

MEAN STACK PROJECT

MEAN STACK EMPLOYEE CRUD APPLICATION

ASWIN J AM.EN.P2MCA22062

DEMO URL: [MEAN-STACK-EMPLOYEE-CRUD-APP DEMO -ASWIN J.mp4](#)

Code Snippets :

Backend:

Inside the backend inside the models folder I have created an employee.js file where I created schema for creating an employee details

Employee.json

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

// Define collection and schema
let Employee = new Schema({
  name: {
    type: String
  },
  email: {
    type: String
  },
  designation: {
    type: String
  },
  phoneNumber: {
    type: Number
  }
})
```

```
}, {  
  collection: 'employees'  
})
```

```
module.exports = mongoose.model('Employee', Employee)
```

here used mongoose .

Inside the backend folder there is routes folder where I created employee.route.js

Employee.route.js

```
const express = require("express");  
const app = express();  
const employeeRoute = express.Router();  
const cors = require("cors");  
  
// CORS OPTIONS  
var whitelist = ["http://localhost:4200", "http://localhost:4000"];  
var corsOptionsDelegate = function (req, callback) {  
  var corsOptions;  
  if (whitelist.indexOf(req.header("Origin")) !== -1) {  
    corsOptions = {  
      origin: "*",  
      methods: "GET,HEAD,PUT,PATCH,POST,DELETE",  
    };  
  } else {  
    corsOptions = { origin: false }; // disable CORS for this  
request  
  }  
  callback(null, corsOptions);  
};  
  
// Employee model  
let Employee = require("../models/Employee");  
  
// Add Employee  
employeeRoute.route("/create").post(async (req, res, next) => {  
  await Employee.create(req.body)
```

```

        .then((result) => {
            res.json({
                data: result,
                message: "Data successfully added!",
                status: 200,
            });
        })
        .catch((err) => {
            return next(err);
        });
    });

// Get All Employees
employeeRoute
    .route("/", cors(corsOptionsDelegate))
    .get(async (req, res, next) => {
        await Employee.find()
            .then((result) => {
                res.writeHead(201, { "Content-Type": "application/json" });
                res.end(JSON.stringify(result));
            })
            .catch((err) => {
                return next(err);
            });
    });

// Get single employee
employeeRoute.route("/read/:id").get(async (req, res, next) => {
    await Employee.findById(req.params.id, req.body)
        .then((result) => {
            res.json({
                data: result,
                message: "Data successfully retrieved.",
                status: 200,
            });
        })
        .catch((err) => {
            return next(err);
        });
    });

// Update employee

```

```

employeeRoute.route("/update/:id").put(async (req, res, next) => {
  await Employee.findByIdAndUpdate(req.params.id, {
    $set: req.body,
  })
  .then((result) => {
    res.json({
      data: result,
      msg: "Data successfully updated.",
    });
  })
  .catch((err) => {
    console.log(err);
  });
});

// Delete employee
employeeRoute.route("/delete/:id").delete(async (req, res) => {
  await Employee.findByIdAndRemove(req.params.id)
  .then(() => {
    res.json({
      msg: "Data successfully updated.",
    });
  })
  .catch((err) => {
    console.log(err);
  });
});

module.exports = employeeRoute;

```

here were I created employee crud functions

In index.js file I connected my node backend server with MongoDB .I created a cluster named cluster0 where I get the connection string and pasted inside async function

```

mongodbConnection() {
  await mongoose.connect(
    "mongodb+srv://aswinjbca2019:abcd@cluster0.gkkush3.mongodb.net
/?retryWrites=true&w=majority",

```

```

    {
      useUrlParser: true,
      useUnifiedTopology: true,
    },
    6000
  );
}

```

Index.js

```

const express = require('express')
const path = require('path')
const mongoose = require('mongoose')
const cors = require('cors')
const bodyParser = require('body-parser')

// Connecting with mongo db
// Connecting MongoDB
async function mongoDbConnection() {
  await mongoose.connect(
    "mongodb+srv://aswinjbca2019:abcd@cluster0.gkkush3.mongodb.net
/?retryWrites=true&w=majority",
    {
      useUrlParser: true,
      useUnifiedTopology: true,
    },
    6000
  );
}

mongoDbConnection().then(() => {
  console.log("MongoDB successfully connected.");
}),
(err) => {
  console.log("Could not connected to database : " + err);
};

// Setting up port with express js
const employeeRoute = require('../backend/routes/employee.route')

```

```

const app = express()
app.use(bodyParser.json())
app.use(
  bodyParser.urlencoded({
    extended: false,
  }),
)
app.use(cors())
app.use(express.static(path.join(__dirname, 'dist/mean-stack-crud-app')))
app.use('/', express.static(path.join(__dirname, 'dist/mean-stack-crud-app')))
app.use('/api', employeeRoute)

// Create port
const port = process.env.PORT || 4000
const server = app.listen(port, () => {
  console.log('Connected to port ' + port)
})

// Find 404 and hand over to error handler
app.use((req, res, next) => {
  next(createError(404))
})

// error handler
app.use(function (err, req, res, next) {
  console.error(err.message) // Log error message in our server's console
  if (!err.statusCode) err.statusCode = 500 // If err has no specified error code, set error code to 'Internal Server Error (500)'
  res.status(err.statusCode).send(err.message) // All HTTP requests must have a response, so let's send back an error with its status code and message
})

```

This is my package.json file

Package.json

```

{
  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.20.2",
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "mongoose": "^7.3.4"
  },
  "devDependencies": {
    "nodemon": "^3.0.1"
  }
}

```

Frontend

Inside the src ,and app ,components folder I created required components for Employee crud app.

Employeecreate component

Employee-create.componet.html

```

<div class="row justify-content-center">
  <div class="col-md-4 register-employee">
    <!-- form card register -->
    <div class="card-body">
      <h2>Employee Register</h2>
      <form [formGroup]="employeeForm" (ngSubmit)="onSubmit()">
        <div class="mb-3">
          <label for="inputName">Name</label>
          <input class="form-control" type="text"
formControlName="name" />
          <!-- error -->
          <div
            class="text-danger"
            *ngIf="submitted && myForm['name'].errors?.['required']"
          >
            Name is required.
          </div>
        </div>

        <div class="mb-3">
          <label for="inputEmail3">Email</label>
          <input class="form-control" type="text"
formControlName="email" />
          <!-- error -->
          <div
            class="text-danger"
            *ngIf="submitted &&
myForm['email'].errors?.['required']"
          >
            Enter your email.
          </div>
          <div
            class="text-danger"
            *ngIf="submitted && myForm['email'].errors?.['pattern']"
          >
            Enter valid email.
          </div>
        </div>

        <div class="mb-3">
          <label for="inputPassword3">Designation</label>
          <select

```



```

        class="custom-select form-control"
        (change)="updateProfile($event.target.value)"
        formControlName="designation"
    >
        <option value="">Choose...</option>
        <option
            *ngFor="let employeeProfile of EmployeeProfile"
            value="{{ employeeProfile }}"
        >
            {{ employeeProfile }}
        </option>
    </select>
    <!-- error -->
    <div
        class="text-danger"
        *ngIf="submitted &&
myForm['designation'].errors?.['required']"
    >
        Choose designation.
    </div>
</div>

<div class="mb-3">
    <label for="inputVerify3">Mobile No</label>
    <input
        class="form-control"
        type="text"
        formControlName="phoneNumber"
    />
    <!-- error -->
    <div
        class="text-danger"
        *ngIf="submitted &&
myForm['phoneNumber'].errors?.['required']"
    >
        Enter your phone number.
    </div>
    <div
        class="text-danger"
        *ngIf="submitted &&
myForm['phoneNumber'].errors?.['pattern']"
    >

```

```

        Enter Numbers Only
    </div>
</div>

    <div class="d-grid">
        <button class="btn btn-primary btn-block "
(click)="openSnackBar('Employee Successfully Created')"
type="submit">
            Register
        </button>
    </div>
</form>
</div>
</div>
<!-- form card register -->
<div class="col-md-4">
    
</div>

</div>

```

Employee-create-component.ts

```

import { Router } from '@angular/router';
import { ApiService } from '../service/api.service';
import { Component, OnInit, NgZone } from '@angular/core';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';
import { MatSnackBar } from '@angular/material/snack-bar';

@Component({
  selector: 'app-employee-create',
  templateUrl: './employee-create.component.html',
  styleUrls: ['./employee-create.component.scss'],
})

export class EmployeeCreateComponent implements OnInit {
  submitted = false;
  employeeForm: FormGroup;

```

```

EmployeeProfile: any = ['Finance', 'BDM', 'HR', 'Sales', 'Admin'];

constructor(
  public fb: FormBuilder,
  private router: Router,
  private ngZone: NgZone,
  private apiService: ApiService,
  private _snackBar: MatSnackBar
) {
  this.mainForm();
}

ngOnInit() {}

mainForm() {
  this.employeeForm = this.fb.group({
    name: ['', [Validators.required]],
    email: [
      '',
      [
        Validators.required,
        Validators.pattern('[a-z0-9._%+-]+@[a-z0-9.-]+.[a-z]{2,3}$'),
      ],
    ],
    designation: ['', [Validators.required]],
    phoneNumber: ['', [Validators.required,
Validators.pattern('^[0-9]+$')]],
  });
}

openSnackBar(message: string): void {
  this._snackBar.open(message, 'Close', {
    duration: 2000,
  });
}

// Choose designation with select dropdown
updateProfile(e) {
  this.employeeForm.get('designation').setValue(e, {
    onlySelf: true,
  });
}

```

```

// Getter to access form control
get myForm() {
  return this.employeeForm.controls;
}

onSubmit() {
  this.submitted = true;
  if (!this.employeeForm.valid) {
    return false;
  } else {
    return
this.apiService.createEmployee(this.employeeForm.value).subscribe({
  complete: () => {
    console.log('Employee successfully created!'),
    this.ngZone.run(() =>
this.router.navigateByUrl('/employees-list'));
  },
  error: (e) => {
    console.log(e);
  },
});
}
}
}

```

Employee-create.component.scss

```

/* employee-create.component.scss */
@keyframes hovering {
  /* Define your keyframes here */
  0% {
    /* Styles at the start of the animation */
    transform: translateY(0);
  }
  50% {
    /* Styles at the middle of the animation */
    transform: translateY(-10px);
  }
}

```

```

    }
    100% {
        /* Styles at the end of the animation */
        transform: translateY(0);
    }
}

/* Apply the animation properties to the image element */
.col-md-4 img {
    animation-duration: 5s;
    animation-timing-function: ease;
    animation-delay: 0s;
    animation-iteration-count: infinite;
    animation-direction: normal;
    animation-fill-mode: none;
    animation-play-state: running;
    animation-name: hovering;
    /* You can remove the rest of the animation properties if not
needed for your case */
}

```

Now I am adding employeelist , employeeedit component,home component,user component ,login component,user component,manageemployee component

Employee-edit.componet.html

```

<div class="row justify-content-center">
  <div class="col-md-4 register-employee">
    <!-- form card register -->
    <div class="card card-outline-secondary">
      <div class="card-header">
        <h3 class="mb-0">Edit Employee</h3>
      </div>

```

```

<div class="card-body">
  <form [formGroup]="editForm" (ngSubmit)="onSubmit()">
    <div class="mb-3">
      <label for="inputName">Name</label>
      <input class="form-control" type="text"
formControlName="name" />
      <div
        class="text-danger"
        *ngIf="submitted &&
myForm['name'].errors?.['required']"
      >
        Name is required.
      </div>
    </div>
    <div class="mb-3">
      <label for="inputEmail3">Email</label>
      <input class="form-control" type="text"
formControlName="email" />
      <!-- error -->
      <div
        class="text-danger"
        *ngIf="submitted &&
myForm['email'].errors?.['required']"
      >
        Enter your email.
      </div>
      <div
        class="text-danger"
        *ngIf="submitted &&
myForm['email'].errors?.['pattern']"
      >
        Enter valid email.
      </div>
    </div>

    <div class="mb-3">
      <label for="inputPassword3">Designation</label>
      <select
        class="custom-select form-control"
        (change)="updateProfile($event.target.value)"
        FormControlName="designation"
      >

```

```

        <option value="">Choose...</option>
        <option
            *ngFor="let employeeProfile of EmployeeProfile"
            value="{{ employeeProfile }}"
        >
            {{ employeeProfile }}
        </option>
    </select>
    <!-- error -->
    <div
        class="text-danger"
        *ngIf="submitted &&
myForm['designation'].errors?.['required']"
    >
        Choose designation.
    </div>
</div>

<div class="mb-3">
    <label for="inputVerify3">Mobile No</label>
    <input
        class="form-control"
        type="text"
        formControlName="phoneNumber"
    />
    <!-- error -->
    <div
        class="text-danger"
        *ngIf="submitted &&
myForm['phoneNumber'].errors?.['required']"
    >
        Enter your phone number.
    </div>
    <div
        class="text-danger"
        *ngIf="submitted &&
myForm['phoneNumber'].errors?.['pattern']"
    >
        Enter Numbers Only
    </div>
</div>

```

```

        <div class="d-grid">
            <button class="btn btn-dark btn-
block" (click)="openSnackBar('Employee Updated
Successfully')" type="submit">
                Update
            </button>
        </div>
    </form>
</div>
</div>
<!-- form -->
</div>
</div>

```

Employee-edit.component.ts

```

import { Employee } from './../../model/employee';
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { ApiService } from './../../service/api.service';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';
import { MatSnackBar } from '@angular/material/snack-bar';

```

```

@Component({
  selector: 'app-employee-edit',
  templateUrl: './employee-edit.component.html',
  styleUrls: ['./employee-edit.component.scss'],
})

```

```

export class EmployeeEditComponent implements OnInit {
  submitted = false;
  editForm: FormGroup;
  employeeData: Employee[];
  EmployeeProfile: any = ['Finance', 'BDM', 'HR', 'Sales', 'Admin'];

```

```

  constructor(
    public fb: FormBuilder,
    private actRoute: ActivatedRoute,
    private apiService: ApiService,
    private router: Router,
    private _snackBar: MatSnackBar

```



```

) {}

ngOnInit() {
  this.updateEmployee();
  let id = this.actRoute.snapshot.paramMap.get('id');
  this.getEmployee(id);
  this.editForm = this.fb.group({
    name: ['', [Validators.required]],
    email: [
      '',
      [
        Validators.required,
        Validators.pattern('[a-z0-9._%+-]+@[a-z0-9.-]+.[a-z]{2,3}$'),
      ],
    ],
    designation: ['', [Validators.required]],
    phoneNumber: ['', [Validators.required,
Validators.pattern('^[0-9]+$')]],
  });
}

openSnackBar(message: string): void {
  this._snackBar.open(message, 'Close', {
    duration: 2000,
  });}

// Choose options with select-dropdown
updateProfile(e) {
  this.editForm.get('designation').setValue(e, {
    onlySelf: true,
  });
}

// Getter to access form control
get myForm() {
  return this.editForm.controls;
}

getEmployee(id) {
  this.apiService.getEmployee(id).subscribe((data) => {
    this.editForm.setValue({
      name: data.data['name'],
      email: data.data['email'],

```

```

        designation: data.data['designation'],
        phoneNumber: data.data['phoneNumber'],
    });
});
}

updateEmployee() {
    this.editForm = this.fb.group({
        name: ['', [Validators.required]],
        email: [
            '',
            [
                Validators.required,
                Validators.pattern('[a-z0-9._%+-]+@[a-z0-9.-]+.[a-z]{2,3}$'),
            ],
        ],
        designation: ['', [Validators.required]],
        phoneNumber: ['', [Validators.required,
Validators.pattern('^[0-9]+$')]],
    });
}

onSubmit() {
    this.submitted = true;
    if (!this.editForm.valid) {
        return false;
    } else {
        if (window.confirm('Are you sure?')) {
            let id = this.actRoute.snapshot.paramMap.get('id');
            this.apiService.updateEmployee(id,
this.editForm.value).subscribe({
                complete: () => {
                    this.router.navigateByUrl('/employees-list');
                    console.log('Content updated successfully!');
                },
                error: (e) => {
                    console.log(e);
                },
            });
        }
    }
}

```

```
}  
}
```

Employee-list.component.html

```
<div class="container">  
  <!-- No data message -->  
  <p *ngIf="Employee.length <= 0" class="no-data text-center">  
    There is no employee added yet!  
  </p>  
  
  <!-- Employee list -->  
  <table class="table table-bordered" *ngIf="Employee.length > 0">  
    <thead class="table-success">  
      <tr>  
        <th scope="col">Employee ID</th>  
        <th scope="col">Name</th>  
        <th scope="col">Email</th>  
        <th scope="col">Designation</th>  
        <th scope="col">Phone No</th>  
        <th scope="col center">Update</th>  
      </tr>  
    </thead>  
    <tbody>  
      <tr *ngFor="let employee of Employee; let i = index">  
        <th scope="row">{{ employee._id }}</th>  
        <td>{{ employee.name }}</td>  
        <td>{{ employee.email }}</td>  
        <td>{{ employee.designation }}</td>  
        <td>{{ employee.phoneNumber }}</td>  
        <td class="text-center edit-block">  
          <span class="edit" [routerLink]="['/edit-employee/',  
employee._id]">  
            <button type="button" class="btn btn-success btn-  
sm">Edit</button>  
          </span>  
          <span class="delete" (click)="removeEmployee(employee,  
i)">  
            <button type="button" class="btn btn-danger btn-  
sm">Delete</button>  
          </span>  
        </td>  
      </tr>  
    </tbody>  
  </table>  
</div>
```

```
        </tr>
      </tbody>
    </table>
  </div>
```

Employee-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ApiService } from './../../service/api.service';
import { MatSnackBar } from '@angular/material/snack-bar';

@Component({
  selector: 'app-employee-list',
  templateUrl: './employee-list.component.html',
  styleUrls: ['./employee-list.component.scss'],
})
export class EmployeeListComponent implements OnInit {
  Employee: any = [];

  constructor(private apiService: ApiService, private _snackBar:
MatSnackBar) {
    this.readEmployee();
  }

  ngOnInit() {}

  readEmployee() {
    this.apiService.getEmployees().subscribe((data) => {
      this.Employee = data;
    });
  }

  openSnackBar(message: string): void {
    this._snackBar.open(message, 'Close', {
      duration: 2000,
    });
  }

  removeEmployee(employee, index) {
    if (window.confirm('Are you sure?')) {
```

```

        this.apiService.deleteEmployee(employee._id).subscribe((data)
=> {
            this.Employee.splice(index, 1);
            this.openSnackBar('Employee Deleted Successfully');
        });
    }
}
}

```

Employee-list.component.scss

```

/* Add this CSS to your styles.css or component-specific styles file
*/

```

```

.container {
    margin: 20px auto;
    padding: 20px;
    width: 1000px;
    border-radius: 10px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
    background-color: #ffffff;
}

.no-data {
    color: #ffffff;
    text-align: center;
    margin-top: 20px;
}

.table {
    width: 50%;
    border-collapse: collapse;
    margin-top: 20px;
}

th, td {
    padding: 15px;
    text-align: left;
}

th:nth-child(2), td:nth-child(2) {
    /* Adjust the width as needed */
}

```

```
width: 200px;
max-width: 200px;
overflow: hidden;
text-overflow: ellipsis;
white-space: nowrap;
}

thead {
  background-color: #007bff;
  color: #fff;
}

tbody tr:nth-child(even) {
  background-color: #f8f9fa;
}

.edit-block {
  display: flex;
  justify-content: center;
  align-items: center;
}

.edit, .delete {
  margin-right: 10px;
}

.btn {
  padding: 10px 20px;
  font-size: 14px;
  border-radius: 5px;
  transition: background-color 0.3s ease;
}

.btn-success {
  background-color: #ffffff;
  color: #3928cf;
}

.btn-danger {
  background-color: #ffffff;
  color: #1337d6;
}
```

```

.btn:hover {
  background-color: #d4d4d4;
}

@media only screen and (max-width: 768px) {
  .container {
    padding: 15px;
  }

  .table th, .table td {
    font-size: 13px;
    padding: 12px;
  }

  .edit-block {
    flex-direction: column;
    align-items: flex-start;
  }

  .edit, .delete {
    margin-right: 0;
    margin-bottom: 8px;
  }
}

```

[Home.component.html](#)

```
<!-- about.component.html -->
```

```

<section class="about-section">
  <div class="container">
    <div class="about-content">
      <div class="image-container">
        
      </div>
      <div class="text-container">
        <h2>About Aswin J</h2>
        <p>
          Aswin J is an Final year Master of Computer
          Application Student at Amrita Vishwa Vidyapeetham, Amritapuri.

```

```

        </p>

    </div>
</div>
</div>
</section>

```

Login.component.ts

```

import { Component } from '@angular/core';

@Component({
  selector: 'app-home',
  templateUrl: './home.component.html',
  styleUrls: ['./home.component.scss']
})
export class HomeComponent {

}

```

Manage-employee.component.html

```

<div class="container">
  <header>
    <h1>Employee CRUD Application</h1>
    <p>Welcome to our Employee CRUD application! This simple
application allows you to perform basic CRUD operations.</p>
  </header>

  <div class="content">
    <div class="overview">
      <h2>Overview</h2>
      <p>This CRUD application provides the following
operations:</p>
      <ul>
        <li><strong>Create:</strong> Add new Employees to the
database.</li>
        <li><strong>Read:</strong> View a list of existing
Employees.</li>

```



```

        <li><strong>Update:</strong> Modify existing Employees in
the database.</li>
        <li><strong>Delete:</strong> Remove Employees from the
database.</li>
    </ul>
</div>
</div>
</div>

```

Manage-employee.component.ts

```

import { Component } from '@angular/core';

@Component({
  selector: 'app-manage-employee',
  templateUrl: './manage-employee.component.html',
  styleUrls: ['./manage-employee.component.scss']
})
export class ManageEmployeeComponent {

}

```

Manage-employee.component.scss

```

.container {
  max-width: 800px;
  margin: 20px auto;
  padding: 20px;
  background-color: #f8f9fa;
  border-radius: 8px;
  box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);
}

header {
  background-color: #007bff;
  color: #fff;
  padding: 20px;
  text-align: center;
}

```

```
    border-radius: 8px 8px 0 0;
}

.content {
    margin-top: 20px;
}

.overview {
    margin-bottom: 20px;
}

.overview h2,
.overview p,
.overview ul,
.overview li {
    color: #333;
}

.overview ul {
    list-style-type: none;
    padding: 0;
}

.overview li {
    margin-bottom: 10px;
}

.crud-actions {
    display: flex;
    flex-wrap: wrap; /* Allow buttons to wrap to the next line on
small screens */
    justify-content: center;
    margin-top: 20px;
}

.btn {
    padding: 12px 24px;
    font-size: 18px;
    text-decoration: none;
    border-radius: 8px;
    transition: background-color 0.3s ease;
    cursor: pointer;
}
```

```

    margin: 5px; /* Add margin between buttons */
}

.btn-primary {
    background-color: #ff5c5c;
    color: #fff;
}

.btn-success {
    background-color: #4caf50;
    color: #fff;
}

.btn-warning {
    background-color: #ffbb33;
    color: #fff;
}

.btn-danger {
    background-color: #dc3545;
    color: #fff;
}

.btn:hover {
    background-color: #555;
}

```

User.component.html

```

<mat-card>
  <mat-card-header>
    <h2>User Listing</h2>
  </mat-card-header>
  <mat-card-content>
    <div class="mat-elevation-z8">
      <table mat-table matSort [dataSource]="dataSource">

        <!-- Position Column -->
        <ng-container matColumnDef="username">
          <th mat-header-cell *matHeaderCellDef> Username
        </th>

```

```

        <td mat-cell *matCellDef="let element">
{{element.id}} </td>
    </ng-container>

    <!-- Name Column -->
    <ng-container matColumnDef="name">
        <th mat-header-cell mat-sort-header
*matHeaderCellDef> Name </th>
        <td mat-cell *matCellDef="let element">
{{element.name}} </td>
    </ng-container>

    <!-- Weight Column -->
    <ng-container matColumnDef="email">
        <th mat-header-cell *matHeaderCellDef> Email </th>
        <td mat-cell *matCellDef="let element">
{{element.email}} </td>
    </ng-container>

    <!-- Symbol Column -->
    <ng-container matColumnDef="status">
        <th mat-header-cell *matHeaderCellDef> Status </th>
        <td mat-cell *matCellDef="let element">
{{element.isactive ? 'Active':'In Active'}} </td>
    </ng-container>
    <ng-container matColumnDef="role">
        <th mat-header-cell *matHeaderCellDef> Role </th>
        <td mat-cell *matCellDef="let element">
{{element.role === '' ? 'Un Assigned':element.role}} </td>
    </ng-container>
    <ng-container matColumnDef="action">
        <th mat-header-cell *matHeaderCellDef> Action </th>
        <td mat-cell *matCellDef="let element">
            <button *ngIf="element.role !== 'admin'"
(click)="updateuser(element.id)" mat-raised-button
color="primary">Update</button>
        </td>
    </ng-container>

    <tr mat-header-row
*matHeaderRowDef="displayedColumns"></tr>

```

```

        <tr mat-row *matRowDef="let row; columns:
displayedColumns;"></tr>
    </table>

    <mat-paginator [pageSizeOptions]="[5, 10, 20]"
        showFirstLastButtons
        aria-label="Select page of periodic
elements">
    </mat-paginator>
</div>
</mat-card-content>
</mat-card>

```

User.component.ts

```

import { AfterViewInit, Component, ViewChild } from '@angular/core';
import { FormBuilder } from '@angular/forms';
import { MatTableDataSource } from '@angular/material/table';
import { MatPaginator } from '@angular/material/paginator';
import { MatSort } from '@angular/material/sort';
import { AuthService } from 'src/app/service/auth-service.service';
import { MatDialog } from '@angular/material/dialog';
import { EmployeeCreateComponent } from '../employee-
create/employee-create.component';
@Component({
  selector: 'app-user',
  templateUrl: './user.component.html',
  styleUrls: ['./user.component.scss']
})
export class UserComponent implements AfterViewInit {
  displayedColumns: string[] = ['username', 'name', 'email',
'status', 'role', 'action'];
  constructor(private service: AuthService, private dialog:
MatDialog) {
    this.LoadUser();
  }
  userlist: any;
  dataSource: any;

```

```

@ViewChild(MatPaginator) paginator!: MatPaginator;
@ViewChild(MatSort) sort!: MatSort;

ngAfterViewInit(): void {

}

LoadUser() {
    this.service.Getall().subscribe(res => {
        this.userlist = res;
        this.dataSource = new MatTableDataSource(this.userlist);
        this.dataSource.paginator = this.paginator;
        this.dataSource.sort = this.sort;
    });
}

updateuser(code: any) {
    this.OpenDialog('1000ms', '600ms', code);
}

OpenDialog(enteranimation: any, exitanimation: any, code: string)
{
    const popup = this.dialog.open(EmployeeCreateComponent, {
        enterAnimationDuration: enteranimation,
        exitAnimationDuration: exitanimation,
        width: '30%',
        data: {
            usercode: code
        }
    });
    popup.afterClosed().subscribe(res => {
        this.LoadUser();
    });
}

}

```

User.component.js

```
<mat-card>
  <mat-card-header>
    <h2>User Listing</h2>
  </mat-card-header>
  <mat-card-content>
    <div class="mat-elevation-z8">
      <table mat-table matSort [dataSource]="dataSource">

        <!-- Position Column -->
        <ng-container matColumnDef="username">
          <th mat-header-cell *matHeaderCellDef> Username
</th>
          <td mat-cell *matCellDef="let element">
{{element.id}} </td>
        </ng-container>

        <!-- Name Column -->
        <ng-container matColumnDef="name">
          <th mat-header-cell mat-sort-header
*matHeaderCellDef> Name </th>
          <td mat-cell *matCellDef="let element">
{{element.name}} </td>
        </ng-container>

        <!-- Weight Column -->
        <ng-container matColumnDef="email">
          <th mat-header-cell *matHeaderCellDef> Email </th>
          <td mat-cell *matCellDef="let element">
{{element.email}} </td>
        </ng-container>

        <!-- Symbol Column -->
        <ng-container matColumnDef="status">
          <th mat-header-cell *matHeaderCellDef> Status </th>
          <td mat-cell *matCellDef="let element">
{{element.isactive ? 'Active': 'In Active'}} </td>
        </ng-container>
        <ng-container matColumnDef="role">
          <th mat-header-cell *matHeaderCellDef> Role </th>
```

```

        <td mat-cell *matCellDef="let element">
{{element.role === ''? 'Un Assigned':element.role}} </td>
    </ng-container>
    <ng-container matColumnDef="action">
        <th mat-header-cell *matHeaderCellDef> Action </th>
        <td mat-cell *matCellDef="let element">
            <button *ngIf="element.role !== 'admin'"
(click)="updateuser(element.id)" mat-raised-button
color="primary">Update</button>
        </td>
    </ng-container>

    <tr mat-header-row
*matHeaderRowDef="displayedColumns"></tr>
    <tr mat-row *matRowDef="let row; columns:
displayedColumns;"></tr>
</table>

    <mat-paginator [pageSizeOptions]="[5, 10, 20]"
        showFirstLastButtons
        aria-label="Select page of periodic
elements">
    </mat-paginator>
</div>
</mat-card-content>
</mat-card>

```

Service

Inside the service folder

Auth-service.service

```

import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';

```



```

@Injectable({
  providedIn: 'root'
})
export class AuthService {

  constructor(private http:HttpClient) {

  }
  apiUrl='http://localhost:4200/user';

  RegisterUser(inputdata:any){
    return this.http.post(this.apiUrl,inputdata)
  }
  GetUserbyCode(id:any){
    return this.http.get(this.apiUrl+'/'+id);
  }
  Getall(){
    return this.http.get(this.apiUrl);
  }
  updateuser(id:any,inputdata:any){
    return this.http.put(this.apiUrl+'/'+id,inputdata);
  }
  getuserrole(){
    return this.http.get('http://localhost:4200/role');
  }
  isloggedin(){
    return sessionStorage.getItem('username')!=null;
  }
  getrole(){
    return
sessionStorage.getItem('role')!=null?sessionStorage.getItem('role')?
.toString():'';
  }
  GetAllCustomer(){
    return this.http.get('http://localhost:3000/customer');
  }
  Getaccessbyrole(role:any,menu:any){
    return
this.http.get('http://localhost:3000/roleaccess?role='+role+'&menu='
+menu)
  }
}

```

```
}
```

Inside model

There employee.ts

```
export class Employee {  
  name: string;  
  email: string;  
  designation: string;  
  phoneNumber: number;  
}
```

App-routing.module.ts

```
import { NgModule } from '@angular/core';  
import { RouterModule, Routes } from '@angular/router';  
import { EmployeeCreateComponent } from './components/employee-  
create/employee-create.component';  
import { AuthGuard } from './auth.guard';  
import { AppComponent } from './app.component';  
import { LoginComponent } from './components/login/login.component';  
import { UserComponent } from './components/user/user.component';  
import { EmployeeListComponent } from './components/employee-  
list/employee-list.component';  
import { EmployeeEditComponent } from './components/employee-  
edit/employee-edit.component';  
import { HomeComponent } from './components/home/home.component';  
import { ManageEmployeeComponent } from './components/manage-  
employee/manage-employee.component';
```

```

const routes: Routes = [
  {component: LoginComponent, path: 'login'},
  { path: '', redirectTo: '/hell', pathMatch: 'full' },
  {component: AppComponent, path: '', canActivate: [AuthGuard]},
  {component: UserComponent, path: 'user', canActivate: [AuthGuard]},
  {component: EmployeeCreateComponent, path: 'customer', canActivate: [AuthGuard]},
  {component: EmployeeEditComponent, path: 'edit-employee/:id', canActivate: [AuthGuard] },
  {component: EmployeeListComponent, path: 'manage', canActivate: [AuthGuard] },
  {component: HomeComponent, path: 'home', canActivate: [AuthGuard]},
  {component: ManageEmployeeComponent, path: 'hell', canActivate: [AuthGuard]}
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})

export class AppRoutingModule { }

```

app.component.html

```

<style>
  :host {
    font-family: -apple-system, BlinkMacSystemFont, "Segoe UI",
    Roboto, Helvetica, Arial, sans-serif, "Apple Color Emoji", "Segoe UI
    Emoji", "Segoe UI Symbol";
    font-size: 14px;
    color: #333;
    box-sizing: border-box;
    -webkit-font-smoothing: antialiased;
    -moz-osx-font-smoothing: grayscale;
  }

  h1,
  h2,

```

```
h3,  
h4,  
h5,  
h6 {  
    margin: 8px 0;  
}  
  
p {  
    margin: 0;  
}  
  
.spacer {  
    flex: 1;  
}  
  
.toolbar {  
    position: absolute;  
    top: 0;  
    left: 0;  
    right: 0;  
    height: 60px;  
    display: flex;  
    align-items: center;  
    background-color: #1976d2;  
    color: white;  
    font-weight: 600;  
}  
  
.toolbar img {  
    margin: 0 16px;  
}  
  
.toolbar #twitter-logo {  
    height: 40px;  
    margin: 0 8px;  
}  
  
.toolbar #youtube-logo {  
    height: 40px;  
    margin: 0 16px;  
}
```

```
.toolbar #twitter-logo:hover,  
.toolbar #youtube-logo:hover {  
  opacity: 0.8;  
}  
  
.content {  
  display: flex;  
  margin: 82px auto 32px;  
  padding: 0 16px;  
  max-width: 960px;  
  flex-direction: column;  
  align-items: center;  
}  
  
svg.material-icons {  
  height: 24px;  
  width: auto;  
}  
  
svg.material-icons:not(:last-child) {  
  margin-right: 8px;  
}  
  
.card svg.material-icons path {  
  fill: #888;  
}  
  
.card-container {  
  display: flex;  
  flex-wrap: wrap;  
  justify-content: center;  
  margin-top: 16px;  
}  
  
.card {  
  all: unset;  
  border-radius: 4px;  
  border: 1px solid #eee;  
  background-color: #fafafa;  
  height: 40px;  
  width: 200px;  
  margin: 0 8px 16px;
```

```

padding: 16px;
display: flex;
flex-direction: row;
justify-content: center;
align-items: center;
transition: all 0.2s ease-in-out;
line-height: 24px;
}

.card-container .card:not(:last-child) {
margin-right: 0;
}

.card.card-small {
height: 16px;
width: 168px;
}

.card-container .card:not(.highlight-card) {
cursor: pointer;
}

.card-container .card:not(.highlight-card):hover {
transform: translateY(-3px);
box-shadow: 0 4px 17px rgba(0, 0, 0, 0.35);
}

.card-container .card:not(.highlight-card):hover .material-icons
path {
fill: rgb(105, 103, 103);
}

.card.highlight-card {
background-color: #1976d2;
color: white;
font-weight: 600;
border: none;
width: auto;
min-width: 30%;
position: relative;
}

```

```
.card.card.highlight-card span {
  margin-left: 60px;
}

svg#rocket {
  width: 80px;
  position: absolute;
  left: -10px;
  top: -24px;
}

svg#rocket-smoke {
  height: calc(100vh - 95px);
  position: absolute;
  top: 10px;
  right: 180px;
  z-index: -10;
}

a,
a:visited,
a:hover {
  color: #1976d2;
  text-decoration: none;
}

a:hover {
  color: #125699;
}

.terminal {
  position: relative;
  width: 80%;
  max-width: 600px;
  border-radius: 6px;
  padding-top: 45px;
  margin-top: 8px;
  overflow: hidden;
  background-color: rgb(15, 15, 16);
}

.terminal::before {
```

```

    content: "\2022 \2022 \2022";
    position: absolute;
    top: 0;
    left: 0;
    height: 4px;
    background: rgb(58, 58, 58);
    color: #c2c3c4;
    width: 100%;
    font-size: 2rem;
    line-height: 0;
    padding: 14px 0;
    text-indent: 4px;
}

.terminal pre {
    font-family: SFMono-Regular,Consolas,Liberation
Mono,Menlo,monospace;
    color: white;
    padding: 0 1rem 1rem;
    margin: 0;
}

.circle-link {
    height: 40px;
    width: 40px;
    border-radius: 40px;
    margin: 8px;
    background-color: white;
    border: 1px solid #eeeeee;
    display: flex;
    justify-content: center;
    align-items: center;
    cursor: pointer;
    box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px rgba(0,
0, 0, 0.24);
    transition: 1s ease-out;
}

.circle-link:hover {
    transform: translateY(-0.25rem);
    box-shadow: 0px 3px 15px rgba(0, 0, 0, 0.2);
}

```



```

footer {
  margin-top: 8px;
  display: flex;
  align-items: center;
  line-height: 20px;
}

footer a {
  display: flex;
  align-items: center;
}

.github-star-badge {
  color: #24292e;
  display: flex;
  align-items: center;
  font-size: 12px;
  padding: 3px 10px;
  border: 1px solid rgba(27,31,35,.2);
  border-radius: 3px;
  background-image: linear-gradient(-180deg,#fafbfc,#eff3f6
90%);
  margin-left: 4px;
  font-weight: 600;
}

.github-star-badge:hover {
  background-image: linear-gradient(-180deg,#f0f3f6,#e6ebf1
90%);
  border-color: rgba(27,31,35,.35);
  background-position: -.5em;
}

.github-star-badge .material-icons {
  height: 16px;
  width: 16px;
  margin-right: 4px;
}

svg#clouds {
  position: fixed;

```

```

    bottom: -160px;
    left: -230px;
    z-index: -10;
    width: 1920px;
}

/* Responsive Styles */
@media screen and (max-width: 767px) {
    .card-container > *:not(.circle-link) ,
    .terminal {
        width: 100%;
    }

    .card:not(.highlight-card) {
        height: 16px;
        margin: 8px 0;
    }

    .card.highlight-card span {
        margin-left: 72px;
    }

    svg#rocket-smoke {
        right: 120px;
        transform: rotate(-5deg);
    }
}

@media screen and (max-width: 575px) {
    svg#rocket-smoke {
        display: none;
        visibility: hidden;
    }
}
</style>

<!-- Toolbar -->
<div class="toolbar" *ngIf="isMenuVisible" role="banner">
     |
Promise<boolean | UrlTree> | boolean | UrlTree {

    if (this.service.isloggedin()) {
      if (route.url.length > 0) {
        let menu = route.url[0].path;
        if (menu == 'user') {
          if (this.service.getrole() == 'admin') {
            return true;
          } else {
            this.router.navigate(['']);
            this.toastr.warning('You dont have access.')
            return false;
          }
        }
      }else{
        return true;
      }
    } else {

```

```

        return true;
    }
}
else {
    this.router.navigate(['login']);
    return false;
}
}
}

```

Assets where I upload the images

Package.json

```

{
  "name": "mean-stack-crud-app",
  "version": "0.0.0",
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
    "watch": "ng build --watch --configuration development",
    "test": "ng test"
  },
  "private": true,
  "dependencies": {
    "@angular/animations": "^16.1.0",
    "@angular/cdk": "^16.2.12",
    "@angular/common": "^16.1.0",
    "@angular/compiler": "^16.1.0",
    "@angular/core": "^16.1.0",
    "@angular/forms": "^16.1.0",
    "@angular/material": "^16.2.12",
    "@angular/platform-browser": "^16.1.0",
    "@angular/platform-browser-dynamic": "^16.1.0",
    "@angular/router": "^16.1.0",
    "@okta/okta-angular": "^6.3.1",

```

```

"@okta/okta-auth-js": "^7.5.0",
"bootstrap": "^5.3.0",
"ngx-toastr": "^18.0.0",
"primeng": "^17.0.0",
"rxjs": "~7.8.0",
"tslib": "^2.3.0",
"zone.js": "~0.13.0"
},
"devDependencies": {
"@angular-devkit/build-angular": "^16.1.4",
"@angular/cli": "~16.1.4",
"@angular/compiler-cli": "^16.1.0",
"@types/jasmine": "~4.3.0",
"jasmine-core": "~4.6.0",
"karma": "~6.4.0",
"karma-chrome-launcher": "~3.2.0",
"karma-coverage": "~2.2.0",
"karma-jasmine": "~5.1.0",
"karma-jasmine-html-reporter": "~2.1.0",
"typescript": "~5.1.3"
}
}

```

Screenshots :

Login



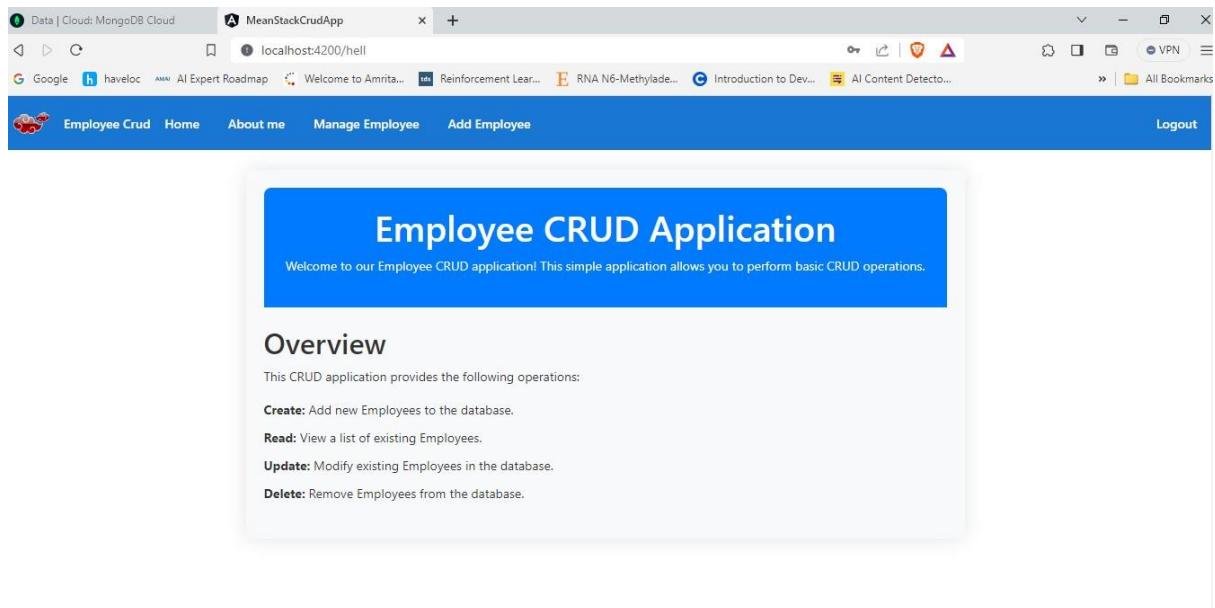
Admin Login

Username*

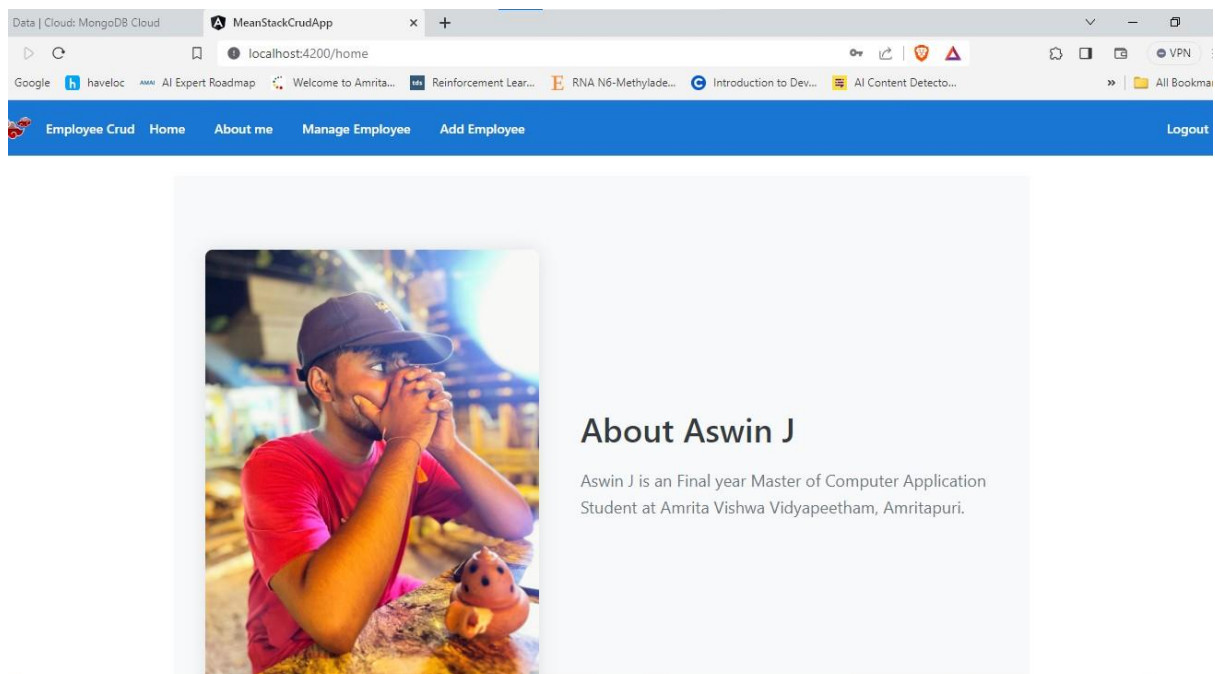
Password*

Login

Home



About me



Manage Employee

localhost:4200/manage

Employee Crud Home About me Manage Employee Add Employee Logout

Employee ID	Name	Email	Designation	Phone No	Update
65746f5fe94c8f2357876d25	Aswin J	aswinj2424@gmail.com	Finance	8301933187	<button>Edit</button> <button>Delete</button>
6574817a5cd04815ee44a0ae	Sudhi	sudhi@gmail.com	Sales	8992833287	<button>Edit</button> <button>Delete</button>

Employee Add

Data | Cloud: MongoDB Cloud MeanStackCrudApp x +

localhost:4200/customer

Employee Crud Home About me Manage Employee Add Employee Logout

Employee Register


Name

Email

Designation

Mobile No

Register



Employee Edit

Google haveloc AI Expert Roadmap Welcome to Amrita... Reinforcement Lear... RNA N6-Methylade... Introduction to Dev... AI Content Detecto... All Bookmarks

Employee Crud Home About me Manage Employee Add Employee Logout

Edit Employee

Name
Aswin J

Email
aswinj2424@gmail.com

Designation
Finance

Mobile No
8301933187

Update

Database created on MongoDB

cloud.mongodb.com/v2/656fe199b346343626370777#/metrics/replicaSet/656fe33b295a9d6d33450c... VPN

Google haveloc AI Expert Roadmap Welcome to Amrita... Reinforcement Lear... RNA N6-Methylade... Introduction to Dev... AI Content Detecto... All Bookmarks

Atlas Aswin's Org Access Manager Billing All Clusters Get Help Aswin

mean-stack-crud Data Services App Services Charts

Overview Search Namespaces

test employees

Database

Data Lake

SERVICES

Device Syno

Triggers

Data API

Data Federation

Atlas Search

Stream Processing

SECURITY

Backup

Database Access

Network Access

Advanced

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 245B TOTAL DOCUMENTS: 2 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema And Patterns Aggregation Search Indexes

Filter: Type a query: { field: 'value' } Reset Apply Options

INSERT DOCUMENT

QUERY RESULTS: 1-2 OF 2

```
{
  "_id": ObjectId("65746f5fe94c8f2357876d25"),
  "name": "Aswin J",
  "email": "aswinj2424@gmail.com",
  "designation": "Finance",
  "phoneNumber": 8301933187,
  "__v": 0
}
```

```
{
  "_id": ObjectId("6574817a5cd9815ee44a8ae"),
  "name": "Sudhi",
  "email": "sudhi@gmail.com",
  "designation": "Sales",
  "phoneNumber": 8992833287,
  "__v": 0
}
```

Aswin J AM.EN.P2MCA22062