

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Calculator</title>
7      <style>
8          body { display: flex; flex-direction: column; padding: 20px; gap: 20px; }
9          #calc { display: grid; grid-template-columns: repeat(4, 50px); gap: 20px; }
10         button { font-size: 20px; text-align: center; padding: 10px; }
11         #display { font-size: 20px; text-align: right; padding: 10px; max-width: fit-content;
12     }
13     </style>
14 </head>
15 <body>
16     <input type="text" id="display" readonly>
17     <div id="calc">
18         <button onclick="clearDisplay()">C</button>
19         <button onclick="appendToDisplay('7')">7</button>
20         <button onclick="appendToDisplay('8')">8</button>
21         <button onclick="appendToDisplay('9')">9</button>
22         <button onclick="appendToDisplay('/')">/</button>
23         <button onclick="appendToDisplay('4')">4</button>
24         <button onclick="appendToDisplay('5')">5</button>
25         <button onclick="appendToDisplay('6')">6</button>
26         <button onclick="appendToDisplay('*')">*</button>
27         <button onclick="appendToDisplay('1')">1</button>
28         <button onclick="appendToDisplay('2')">2</button>
29         <button onclick="appendToDisplay('3')">3</button>
30         <button onclick="appendToDisplay('-')">-</button>
31         <button onclick="appendToDisplay('0')">0</button>
32         <button onclick="appendToDisplay('.')">.</button>
33         <button onclick="calculate()">=</button>
34         <button onclick="appendToDisplay('+')">+</button>
35     </div>
36     <script src="calculator.js"></script>
37 </body>
38 </html>
```



127.0.0.1:5500/calculator/calculator.html

24\*24

C

7

8

9

/

4

5

6

\*

1

2

3

-

0

.

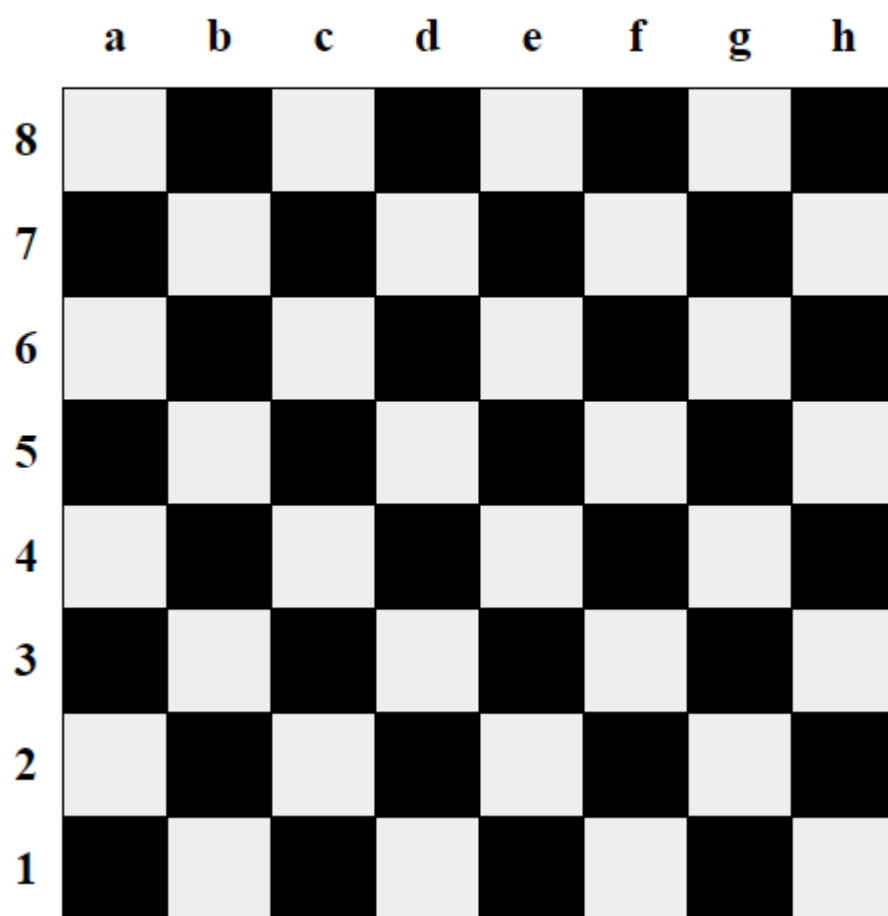
=

+

## calculator\calculator.js

```
1  function appendToDisplay(value) {  
2      document.getElementById("display").value += value;  
3  }  
4  
5  function clearDisplay() {  
6      document.getElementById("display").value = "";  
7  }  
8  
9  function calculate(){  
10     let result = eval(document.getElementById("display").value);  
11     document.getElementById("display").value = result;  
12 }  
13
```

127.0.0.1:5500/chessboard/chessboard.html



## chessboard\chessboard.html

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title></title>
5          <meta charset="UTF-8">
6          <style>
7              .chess-board { border-spacing: 0; border-collapse: collapse; }
8              .chess-board th { padding: .5em; }
9              .chess-board td { border: 1px solid; width: 2em; height: 2em; }
10             .chess-board .light { background: #eee; }
11             .chess-board .dark { background: #000; }
12         </style>
13     </head>
14     <body>
15         <table class="chess-board">
16             <tbody>
17                 <tr>
18                     <th></th>
19                     <th>a</th>
20                     <th>b</th>
21                     <th>c</th>
22                     <th>d</th>
23                     <th>e</th>
24                     <th>f</th>
25                     <th>g</th>
26                     <th>h</th>
27                 </tr>
28                 <tr>
29                     <th>8</th>
30                     <td class="light"></td>
31                     <td class="dark"></td>
32                     <td class="light"></td>
33                     <td class="dark"></td>
34                     <td class="light"></td>
35                     <td class="dark"></td>
36                     <td class="light"></td>
37                     <td class="dark"></td>
38                 </tr>
39                 <tr>
40                     <th>7</th>
41                     <td class="dark"></td>
42                     <td class="light"></td>
43                     <td class="dark"></td>
44                     <td class="light"></td>
45                     <td class="dark"></td>
46                     <td class="light"></td>
47                     <td class="dark"></td>
48                     <td class="light"></td>
49                 </tr>
50             </tbody>
51         </table>
52     </body>
```

127.0.0.1:5500/client\_pages/register.html

Register

Username:

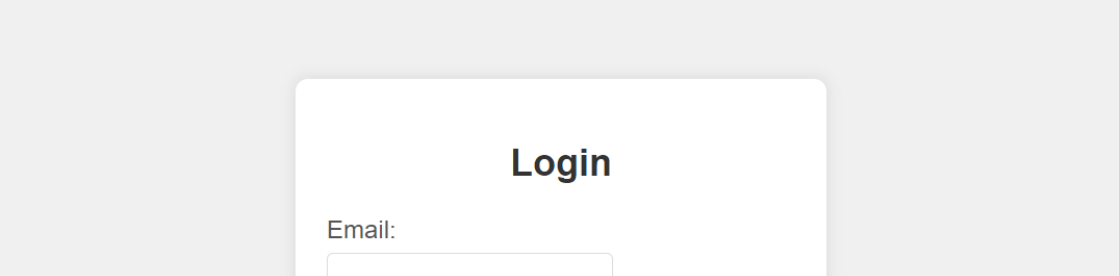
Email:

Password:

Register

## client\_pages\register.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <link rel="stylesheet" href="styles.css">
7   <title>Register</title>
8 </head>
9 <body>
10   <div class="form-container">
11     <h2>Register</h2>
12     <form onSubmit="validate()">
13       <div class="form-group">
14         <label for="username">Username:</label>
15         <input type="text" id="username" name="username" required>
16       </div>
17       <div class="form-group">
18         <label for="email">Email:</label>
19         <input type="email" id="email" name="email" required>
20       </div>
21       <div class="form-group">
22         <label for="password">Password:</label>
23         <input type="password" id="password" name="password" required>
24       </div>
25       <button type="submit">Register</button>
26     </form>
27   </div>
28 </body>
29 </html>
30
```



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5500/client\_pages/login.html'. The browser's navigation bar includes back, forward, and refresh buttons. The login page itself is a simple white card with rounded corners on a light gray background. It features a bold 'Login' title, followed by an 'Email:' label and a text input field. Below that is a 'Password:' label and another text input field. At the bottom of the card is a prominent green button labeled 'Login'.



## client\_pages\login.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <link rel="stylesheet" href="styles.css">
7      <title>Login</title>
8  </head>
9  <body>
10     <div class="form-container">
11         <h2>Login</h2>
12         <form onSubmit="verify()">
13             <div class="form-group">
14                 <label for="email">Email:</label>
15                 <input type="email" id="email" name="email" required>
16             </div>
17             <div class="form-group">
18                 <label for="password">Password:</label>
19                 <input type="password" id="password" name="password" required>
20             </div>
21             <button type="submit">Login</button>
22         </form>
23     </div>
24 </body>
25 </html>
26
```



```
1  body {
2      font-family: Arial, sans-serif;
3      background-color: #f0f0f0;
4      display: flex;
5      justify-content: center;
6      align-items: center;
7      height: 100vh;
8      margin: 0;
9  }
10
11  .form-container {
12      background-color: white
13      padding: 20px;
14      border-radius: 8px;
15      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
16      width: 300px;
17  }
18
19  h2 {
20      text-align: center;
21      color: #333;
22  }
23
24  .form-group {
25      margin-bottom: 15px;
26  }
27
28  label {
29      display: block;
30      margin-bottom: 5px;
31      color: #555;
32  }
33
34  input[type="email"],
35  input[type="password"],
36  input[type="text"]{
37      /* width: 100%; */
38      padding: 10px;
39      border: 1px solid #ddd;
40      border-radius: 4px;
41  }
42
43  button {
44      width: 100%;
45      padding: 10px;
46      background-color: #28a745;
47      border: none;
48      border-radius: 4px;
49      color: white
50      font-size: 16px;
51      cursor: pointer;
52  }
53
54  button:hover{
55      background-color: #218838;
```

}



## server\index.js

```
1  const express = require('express');
2  const bodyParser = require('body-parser');
3  const mongoose = require('mongoose');
4  const alienRoutes = require('./controller/controller')
5
6  const app =
  express();
7
8  app.use(bodyParser.json());
9
10
11  const url = "mongodb://localhost:27017/alien"
12  // const url = "mongodb://127.0.0.1:27020,127.0.0.1:27021,127.0.0.1:27022/alien?replicaSet=m101"
13  mongoose.connect(url)
14  .then(() => {
15    console.log('Connected to MongoDB');
16  }).catch(err => {
17    console.error('Failed to connect to MongoDB', err);
18  });
19
20  app.use('/api/alien', alienRoutes);
21
22  const PORT = 3000;
23  app.listen(PORT, () => {
24    console.log(`Server running on port ${PORT}`);
25  });
26
```

Overview

RESTful API Basics #blue

POST localhost:3000/api/alien

Beta



localhost:3000/api/alien/

Save

Share

POST

localhost:3000/api/alien/

Send

Params

Authorization

Headers (9)

Body

Scripts

Tests

Settings

Cookies

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON

Beautify

```
1 {  
2   "name": "Blaxk",  
3   "age": 24,  
4   "grade": "z"  
5 }
```

Body

Cookies

Headers (7)

Test Results

201 Created

227 ms

318 B



Pretty

Raw

Preview

Visualize

JSON



```
1 {  
2   "name": "Blaxk",  
3   "age": 24,  
4   "grade": "z",  
5   "_id": "67305da04cee93cbc7564008",  
6   "__v": 0  
7 }
```

## server\controller\controller.js

```
1  const express = require('express');
2  const router = express.Router();
3      const Alien =
require('./models/alien'); 4
5  router.post('/', async (req, res) => {
6      try {
7          const alien = new Alien(req.body);
8          await alien.save();
9          res.status(201).send(alien);
10     } catch (error) {
11         res.status(400).send(error);
12     }
13 });
14
15 router.get('/', async (req, res) => {
16     try {
17         const aliens = await Alien.find();
18         res.status(200).send(aliens);
19     } catch (error) {
20         res.status(500).send(error);
21     }
22 });
23
24 router.get('/:id', async (req, res) => {
25     try {
26         const alien = await Alien.findById(req.params.id);
27         if (!alien) return res.status(404).send();
28         res.status(200).send(alien);
29     } catch (error) {
30         res.status(500).send(error);
31     }
32 });
33
34 router.patch('/:id', async (req, res) => {
35     try {
36         const alien = await Alien.findById(req.params.id);
37         if (!alien) return res.status(404).send();
38         alien.grade = req.body.grade;
39         alien.age = req.body.age;
40         updatedAlien = await alien.save();
41         res.status(200).send(updatedAlien);
42     } catch (error) {
43         res.status(400).send(error);
44     }
45 });
46
47 router.delete('/:id', async (req, res) => {
48     try {
49         const alien = await Alien.findByIdAndDelete(req.params.id);
50         if (!alien) return res.status(404).send();
51         res.status(200).send(alien);
52     } catch (error) {
53         res.status(500).send(error);
54     }
55 });
56
57 module.exports = router;
```

Compass

My Queries

CONNECTIONS (1)

Search connections

localhost:27017

admin

aliens

aliens

config

dbnew

local

test

aliens



localhost:27017 > aliens > aliens

Open MongoDB shell

Documents 0

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 - 1 of 1

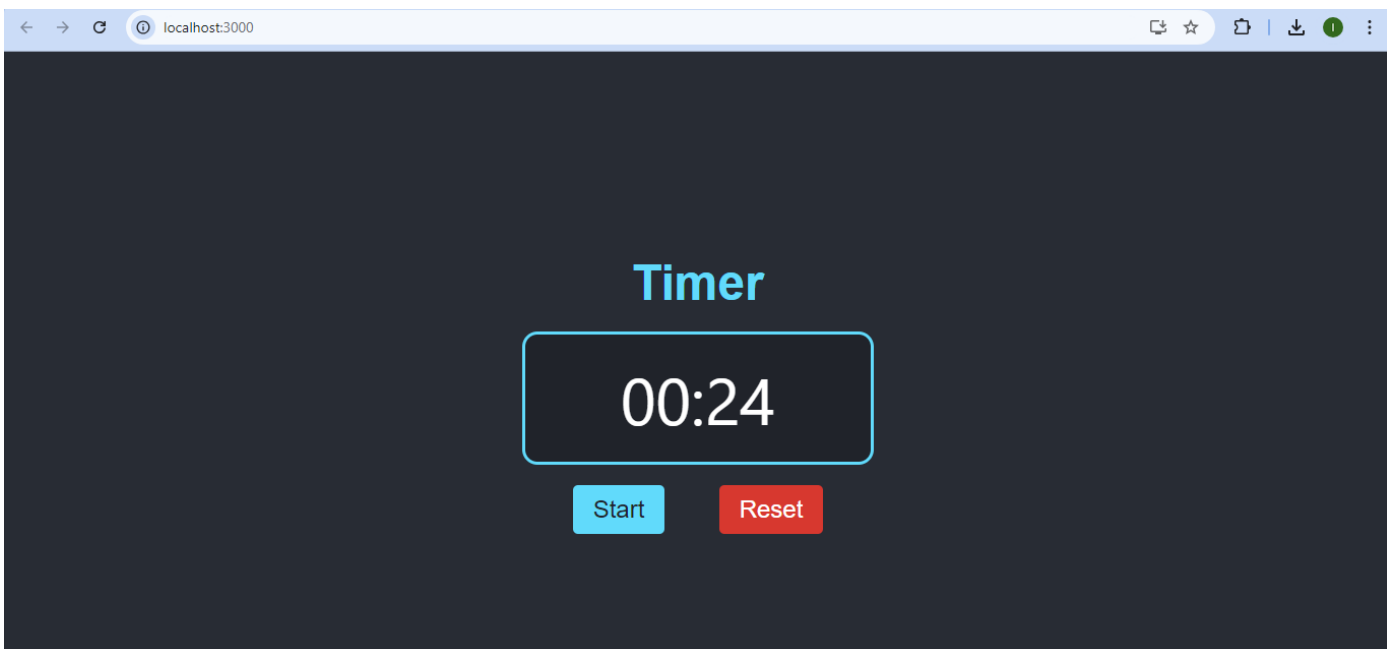


```
{
  "_id": ObjectId('67305da04cee93cbc7564008'),
  "name": "Blaxk",
  "age": 24,
  "grade": "z",
  "__v": 0
}
```



## server\models\alien.js

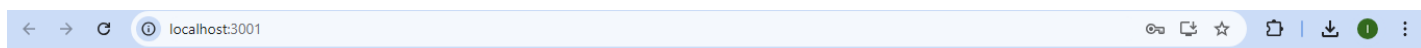
```
1 const mongoose =
  require('mongoose');
3 const alienSchema = new mongoose.Schema({
4   name: {
5     type: String,
6     required: true
7   },
8   age: {
9     type: Number,
10    required: true
11  },
12  grade: {
13    type: String,
14    required: true
15  }
16 });
17
18 const Alien = mongoose.model('Alien', alienSchema); 19
20 module.exports =
  Alien; 21
```



```

1  import React,{ useState, useEffect } from 'react';
2  import './Timer.css';
3
4  const Timer = () => {
5    const [time, setTime] = useState(0);
6    const [isRunning, setIsRunning] =
    useState(false);
7
8    useEffect(() => {
9      let interval
    = null;
10
11      if (isRunning) {
12        interval = setInterval(() => {
13          setTime((prevTime) => prevTime + 1);
14        }, 1000);
15      } else if (!isRunning && time !== 0) {
16        clearInterval(interval);
17      }
18
19      return () => clearInterval(interval);
20    },
    [isRunning, time]);
21
22    const handleStartPause = () => {
23      setIsRunning(!isRunning);
24    };
25
26    const handleReset = () => {
27      setTime(0);
28      setIsRunning(false);
29    };
30
31    const formatTime = (time) => {
32      const getSeconds = `0${time % 60}`.slice(-2);
33      const minutes = Math.floor(time / 60);
34      const getMinutes = `0${minutes % 60}`.slice(-2);
35      const getHours = `0${Math.floor(time / 3600)}`.slice(-2);
36
37      if (getHours > 0) {
38        return `${getHours}:${getMinutes}:${getSeconds}`;
39      } else {
40        return `${getMinutes}:${getSeconds}`;
41      }
42    };
43
44    return (
45      <div className="timer-container">
46        <h1 className="timer-title">Timer</h1>
47        <div className="timer-display">{formatTime(time)}</div>
48        <div className="timer-controls">
49          <button className="timer-button start-pause" onClick={handleStartPause}>
50            {isRunning ? 'Pause' : 'Start'}
51          </button>
52          <button className="timer-button reset" onClick={handleReset}>
53            Reset
54          </button>
55        </div>
56      </div>
57    );
58  };

```



## Password Strength Checker

.....

**Password Strength: Strong**

```

1  import React, {useState} from 'react';
2      import './PasswordStrength.css'; // Import
CSS for styling
3
4  const PasswordStrengthChecker = () => {
5      const [password, setPassword] = useState("");
6      const [strength, setStrength]
= useState("");
7
8      const checkPasswordStrength = (pwd) => {
9          let strengthLevel = 0;
10
11         if (pwd.length >= 8) strengthLevel += 1;
12         if (/[A-Z]/.test(pwd)) strengthLevel += 1;
13         if (/[a-z]/.test(pwd)) strengthLevel += 1;
14         if (/[\d-9]/.test(pwd)) strengthLevel += 1;
15         if (/^[A-Za-z0-9]/.test(pwd)) strengthLevel += 1;
16
17         switch (strengthLevel) {
18             case 0:
19                 setStrength("");
20                 break;
21             case 1:
22                 setStrength("Very Weak");
23                 break;
24             case 2:
25                 setStrength("Weak");
26                 break;
27             case 3:
28                 setStrength("Moderate");
29                 break;
30             case 4:
31                 setStrength("Strong");
32                 break;
33             case 5:
34                 setStrength("Very Strong");
35                 break;
36             default:
37                 setStrength("");
38         }
39     };
40
41     const handlePasswordChange = (e) => {
42         const newPassword = e.target.value;
43         setPassword(newPassword);
44         checkPasswordStrength(newPassword);
45     };
46
47     return (
48         <div className="password-checker-container">
49             <h1>Password Strength Checker</h1>
50             <input
51                 type="password"
52                 placeholder="Enter password"
53                 value={password}
54                 onChange={handlePasswordChange}
55                 className="password-input"
56             />
57             <div className={`strength-indicator ${strength.toLowerCase()}`>
58                 {strength} && <p>Password Strength: {strength}</p>
59             </div>
60         </div>
61     );
62 };
63
64 export default PasswordStrengthChecker;

```

Overview

RESTful API Basics #blu

POST localhost:7000/login



Beta



localhost:7000/login

Save



Share

POST



localhost:7000/login

Send



Params

Authorization

Headers (9)

Body

Scripts

Tests

Settings

Cookies

☐ none☐ form-data☐ x-www-form-urlencoded☒ raw☐ binary☐ GraphQL

JSON



Beautify

```
1 {
2   "username": "Onner"
3 }
```

Body Cookies Headers (7) Test Results

200 OK

112 ms

373 B



Pretty

Raw

Preview

Visualize

JSON



```
1 {
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bm90aXIiOiJ0aW0tMjUuZXIiLCJpYXQiOiE3MzEyMjMzF9.w9XLeRfpJXVlc01KZgca-UbjvL8PJQ3VXFcApJBjd0w"
3 }
```

Overview

RESTful API Basics #blu

GET localhost:7000/posts



Beta



localhost:7000/posts

Save



Share

GET



localhost:7000/posts

Send



Params

Authorization

Headers (9)

Body

Scripts

Tests

Settings

Cookies

<input checked="" type="checkbox"/>	Host	<calculated when request is sent>	
<input checked="" type="checkbox"/>	User-Agent	PostmanRuntime/7.42.0	
<input checked="" type="checkbox"/>	Accept	/*/*	
<input checked="" type="checkbox"/>	Accept-Encoding	gzip, deflate, br	
<input checked="" type="checkbox"/>	Connection	keep-alive	
<input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bm90aXIiOiJ0aW0tMjUuZXIiLCJpYXQiOiE3MzEyMjMzF9.w9XLeRfpJXVlc01KZgca-UbjvL8PJQ3VXFcApJBjd0w	
	Key	Value	Description

Body Cookies Headers (7) Test Results

200 OK

27 ms

272 B



Pretty

Raw

Preview

Visualize

JSON



```
1 [
2   {
3     "name": "Onner",
4     "title": "By Onner"
5   }
6 ]
```

jwt\_demo\server1.js

```
1
2  const express = require('express')
3  const app = express()
4    const jwt =
require('jsonwebtoken') 5
6  app.use(express.json())
7  require('dotenv').config()
8
9  const posts = [
10    {
11      name: "Onner",
12      title: "By Onner"
13    },
14    {
15      name: "Twoer",
16      title: "By"
17    }
18  ]
19
20  const authenticateToken=(req, res, next)=> {
21    const authHeader = req.headers['authorization']
22    const token = authHeader && authHeader.split(' ')[1]
23    if (!token) return
res.sendStatus(401) 24
25    jwt.verify(token, process.env.ACCESS_TOKEN, (err, user) => {
26      if (err)
27        {
28          return res.sendStatus(403)
29        }
30      req.user = user
31      next()
32    })
33  }
34  app.post('/login', (req, res) => {
35    const username = req.body.username
36    const user = { name:
username } 37
38    const accessToken = jwt.sign(user, process.env.ACCESS_TOKEN)
39
40    res.json({ token: accessToken})
41  })
42
43  app.use(authenticateToken)
44  app.get('/posts', (req, res) => {
45    console.log(req.user.name)
46    res.json(posts.filter(post => post.name ===
req.user.name)) 47
48  })
```

49

50 app.listen(7000)

DEPARTMENT OF INFORMATION TECHNOLOGY

CLASS TIME TABLE FOR THE AY 2024-25 ODD SEMESTER

CLASS: BE - V SEMESTER IT-1

ROOM NO: L-305

W.E.F.: 30/07/2024

PERIOD	I	II	III		IV	V	VI
DAY/TIME	(9:10 TO 10:10 AM)	(10:10 TO 11:10 AM)	(11:15 TO 12:15 PM)	LUNCH BREAK (12:15 TO 1:00 PM)	(1:00 TO 2:00 PM)	(2:00 TO 3:00 PM)	(3:05 TO 4:05 PM)
MONDAY	CT LAB (B1) / ML LAB (B2)	PE-2	CN		EAD	SE	CN LAB (B1) / EAD LAB (B2)
TUESDAY	PE-2	ML	EAD		SE	LIBRARY (B2)	CC LAB (B1)
WEDNESDAY	CN	EAD	FLAT		ML	FLAT	SPORTS (B2)
THURSDAY	ML	CN	CT LAB (B2) / ML LAB (B1)		FLAT	LIBRARY (B1)	CC LAB (B2)
FRIDAY	CN LAB (B2) / EAD LAB (B1)	PE-2	FLAT		SE	LIBRARY (B1)	SPORTS (B1)
SATURDAY	MENTORING						



time\_table\time\_table.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6     <title>Class Timetable</title>
7     <link
8       href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"
9       rel="stylesheet"
10    />
11    <style>
12      * {
13        text-transform: uppercase;
14      }
15      .timetable {
16        font-size: 14px;
17        text-align: center;
18        border: 3px solid black;
19        margin-top: 30px;
20      }
21      .timetable th,
22      .timetable td {
23        border: 3px solid black;
24        padding: 10px;
25      }
26      .lunch-break {
27        writing-mode: vertical-rl;
28        text-align: center;
29        font-weight: bold;
30      }
31    </style>
32  </head>
33  <body>
34    <div class="container">
35      <h2 class="text-center my-4">Department of Information Technology</h2>
36      <h4 class="text-center">
37        Class Time Table for the AY 2024-25 ODD Semester
38      </h4>
39      <h5 class="text-center">Class: BE - V Semester IT-1</h5>
40      <h6 class="text-center">Room No: L-305</h6>
41      <h6 class="text-center">W.E.F.: 30/07/2024</h6> 42
42      <table class="table table-bordered timetable">
43        <thead>
44          <tr>
45            <th rowspan="2">Period</th>
46            <th>I</th>
47            <th>II</th>
48            <th>III</th>
49            <th class="lunch-break" rowspan="6"></th>
50            <th>IV</th>
51            <th>V</th>
52            <th>VI</th>
53          </tr>
54        </thead>
55        <tbody>
56          <tr>
57            <th>Day/Time</th>
58            <th>(9:10 to 10:10 AM)</th>
59            <th>(10:10 to 11:10 AM)</th>
```



```

61         <th class="lunch-break" rowspan="6">
62             <span style="color: red">Lunch Break </span>
63             (12:15 to 1:00 PM)
64         </th>
65         <th>(1:00 to 2:00 PM)</th>
66         <th>(2:00 to 3:00 PM)</th>
67         <th>(3:05 to 4:05 PM)</th>
68     </tr>
69     <tr>
70         <th>Monday</th>
71         <td>CT Lab (B1) / ML Lab (B2)</td>
72         <td>PE-2</td>
73         <td>CN</td>
74         <td>EAD</td>
75         <td>SE</td>
76         <td>CN Lab (B1) / EAD Lab (B2)</td>
77     </tr>
78     <tr>
79         <th>Tuesday</th>
80         <td>PE-2</td>
81         <td>ML</td>
82         <td>EAD</td>
83         <td>SE</td>
84         <td>Library (B2)</td>
85         <td>CC Lab (B1)</td>
86     </tr>
87 </tbody>
88 </table>
89 </div>
90 </div>
91
92 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"></script>
93 </body>
94 </html>

```

TERMINAL1

```
start mongod --replSet m101 -logpath \data\rs1\1.log --dbpath \data\rs1 --port 27020
```

TERMINAL2

```
start mongod --replSet m101 -logpath \data\rs2\2.log --dbpath \data\rs2 --port 27021
```

TERMINAL3

```
start mongod --replSet m101 -logpath \data\rs3\3.log --dbpath \data\rs3 --port 27022
```

TERMINAL1

```
mongosh --port 27020
```

```
config = { _id: "m101", members:[  
    { _id : 0, host : "localhost:27020"},  
    { _id : 1, host : "localhost:27021"},  
    { _id : 2, host : "localhost:27022"} ]  
};
```

```
rs.initiate(config);  
rs.status();
```

```
use new  
db.new_col.find();
```

TERMINAL2

```
mongosh --port 27021  
rs.secondaryOk()
```

TERMINAL3

```
mongosh --port 27022  
rs.secondaryOk()
```

TERMINAL1

```
use admin;  
db.shutdownServer();
```

open a TERMINAL2 & TERMINAL3

```
rs.status();
```