**Get all BlogPost -API**

**IBlogpostRepository**

Task<IEnumerable<BlogPost>> GetAllAsync();

**BlogPostRepository**

using codepulseapi.API.Data;

using codepulseapi.API.Models.Domain;

using codepulseapi.API.Repositories.Interface;

namespace codepulseapi.API.Repositories.Repository

{

    public class BlogPostRepository : IBlogPostRepository

    {

        private readonly ApplicationDbContext dbContext;

        public BlogPostRepository(ApplicationDbContext dbContext)

        {

            this.dbContext = dbContext;

        }

        public async Task<IEnumerable<BlogPost>> GetAllAsync()

        {

            return await dbContext.BlogPosts.Include(x => x.Categories).ToListAsync();

        }

    }

}

**BlogPostController**

using codepulseapi.API.Models.Domain;

using codepulseapi.API.Models.DTO;

using codepulseapi.API.Repositories.Interface;

using codepulseapi.API.Repositories.Repository;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.IdentityModel.Tokens;

namespace codepulseapi.API.Controllers

{

    [Route("api/[controller]")]

    [ApiController]

    public class BlogPostsController : ControllerBase

    {

        private readonly IBlogPostRepository blogPostRepository;

        public BlogPostsController(IBlogPostRepository blogPostRepository)

        {

            this.blogPostRepository = blogPostRepository;

        }

[HttpGet]

        public async Task<IActionResult> GetAllBlogPosts()

        {

           var blogPosts = await blogPostRepository.GetAllAsync();

           var response = new List<BlogPostDto>();

           foreach (var blogPost in blogPosts)

           {

            response.Add(new BlogPostDto

            {

                Id = blogPost.Id,

                Title = blogPost.Title,

                ShortDescription = blogPost.ShortDescription,

                Content = blogPost.Content,

                FeaturedImageUrl = blogPost.FeaturedImageUrl,

                UrlHandle = blogPost.UrlHandle,

                PublishedDate = blogPost.PublishedDate,

                Author = blogPost.Author,

                IsVisible = blogPost.IsVisible,Categories = blogPost.Categories.Select(x=> new CategoryDto

                {

                    Id = x.Id,

                    Name = x.Name,

                    UrlHandle = x.UrlHandle

                }).ToList()

            });

           }

           return Ok(response);

        }

**Add BlogPost -UI**

getAllBlogPosts() : Observable<BlogPost[]>{

    return this.http.get<BlogPost[]>('http://localhost:5125/api/blogposts');

  }

**Add Blog Post.component.ts**

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

import { AddBlogPost } from '../models/add-blog-post.model';

@Component({

  selector: 'app-add-blogpost',

  templateUrl: './add-blogpost.component.html',

  styleUrls: ['./add-blogpost.component.css']

})

export class AddBlogpostComponent {

  model: AddBlogPost;

  constructor() {

    this.model ={

      title: '',

      shortDescription: '',

      content:'',

      featuredImageUrl:'',

      urlHandle: '',

      author: '',

      isVisible: true,

      publishedDate: new Date()

    }

  }

   ngOnInit(): void {

   this.categories$= this.categoryService.getAllCategories();

  }

  onFormSubmit(): void {

    console.log(this.model);

    this.blogPostService.createBlogPost(this.model)

      .subscribe({

        next: (response)=> {

          this.router.navigateByUrl('/admin/blogposts');

        }

      });

  }

}

**AddBlogPost.component.html**

div class="container">

    <h1 class="mt-3"> Add BlogPost</h1>

    <form #form="ngForm" (ngSubmit)="onFormSubmit()">

        <div class="mt-2 mb-5">

            <div class="mt-3">

                <label for="title" class="form-label"> Title </label>

                <input type="text"

                id="title"

                name="title"

                class="form-control"

                [(ngModel)]="model.title">

            </div>

            <div class="mt-3">

                <label for="urlHandle" class="form-label"> Url Handle </label>

                <input type="text"

                id="urlHandle"

                name="urlHandle"

                class="form-control"

                [(ngModel)]="model.urlHandle">

            </div>

            <div class="mt-3">

                <label for="shortDescription" class="form-label"> Short Description </label>

                <input type="text"

                id="shortDescription"

                name="shortDescription"

                class="form-control"

                [(ngModel)]="model.shortDescription">

            </div>

            <div class="mt-3">

                <label for="content" class="form-label"> Content </label>

                <textarea name="content" id="content"

                class="form-control"

                [(ngModel)]="model.content"></textarea>

            </div>

            <div class="mt-3">

                <label for="featuredImageUrl" class="form-label"> Featured Image Url </label>

                <input type="text"

                id="featuredImageUrl"

                name="featuredImageUrl"

                class="form-control"

                [(ngModel)]="model.featuredImageUrl">

            </div>

            <div class="mt-3">

                <label for="publishedDate" class="form-label"> Published Date </label>

                <input type="date"

                id="publishedDate"

                name="publishedDate"

                class="form-control"

                value="{{model.publishedDate}}"

                [ngModel]="model.publishedDate | date: 'yyyy-MM-dd' "

                (ngModelChange)="model.publishedDate = $event">

            </div>

            <div class="mt-3">

                <label for="author" class="form-label"> Author </label>

                <input type="text"

                id="author"

                name="author"

                class="form-control"

                [(ngModel)]="model.author">

            </div>

            <div class="form-check mt-3">

                <input class="form-check-input" type="checkbox"  id="isVisible"

                [(ngModel)]="model.isVisible"

                name="isVisible">

                <label class="form-check-label" for="flexCheckDefault">

                  Is Visible?

                </label>

              </div>

              <div class="mt-3">

                <button type="submit" class="btn btn-primary"> Save </button>

              </div>

        </div>

    </form>

</div>

**Live Mark down preview**

 <div class="mt-3">

                <label for="content" class="form-label"> Content </label>

                <div class="d-flex flex-row">

                    <div class="col-6 p-2">

                        <textarea

                        id="content"

                        name="content"

                        class="form-control"

                        style="height: 350px; max-height: 350px;"

                        [(ngModel)]="model.content">

                    </textarea>

                    </div>

                    <div class="col-6 p-2" style="max-height: 350px; overflow-y: scroll ;">

                      <markdown [data]="model.content"></markdown>

                    </div>

                </div>

            </div>

**Image preview**

 <div class="mt-3">

                <label for="featuredImageUrl" class="form-label"> Featured Image Url </label>

                 <ng-container \*ngIf="model.featuredImageUrl !== ''">

                    <div>

                        <img [src] = "model.featuredImageUrl" height="100" class="my-2">

                    </div>

                </ng-container>

                <input type="text"

                id="featuredImageUrl"

                name="featuredImageUrl"

                class="form-control"

                [(ngModel)]="model.featuredImageUrl">

            </div>