

**Name:** Devanshu Surana

**PRN:** 1032210755

**Roll No.:** 23

**Panel:** C, **Batch:** C1

## **FSD Lab 07**

**Aim:** Develop a full stack web application using MERN stack to perform CRUD operations.

### **Objectives:**

1. To develop full-stack web projects using the MERN stack.
2. To learn database connectivity using fetch api.
3. To perform insert, update, delete and search operations on database.

### **Theory:**

#### **1. What is MERN stack?**

**Definition:** MERN stack is a popular acronym in web development that stands for MongoDB, Express.js, React.js, and Node.js. It represents a full-stack JavaScript framework used for building dynamic and robust web applications. Each component of the MERN stack plays a specific role in the development process:

**MongoDB:** A NoSQL database that stores data in a JSON-like format.

**Express.js:** A web application framework for Node.js that simplifies the creation of server-side applications.

**React.js:** A JavaScript library for building user interfaces, particularly for single-page applications where data can change without reloading the page.

**Node.js:** A JavaScript runtime environment that executes server-side code.

## 2. Use of Fetch API:

**Definition:** The Fetch API is a modern web API for making HTTP requests and handling responses. It provides a more flexible and powerful alternative to the older XMLHttpRequest. The Fetch API is native to modern browsers and is used in conjunction with JavaScript Promises.

**Usage:** The Fetch API is used to make asynchronous requests to a server. It returns a Promise that resolves to the Response to that request, whether it is successful or not. The basic syntax looks like this:

```
fetch(url)
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```

## FAQ:

### 1. What makes MERN stack the fastest growing tech stack?

**ANS: JavaScript Throughout:** One key factor is the use of JavaScript across the entire stack (both client and server). Developers can use the same language for both front-end and back-end development, streamlining the development process and reducing context switching.

**Component Reusability:** React.js, a part of the MERN stack, promotes the concept of reusable components. This leads to more modular and maintainable code, enabling developers to efficiently build and scale applications.

**Isomorphic/Universal JavaScript:** MERN allows for the use of JavaScript on both the client and server sides. This enables server-side rendering and improves performance, especially in terms of initial page load times.

**Active Community and Ecosystem:** The MERN stack has a large and active developer community. There are numerous libraries, tools, and resources available, making it easier for developers to find solutions and best practices.

**Scalability and Flexibility:** MongoDB, a NoSQL database in the stack, provides flexibility in handling different types of data. Node.js, with its

non-blocking I/O, contributes to the scalability of applications, making them capable of handling a large number of simultaneous connections.

**Rapid Development:** The combination of a powerful front-end library (React.js) and a fast, event-driven server (Node.js) allows for rapid development of modern, dynamic web applications.

## Output Screenshots:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.22631.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\anujm>cd C:\Users\anujm\OneDrive\Desktop

C:\Users\anujm\OneDrive\Desktop>cd FLIGHT

C:\Users\anujm\OneDrive\Desktop\FLIGHT>npm init -y
Wrote to C:\Users\anujm\OneDrive\Desktop\FLIGHT\package.json:

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

C:\Users\anujm\OneDrive\Desktop\FLIGHT>npm install express mongoose body-parser cors

added 88 packages, and audited 89 packages in 12s

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

found 0 vulnerabilities

C:\Users\anujm\OneDrive\Desktop\FLIGHT>cd C:\Users\anujm\OneDrive\Desktop\FLIGHT
```

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.22631.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\anujm>cd C:\Users\anujm\OneDrive\Desktop\FLIGHT

C:\Users\anujm\OneDrive\Desktop\FLIGHT>npx create-react-app client

Creating a new React app in C:\Users\anujm\OneDrive\Desktop\FLIGHT\client.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...
```

```
C:\Users\anujm\OneDrive\Desktop\FLIGHT\client>cd C:\Users\anujm\OneDrive\Desktop\FLIGHT

C:\Users\anujm\OneDrive\Desktop\FLIGHT>node server.js
(node:12464) [DEP0040] DeprecationWarning: The 'punycode' module is deprecated. Please use a userland alternative instead.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:12464) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4
(node:12464) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver ver
Server is running on port 5000
^C
C:\Users\anujm\OneDrive\Desktop\FLIGHT>
```

```

C:\WINDOWS\system32\cmd.exe - node server.js
Microsoft Windows [Version 10.0.22631.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\anujm>cd C:\Users\anujm\OneDrive\Desktop\FLIGHT

C:\Users\anujm\OneDrive\Desktop\FLIGHT>node server.js
(node:22904) [DEP0040] DeprecationWarning: The 'punycode' module is deprecated. Please use a userland alternative instead.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:22904) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0
(node:22904) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4
Server is running on port 5000

```

```

Windows PowerShell
Microsoft Windows [Version 10.0.22631.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\anujm>cd C:\Users\anujm\OneDrive\Desktop\FLIGHT\client

C:\Users\anujm\OneDrive\Desktop\FLIGHT\client>npm start

> client@0.1.0 start
> react-scripts start

```

## server.js

```

JS server.js  X  JS App.js  # App.css

JS server.js > app.post('/passengers/add') callback
1  const express = require('express');
2  const mongoose = require('mongoose');
3  const bodyParser = require('body-parser');
4  const cors = require('cors');
5
6  const app = express();
7  const PORT = process.env.PORT || 5000;
8
9  app.use(cors());
10 app.use(bodyParser.json());
11
12 // MongoDB connection setup
13 mongoose.connect('mongodb://localhost:27017/flightDB', {
14   useNewUrlParser: true,
15   useUnifiedTopology: true,
16 });
17
18 const flightBookingSchema = new mongoose.Schema({
19   passengerName: String,
20   from: String,
21   to: String,
22   date: String,
23   departureDate: String,
24   arrivalDate: String,
25   phoneNumber: String,
26   email: String,
27 });
28
29 const FlightBooking = mongoose.model('FlightBooking', flightBookingSchema);
30
31 // Routes for CRUD operations
32
33 // Insert Passenger details
34 app.post('/passengers/add', async (req, res) => {
35   try {
36     const newPassenger = new FlightBooking(req.body);
37     await newPassenger.save();
38     res.status(201).json(newPassenger);
39   } catch (error) {}
40   res.status(500).json({ error: error.message });
41 }

```

JS server.js X JS App.js # App.css

JS server.js > app.post('/passengers/add') callback

```
43
44 // Get all passengers
45 app.get('/passengers', async (req, res) => {
46   try {
47     const passengers = await FlightBooking.find();
48     res.status(200).json(passengers);
49   } catch (error) {
50     res.status(500).json({ error: error.message });
51   }
52 });
53
54 // Delete passenger by phone number
55 app.delete('/passengers/delete/:phoneNumber', async (req, res) => {
56   try {
57     const deletedPassenger = await FlightBooking.findOneAndDelete({
58       phoneNumber: req.params.phoneNumber,
59     });
60     res.status(200).json(deletedPassenger);
61   } catch (error) {
62     res.status(500).json({ error: error.message });
63   }
64 });
65
66 // Update passenger details by phone number
67 app.post('/passengers/update/:phoneNumber', async (req, res) => {
68   try {
69     const updatedPassenger = await FlightBooking.findOneAndUpdate(
70       { phoneNumber: req.params.phoneNumber },
71       req.body,
72       { new: true }
73     );
74     res.status(200).json(updatedPassenger);
75   } catch (error) {
76     res.status(500).json({ error: error.message });
77   }
78 });
79
```

```
80 // Get passenger by phone number
81 app.get('/passengers/:phoneNumber', async (req, res) => {
82   try {
83     const passenger = await FlightBooking.findOne({
84       phoneNumber: req.params.phoneNumber,
85     });
86     res.status(200).json(passenger);
87   } catch (error) {
88     res.status(500).json({ error: error.message });
89   }
90 });
91
92 app.listen(PORT, () => {
93   console.log(`Server is running on port ${PORT}`);
94 });
95
```

# App.js

```
JS server.js JS App.js # App.css
client > src > JS App.js > ...
1 import React, { useState, useEffect } from 'react';
2 import axios from 'axios';
3 import './App.css';
4
5 function App() {
6   const [passengerDetails, setPassengerDetails] = useState({
7     passengerName: '',
8     from: '',
9     to: '',
10    date: '',
11    departureDate: '',
12    arrivalDate: '',
13    phoneNumber: '',
14    email: '',
15  });
16
17  const [passengerList, setPassengerList] = useState([]);
18  const [updatePhone, setUpdatePhone] = useState('');
19  const [updatedDetails, setUpdatedDetails] = useState({});
20  const [isUpdateMode, setIsUpdateMode] = useState(false);
21
22  useEffect(() => {
23    // Fetch all passengers on component mount
24    axios.get('http://localhost:5000/passengers')
25      .then((response) => {
26        setPassengerList(response.data);
27      });
28  }, []);
29
30  const handleInputChange = (e) => {
31    const { name, value } = e.target;
32    setPassengerDetails({
33      ...passengerDetails,
34      [name]: value,
35    });
36  };
37}
```

```
JS server.js JS App.js # App.css
client > src > JS App.js > ...
37
38 const handleSubmit = (e) => {
39   e.preventDefault();
40
41   const endpoint = isUpdateMode
42     ? `http://localhost:5000/passengers/update/${updatePhone}`
43     : 'http://localhost:5000/passengers/add';
44
45   axios.post(endpoint, passengerDetails)
46     .then(() => {
47       setIsUpdateMode(false);
48       setUpdatedDetails({});
49       setUpdatePhone('');
50
51       // Refresh passenger list
52       axios.get('http://localhost:5000/passengers')
53         .then((response) => {
54           setPassengerList(response.data);
55         });
56
57       // Clear form fields
58       setPassengerDetails({
59         passengerName: '',
60         from: '',
61         to: '',
62         date: '',
63         departureDate: '',
64         arrivalDate: '',
65         phoneNumber: '',
66         email: '',
67       });
68     })
69     .catch((error) => {
70       console.error("Error submitting passenger details:", error);
71     });
72 };
```

```

JS server.js JS App.js # App.css
client > src > JS App.js > ...
74 const handleDelete = (phoneNumber) => {
75   // Delete passenger by phone number
76   axios.delete(`http://localhost:5000/passengers/delete/${phoneNumber}`)
77     .then(() => {
78       // Refresh passenger list
79       axios.get('http://localhost:5000/passengers')
80         .then((response) => {
81           setPassengerList(response.data);
82         });
83     });
84 };
85
86 const handleUpdateSearch = () => {
87   console.log("Searching for phone number:", updatePhone);
88
89   // Search for passenger by phone number
90   axios.get(`http://localhost:5000/passengers/${updatePhone}`)
91     .then((response) => {
92       setUpdatedDetails(response.data);
93       setPassengerDetails(response.data); // Set the form fields with the retrieved data
94       setIsUpdateMode(true);
95     })
96     .catch((error) => {
97       console.error("Error fetching passenger details for update:", error);
98       setIsUpdateMode(false);
99     });
100 };
101

```

```

JS server.js JS App.js # App.css
client > src > JS App.js > App
100 };
101
102 return (
103   <div className="App">
104     <h1>Flight Booking Management System</h1>
105     { /* Flight Booking form */ }
106     <form onSubmit={handleSubmit}>
107       <div className="form-group">
108         <label>Passenger Name:</label>
109         <input
110           type="text"
111           name="passengerName"
112           value={passengerDetails.passengerName}
113           onChange={handleInputChange}
114           required
115         />
116       </div>
117       <div className="form-group">
118         <label>From:</label>
119         <input
120           type="text"
121           name="from"
122           value={passengerDetails.from}
123           onChange={handleInputChange}
124           required
125         />
126       </div>
127       <div className="form-group">
128         <label>To:</label>
129         <input
130           type="text"
131           name="to"
132           value={passengerDetails.to}
133           onChange={handleInputChange}
134           required
135         />
136       </div>
137     </form>
138   </div>
139 );
140

```

```

JS server.js JS App.js # App.css
client > src > JS App.js > App
141 <div className="form-group">
142   <label>Date:</label>
143   <input
144     type="text"
145     name="date"
146     value={passengerDetails.date}
147     onChange={handleInputChange}
148     required
149   />
150 </div>
151
152 <div className="form-group">
153   <label>Departure Date:</label>
154   <input
155     type="text"
156     name="departureDate"
157     value={passengerDetails.departureDate}
158     onChange={handleInputChange}
159     required
160   />
161 </div>
162
163 <div className="form-group">
164   <label>Arrival Date:</label>
165   <input
166     type="text"
167     name="arrivalDate"
168     value={passengerDetails.arrivalDate}
169     onChange={handleInputChange}
170     required
171   />
172 </div>
173
174 <div className="form-group">
175   <label>Phone Number:</label>
176   <input
177     type="text"
178     name="phoneNumber"
179     value={passengerDetails.phoneNumber}
180     onChange={handleInputChange}
181     required

```

```
JS server.js JS App.js # App.css
client > src > JS App.js > App
141     <div className="form-group">
142       <label>Date:</label>
143       <input
144         type="text"
145         name="date"
146         value={passengerDetails.date}
147         onChange={handleInputChange}
148         required
149       />
150     </div>
151
152     <div className="form-group">
153       <label>Departure Date:</label>
154       <input
155         type="text"
156         name="departureDate"
157         value={passengerDetails.departureDate}
158         onChange={handleInputChange}
159         required
160       />
161     </div>
162
163     <div className="form-group">
164       <label>Arrival Date:</label>
165       <input
166         type="text"
167         name="arrivalDate"
168         value={passengerDetails.arrivalDate}
169         onChange={handleInputChange}
170         required
171       />
172     </div>
173
174     <div className="form-group">
175       <label>Phone Number:</label>
176       <input
177         type="text"
178         name="phoneNumber"
179         value={passengerDetails.phoneNumber}
180         onChange={handleInputChange}
181         required
```

```
JS server.js JS App.js # App.css
client > src > JS App.js > App
181       required
182     />
183   </div>
184
185   <div className="form-group">
186     <label>Email:</label>
187     <input
188       type="email"
189       name="email"
190       value={passengerDetails.email}
191       onChange={handleInputChange}
192       required
193     />
194   </div>
195
196   <button type="submit">{isUpdateMode ? 'Update Passenger' : 'Add Passenger'}</button>
197 </form>
198
```



```

JS server.js  JS App.js  # App.css
client > src > JS App.js > App
196     <button type="submit">{isUpdateMode ? 'Update Passenger' : 'Add Passenger'}</button>
197   </form>
198
199   {/* Display Flight Booking details in a tabular format */}
200   <table>
201     <thead>
202       <tr>
203         <th>Passenger Name</th>
204         <th>From</th>
205         <th>To</th>
206         <th>Date</th>
207         <th>Departure Date</th>
208         <th>Arrival Date</th>
209         <th>Phone Number</th>
210         <th>Email</th>
211         <th>Action</th>
212       </tr>
213     </thead>
214     <tbody>
215       {passengerList.map((passenger) => (
216         <tr key={passenger.phoneNumber}>
217           <td>{passenger.passengerName}</td>
218           <td>{passenger.from}</td>
219           <td>{passenger.to}</td>
220           <td>{passenger.date}</td>
221           <td>{passenger.departureDate}</td>
222           <td>{passenger.arrivalDate}</td>
223           <td>{passenger.phoneNumber}</td>
224           <td>{passenger.email}</td>
225           <td>
226             <button onClick={() => handleUpdateSearch(passenger.phoneNumber)}>Update</button>
227             <button onClick={() => handleDelete(passenger.phoneNumber)}>Delete</button>
228           </td>
229         </tr>
230       )]}
231     </tbody>
232   </table>
233
234   {/* Update Passenger form */}
235   {isUpdateMode && (
236     <div>

```

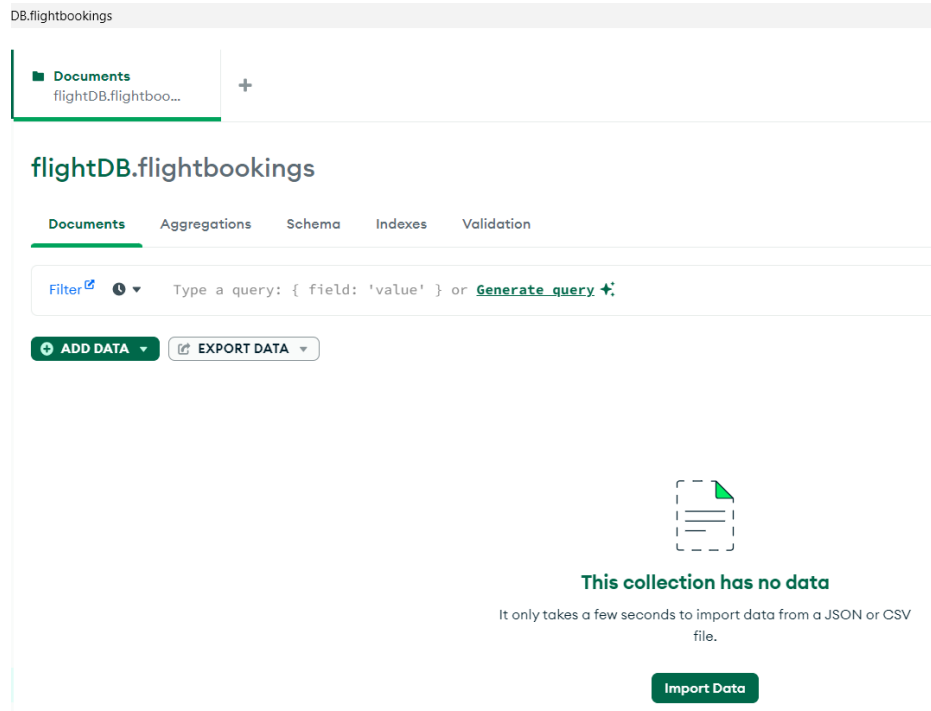
```

234   {/* Update Passenger form */}
235   {isUpdateMode && (
236     <div>
237       <h2>Update Passenger Details</h2>
238       <input
239         type="text"
240         placeholder="Enter Phone Number"
241         value={updatePhone}
242         onChange={(e) => setUpdatePhone(e.target.value)}
243       />
244       <button onClick={handleUpdateSearch}>Search</button>
245
246       {Object.keys(updatedDetails).length > 0 && (
247         <div>
248           {/* Your update form fields go here */}
249         </div>
250       )}
251     </div>
252   )}
253 </div>
254 );
255 }
256
257 export default App;
258

```

# App.css

```
JS server.js JS App.js # App.css X
client > src > # App.css > .form-group
1  .form-group {
2    margin-bottom: 6px;
3  }
4
5  label {
6    display: block;
7    margin-bottom: 2px;
8  }
9
10 input {
11   width: calc(50% - 8px);
12   padding: 4px;
13   box-sizing: border-box;
14   margin-bottom: 4px;
15 }
16
17 table {
18   width: 100%;
19   border-collapse: collapse;
20   margin-top: 12px;
21 }
22
23 th, td {
24   border: 1px solid #ddd;
25   padding: 4px;
26   text-align: left;
27 }
28
29 th {
30   background-color: #f2f2f2;
31 }
32
33 button {
34   margin-top: 4px;
35   padding: 4px;
36   cursor: pointer;
37 }
38
```



## Insert:

React App

http://localhost:3000

### Flight Booking Management System

Passenger Name:

From:

To:

Date:

Departure Date:

Arrival Date:

Phone Number:

Email:

Add Passenger

ADD DATAEXPORT DATA

1 - 1 of 1

#	flightbookings	_id ObjectId	passengerName String	from String	to String	date String	departureDate Stri	
1		ObjectId("655ee9d19cd6c0a2238...	"Abhilash Kashid"	"Pune"	"Mumbai"	"23/11/23"	"24/11/23"	

Update:

Flight Booking Management System

Passenger Name:

Abhilash Kashid

From:

Delhi

To:

Bangalore

Date:

23/11/23

Departure Date:

24/11/23

Arrival Date:

25/11/23

Phone Number:

9875689756

Email:

abhilaskashid456@gmail.com

Update Passenger

Add Passenger

Passenger Name	From	To	Date	Departure Date	Arrival Date	Phone Number	Email
Abhilash Kashid	Delhi	Bangalore	23/11/23	24/11/23	25/11/23	9875689756	abhilaskashid456@gmail.com

Updated:

flightDB.flightbookings

11DOCUMENTSINDEXES

DocumentsAggregationsSchemaIndexesValidation

FilterType a query: { field: 'value' } or Generate query

ExplainResetFindOptions

ADD DATAEXPORT DATA

1 - 1 of 1

#	flightbookings	_id ObjectId	passengerName String	from String	to String	date String	departureDate Stri	
1		ObjectId("655ee9d19cd6c0a2238...	"Abhilash Kashid"	"Delhi"	"Bangalore"	"23/11/23"	"24/11/23"	

Deleted:

Documents

flightDB.flightboo...

+

## flightDB.flightbookings

0

DOCUMENTS

Documents

Aggregations

Schema

Indexes

Validation

Filter

Type a query: { field: 'value' } or [Generate query](#)

Explain

Reset

Find

C

ADD DATA

EXPORT DATA

0 - 0 of 0



### This collection has no data

It only takes a few seconds to import data from a JSON or CSV file.

Import Data