Models/entry-schema.js  
const mongoose = require('mongoose');

const entrySchema = mongoose.Schema({

    date: String,

    entry: String

});

module.exports = mongoose.model('DiaryEntry', entrySchema);

models/rest.js  
const express = require('express');

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

const DiaryEntryModel = require('./entry-schema');

const mongoose = require('mongoose');

const UserModel = require('./user-model');

const bcrypt = require('bcrypt');

const app = express();

mongoose.connect("mongodb://127.0.0.1:27017/login")

    .then(() => {

        console.log('Connected to MongoDB')

    })

    .catch(() => {

        console.log('Error connecting to MongoDB');

    })

app.use(bodyParser.json());

app.use((req, res, next) => {

    res.setHeader('Access-Control-Allow-Origin', '\*');

    res.setHeader('Access-Control-Allow-Headers', 'Origin, X-Requested-With, Content-Type, Accept');

    res.setHeader('Access-Control-Allow-Methods', 'GET, POST, PUT, DELETE, OPTIONS');

    next();

})

app.delete('/remove-entry/:id', (req, res) => {

    DiaryEntryModel.deleteOne({\_id: req.params.id})

    .then(() => {

        res.status(200).json({

            message: 'Post Deleted'

        })

    })

})

app.put('/update-entry/:id', (req, res) => {

    const updatedEntry = new DiaryEntryModel({\_id: req.body.id, date: req.body.date, entry: req.body.entry})

    DiaryEntryModel.updateOne({\_id: req.body.id}, updatedEntry)

        .then(() => {

            res.status(200).json({

                message: 'Update completed'

            })

        })

})

app.post('/add-entry', (req,res) => {

    const diaryEntry = new DiaryEntryModel({date: req.body.date, entry: req.body.entry});

    diaryEntry.save()

        .then(() => {

            res.status(200).json({

                message: 'Post submitted'

            })

        })

})

app.get('/diary-entries',(req, res, next) => {

    DiaryEntryModel.find()

    .then((data) => {

        res.json({'diaryEntries': data});

    })

    .catch(() => {

        console.log('Error fetching entries')

    })

})

app.post('/sign-up', (req,res) => {

    bcrypt.hash(req.body.password, 10)

        .then(hash => {

            const userModel = new UserModel({

                username: req.body.username,

                password: hash

            })

            userModel.save()

            .then(result => {

                res.status(201).json({

                    message: 'User created',

                    result: result

                })

            })

            .catch(err => {

                res.status(500).json({

                    error: err

                })

            })

        })

})

module.exports = app;

models/server.js

const http = require('http');

const express = require('./rest.js');

const server = http.createServer(express)

server.listen(3000);

models/user-model.js

const mongoose = require('mongoose');

const uniqueValidator = require('mongoose-unique-validator');

const userSchema = mongoose.Schema({

    username: {type: String, required: true, unique: true},

    password: {type: String, required: true}

});

userSchema.plugin(uniqueValidator);

module.exports = mongoose.model("UserModel", userSchema);

signup.html  
<div class="row">

    <div class="col">

        <form (ngSubmit)="onSubmit()" [formGroup]='signupForm'>

            <div class="form-group">

                <label for="username">Username</label>

                <input class="form-control" formControlName="username" type="text">

                <label for="password">Password</label>

                <input class="form-control" formControlName="password" type="password">

            </div>

            <button [disabled]="!signupForm.valid" type="submit" class="btn btn-success">Sign Up</button>

        </form>

    </div>

</div>

Sign-up.ts  
import { Component, OnInit } from '@angular/core';

import { FormControl, FormGroup, Validators } from '@angular/forms';

import { AuthService } from '../shared/auth-service';

@Component({

  selector: 'app-sign-up',

  templateUrl: './sign-up.component.html',

  styleUrls: ['./sign-up.component.css']

})

export class SignUpComponent implements OnInit {

  signupForm!: FormGroup;

  constructor(private authService: AuthService) { }

  ngOnInit(): void {

    this.signupForm = new FormGroup({

      'username': new FormControl('', [Validators.required]),

      'password': new FormControl('', [Validators.required])

    })

  }

  onSubmit(){

    this.authService.signupUser(this.signupForm.value.username, this.signupForm.value.password);

  }

}

Shared/auth.ts

export interface AuthModel{

    username: string;

    password: string;

}

Shared/auth-service.ts  
import { HttpClient } from "@angular/common/http";

import { Injectable } from "@angular/core";

import { AuthModel } from "./auth-model";

@Injectable({providedIn:"root"})

export class AuthService{

    constructor(private http: HttpClient){}

    signupUser(username: string, password: string){

        const authData: AuthModel = {username: username, password: password};

        this.http.post('http://localhost:3000/sign-up/', authData).subscribe(res => {

            console.log(res);

        })

    }

}

Shared/dairy-entry-model.ts  
export class DiaryEntry{

    public constructor(public id: string, public date: string, public entry: string){}

}

Diary-data.component.ts  
import { HttpClient } from "@angular/common/http";

import { Injectable } from "@angular/core";

import { map, Subject } from "rxjs";

import { DiaryEntry } from "./dairy-entry.model";

@Injectable({providedIn:"root"})

export class DiaryDataService{

    public maxId!: number;

    constructor(private http: HttpClient){}

    updateEntry(id: string, entry: DiaryEntry) {

      this.http.put<{message: string}>('http://localhost:3000/update-entry/' + id, entry).subscribe((jsonData) => {

        console.log(jsonData.message);

        this.getDiaryEntries();

      })

    }

    public diarySubject = new Subject<DiaryEntry[]>();

    private diaryEntries: DiaryEntry[] = [];

    onDeleteEntry(id: string){

        this.http.delete<{message: string}>('http://localhost:3000/remove-entry/' + id).subscribe((jsonData) => {

        console.log(jsonData.message);

        this.getDiaryEntries();

        })

    }

    getDiaryEntries(){

        this.http.get<{diaryEntries: any}>('http://localhost:3000/diary-entries')

        .pipe(map((responseData) => {

            return responseData.diaryEntries.map((entry: {date: string; entry: string; \_id: string}) => {

                return {

                    date: entry.date,

                    entry: entry.entry,

                    id: entry.\_id

                }

            })

        }))

        .subscribe((updateResponse) => {

            this.diaryEntries = updateResponse;

            this.diarySubject.next(this.diaryEntries);

        })

    }

    getDiaryEntry(id: string){

        const index = this.diaryEntries.findIndex(el => {

            return el.id == id;

        })

        return this.diaryEntries[index];

    }

    onAddDiaryEntry(diaryEntry: DiaryEntry){

     this.http.post<{message: string}>('http://localhost:3000/add-entry', diaryEntry).subscribe((jsonData) => {

                console.log(diaryEntry);

                this.getDiaryEntries();

            })

    }

}

Diary-form.html  
<div class="row">

    <div class="col">

        <form (ngSubmit)="onSubmit()" [formGroup]='diaryForm'>

            <div class="form-group">

                <label for="date">Date</label>

                <input class="form-control" formControlName="date" type="text">

                <label for="entry">Entry</label>

                <input class="form-control" formControlName="entry" type="text">

            </div>

            <button [disabled]="!diaryForm.valid" type="submit" class="btn btn-success">Submit</button>

        </form>

    </div>

</div>

diary-form.component.html  
import { Component, OnInit } from '@angular/core';

import { FormControl, FormGroup, Validators } from '@angular/forms';

import { ActivatedRoute, ParamMap, Router } from '@angular/router';

import { DiaryEntry } from '../shared/dairy-entry.model';

import { DiaryDataService } from '../shared/diary-data.component';

@Component({

  selector: 'app-diary-form',

  templateUrl: './diary-form.component.html',

  styleUrls: ['./diary-form.component.css']

})

export class DiaryFormComponent implements OnInit {

  editMode = false;

  private paramId: string = '';

  diaryEntry!: DiaryEntry;

  diaryForm! : FormGroup;

  constructor(private diaryDataService: DiaryDataService, private router: Router, private activatedRoute: ActivatedRoute) { }

  ngOnInit(): void {

    this.activatedRoute.paramMap.subscribe((paramMap: ParamMap) => {

      if(paramMap.has('id')){

        this.editMode = true;

        this.paramId = paramMap.get('id')!;

        this.diaryEntry = this.diaryDataService.getDiaryEntry(this.paramId);

      }

      else{

        this.editMode = false;

      }

    })

    this.diaryForm = new FormGroup({

      'date': new FormControl(this.editMode ? this.diaryEntry.date : '', [Validators.required]),

      'entry': new FormControl(this.editMode ? this.diaryEntry.entry : '', [Validators.required])

    })

  }

  onSubmit(){

    const entry = new DiaryEntry('', this.diaryForm.value.date, this.diaryForm.value.entry);

    if(this.editMode){

      entry.id = this.paramId;

      this.diaryDataService.updateEntry(this.paramId, entry);

    }

    else{

      this.diaryDataService.onAddDiaryEntry(entry);

    }

    this.router.navigateByUrl("");

  }

}

Diary.html  
<div class="row">

    <div class="col">

        <h2>Welcome to the Diary</h2>

        <table class="table table-bordered">

            <thead>

            <tr>

                <th>Date</th>

                <th>Entry</th>

                <th>Action</th>

            </tr>

            </thead>

            <tbody>

                <tr \*ngFor='let diary of diaryEntries; let i = index'>

                    <td>{{diary.date}}</td>

                    <td>{{diary.entry}}</td>

                    <td>

                        <button class="btn btn-primary" (click)="onEdit(diary.id)">Edit</button>

                        <button class="btn btn-danger" (click)='onDelete(diary.id)'>Delete</button>

                    </td>

                </tr>

            </tbody>

        </table>

    </div>

</div>

Diary.ts  
import { Component, OnDestroy, OnInit } from '@angular/core';

import { Router } from '@angular/router';

import { Subscription } from 'rxjs';

import { DiaryEntry } from '../shared/dairy-entry.model';

import { DiaryDataService } from '../shared/diary-data.component';

@Component({

  selector: 'app-diary',

  templateUrl: './diary.component.html',

  styleUrls: ['./diary.component.css']

})

export class DiaryComponent implements OnInit, OnDestroy {

  constructor(private diaryDataService: DiaryDataService, private router: Router) { }

  ngOnDestroy(): void {

    this.diaryEntriesSub.unsubscribe();

  }

  diaryEntries: DiaryEntry[] = [];

  diaryEntriesSub = new Subscription();

  ngOnInit(): void {

    this.diaryDataService.getDiaryEntries();

    this.diaryEntriesSub = this.diaryDataService.diarySubject.subscribe(entries => {

      this.diaryEntries = entries;

    })

  }

  onDelete(id: string){

    this.diaryDataService.onDeleteEntry(id);

  }

  onEdit(id: string){

    this.router.navigate(['edit',id]);

  }

}