 1px solid #ccc;
```

```
}
table {
    width: 60%;
    margin: 20px auto;
    border-collapse: collapse;
}
th, td {
    padding: 10px;
    border: 1px solid #ddd;
    text-align: center;
}
button {
    padding: 8px 12px;
    border: none;
    color: white;
    border-radius: 5px;
}
button:nth-child(1) { background-color: #28a745; }
button:nth-child(2) { background-color: #dc3545; }
```

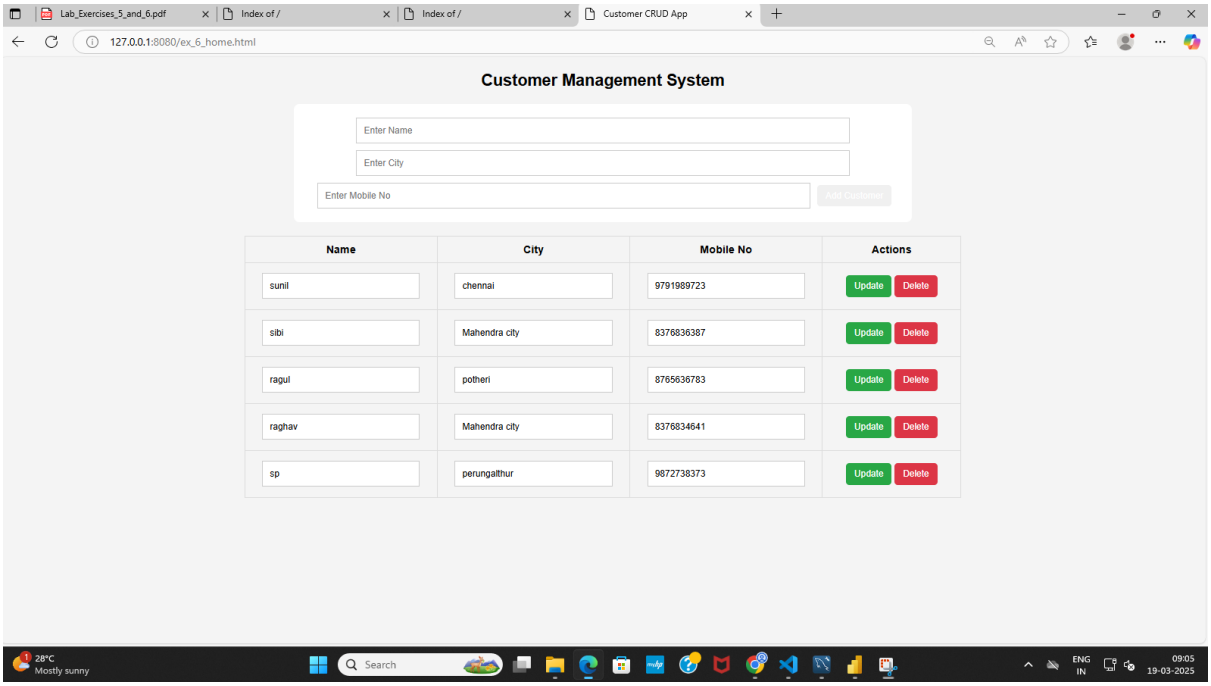
Ex_6_server.js

```
require('dotenv').config();
const express = require('express');
const mysql = require('mysql2');
const cors = require('cors');
const app = express();
app.use(express.json());
app.use(cors());
const db = mysql.createConnection({
    host: process.env.DB_HOST,
    user: process.env.DB_USER,
    password: process.env.DB_PASS,
    database: process.env.DB_NAME
});
```

```
db.connect(err => {
    if (err) {
        console.error("❌ MySQL Connection Error:", err.message);
        process.exit(1);
    }
    console.log('✅ MySQL Connected');
});
app.post('/customers', (req, res) => {
    const { name, city, mobile } = req.body;
    const sql = 'INSERT INTO customers (name, city, mobile) VALUES (?, ?, ?)';
    db.query(sql, [name, city, mobile], (err, result) => {
        if (err) return res.status(500).json({ error: err.message });
        res.json({ id: result.insertId, name, city, mobile });
    });
});
app.get('/customers', (req, res) => {
    const sql = 'SELECT * FROM customers';
    db.query(sql, (err, results) => {
        if (err) return res.status(500).json({ error: err.message });
        res.json(results);
    });
});
app.put('/customers/:id', (req, res) => {
    const { name, city, mobile } = req.body;
    const sql = 'UPDATE customers SET name = ?, city = ?, mobile = ? WHERE id = ?';
    db.query(sql, [name, city, mobile, req.params.id], (err) => {
        if (err) return res.status(500).json({ error: err.message });
        res.json({ message: 'Customer updated successfully' });
    });
});
app.delete('/customers/:id', (req, res) => {
    const sql = 'DELETE FROM customers WHERE id = ?';
    db.query(sql, [req.params.id], (err) => {
```

```
        if (err) return res.status(500).json({ error: err.message });
        res.json({ message: 'Customer deleted successfully' });
    });
});
app.listen(5000, () => console.log('🚀 Server running on port 5000'));
```

Output:


Lab_Exercises_5_and_6.pdf
Index of /
Index of /
Customer CRUD App
127.0.0.1:8080/ex_6_home.html
Customer Management System
Enter Name
Enter City
Enter Mobile No
Add Customer
Name
City
Mobile No
Actions
sunil
chennai
9791989723
Update
Delete
sibi
Mahendra city
8376836387
Update
Delete
ragul
potheri
8765636783
Update
Delete
raghav
Mahendra city
8376834641
Update
Delete
sp
perungalthur
9872738373
Update
Delete
28°C
Mostly sunny
Search
ENG
IN
09:05
19-03-2025

PROGRAM:

Exercise 7.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

App.css

```
.App {
  text-align: center;
}
.App-logo {
  height: 40vmin;
  pointer-events: none;
}
@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
```

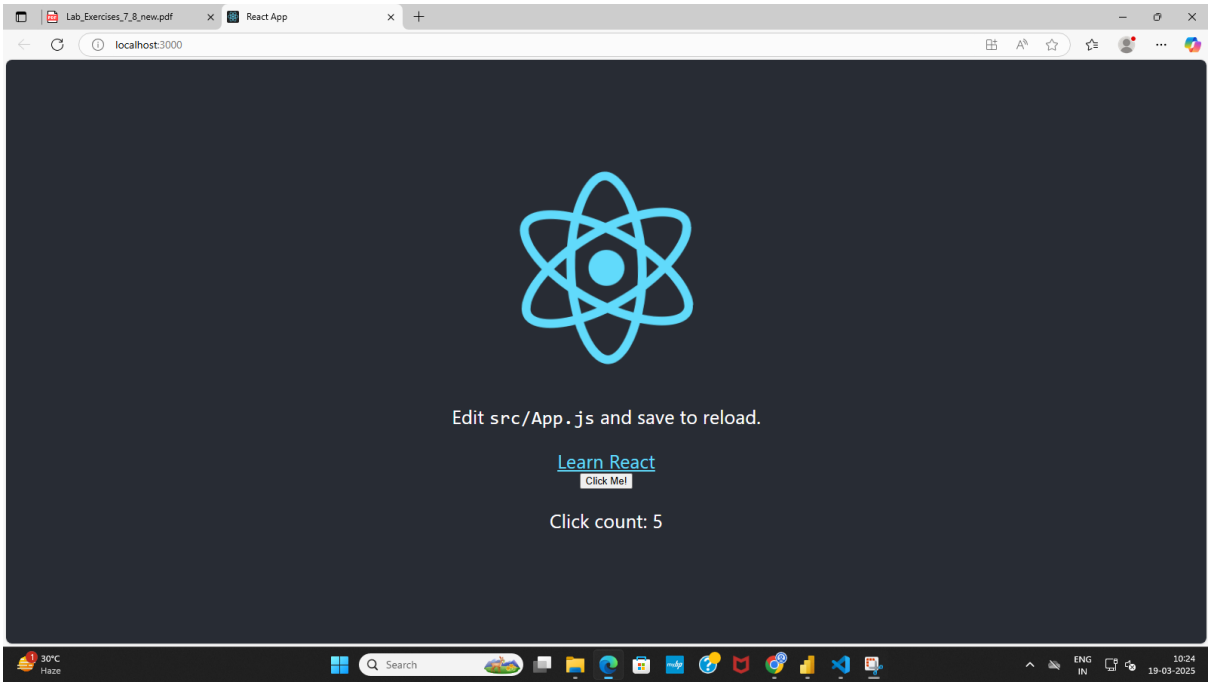
```
  }
}
.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}
.App-link {
  color: #61dafb;
}
@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```

App.js

```
import React, { useState } from 'react';  // Import useState from React
import logo from './logo.svg';
import './App.css';
function App() {
  const [count, setCount] = useState(0);
  const handleClick = () => {
    setCount(count + 1);  // Increase the count by 1 each time the button is clicked
  };
```

```
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
        {/* Button to trigger count update */}
        <button onClick={handleClick}>Click Me!</button>
        {/* Display the click count */}
        <p>Click count: {count}</p>
      </header>
    </div>
  );
}
export default App;
```

Output:


Lab_Exercises_7_8_new.pdf
React App
localhost:3000
Edit src/App.js and save to reload.
Learn React
Click Me!
Click count: 5

PROGRAM:

Exercise 8:

App.js

```
import React, { useState } from 'react';
import './App.css';
function App() {
  const [todos, setTodos] = useState([]);
  const [input, setInput] = useState('');
  const [editIndex, setEditIndex] = useState(null);
  const [editText, setEditText] = useState('');
  const handleInputChange = (e) => {
    setInput(e.target.value);
  };
  const addTodo = () => {
    if (input.trim() !== '') {
      setTodos([...todos, { text: input, completed: false }]);
      setInput('');
    }
  };
  const toggleComplete = (index) => {
    const updatedTodos = todos.map((todo, i) =>
      i === index ? { ...todo, completed: !todo.completed } : todo
    );
    setTodos(updatedTodos);
  };
  const deleteTodo = (index) => {
    const updatedTodos = todos.filter((_, i) => i !== index);
    setTodos(updatedTodos);
  };
  const editTodo = (index) => {
  setEditIndex(index);
    setEditText(todos[index].text);
  };
```

```
const saveEdit = () => {
  if (editText.trim() !== '') {
    const updatedTodos = todos.map((todo, i) =>
      i === editIndex ? { ...todo, text: editText } : todo
    );
    setTodos(updatedTodos);
    setEditIndex(null);
    setEditText('');
  }
};
return (
  <div className="App">
    <h1>To-Do App</h1>
    <div className="todo-input">
      <input
        type="text"
        value={input}
        onChange={handleInputChange}
        placeholder="Enter a new task"
      />
      <button className="add-btn" onClick={addTodo}>Add</button>
    </div>
    {/* Edit todo */}
    {editIndex !== null && (
      <div className="edit-todo">
        <input
          type="text"
          value={editText}
          onChange={(e) => setEditText(e.target.value)}
          placeholder="Edit your task"
        />
        <button className="save-btn" onClick={saveEdit}>Save</button>
      </div>
```

```
      )}
      <table className="todo-table">
        <thead>
          <tr>
            <th>Task</th>
            <th>Actions</th>
          </tr>
        </thead>
        <tbody>
          {todos.map((todo, index) => (
            <tr key={index}>
              <td className={todo.completed ? 'completed' : ''}>
                <span onClick={() => toggleComplete(index)}>{todo.text}</span>
              </td>
              <td>
                <button className="edit-btn" onClick={() => editTodo(index)}>Edit</button>
                <button className="delete-btn" onClick={() =>
deleteTodo(index)}>Delete</button>
              </td>
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  );
}
export default App;
```

App.css

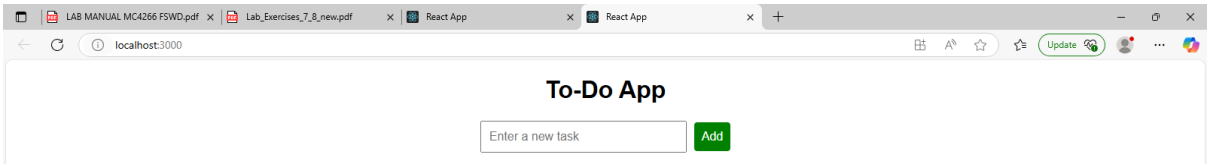
```
.App {
  text-align: center;
  margin-top: 20px;
  font-family: Arial, sans-serif;
```

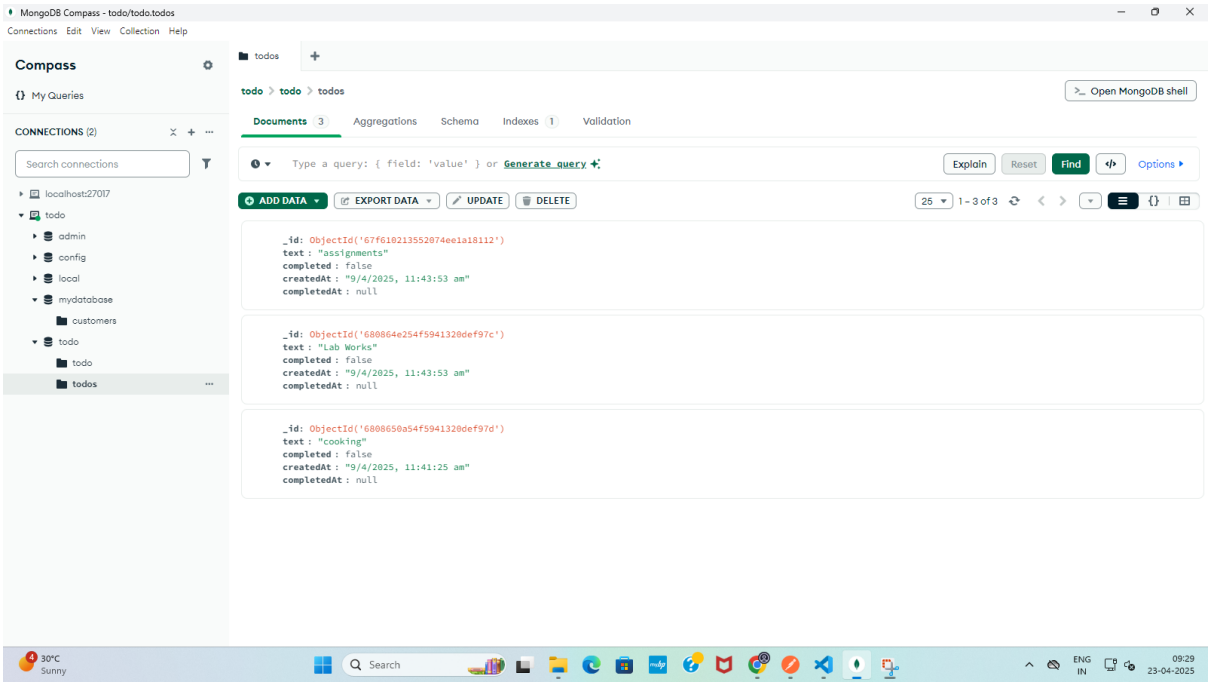
```
}
.todo-input {
  margin-bottom: 20px;
}
input {
  padding: 10px;
  font-size: 16px;
  width: 250px;
}
button {
  padding: 10px;
  font-size: 16px;
  cursor: pointer;
  margin-left: 10px;
  border: none;
  border-radius: 4px;
}
.add-btn {
  background-color: green;
  color: white;
}
.add-btn:hover {
  background-color: darkgreen;
}
.delete-btn {
  background-color: red;
  color: white;
}
.delete-btn:hover {
  background-color: darkred;
}
.edit-btn {
  background-color: orange;
```

```
  color: white;
}
.edit-btn:hover {
  background-color: darkorange;
}
.todo-table {
  width: 80%;
  margin: 0 auto;
  border-collapse: collapse;
}
th, td {
  padding: 10px;
  text-align: left;
  border: 1px solid #ddd;
}
th {
  background-color: #f2f2f2;
}
.completed {
  text-decoration: line-through;
  color: gray;
}
.edit-todo {
  margin-top: 20px;
}
.edit-todo input {
  padding: 10px;
  font-size: 16px;
  width: 250px;
}
.save-btn {
  background-color: blue;
  color: white;
```

```
  cursor: pointer;
  padding: 10px;
  margin-left: 10px;
}
.save-btn:hover {
  background-color: darkblue;
}
```

Output:


LAB MANUAL MC4266 FSWD.pdf
Lab_Exercises_7_8_new.pdf
React App
React App
localhost:3000
Update
To-Do App
Enter a new task
Add


MongoDB Compass - todo/todo.todos
Connections Edit View Collection Help
Compass
My Queries
CONNECTIONS (2)
Search connections
localhost:27017
todo
admin
config
local
mydatabase
customers
todo
todo
todos
todos
todo > todo > todos
Open MongoDB shell
Documents 3
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 3 of 3
_id: ObjectId('67f610213552074ee1a18112')
text : "assignments"
completed : false
createdAt : "9/4/2025, 11:43:53 am"
completedAt : null
_id: ObjectId('680864e254f5941320def97c')
text : "Lab Works"
completed : false
createdAt : "9/4/2025, 11:43:53 am"
completedAt : null
_id: ObjectId('6808650a54f5941320def97d')
text : "cooking"
completed : false
createdAt : "9/4/2025, 11:41:25 am"
completedAt : null
30°C
Sunny
Search
ENG
IN
09:29
23-04-2025

PROGRAM

Exercise 9

models/user.js

```
const mongoose = require('mongoose');

const UserSchema = new mongoose.Schema({
    username: { type: String, required: true, unique: true },
    password: { type: String, required: true }
});

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

routes/user.js

```
import express from 'express';
import { getAllUsers, login, logout, signUp } from "../controllers/user.js";
import { checkRole, checkToken } from '../middlewares/middlewares.js';
const router = express.Router();

router.post("/signUp", signUp);
router.post("/login", login);
router.post("/logout", checkToken, logout);
router.get('/getAllUsers', checkToken, checkRole(['admin', 'manager']), getAllUsers);

export default router;
```

Server.js

```
const express = require('express');
const connectDB = require('./config/db');
const cookieParser = require('cookie-parser');
const authRoutes = require('./routes/auth');
require('dotenv').config();

const app = express();
const PORT = process.env.PORT || 2222;

app.use(express.json());
app.use(cookieParser());
```

```
app.use('/api/auth', authRoutes);

connectDB();
app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
```

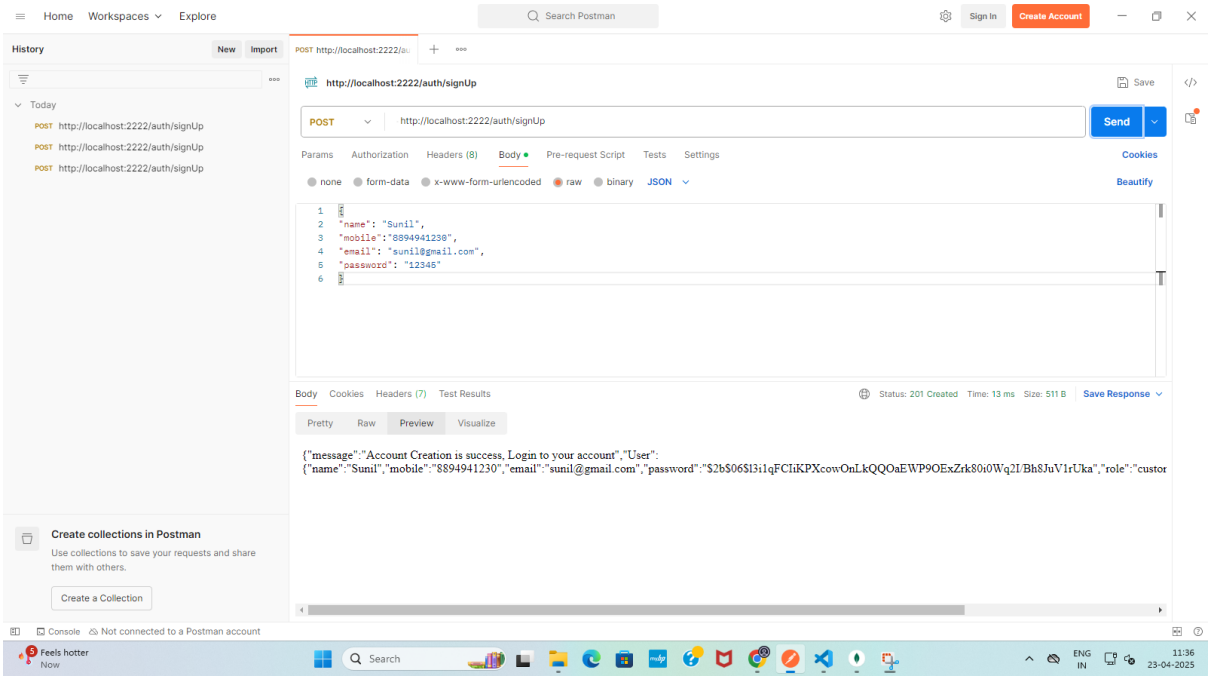
controller/user.js

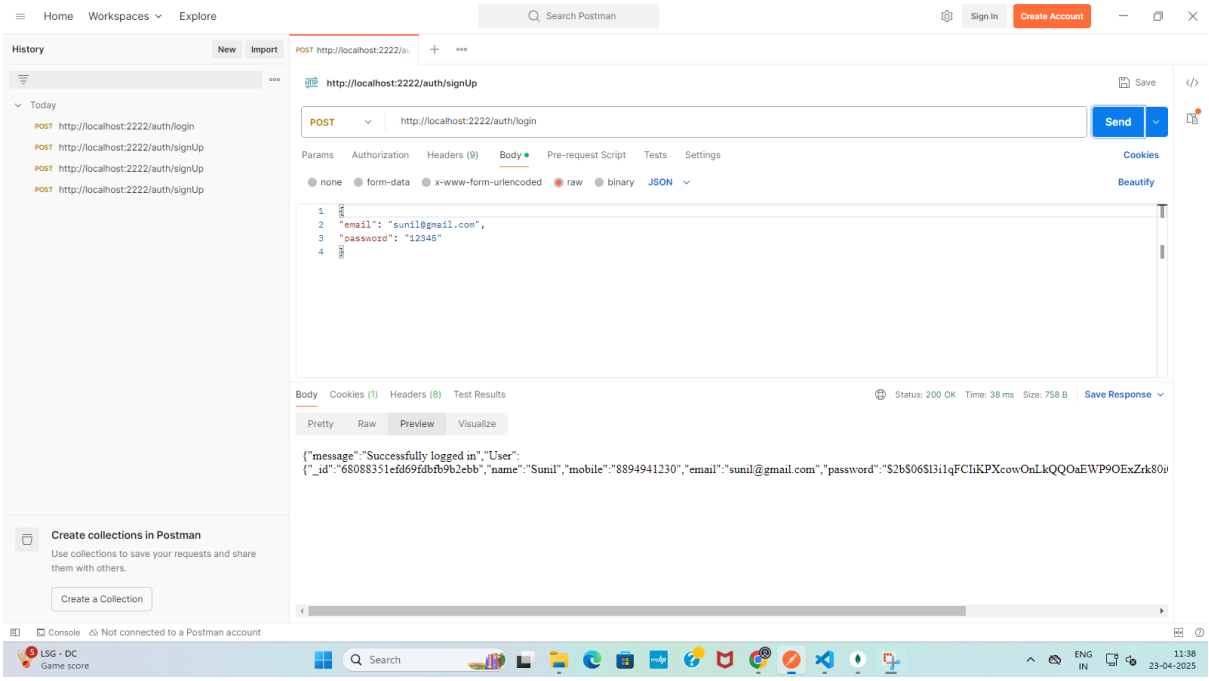
```
import bcrypt from 'bcrypt';
import User from "../models/user.js";
import { CreateToken } from '../middlewares/middlewares.js';
import jsonwebtoken from 'jsonwebtoken';

export const signUp = async (req, res) => {

  const { name, mobile, email, password, role } = req.body;
  if (!name || !mobile || !email || !password) {
    return res.status(422).json({ message: "All feilds should be filled" })
  }
  try {
    let existingUser;
    try {
      existingUser = await User.findOne({ $or: [{ email: email }, { mobile: mobile }] });
    } catch (err) {
      console.error(err);
    }

    if (existingUser) {
      if (existingUser.email == email) {
        return res.status(409).json({ message: "A User is already signUp with this email" })
      }
      else if (existingUser.mobile == mobile) {
        return res.status(409).json({ message: "A User is already signUp with this mobile" })
      }
    }

    const salt = await bcrypt.genSalt(6)
    const hashedpassword = await bcrypt.hash(password, salt);

    const user = new User({
      name,
      mobile,
```

```
      email,
      password: hashedpassword,
      role: role,
    });

    await user.save();
    return res.status(201).json({ message: "Account Creation is success, Login to your account", User: user
})

  } catch (err) {
    console.error(err)
    return res.status(400).json({ message: "Error in saving user in DB" });
  }

}

export const login = async (req, res) => {

  const { email, password } = req.body;

  if (!email || !password) {
    return res.status(422).json({ message: "All feilds should be filled" })
  }

  let loggedUser;

  try {
    loggedUser = await User.findOne({ email: email });

    if (!loggedUser) {
      return res.status(404).json({ message: "Email is not found, Check it and try again" })
    }
    const isPasswordCorrect = bcrypt.compareSync(password, loggedUser.password);
    if (!isPasswordCorrect) {
      return res.status(400).json({ message: "Invalid password, Check it and try again" })
    }
    const token = CreateToken(loggedUser._id);
      res.cookie(String(loggedUser._id), token, {
      path: "/",
      expires: new Date(Date.now() + 1000 * 59),
      httpOnly: true      sameSite: "lax"
    })
```

```
    return res.status(200).json({ message: "Successfully logged in", User: loggedUser })
  } catch (err) {
    console.log(err)
  }
}

export const logout = (req, res) => {
  const cookies = req.headers.cookie
  const previousToken = cookies.split("=")[1];

   if (!previousToken) {
    return res.status(400).json({ message: "Couldn't find token" });
  }
  jsonwebtoken.verify(String(previousToken), process.env.JWTAUTHSECRET, (err, user) => {
    if (err) {
      console.log(err);
      return res.status(403).json({ message: "Authentication failed" });    }
    res.clearCookie(`${user.id}`);
    req.cookies[`${user.id}`] = "";
    return res.status(200).json({ message: "Successfully Logged Out" });
  });
};

export const getAllUsers = async (req, res) => {
  try {
    const allusers = await User.find();
    if (!allusers) {
      return res.status(404).json({ message: "There are not any users" });
    }
    else {
      res.status(200).json({ allusers })
    }
  } catch (error) {
    console.log(error);
    return res.status(500).json({ message: "Error in getting the Users" })
  }
}
```

Output


Home
Workspaces
Explore
Search Postman
Sign In
Create Account
History
New
Import
Today
POST http://localhost:2222/auth/signUp
POST http://localhost:2222/auth/signUp
POST http://localhost:2222/auth/signUp
http://localhost:2222/auth/signUp
Save
POST
http://localhost:2222/auth/signUp
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
{
"name": "Sunil",
"mobile":"8894941230",
"email": "sunil@gmail.com",
"password": "12345"
}
Body
Cookies
Headers (7)
Test Results
Status: 201 Created
Time: 13 ms
Size: 511 B
Save Response
Pretty
Raw
Preview
Visualize
{"message":"Account Creation is success, Login to your account","User":
{"name":"Sunil","mobile":"8894941230","email":"sunil@gmail.com","password":"$2b$06$l3i1qFCIiKPXcowOnLkQQOaEWP9OExZrk80i0Wq2I/Bh8JuV1rUka","role":"custor
Create collections in Postman
Use collections to save your requests and share them with others.
Create a Collection
Console
Not connected to a Postman account
Feels hotter
Now
Search
ENG
IN
11:36
23-04-2025


Home
Workspaces
Explore
Search Postman
Sign In
Create Account
History
New
Import
Today
POST http://localhost:2222/auth/login
POST http://localhost:2222/auth/signUp
POST http://localhost:2222/auth/signUp
POST http://localhost:2222/auth/signUp
http://localhost:2222/auth/signUp
Save
POST
http://localhost:2222/auth/login
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
{
"email": "sunil@gmail.com",
"password": "12345"
}
Body
Cookies (1)
Headers (8)
Test Results
Status: 200 OK
Time: 38 ms
Size: 758 B
Save Response
Pretty
Raw
Preview
Visualize
{"message":"Successfully logged in","User":
{"_id":"68088351efd69fdbfb9b2ebb","name":"Sunil","mobile":"8894941230","email":"sunil@gmail.com","password":"$2b$06$l3i1qFCIiKPXcowOnLkQQOaEWP9OExZrk80i
Create collections in Postman
Use collections to save your requests and share them with others.
Create a Collection
Console
Not connected to a Postman account
LSG - DC
Game score
Search
ENG
IN
11:38
23-04-2025

PROGRAM

Ex10:

Step 1: Prepare your computer for Virtualization:

• Enable Processor Virtualization: Ensure Virtualization is enabled on your computer. See the Virtualization Error (VT-d/VT-x or AMD-V) for troubleshooting support.

• Review File Sync Services for tools like OneDrive, Nextcloud, DropBox Sync, iCloud, etc. If you are using a data synchronization service, make sure it DOES NOT (or at least not frequently) synchronize the folder in which your hypervisor imports and installs the Virtual Machines.

• File sync services can cause a dramatic fall-off in performance for your entire system as these services try to synchronize these massive files that are getting updated constantly while you are using the Virtual Machines.

• Sufficient Disk Space: Virtual Machines require a significant amount of Disk space (10 GB or more each is typical). Ensure you have sufficient space on your computer.

• Admin Privileges: Installing a hypervisor on a host in most cases requires admin privileges.

Step 2: Install Hypervisor (Virtualization Tool):

Installing a hypervisor on your host is usually quite simple. In most cases, the install program will ask only a couple of questions, such as where to install the hypervisor software.

Step 3: Import a Virtual Machine:

• The first step is to download the Virtual Machine for your course from our Course Virtual Machines page. This will download an .ova file. The .ova file is actually a compressed (zipped) tarball of a Virtual Machine exported from Virtual Box.

• Once the Virtual Machine has been imported, it will normally show up in the guest list within your hypervisor tool.

Step 4: Start the Virtual Machine:

To start up a Virtual Machine guest in most hypervisors, you simply click on the desired guest and click the Start button (often double-clicking the guest icon will work as well).

Step 5: Using the Virtual Machine:

• Sharing files between the guest and host: To learn about different ways of sharing files, check out this guide.

• Run a command with sudo (root) privileges: Open a terminal and type any command with sudo in front to run that command as root.

• Example: sudo apt-get install vim – will install the vim text editor package on an Ubuntu Linux Virtual Machine.

• Find the IP address of your guest: Open a terminal and type ifconfig | more – The | more (pronounced “pipe more”) will “pipe” the output of the ifconfig command to the more command, which will show the results one page at a time, so it doesn’t scroll by before you see it all.

• If you have a Host-Only Network IP address, you will see an IP of 192.168.56.101 (or something similar). Check the Trouble-Shooting section below for more information about the Host-Only Network.

Step 6: Shut down the Virtual Machine:

When you are done using a guest Virtual Machine, regardless of hypervisor, you need to shut it down properly. This can be done in three ways:

1. Press the shutdown button found on the desktop, taskbar, or task menu of the guest operating system.

2. Open a terminal and type the command: sudo shutdown -h now

3. In the guest window, click Machine (menu) -> ACPI Shut down – This will simulate the power button being pressed

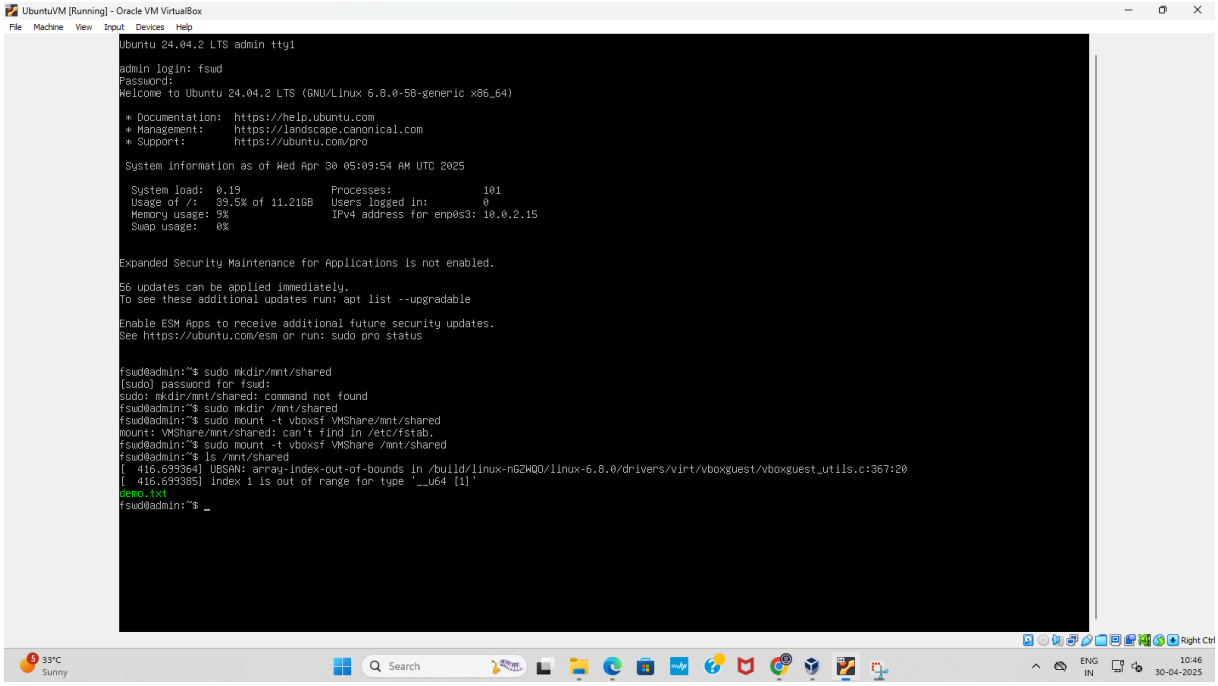
Output:

Bash

ssh [USERNAME]@[IP_ADDRESS]

Enter Password: Enter the password for the specified user.

SSH Connection: A screenshot of the terminal window on the host computer, displaying a successful SSH connection to the guest OS.



PROGRAM

Ex11_ server.js:

```
const http = require('http');
const hostname = '0.0.0.0';
const port = 8080;
const server = http.createServer((req, res) => {
  if (req.method === 'GET' && req.url === '/ping') {
    res.statusCode = 200;
    res.setHeader('Content-Type', 'application/json');
    res.end(JSON.stringify({ message: 'pong' }));
  } else {
    res.statusCode = 404;
    res.end('Not Found');
  }
});
server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Dockerfile:

```
# Use official Node.js image from Docker Hub
FROM node:16
# Set the working directory inside the container
WORKDIR /usr/src/app
# Copy the server.js file to the working directory
COPY server.js .
# Expose the port the app will run on
EXPOSE 8080
# Run the Node.js server
CMD ["node", "server.js"]
```

Output:

After running the server, you should be able to access the endpoint via http://localhost:8080/ping, and it should respond with the following JSON message:

json

{

"message": "pong"

}