


**AKDENİZ ÜNİVERSİTESİ**  
**İTERNET PROGRAMCILIĞI 2 FİNAL PROJESİ**

Öğrenci No	20201132024
Adı Soyadı	AHMET CAN KARTAL
Proje Adı	<b>SORU CEVAP BLOGU HAZIRLAMAK</b>
Github Link	<a href="https://github.com/ahmetcankartal/INTERNETPROGRAMCILIGI2-FINAL/">https://github.com/ahmetcankartal/INTERNETPROGRAMCILIGI2-FINAL/</a>
Youtube Link	<a href="https://youtu.be/oD9np3hCzdA">https://youtu.be/oD9np3hCzdA</a>

# RESİMLER

[Find Any Answer](#) [Anasayfa](#) [hakkında](#) [iletişim](#) [yönetim paneline giriş](#)  [Search](#)



**SORU 1**  
SORU ÖZET 1


01.01.0001

Asp.Net

1

0

devamını oku



**SORu Cevap 34343**  
makale özet 1adasddd


01.01.0001

Spor

2

1

devamını oku



**BASLIKBASLIK**

**Kategori**

[Asp.Net](#)  
[Angular](#)  
[Spor](#)  
[Sanat](#)

**En çok okunanlar**

[ahmet can kartal 3](#)  
[BASLIKBASLIK](#)  
[ahmet can kartal 5](#)  
[dasdasdasdasdaas 7](#)

Her Sorunuza Cevap Bulun.



asdasdasdasdad

01.01.0001 Asp.Net 53 0

devamını oku



dasdadasdasddaas 7

asdasdasdasdad

01.01.0001 Spor 7 0

devamını oku



ahmet can kartal 3

asdasdasdasdad

01.01.0001 Asp.Net 687 0

devamını oku

dasdadasdasddaas 7

ahmet can kartal 6

#### Arşiv

- 2022 Nisan(1)
- 2017 Mayıs(1)
- 2018 Mayıs(1)
- 2021 Mayıs(1)
- 2022 Mayıs(2)
- 2022 Haziran(1)

## İletişim

isminiz

email adresiniz

konu

mesajınız

Gönder

### Kategori

- ➔ Asp.Net
- ➔ Angular
- ➔ Spor
- ➔ Sanat

### En çok okunanlar

- ➔ ahmet can kartal 3
- ➔ BASLIKBASLIK
- ➔ ahmet can kartal 5
- ➔ dasdasdasdasdaas 7
- ➔ ahmet can kartal 6

## Yönetim paneline hoş

SORU EKLE

SORULARI LİSTELE










kategori seç


kategori alanı boş bırakılamaz

## SORU 1

**SORU ÖZET 1**

Paragraph

AI A<sup>2</sup> B I U S         



**SORUNA RESİM EKLE**



Dosya Seç

ç Dosy...ilmedi

Resim	Başlık	Kategori	yorum sayısı	Gösterim sayısı	Tarih	işlemler
	SORU 1	Asp.Net	0	1	01.06.2022	<a href="#">güncelle</a> <a href="#">sil</a>
	SOru Cevap 34343	Spor	1	2	31.05.2022	<a href="#">güncelle</a> <a href="#">sil</a>
	BASLIKBASLIK	Asp.Net	0	53	30.05.2022	<a href="#">güncelle</a> <a href="#">sil</a>
	dasdasdasdasddaas 7	Spor	0	7	30.04.2022	<a href="#">güncelle</a> <a href="#">sil</a>
	ahmet can kartal 3	Asp.Net	0	687	30.05.2021	<a href="#">güncelle</a> <a href="#">sil</a>

Items per page: 5

1 - 5 of 7

&lt; &lt; &gt; &gt;

makale özet 1adasddd

Paragraph

AI

A<sup>2</sup>

B

I

↩

asdasdsadasddasdasdasd

SORU GÜNCELLE



Dosya Seç

Dosy...ilmedi



## Admin Panel Giriř

email

ahmet@mail.com

řifre

\*\*\*\*

giriř

### Kategori

- ➔ Asp.Net
- ➔ Angular
- ➔ Spor
- ➔ Sanat

### En çok okunanlar

- ahmet can kartal 3
- BASLIKBASLIK
- ahmet can kartal 5
- dasdasdasdasddaas 7
- ahmet can kartal 6

# API KODLAMALARI

ARTICLESCONTROLLER.  
CS

```
using System;

using System.Collections.Generic;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using sorucevapblog.API.Models;

using sorucevapblog.API.Responses;


namespace sorucevapblog.API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ArticlesController : ControllerBase
    {
        private readonly sorucevapblogdbContext _context;

        public ArticlesController(sorucevapblogdbContext context)
        {
            _context = context;
        }

        // GET: api/Articles
        [HttpGet]
        public IActionResult GetArticle()
        {
            var articles =
                _context.Article.Include(a=>a.Category).Include(b=>b.Comment).OrderByDesce
```

```

        nding(x => x.PublishDate).ToList().Select(y=>new ArticleResponse()
        {
            Id=y.Id,
            Title=y.Title,
            Picture=y.Picture,
            Category=new CategoryResponse() {
Id=y.Category.Id,Name=y.Category.Name},
            CommentCount=y.Comment.Count,
            ViewCount=y.ViewCount,
            PublishDate=y.PublishDate

        });
        return Ok(articles);
    }

```

```

[HttpGet("{page}/{pageSize}")]
public IActionResult GetArticle(int page=1,int pageSize=5)
{
    System.Threading.Thread.Sleep(3000);
    try
    {
        IQueryable<Article> query;

        query = _context.Article.Include(x => x.Category).Include(y =>
y.Comment).OrderByDescending(z => z.PublishDate);

        int totalCount = query.Count();

        var articlesResponse = query.Skip((pageSize * (page -
1))).Take(5).ToList().Select(x => new ArticleResponse()
        {

            Id = x.Id,

```

```

        Title = x.Title,

        ContentMain = x.ContentMain,

        ContentSummary = x.ContentSummary,

        Picture = x.Picture,

        ViewCount = x.ViewCount,

        CommentCount = x.Comment.Count,

        Category = new CategoryResponse() { Id = x.Category.Id, Name =
x.Category.Name }

    });

    var result = new
    {
        TotalCount = totalCount,
        Articles = articlesResponse
    };

    return Ok(result);
}

catch(SystemException ex)
{
    return BadRequest(ex.Message);
}

}

[HttpGet]

[Route("GetArticlesWithCategory/{categoryId}/{page}/{pageSize}")]

public IActionResult GetArticlesWithCategory(int categoryId,int page=1,int
pageSize=5)

```

```

{

    IQueryable<Article> query = _context.Article.Include(x =>
x.Category).Include(y => y.Comment).Where(z => z.CategoryId ==
categoryId).OrderByDescending(x => x.PublishDate);

    var queryResult = ArticlesPagination(query, page, pageSize);

    var result = new
    {
        TotalCount = queryResult.Item2,
        Articles = queryResult.Item1
    };
    return Ok(result);

}

[HttpGet]
[Route("SearchArticles/{searchText}/{page}/{pageSize}")]
public IActionResult SearchArticles(string searchText,int page=1,int
pageSize=5)
{
    IQueryable<Article> query;

    query = _context.Article.Include(x => x.Category).Include(y =>
y.Comment).Where(z => z.Title.Contains(searchText)).OrderByDescending(f =>
f.PublishDate);

    var resultQuery = ArticlesPagination(query, page, pageSize);

    var result = new
    {
        Articles = resultQuery.Item1,
        TotalCount = resultQuery.Item2
    };

    return Ok(result);
}

```

```
}
```

```
[HttpGet]
```

```
[Route("GetArticlesByMostView")]
```

```
public IActionResult GetArticlesByMostView()
```

```
{
```

```
    System.Threading.Thread.Sleep(2000);
```

```
    var articles = _context.Article.OrderByDescending(x =>  
x.ViewCount).Take(5).Select(x => new ArticleResponse()
```

```
    {
```

```
        Title=x.Title,
```

```
        Id=x.Id,
```

```
    });
```

```
    return Ok(articles);
```

```
}
```

```
[HttpGet]
```

```
[Route("GetArticlesArchive")]
```

```
public IActionResult GetArticlesArchive()
```

```
{
```

```
    System.Threading.Thread.Sleep(1000);
```

```
    var query = _context.Article.GroupBy(x => new { x.PublishDate.Year,  
x.PublishDate.Month }).Select(y =>
```

```
        new {
```

```
            year = y.Key.Year,
```

```
            month = y.Key.Month,
```

```
            count = y.Count(),
```

```
            monthName = new DateTime(y.Key.Year, y.Key.Month,  
1).ToString("MMMM", CultureInfo.CreateSpecificCulture("tr"))
```

```
        });
```

```
return Ok(query);
```

```
}
```

```
[HttpGet]
```

```
[Route("GetArticleArchiveList/{year}/{month}/{page}/{pageSize}")]
```

```
public IActionResult GetArticleArchiveList(int year,int month,int page,int  
pageSize)
```

```
{
```

```
    System.Threading.Thread.Sleep(1700);
```

```
    IQueryable<Article> query;
```

```
    query = _context.Article.Include(x => x.Category).Include(y =>  
y.Comment).Where(z => z.PublishDate.Year == year && z.PublishDate.Month  
== month).OrderByDescending(f => f.PublishDate);
```

```
    var resultQuery = ArticlesPagination(query, page, pageSize);
```

```
    var result = new
```

```
{
```

```
        Articles = resultQuery.Item1,
```

```
        TotalCount = resultQuery.Item2
```

```
};
```

```
return Ok(result);
```

```
}
```

```
// GET: api/Articles/5
```

```
[HttpGet("{id}")]
```

```
public IActionResult GetArticle([FromRoute] int id)
```

```
{

    System.Threading.Thread.Sleep(2000);

    var article = _context.Article.Include(x => x.Category).Include(y =>
y.Comment).FirstOrDefault(z => z.Id == id);

    if(article==null)

    {

        return NotFound();

    }

    ArticleResponse articleResponse = new ArticleResponse()

    {

        Id = article.Id,

        Title = article.Title,

        ContentMain = article.ContentMain,

        ContentSummary = article.ContentSummary,

        Picture = article.Picture,

        PublishDate = article.PublishDate,

        ViewCount = article.ViewCount,

        Category = new CategoryResponse() { Id = article.Category.Id, Name
= article.Category.Name },

        CommentCount = article.Comment.Count

    };

    return Ok(articleResponse);

}

// PUT: api/Articles/5
```



```
[HttpPut("{id}")]
```

```
public async Task<ActionResult> PutArticle( int id, Article article)
```

```
{
```

```
    Article firstArticle = _context.Article.Find(id);
```

```
    firstArticle.Title = article.Title;
```

```
    firstArticle.ContentSummary = article.ContentSummary;
```

```
    firstArticle.ContentMain = article.ContentMain;
```

```
    firstArticle.CategoryId = article.Category.Id;
```

```
    firstArticle.Picture = article.Picture;
```

```
    try
```

```
    {
```

```
        await _context.SaveChangesAsync();
```

```
    }
```

```
    catch (DbUpdateConcurrencyException)
```

```
    {
```

```
        if (!ArticleExists(id))
```

```
        {
```

```
            return NotFound();
```

```
        }
```

```
    else
```

```
    {
```

```
        throw;
```

```
    }

    }

    return NoContent();
}

// POST: api/Articles
[HttpPost]
public async Task<IActionResult> PostArticle( Article article)
{
    if(article.Category !=null)
    {
        article.CategoryId = article.Category.Id;
    }
    article.Category = null;
    article.ViewCount = 0;
    article.PublishDate = DateTime.Now;

    _context.Article.Add(article);
    await _context.SaveChangesAsync();
    return Ok();
}

// DELETE: api/Articles/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteArticle([FromRoute] int id)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
}
```

```
var article = await _context.Article.FindAsync(id);

if (article == null)
{
    return NotFound();
}

_context.Article.Remove(article);
await _context.SaveChangesAsync();

return Ok(article);
}
```

```
private bool ArticleExists(int id)
{
    return _context.Article.Any(e => e.Id == id);
}
```

```
[Route("ArticleViewCountUp/{id}")]
[HttpGet()]
```

```
public IActionResult ArticleViewCountUp(int id)
{
    Article article = _context.Article.Find(id);
    article.ViewCount += 1;
    _context.SaveChanges();
    return Ok();
}
```

```
public System.Tuple<IEnumerable<ArticleResponse>, int>
```

```
ArticlesPagination(IQueryable<Article> query, int page, int pageSize)

{

    int totalCount = query.Count();

    var articlesResponse = query.Skip((pageSize * (page - 1))).Take(5).ToList().Select(x => new ArticleResponse()

    {

        Id = x.Id,

        Title = x.Title,

        ContentMain = x.ContentMain,

        ContentSummary = x.ContentSummary,

        Picture = x.Picture,

        ViewCount = x.ViewCount,

        CommentCount = x.Comment.Count,

        Category = new CategoryResponse() { Id = x.Category.Id, Name = x.Category.Name }

    });

    return new System.Tuple<IEnumerable<ArticleResponse>, int>(articlesResponse, totalCount);

}

[HttpPost]

[Route("SaveArticlePicture")]
```

```
public async Task<IActionResult> SaveArticlePicture(IFormFile picture)
{
    var fileName = Guid.NewGuid().ToString() +
Path.GetExtension(picture.FileName);

    var path = Path.Combine(Directory.GetCurrentDirectory(),
"wwwroot/articlePictures", fileName);

    using (var stream = new FileStream(path, FileMode.Create))
    {
        await picture.CopyToAsync(stream);
    };

    var result = new
    {
        path = "https://" + Request.Host + "/articlePictures/" + fileName
    };

    return Ok(result);
}
}
```

## COMMENTS\_CONTROLLER.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using sorucevapblog.API.Models;

namespace sorucevapblog.API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class CommentsController :
ControllerBase
    {
        private readonly sorucevapblogdbContext
_context;

        public
CommentsController(sorucevapblogdbContext context)
        {
            _context = context;
        }

        // GET: api/Comments
        [HttpGet]
        public IEnumerable<Comment> GetComment()
        {
            return _context.Comment;
        }

        // GET: api/Comments/5
        [HttpGet("{id}")]
        public IActionResult GetCommentList( int
id)
        {
            var comments =
_context.Comment.Where(a => a.ArticleId==
id).ToList();
            if(comments== null)
            {
                return NotFound();
            }
            return Ok(comments);
        }

        // PUT: api/Comments/5
        [HttpPut("{id}")]
        public async Task<IActionResult>
PutComment([FromRoute] int id, [FromBody] Comment
comment)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            if (id != comment.Id)
            {
                return BadRequest();
            }

            _context.Entry(comment).State =
EntityType.Modified;
        }
    }
}
```

```

        try
        {
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!CommentExists(id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return NoContent();
    }

    // POST: api/Comments
    [HttpPost]
    public async Task<IActionResult>
    PostComment([FromBody] Comment comment)
    {
        System.Threading.Thread.Sleep(2500);
        comment.PublishDate = System.DateTime.Now;

        _context.Comment.Add(comment);
        await _context.SaveChangesAsync();

        return CreatedAtAction("GetComment",
new { id = comment.Id }, comment);
    }

    // DELETE: api/Comments/5
    [HttpDelete("{id}")]
    public async Task<IActionResult>
    DeleteComment([FromRoute] int id)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        var comment = await
_context.Comment.FindAsync(id);
        if (comment == null)
        {
            return NotFound();
        }

        _context.Comment.Remove(comment);
        await _context.SaveChangesAsync();

        return Ok(comment);
    }

    private bool CommentExists(int id)
    {
        return _context.Comment.Any(e => e.Id
== id);
    }
}

```

## AUTHCONTROLLER.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using sorucevapblog.API.Models;

namespace sorucevapblog.API.Controllers
{
    [Route("api/[controller]/[action]")]
    [ApiController]
    public class AuthController : ControllerBase
    {
        [HttpPost]
        public IActionResult
IsAuthenticated(AdminUser adminUser)
        {
            bool status = false;

            if(adminUser.Email=="ahmet@mail.com"&&
adminUser.Password=="1234")
            {
                status = true;
            }

            var result = new
            {
                status = status
            };
            return Ok(result);
        }
    }
}
```



## CATEGORIESCONTROLLER.CS

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using sorucevapblog.API.Models;

namespace sorucevapblog.API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class CategoriesController :
ControllerBase
    {
        private readonly sorucevapblogdbContext
_context;

        public
CategoriesController(sorucevapblogdbContext
context)
        {
            _context = context;
        }

        // GET: api/Categories
        [HttpGet]
        public IEnumerable<Category> GetCategory()
        {
            System.Threading.Thread.Sleep(3000);

            return _context.Category;
        }

        // GET: api/Categories/5
        [HttpGet("{id}")]
        public async Task<IActionResult>
GetCategory([FromRoute] int id)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            var category = await
_context.Category.FindAsync(id);

            if (category == null)
            {
                return NotFound();
            }

            return Ok(category);
        }

        // PUT: api/Categories/5
        [HttpPut("{id}")]
        public async Task<IActionResult>
PutCategory([FromRoute] int id, [FromBody]
Category category)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

```

```

        if (id != category.Id)
        {
            return BadRequest();
        }

        _context.Entry(category).State =
EntityState.Modified;

        try
        {
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!CategoryExists(id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return NoContent();
    }

    // POST: api/Categories
    [HttpPost]
    public async Task<IActionResult>
PostCategory([FromBody] Category category)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        _context.Category.Add(category);
        await _context.SaveChangesAsync();

        return CreatedAtAction("GetCategory",
new { id = category.Id }, category);
    }

    // DELETE: api/Categories/5
    [HttpDelete("{id}")]
    public async Task<IActionResult>
DeleteCategory([FromRoute] int id)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        var category = await
_context.Category.FindAsync(id);
        if (category == null)
        {
            return NotFound();
        }

        _context.Category.Remove(category);
        await _context.SaveChangesAsync();

        return Ok(category);
    }

    private bool CategoryExists(int id)
    {
        return _context.Category.Any(e => e.Id

```

	<pre>== id);     }     } }</pre>
--	--

ADMINUSER.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace sorucevapblog.API.Models
{
    public class AdminUser
    {
        public string Email { get; set; }
        public string Password { get; set; }
    }
}
```

ARTICLE.CS

```
using System;
using System.Collections.Generic;

namespace sorucevapblog.API.Models
{
    public partial class Article
    {
        public Article()
        {
            Comment = new HashSet<Comment>();
        }

        public int Id { get; set; }
        public string Title { get; set; }
        public string ContentSummary { get; set; }
        public string ContentMain { get; set; }
        public DateTime PublishDate { get; set; }
        public string Picture { get; set; }
        public int CategoryId { get; set; }
        public int ViewCount { get; set; }

        public Category Category { get; set; }
        public ICollection<Comment> Comment { get;
set; }
    }
}
```

CATEGORY.CS

```
using System;
using System.Collections.Generic;

namespace sorucevapblog.API.Models
{
    public partial class Category
    {
        public Category()
        {
            Article = new HashSet<Article>();
        }

        public int Id { get; set; }
        public string Name { get; set; }

        public ICollection<Article> Article { get;
set; }
    }
}
```

COMMENT.CS

```
using System;
using System.Collections.Generic;

namespace sorucevapblog.API.Models
{
    public partial class Comment
    {
        public int Id { get; set; }
        public int ArticleId { get; set; }
        public string Name { get; set; }
        public string ContentMain { get; set; }
        public DateTime PublishDate { get; set; }

        public virtual Article Article { get;
set; }
    }
}
```

ARTICLERESPONSE.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace sorucevapblog.API.Responses
{
    public class ArticleResponse
    {
        public int Id { get; set; }

        public string Title { get; set; }

        public string ContentMain { get; set; }

        public string ContentSummary { get; set; }

        public DateTime PublishDate { get; set; }

        public string Picture { get; set; }

        public int ViewCount { get; set; }

        public int CommentCount { get; set; }

        public CategoryResponse Category { get;
set; }

    }
}
```



CATEGORYRESPONSE.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace sorucevapblog.API.Responses
{
    public class CategoryResponse
    {
        public int Id { get; set; }

        public string Name { get; set; }
    }
}
```

## MODULES(MATERIAL)

src\app\modules\material.module.ts

```
import { NgModule } from "@angular/core";
import { CommonModule } from
"@angular/common";
import { BrowserAnimationsModule } from
"@angular/platform-browser/animations";
import { ReactiveFormsModule, FormsModule }
from "@angular/forms";
import {
  MatButtonModule,
  MatInputModule,
  MatTableModule,
  MatPaginatorModule,
  MatAutocompleteModule
} from "@angular/material";

@NgModule({
  declarations: [],
  imports: [
    CommonModule,
    BrowserAnimationsModule,
    ReactiveFormsModule,
    FormsModule,
    MatButtonModule,
    MatInputModule,
    MatTableModule,
    MatPaginatorModule,
    MatAutocompleteModule
  ],
  exports: [
    CommonModule,
```

	<pre>BrowserAnimationsModule, ReactiveFormsModule, FormsModule, MatButtonModule, MatInputModule, MatTableModule, MatPaginatorModule, MatAutocompleteModule ] })) export class MaterialModule {}</pre>
--	---



# SERVISLER

src\app\services\article.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { ArticlePg } from '../models/article-pg';
import { tap } from 'rxjs/operators';
import { Article } from '../models/article';
import { Archive } from '../models/archive';
@Injectable({
  providedIn: 'root'
})
export class ArticleService {
  constructor(private httpClient: HttpClient) {}
  public loading: boolean = true;
  private apiUrl: string =
'https://localhost:44386/api/articles';

  getArticlesWithoutPg() {
    return this.httpClient.get<Article[]>(this.apiUrl);
  }

  getArticles(page: number, pageSize: number) {
    let api = `${this.apiUrl}/${page}/${pageSize}`;

    return this.httpClient.get<ArticlePg>(api).pipe(
      tap(x => {
        this.loading = false;
        console.log(x.articles);
      })
    );
  }

  getSearchArticles(searchText: string, page: number, pageSize:
number) {
    let api =
`${this.apiUrl}/SearchArticles/${searchText}/${page}/${pageSize}
`;

    return this.httpClient.get<ArticlePg>(api).pipe(
      tap(x => {
        this.loading = false;
      })
    );
  }

  getArticlesWithCategory(categoryId: number, page: number,
pageSize: number) {
    let api =
`${this.apiUrl}/GetArticlesWithCategory/${categoryId}/${page}/${
pageSize}`;

    return this.httpClient.get<ArticlePg>(api).pipe(
      tap(x => {
        this.loading = false;
      })
    );
  }
}
```

```

    );
}

getArticle(id: number) {
    let api = `${this.apiUrl}/${id}`;
    return this.httpClient.get<Article>(api).pipe(
        tap(x => {
            this.loading = false;
        })
    );
}

getArticlesByMostView() {
    let api = `${this.apiUrl}/GetArticlesByMostView`;

    return this.httpClient.get<Article[]>(api);
}

getArticlesArchive() {
    let api = `${this.apiUrl}/GetArticlesArchive`;
    return this.httpClient.get<Archive[]>(api);
}

getArticleArchiveList(
    year: number,
    month: number,
    page: number,
    pageSize: number
) {
    let api =
`${this.apiUrl}/GetArticleArchiveList/${year}/${month}/${page}/${
    pageSize}`;

    return this.httpClient.get<ArticlePg>(api).pipe(
        tap(x => {
            this.loading = false;
        })
    );
}

ArticleViewCountUp(id: number) {
    let api = `${this.apiUrl}/ArticleViewCountUp/${id}`;
    return this.httpClient.get(api);
}

saveArticlePicture(image) {
    return this.httpClient.post<any>(
        `${this.apiUrl}/SaveArticlePicture`,
        image
    );
}

addArticle(article: Article) {
    return this.httpClient.post(this.apiUrl, article);
}

updateArticle(id: number, article: Article) {
    return this.httpClient.put(`${this.apiUrl}/${id}`,
article);
}

```

```
}

deleteArticle(id: number) {
  return this.httpClient.delete(`${this.apiUrl}/${id}`);
}
}
```

src\app\services\category.service.ts

```
import { Injectable } from '@angular/core';

import { HttpClient } from
 '@angular/common/http';
import { Category } from
 '../models/category';

@Injectable({
  providedIn: 'root'
})
export class CategoryService {
  private apiUrl: string =
    'https://localhost:44386/api/categories';
  constructor(private httpCleint: HttpClient)
  {}

  public getCategories() {
    return
    this.httpCleint.get<Category[]>(this.apiUrl);
  }

  public getCategoryById(id: number) {

    let url = `${this.apiUrl}/${id}`;
    return
    this.httpCleint.get<Category>(url);
  }
}
```



src\app\services\comment.service.ts

```
import { Injectable } from "@angular/core";
import { Comment } from "../models/comment";
import { HttpClient } from "@angular/common/http";
import { tap } from "rxjs/operators";

@Injectable({
  providedIn: "root"
})
export class CommentService {
  constructor(private httpClient: HttpClient) {}

  private apiUrl: string =
    "https://localhost:44386/api/comments";
  loading: boolean;
  addComment(comment: Comment) {
    this.loading = true;
    return this.httpClient.post(this.apiUrl,
comment).pipe(
      tap(x => {
        this.loading = false;
      })
    );
  }

  commentList(id: number) {
    return
this.httpClient.get<Comment[]>(`${this.apiUrl}/${id}`);
  }
}
```

src\app\services\myvalidation.service.ts

```
import { Injectable } from "@angular/core";
import { AbstractControl } from
"@angular/forms";

@Injectable({
  providedIn: "root"
})
export class MyvalidationService {
  constructor() {}

  GetValidationMessages(f: AbstractControl,
name: string) {
    if (f.errors) {
      for (let errroName in f.errors) {
        if (errroName == "required") return
`${name} alanı boş bırakılamaz`;
        else if (errroName == "email") return
`email formatı yanlış`;
        else if (errroName == "minlength")
          return `${name} alanız en az 5
karakter olmalıdır.`;
      }
    }
  }
}
```

src\app\services\auth.service.ts

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

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  constructor(private httpClient: HttpClient) {}

  IsAuthenticated(email: string, password: string)
  {
    let adminUser = { email: email, password:
password };

    return this.httpClient.post<any>(
'https://localhost:44386/api/Auth/IsAuthenticated',
  adminUser
  );
  }
}
```

src\app\services\auth-guard.service.ts

```
import { Injectable } from "@angular/core";
import { CanActivate, Router } from
"@angular/router";
import { AuthService } from "../auth.service";
import { map } from "rxjs/operators";

@Injectable({
  providedIn: "root"
})
export class AuthGuardService implements
CanActivate {
  canActivate(
    route:
import("@angular/router").ActivatedRouteSnapshot,
    state:
import("@angular/router").RouterStateSnapshot
  ):
    | boolean
    | import("@angular/router").UrlTree
    | import("rxjs").Observable<boolean |
import("@angular/router").UrlTree>
    | Promise<boolean |
import("@angular/router").UrlTree> {
    let email = localStorage.getItem("email");
    let password =
localStorage.getItem("password");

    return
this.authService.IsAuthenticated(email,
password).pipe(
  map(x => {
    if (x.status == true) {
      return true;
    } else {
      this.router.navigate(["/adminlogin"]);
      return false;
    }
  })
);
}

constructor(private authService: AuthService,
private router: Router) {}
}
```

src\app\services\helper.service.ts

```
import { Injectable } from "@angular/core";
import { HttpClient } from "@angular/common/http";
import { Contact } from "../models/contact";

@Injectable({
  providedIn: "root"
})
export class HelperService {
  constructor(private httpClient: HttpClient) {}

  private apiUrl: string =
    "https://localhost:44386/api/helper";

  sendContactEmail(contact: Contact) {
    return
    this.httpClient.post(`${this.apiUrl}/SendContactEmail`,
    contact);
  }
}
```

src\app\admin-pages\admin-home\admin-home.component.ts

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-admin-home',
  templateUrl: './admin-home.component.html',
  styleUrls: ['./admin-home.component.css']
})
export class AdminHomeComponent implements OnInit {

  constructor() { }

  ngOnInit() {
  }

}
```

src\app\admin-pages\article\article-add\article-add.component.html

```
<div class="row">
  <div class="col-md-9">
    <form
      *ngIf="info == null"
      (ngSubmit)="onSubmit()"
      [formGroup]="articleForm"
    >
      <div class="form-container">
        <mat-form-field>
          <input
            type="text"
            placeholder="kategori seç"
            FormControlName="category"
            matInput
            [matAutocomplete]="auto"
          />
          <mat-error
            *ngIf="getControls.category.invalid">
              {{
                myvalidationService.GetValidationMessages(
                  getControls.category,
                  "kategori"
                )
              }}
            </mat-error>

            <mat-autocomplete

              [displayWith]="displayCategoryName"
              #auto="matAutocomplete"
            >
              <mat-option *ngFor="let item of
categories" [value]="item">
                {{ item.name }}
              </mat-option>
            </mat-autocomplete>
          </mat-form-field>
          <mat-form-field>
            <input FormControlName="title"
matInput placeHolder="SORU İSMİ" />
            <mat-error
              *ngIf="getControls.title.invalid">
                {{
                  myvalidationService.GetValidationMessages(
                    getControls.title,
                    "SORU İSMİ"
                  )
                }}
              </mat-error>
            </mat-form-field>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

```

<mat-form-field>
  <input
    FormControlName="contentSummary"
    matInput
    placeholder="SORU İÇERİK ÖZET"
  />
  <mat-error
*ngIf="getControls.contentSummary.invalid">
    {{
myvalidationService.GetValidationMessages(
    getControls.contentSummary,
    "SORU ÖZET"
  )
    }}
  </mat-error>
</mat-form-field>
<div style="border: 1px solid black"
class="mb-2">
  <ckeditor
    (ready)="onReady($event)"
    FormControlName="contentMain"
    [editor]="Editor"
  ></ckeditor>
</div>
<button [disabled]="loading" mat-
raised-button color="primary">
  SORU EKLE
  <i
    *ngIf="loading"
    class="spinner-border spinner-
border-sm"
    role="status"
    aria-hidden="true"
  ></i>
</button>
</div>
</form>

<div role="alert" *ngIf="info != null"
class="alert alert-danger mt-3">
  {{ info }}
</div>
</div>
<div class="col-md-3">
  <div class="card text-white bg-primary ">
    <div class="card-header">SORUNA RESİM
EKLE</div>

    <div class="card-body">
      <img
        [src]="picture ||
'assets/article_empty.jpg'"
        class="img-thumbnail"
      />

```



	<pre>&lt;/div&gt; &lt;div class="card-footer"&gt;   &lt;input     (change)="upload(\$event)"     type="file"     class="form-control"     name="image"   /&gt; &lt;/div&gt; &lt;/div&gt; &lt;/div&gt; &lt;/div&gt;</pre>
--	--

src\app\admin-  
pages\article\article-  
add\article-  
add.component.ts

```
import { Component, OnInit } from "@angular/core";
import { ArticleService } from
"src/app/services/article.service";
import {
  FormGroup,
  FormControl,
  Validator,
  AbstractControl,
  Validators
} from "@angular/forms";
import { CategoryService } from
"src/app/services/category.service";
import { Category } from "src/app/models/category";
import { MyvalidationService } from
"src/app/services/myvalidation.service";
import { Router } from "@angular/router";
import * as DecoupledEditor from "@ckeditor/ckeditor5-build-
decoupled-document";

@Component({
  selector: "app-article-add",
  templateUrl: "../article-add.component.html",
  styleUrls: ["../article-add.component.css"]
})
export class ArticleAddComponent implements OnInit {
  public Editor = DecoupledEditor;
  public onReady(editor) {
    editor.ui
      .getEditableElement()
      .parentElement.insertBefore(
        editor.ui.view.toolbar.element,
        editor.ui.getEditableElement()
      );
  }

  fileData: File = null;
  picture: string = null;
  articleForm: FormGroup;
  success: boolean;
  loading: boolean;
  info: string;
  categories: Category[];
  constructor(
    private articleService: ArticleService,
    private categoryService: CategoryService,
    public myvalidationService: MyvalidationService,
    private router: Router
  ) {}

  ngOnInit() {
    this.getCategory();

    this.articleForm = new FormGroup({
```

```

        title: new FormControl("SORU 1", Validators.required),
        contentSummary: new FormControl("SORU ÖZET 1",
Validators.required),
        contentMain: new FormControl("", Validators.required),
        category: new FormControl("", Validators.required),
        picture: new FormControl("")
    });
}

get getControls() {
    return this.articleForm.controls;
}

onSubmit() {
    if (this.articleForm.valid) {
        this.loading = true;

this.articleService.addArticle(this.articleForm.value).subscribe(
    data => {
        this.success = true;
        // alert("geldi");
        this.router.navigateByUrl("/admin/makale/liste");
    },
    error => {
        this.success = false;
        this.info = "bir hata meydana geldi:";
        console.log(error);
    }
);
    }
}

displayCategoryName(category) {
    return category.name;
}

getCategory() {
    this.categoryService.getCategories().subscribe(result => {
        this.categories = result;
    });
}

upload(files) {
    this.fileData = files.target.files[0];

    let formData = new FormData();

    formData.append("picture", this.fileData);

this.articleService.saveArticlePicture(formData).subscribe(result
=> {
    console.log(result.path);
    this.picture = result.path;

    this.articleForm.controls.picture.setValue(this.picture);
});
}

```

	<pre>} }</pre>
--	--------------------

src\app\admin-pages\article\article-list\article-list.component.html

```
<div>
  <table style="width: 100%" mat-table
[dataSource]="dataSource">

    <ng-container matColumnDef="picture">
      <th mat-header-cell
*matHeaderCellDef>Resim</th>
      <td mat-cell *matCellDef="let article">
        <img
          class="img-thumbnail"
          style="width: 50px; height: 50px;"
          [src]="article.picture ||
'assets/article_empty.jpg'"
        />
      </td>
    </ng-container>

    <ng-container matColumnDef="title">
      <th mat-header-cell
*matHeaderCellDef>Başlık</th>
      <td mat-cell *matCellDef="let article">
        {{ article.title }}
      </td>
    </ng-container>

    <ng-container matColumnDef="category">
      <th mat-header-cell
*matHeaderCellDef>Kategori</th>
      <td mat-cell *matCellDef="let article">
        <strong> {{ article.category.name
}}</strong>
      </td>
    </ng-container>
    <ng-container
matColumnDef="commentCount">
      <th mat-header-cell
*matHeaderCellDef>yorum sayısı</th>
      <td mat-cell *matCellDef="let article">
        {{ article.commentCount }}
      </td>
    </ng-container>

    <ng-container matColumnDef="viewCount">
      <th mat-header-cell
*matHeaderCellDef>Gösterim sayısı</th>
      <td mat-cell *matCellDef="let article">
        {{ article.viewCount }}
      </td>
    </ng-container>
    <ng-container matColumnDef="publishDate">
      <th mat-header-cell
*matHeaderCellDef>Tarih</th>
      <td mat-cell *matCellDef="let article">
```

```

        {{ article.publishDate | date:
"dd.MM.yyyy" }}
    </td>
</ng-container>

    <ng-container matColumnDef="action">
        <th mat-header-cell
*matHeaderCellDef>işlemler</th>
        <td mat-cell *matCellDef="let article">
            <button

[routerLink]="['/admin/makale/guncelle',
article.id]"
                mat-raised-button
                color="primary"
            >
                güncelle
            </button>

            <button
                class="ml-1"
                (click)="deleteArticle(article.id)"
                mat-raised-button
                color="warn"
            >
                sil
            </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
    ></mat-paginator>
</div>

```

src\app\admin-  
pages\article\article-list\article-  
list.component.ts

```
import { Component, OnInit, ViewChild } from
"@angular/core";
import { DataSource } from "@angular/cdk/table";
import { Article } from "src/app/models/article";
import { MatPaginator, MatTableDataSource } from
"@angular/material";
import { ArticleService } from
"src/app/services/article.service";

@Component({
  selector: "app-article-list",
  templateUrl: "./article-list.component.html",
  styleUrls: ["./article-list.component.css"]
})
export class ArticleListComponent implements OnInit {
  displayedColumns: string[] = [
    "picture",
    "title",
    "category",
    "commentCount",
    "viewCount",
    "publishDate",
    "action"
  ];
  dataSource;
  articles: Article[];
  @ViewChild(MatPaginator, { static: true }) paginator:
MatPaginator;

  constructor(private articleService: ArticleService) {}

  ngOnInit() {
    this.articleService.getArticlesWithoutPg().subscribe(data
=> {
      this.articles = data;
      this.dataSource = new
MatTableDataSource<Article>(data);

      this.dataSource.paginator = this.paginator;
    });
  }

  deleteArticle(id) {
    this.articleService.deleteArticle(id).subscribe(data
=> {
      let article = this.articles.filter(x => x.id ==
id)[0];
      let index = this.articles.indexOf(article);

      this.articles.splice(index, 1);
      this.dataSource = new
MatTableDataSource<Article>(this.articles);
```

```
        this.dataSource.paginator = this.paginator;
    });
}
}
```



src\app\admin-pages\article\article-update\article-update.component.html

```
<div class="row">
  <div class="col-md-9">
    <div *ngIf="!contentLoading">
      <form
        *ngIf="info == null"
        (ngSubmit)="onSubmit()"
        [formGroup]="articleForm"
      >
        <div class="form-container">
          <mat-form-field>
            <input
              type="text"
              placeholder="kategori seç"
              FormControlName="category"
              matInput
              [matAutocomplete]="auto"
            />
            <mat-error
              *ngIf="getControls.category.invalid">
                {{
                  myvalidationService.GetValidationMessages(
                    getControls.category,
                    "kategori"
                  )
                }}
            </mat-error>

            <mat-autocomplete

              [displayWith]="displayCategoryName"
              #auto="matAutocomplete"
            >
              <mat-option *ngFor="let item of
categories" [value]="item">
                {{ item.name }}- {{ item.id
              }}
            </mat-option>
          </mat-autocomplete>
        </mat-form-field>
        <mat-form-field>
          <input FormControlName="title"
matInput placeholder="SORU İSMİ" />
          <mat-error
            *ngIf="getControls.title.invalid">
              {{
                myvalidationService.GetValidationMessages(
                  getControls.title,
                  "SORU İSMİ"
                )
              }}
            </mat-error>

```

```

        </mat-form-field>

        <mat-form-field>
            <input
formControlName="contentSummary"
            matInput
            placeholder="SORU İÇERİK ÖZET"
            />
            <mat-error
*ngIf="getControls.contentSummary.invalid">
                {{
myvalidationService.GetValidationMessages(
                    getControls.contentSummary,
                    "SORU ÖZET"
                )
            }}
        </mat-error>
    </mat-form-field>
    <div style="border: 1px solid
black" class="mb-2">
        <ckeditor
            (ready)="onReady($event)"
            formControlName="contentMain"
            [editor]="Editor"
        ></ckeditor>
    </div>
    <button [disabled]="loading" mat-
raised-button color="primary">
        SORU GÜNCELLE
        <i
            *ngIf="loading"
            class="spinner-border spinner-
border-sm"
            role="status"
            aria-hidden="true"
        ></i>
    </button>
</div>
</form>
</div>
    <div *ngIf="contentLoading" class="text-
center">
        <h3>SORU YÜKLENİYOR....</h3>
        <div class="spinner-grow text-primary"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div class="spinner-grow text-
secondary" role="status">
            <span class="sr-
only">Loading...</span>
        </div>
    </div>

```

```

        <div class="spinner-grow text-success"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div class="spinner-grow text-danger"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div class="spinner-grow text-warning"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div class="spinner-grow text-info"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div class="spinner-grow text-light"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div class="spinner-grow text-dark"
role="status">
            <span class="sr-
only">Loading...</span>
        </div>
        <div role="alert" *ngIf="info != null"
class="alert alert-danger mt-3">
            {{ info }}
        </div>
    </div>

    <div class="col-md-3">
        <div class="card text-white bg-primary ">
            <div class="card-header">SORUNA RESİM
EKLE</div>

            <div class="card-body">
                <img
                    [src]="picture ||
'assets/article_empty.jpg'"
                    class="img-thumbnail"
                />
            </div>
            <div class="card-footer">
                <input
                    (change)="upload($event)"
                    type="file"
                    class="form-control"
                    name="image"

```

	<pre>        /&gt;       &lt;/div&gt;     &lt;/div&gt;   &lt;/div&gt; &lt;/div&gt;</pre>
--	--

src\app\admin-  
pages\article\article-  
update\article-  
update.component.ts

```
import { Component, OnInit } from "@angular/core";
import { ArticleService } from
"src/app/services/article.service";
import {
  FormGroup,
  FormControl,
  Validator,
  AbstractControl,
  Validators
} from "@angular/forms";
import { CategoryService } from
"src/app/services/category.service";
import { MyvalidationService } from
"src/app/services/myvalidation.service";
import {
  Router,
  ActivatedRouteSnapshot,
  ActivatedRoute
} from "@angular/router";
import { Category } from "src/app/models/category";
import * as DecoupledEditor from "@ckeditor/ckeditor5-build-
decoupled-document";
@Component({
  selector: "app-article-update",
  templateUrl: "../article-update.component.html",
  styleUrls: ["../article-update.component.css"]
})
export class ArticleUpdateComponent implements OnInit {
  public Editor = DecoupledEditor;
  public onReady(editor) {
    editor.ui
      .getEditableElement()
      .parentElement.insertBefore(
        editor.ui.view.toolbar.element,
        editor.ui.getEditableElement()
      );
  }

  fileData: File = null;
  picture: string = null;
  articleForm: FormGroup;
  success: boolean;
  loading: boolean;
  info: string;
  categories: Category[];
  articleId: number;
  contentLoading: boolean = true;
  constructor(
    private articleService: ArticleService,
    private categoryService: CategoryService,
    public myvalidationService: MyvalidationService,
    private router: Router,
    private route: ActivatedRoute
  ) {}
}
```

```

) {}

ngOnInit() {
  this.getCategory();
  this.articleId =
Number(this.route.snapshot.paramMap.get("id"));

  this.articleService.getArticle(this.articleId).subscribe(data
=> {
    this.picture = data.picture;

    this.getControls.title.setValue(data.title);

this.getControls.contentSummary.setValue(data.contentSummary);
    this.getControls.contentMain.setValue(data.contentMain);
    this.getControls.category.setValue(data.category);
    this.contentLoading = false;
  });

  this.articleForm = new FormGroup({
    title: new FormControl("", Validators.required),
    contentSummary: new FormControl("", Validators.required),
    contentMain: new FormControl("", Validators.required),
    category: new FormControl("", Validators.required),
    picture: new FormControl("")
  });
}
displayCategoryName(category) {
  return category.name;
}
getCategory() {
  this.categoryService.getCategories().subscribe(result => {
    this.categories = result;
  });
}
get getControls() {
  return this.articleForm.controls;
}

onSubmit() {
  if (this.articleForm.valid) {
    this.loading = true;
    this.articleService
      .updateArticle(this.articleId, this.articleForm.value)
      .subscribe(
        data => {
          this.success = true;

          this.router.navigateByUrl("/admin/makale/liste");
        },
        error => {
          this.success = false;
          this.info = "bir hata meydana geldi.";
          console.log(error);
        }
      );
  }
}

```

```
    );  
  }  
}  
  
upload(files) {  
  this.fileData = files.target.files[0];  
  
  let formData = new FormData();  
  
  formData.append("picture", this.fileData);  
  
  this.articleService.saveArticlePicture(formData).subscribe(result  
=> {  
    console.log(result.path);  
    this.picture = result.path;  
  
    this.articleForm.controls.picture.setValue(this.picture);  
  });  
}
```

src\app\admin-pages\admin.module.ts

```
import { NgModule } from "@angular/core";
import { CommonModule } from
"@angular/common";
import { AppRoutingModuleModule } from "../app-
routing.module";
import { MaterialModule } from
"../modules/material.module";
import { ComponentsModule } from
"../components/components.module";
import { CKEditorModule } from
"@ckeditor/ckeditor5-angular";
import { AdminLayoutComponent } from
"../layout/admin-layout/admin-
layout.component";
import { AdminNavComponent } from
"../nav/admin-nav/admin-nav.component";
import { AdminHomeComponent } from "../admin-
home/admin-home.component";
import { AdminArticleComponent } from
"./article/article/article.component";
import { ArticleAddComponent } from
"./article/article-add/article-
add.component";
import { ArticleUpdateComponent } from
"./article/article-update/article-
update.component";
import { ArticleListComponent } from
"./article/article-list/article-
list.component";

@NgModule({
  declarations: [
    AdminLayoutComponent,
    AdminNavComponent,
    AdminHomeComponent,
    AdminArticleComponent,
    ArticleAddComponent,
    ArticleUpdateComponent,
    ArticleListComponent
  ],
  imports: [
    CommonModule,
    AppRoutingModuleModule,
    MaterialModule,
    ComponentsModule,
    CKEditorModule
  ]
})
export class AdminModule {}
```



src\app\components\add-comment\add-comment.component.html

```
<div class="card bg-light mt-3">
  <div class="card-body">
    <div
      *ngIf="info != null"
      class="alert"
      [ngClass]="{ 'alert-primary': success, 'alert-danger': !success }"
    >
      {{ info }}
    </div>

    <form
      *ngIf="info == null"
      (ngSubmit)="onSubmit()"
      [formGroup]="commentForm"
    >
      <div class="form-container">
        <mat-form-field>
          <input formControlName="name" matInput
placeholder="isim..." />
          <mat-error *ngIf="getControls.name.invalid">{{
myvalidationService.GetValidationMessages(getControls.name,
"isim")
          }}</mat-error>
        </mat-form-field>

        <mat-form-field>
          <textarea
            rows="5"
            formControlName="contentMain"
            matInput
            placeholder="yorumunuz..."
          ></textarea>
          <mat-error
*ngIf="getControls.contentMain.invalid">{{
            myvalidationService.GetValidationMessages(
              getControls.contentMain,
              "yorum"
            )
          }}</mat-error>
        </mat-form-field>

        <button
          [disabled]="commentService.loading"
          mat-raised-button
          color="primary"
        >
          yorum ekle
          <i
            *ngIf="commentService.loading"
            class="spinner-border spinner-border-sm"
            role="status"
          >

```

	<pre>        aria-hidden="true"     &gt;&lt;/i&gt;     &lt;/button&gt; &lt;/div&gt; &lt;/form&gt; &lt;/div&gt; &lt;/div&gt;</pre>
--	---

src\app\components\add-comment\add-comment.component.ts

```
import {
  Component,
  OnInit,
  ViewChild,
  Output,
  EventEmitter
} from '@angular/core';
import { FormGroup, FormControl, Validators } from
'@angular/forms';
import { CommentService } from
'src/app/services/comment.service';
import { ActivatedRoute } from '@angular/router';
import { MyvalidationService } from
'src/app/services/myvalidation.service';
import { ListCommentsComponent } from '../list-comments/list-
comments.component';

@Component({
  selector: 'app-add-comment',
  templateUrl: './add-comment.component.html',
  styleUrls: ['./add-comment.component.css']
})
export class AddCommentComponent implements OnInit {
  commentForm: FormGroup;
  success: boolean;
  info: string;
  @Output() Reload: EventEmitter<string> = new EventEmitter();
  constructor(
    public commentService: CommentService,
    private route: ActivatedRoute,
    public myvalidationService: MyvalidationService
  ) {}

  ngOnInit() {
    this.commentForm = new FormGroup({
      name: new FormControl('', Validators.required),
      contentMain: new FormControl('', Validators.required),
      articleId: new FormControl('')
    });
  }

  get getControls() {
    return this.commentForm.controls;
  }

  onSubmit() {

    if (this.commentForm.valid) {
      let id = Number(this.route.snapshot.paramMap.get('id'));

      this.commentForm.controls.articleId.setValue(id);
    }
  }
}
```

```
this.commentService.addComment(this.commentForm.value).subscribe(  
  data => {  
    this.success = true;  
    this.info = 'Yorumunuz başarıyla eklenmiştir.';  
    this.Reload.emit();  
  },  
  error => {  
    this.success = false;  
    this.info =  
      'Bir hata meydana geldi. Lütfen daha sonra tekrar  
deneyiniz..';  
  }  
);  
}  
}
```

**src\app\components\articles\articles.component.html**

```
<div *ngIf="articleService.loading">
  <main *ngFor="let item of
createRange()" class="page mb-2">

    <div class="page-content">

      <div class="placeholder-
content">

        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
        <div class="placeholder-
content_item"></div>
      </div>
    </div>
  </main>
</div>

<div *ngIf="articles.length > 0; else
emptyTemplate">
  <div
    *ngFor="
      let item of articles
      | paginate
      : {
        itemsPerPage: pageSize,
        currentPage: page,
        totalItems: totalCount
      }
    "
  >
    <ng-container
      *ngTemplateOutlet="
        horizontalTemplate;
        context: {
          id: item.id,
```

```

        commentCount:
item.commentCount,
        viewCount: item.viewCount,
        categoryName:
item.category.name,
        publishDate:
item.publishDate,
        contentSummary:
item.contentSummary,
        title: item.title,
        picture: item.picture
    }
    "
    ></ng-container>
</div>
</div>

<ng-template #emptyTemplate>
    <div *ngIf="!articleService.loading"
class="alert alert-danger"
role="alert">
        Soru bulunamadı....
    </div>
</ng-template>

<ng-template
    #horizontalTemplate
    let-id="id"
    let-commentCount="commentCount"
    let-viewCount="viewCount"
    let-categoryName="categoryName"
    let-publishDate="publishDate"
    let-contentSummary="contentSummary"
    let-title="title"
    let-picture="picture"
>
    <div class="card bg-light border-
primary mb-2">
        <div class="card-body">
            <div class="row">
                <div class="col-md-4">
                    <img [src]="picture ||
default_article" class="card-img" />
                </div>
                <div class="col-md-8">
                    <h5 class="card-title">{{
title }}</h5>
                    <p class="card-text">{{
contentSummary }}</p>
                </div>
            </div>
        </div>
        <div class="card-footer bg-
transparent border-primary">
            <span class="badge badge-pill

```

```

badge-dark">
    <i class="fa fa-
calendar"></i> {{ publishDate | date:
"dd.MM.yyyy" }}
    </span>
    <span class="mr-1"></span>
    <span class="badge badge-pill
badge-success">
        <i class="fa fa-folder"></i>
{{ categoryName }}
    </span>
    <span class="mr-1"></span>
    <span class="badge badge-pill
badge-success">
        <i class="fa fa-eye"></i> {{
viewCount }}
    </span>
    <span class="mr-1"></span>

    <span class="badge badge-pill
badge-primary">
        <i class="fa fa-comment"></i>
{{ commentCount }}
    </span>

    <a
        [routerLink]="['/makale',
title | urlformat, id]"
        class="btn btn-primary btn-sm
float-right"
    >
        devamını oku
    </a>
</div>
</div>
</ng-template>

<pagination-controls
    *ngIf="totalCount > pageSize"
    responsive="true"
    (pageChange)="pageChanged($event)"
    previousLabel="geri"
    nextLabel="ileri"
></pagination-controls>

```

src\app\components\articles\articles.  
component.ts

```
import { Component, OnInit, Input } from
"@angular/core";
import { Article } from "src/app/models/article";
import { Router, ActivatedRoute } from
"@angular/router";
import { ArticleService } from
"src/app/services/article.service";
@Component({
  selector: "app-articles",
  templateUrl: "./articles.component.html",
  styleUrls: ["./articles.component.css"]
})
export class ArticlesComponent implements OnInit {
  @Input() totalCount: number;
  @Input() articles: Article[];
  @Input() page: number;
  @Input() pageSize: number;
  @Input() loadingItem: number;
  @Input() typeList: string;
  default_article: string =
"assets/article_empty.jpg";

  constructor(
    private router: Router,
    private route: ActivatedRoute,
    private articleService: ArticleService
  ) {}

  createRange() {
    var items: number[] = [];
    for (var i = 1; i <= this.loadingItem; i++) {
      items.push(i);
    }
    return items;
  }

  ngOnInit() {
    this.articleService.loading = true;
  }

  pageChanged(event) {
    this.articleService.loading = true;
    this.page = event;
    switch (this.typeList) {
      case "home":

this.router.navigateByUrl(`/sayfa/${this.page}`);
      break;
      case "category":
        let categoryName =
this.route.snapshot.paramMap.get("name");
        let categoryId =
this.route.snapshot.paramMap.get("id");
```



```
        this.router.navigateByUrl(
`/kategori/${categoryName}/${categoryId}/sayfa/${this
.page}`
        );

        break;

        case "search":
            let searchText =
this.route.snapshot.queryParamMap.get("s");

this.router.navigateByUrl(`/arama/sayfa/${this.page}?
s=${searchText}`);

            break;

        case "archive":
            let year =
this.route.snapshot.paramMap.get("year");
            let month =
this.route.snapshot.paramMap.get("month");

this.router.navigateByUrl(`/arsiv/${year}/${month}/sa
yfa/${this.page}`);

            break;

        default:
            break;
    }
}
}
```

src\app\components\list-comments\list-comments.component.html

```
<h3>Yorumlar({{ comments.length }})</h3>
<hr />
<div *ngIf="loading" class="d-flex justify-content-center">
  <div class="spinner-border" role="status">
    <span class="sr-only">Loading...</span>
  </div>
</div>
<div id="commentsWrapper" *ngIf="!loading">
  <div *ngFor="let item of comments"
class="card bg-light mt-3">
    <div class="card-header">
      <strong><i class="fa fa-user-circle"></i> {{ item.name }}</strong>
    </div>
    <div class="card-body">
      <p class="card-text">{{
item.contentMain }}</p>
    </div>
  </div>
</div>
```

src\app\components\list-  
comments\list-  
comments.component.ts

```
import { Component, OnInit } from "@angular/core";
import { CommentService } from
"src/app/services/comment.service";
import { ActivatedRoute } from "@angular/router";
import { Comment } from "src/app/models/comment";

@Component({
  selector: "app-list-comments",
  templateUrl: "../list-comments.component.html",
  styleUrls: ["../list-comments.component.css"]
})
export class ListCommentsComponent implements OnInit
{
  comments: Comment[] = [];
  loading: boolean;

  constructor(
    private commentService: CommentService,
    private route: ActivatedRoute
  ) {}

  ngOnInit() {
    this.loading = true;
    let id =
Number(this.route.snapshot.paramMap.get("id"));

this.commentService.commentList(id).subscribe(data =>
{
  this.comments = data;
  this.loading = false;
});
}

public reload()
{
  this.loading = true;
  let id =
Number(this.route.snapshot.paramMap.get('id'));

  this.commentService.commentList(id).subscribe(data
=> {
    this.comments = data;
    this.loading = false;
  });
}
}
```



src\app\components\menu-archive\menu-  
archive.component.html

```
<div class="card border-primary mt-2">
  <div class="card-header">
    <strong>Arşiv</strong>
  </div>
  <div class="card-body">
    <div *ngIf="archives.length == 0"
class="d-flex justify-content-center">
      <div class="spinner-border text-
primary" role="status">
        <span class="sr-
only">Loading...</span>
      </div>
    </div>

    <h3 *ngFor="let item of archives">
      <a
        [routerLink]="['/arsiv', item.year,
item.month]"
        class="btn btn-link text-left"
        ><i class="fa fa-angle-right"></i> {{
item.year }}
        {{ item.monthName }}({{ item.count
}})</a>
      >
    </h3>
  </div>
</div>
```

src\app\components\menu-  
archive\menu-archive.component.ts

```
import { Component, OnInit } from "@angular/core";
import { Archive } from "src/app/models/archive";
import { ArticleService } from
"src/app/services/article.service";

@Component({
  selector: "app-menu-archive",
  templateUrl: "../menu-archive.component.html",
  styleUrls: ["../menu-archive.component.css"]
})
export class MenuArchiveComponent implements OnInit
{
  archives: Archive[] = [];
  constructor(private articleService:
ArticleService) {}

  ngOnInit() {

this.articleService.getArticlesArchive().subscribe(
  data => {
    this.archives = data;
  },
  error => {
    console.log("bir hata meydana geldi:" +
error);
  }
);
}
}
```

src\app\components\menu-article-most-view\menu-article-most-view.component.html

```
<div class="card border-primary mt-2">
  <div class="card-header">
    <strong>En çok okunanlar</strong>
  </div>
  <div class="card-body">
    <div *ngIf="articles.length == 0"
class="d-flex justify-content-center">
      <div class="spinner-border text-
primary" role="status">
        <span class="sr-
only">Loading...</span>
      </div>
    </div>

    <h3 *ngFor="let item of articles">
      <a
        [routerLink]="['/makale', item.title
| urlformat, item.id]"
        class="btn btn-link text-left"
        ><i class="fa fa-angle-right"></i> {{
item.title }}</a>
      >
    </h3>
  </div>
</div>
```

src\app\components\menu-  
article-most-view\menu-article-  
most-view.component.ts

```
import { Component, OnInit } from "@angular/core";
import { ArticleService } from
"src/app/services/article.service";
import { Article } from "src/app/models/article";

@Component({
  selector: "app-menu-article-most-view",
  templateUrl: "../menu-article-most-view.component.html",
  styleUrls: ["../menu-article-most-view.component.css"]
})
export class MenuArticleMostViewComponent implements
OnInit {
  articles: Article[] = [];
  constructor(private articleService: ArticleService) {}

  ngOnInit() {

this.articleService.getArticlesByMostView().subscribe(data
=> {
  this.articles = data;
});
}
}
```



src\app\components\menu-category\menu-category.component.html

```
<div class="card border-primary">
  <div class="card-header">Kategori</div>
  <div class="card-body text-primary">
    <div *ngIf="categories.length == 0"
class="d-flex justify-content-center">
      <div class="spinner-border text-
primary" role="status">
        <span class="sr-
only">Loading...</span>
      </div>
    </div>

    <h3 *ngFor="let item of categories">
      <strong>

        <a
          [routerLink]="['/kategori',
item.name | urlformat, item.id]"
          class="btn btn-link"
          ><i class="fa fa-arrow-circle-
right"></i> {{ item.name }}</a
        >
      </strong>
    </h3>
  </div>
</div>
```

src\app\components\menu-category\menu-category.component.ts

```
import { Component, OnInit } from "@angular/core";
import { CategoryService } from
"../../services/category.service";
import { Category } from "src/app/models/category";
@Component({
  selector: "app-menu-category",
  templateUrl: "./menu-category.component.html",
  styleUrls: ["./menu-category.component.css"]
})
export class MenuCategoryComponent implements
OnInit {
  categories: Category[] = [];
  constructor(private categoryService:
CategoryService) {}

  ngOnInit() {

this.categoryService.getCategories().subscribe(data
=> {
  this.categories = data;
});
}
}
```

src\app\components\components.module.ts

```
import { NgModule } from "@angular/core";

import { RouterModule } from
"@angular/router";
import { CommonModule } from
"@angular/common";
import { NgxPaginationModule } from "ngx-
pagination";

import { MenuCategoryComponent } from
"./menu-category/menu-category.component";
import { PageTitleComponent } from "./page-
title/page-title.component";
import { ArticlesComponent } from
"./articles/articles.component";
import { UrlformatPipe } from
"../pipes/urlformat.pipe";
import { MenuArticleMostViewComponent } from
"./menu-article-most-view/menu-article-most-
view.component";
import { MenuArchiveComponent } from "../menu-
archive/menu-archive.component";
import { AddCommentComponent } from "../add-
comment/add-comment.component";
import { MaterialModule } from
"../modules/material.module";
import { ListCommentsComponent } from
"./list-comments/list-comments.component";

@NgModule({
  declarations: [
    MenuCategoryComponent,
    PageTitleComponent,
    ArticlesComponent,
    UrlformatPipe,
    MenuArticleMostViewComponent,
    MenuArchiveComponent,
    AddCommentComponent,
    ListCommentsComponent
  ],
  imports: [CommonModule, RouterModule,
NgxPaginationModule, MaterialModule],
  exports: [
    MenuCategoryComponent,
    PageTitleComponent,
    ArticlesComponent,
    UrlformatPipe,
    MenuArticleMostViewComponent,
    MenuArchiveComponent,
    AddCommentComponent,
    ListCommentsComponent
  ]
})
```

	<pre>export class ComponentsModule {}</pre>
--	---

src\app\layout\admin-layout\admin-  
layout.component.html

```
        <app-admin-nav></app-admin-nav>  
<div class="container">  
  <div class="row">  
    <div class="col-md-12">  
      <router-outlet></router-outlet>  
    </div>  
  </div>  
</div>
```

src\app\layout\main-layout\main-  
layout.component.html

```
<app-main-nav></app-main-nav>

<div class="container" style="margin-bottom:
50px;">
  <div class="row mt-4">
    <div class="col-md-8">
      <router-outlet></router-outlet>
    </div>
    <div class="col-md-4 ">
      <app-menu-category></app-menu-category>
      <app-menu-article-most-view></app-menu-
article-most-view>
      <app-menu-archive></app-menu-archive>
    </div>
  </div>
</div>

<app-footer-nav></app-footer-nav>
```

src\lapp\nav\admin-nav\admin-  
nav.component.html

```
<nav class="navbar navbar-expand-lg navbar-  
dark bg-dark">  
  <div class="container">  
    <button  
      class="navbar-toggler"  
      type="button"  
      data-toggle="collapse"  
      data-target="#navbarTogglerDemo01"  
      aria-controls="navbarTogglerDemo01"  
      aria-expanded="false"  
      aria-label="Toggle navigation"  
    >  
      <span class="navbar-toggler-  
icon"></span>  
    </button>  
    <div class="collapse navbar-collapse"  
id="navbarTogglerDemo01">  
      <a class="navbar-brand" href="#">ADMIN  
YÖNETİM PANELİ</a>  
      <ul class="navbar-nav mr-auto mt-2 mt-  
lg-0">  
        <li class="nav-item">  
          <a class="nav-link"  
routerLink="/admin"  
          >Anasayfa <span class="sr-  
only">(current)</span></a>  
        >  
      </li>  
  
        <li class="nav-item dropdown">  
          <a  
            class="nav-link dropdown-toggle"  
            href="#"  
            id="navbarDropdownMenuLink"  
            role="button"  
            data-toggle="dropdown"  
            aria-haspopup="true"  
            aria-expanded="false"  
          >  
            SORU  
          </a>  
          <div class="dropdown-menu" aria-  
labelledby="navbarDropdownMenuLink">  
            <a class="dropdown-item"  
routerLink="/admin/makale/ekle"  
            >SORU EKLE</a>  
            >  
            <a class="dropdown-item"  
routerLink="/admin/makale/liste" href="#"  
            >SORULARI LİSTELE</a>  
            >  
          </div>  
        </li>
```

	<pre>        &lt;li class="nav-item"&gt;           &lt;a class="nav-link" routerLink="/"&gt;             &lt;i class="fa fa-home"&gt;&lt;/i&gt; BLOG ANASAYFASINA DÖN!           &lt;span class="sr- only"&gt;(current)&lt;/span&gt;&lt;/a&gt;         &gt;       &lt;/li&gt;     &lt;/ul&gt;   &lt;/div&gt; &lt;/div&gt; &lt;/nav&gt;</pre>
--	--



src\lapp\nav\main-nav\main-  
nav.component.html

```
<nav class="navbar navbar-expand-lg  
navbar-dark bg-primary">  
  <div class="container">  
    <button  
      class="navbar-toggler"  
      type="button"  
      data-toggle="collapse"  
      data-target="#navbarTogglerDemo01"  
      aria-controls="navbarTogglerDemo01"  
      aria-expanded="false"  
      aria-label="Toggle navigation"  
    >  
      <span class="navbar-toggler-  
icon"></span>  
    </button>  
    <div class="collapse navbar-collapse"  
id="navbarTogglerDemo01">  
      <a class="navbar-brand" href="#">Find  
Any Answer</a>  
      <ul class="navbar-nav mr-auto mt-2 mt-  
lg-0">  
        <li class="nav-item" [ngClass]="{  
active: pageActive == 1 }">  
          <a class="nav-link" routerLink="/">  
            >Anasayfa <span class="sr-  
only">(current)</span></a>  
          </li>  
  
          <li class="nav-item " [ngClass]="{  
active: pageActive == 2 }">  
            <a class="nav-link"  
routerLink="/hakkimda"  
            >hakkimda <span class="sr-  
only">(current)</span></a>  
            </li>  
  
            <li class="nav-item " [ngClass]="{  
active: pageActive == 3 }">  
              <a class="nav-link"  
routerLink="/iletisim"  
              >iletisim <span class="sr-  
only">(current)</span></a>  
              </li>  
  
              <li class="nav-item ">  
                <a class="nav-link"  
routerLink="/admin"  
                ><i class="fa fa-tools"></i>  
yönetim paneline giriş  
                <span class="sr-  
only">(current)</span></a>
```

```
        >
      </li>
    </ul>
    <form class="form-inline my-2 my-lg-0">
      <input
        #searchText
        class="form-control mr-sm-2"
        type="search"
        placeholder="Search"
        aria-label="Search"
      />
      <button
        (click)="search(searchText.value)"
        class="btn btn-outline-success my-2
my-sm-0"
        type="button"
      >
        Search
      </button>
    </form>
  </div>
</div>
</nav>
```

src\app\nav\main-nav\main-nav.component.ts

```
import { Component, OnInit } from "@angular/core";
import { Router, NavigationEnd } from "@angular/router";
enum MainPage {
  home = 1,
  about_me = 2,
  contact = 3
}
@Component({
  selector: "app-main-nav",
  templateUrl: "../main-nav.component.html",
  styleUrls: ["../main-nav.component.css"]
})
export class MainNavComponent implements OnInit {
  pageActive: MainPage;
  constructor(private router: Router) {
    this.router.events.subscribe(x => {
      if (x instanceof NavigationEnd) {
        if (x.url.indexOf("anasayfa") > 0) {
          this.pageActive = MainPage.home;
        } else if (x.url.indexOf("hakkimda") > 0) {
          this.pageActive = MainPage.about_me;
        } else if (x.url.indexOf("iletisim") > 0) {
          this.pageActive = MainPage.contact;
        } else {
          this.pageActive = MainPage.home;
        }
      }
    });
  }

  ngOnInit() {}

  search(searchText) {
    if (searchText == "" || searchText == null || searchText
    == undefined) {
      return false;
    }

    this.router.navigateByUrl(`/arama/sayfa/1?s=${searchText}`);
  }
}
```

src\app\pages\admin-login\admin-  
login.component.html

```
<app-page-title [title]='Admin Panel  
Giriş'></app-page-title>  
<div class="example-container">  
  <mat-form-field>  
    <input #emailInput matInput  
placeholder="email" />  
  </mat-form-field>  
  
  <mat-form-field>  
    <input type="password" #passwordInput  
matInput placeholder="şifre" />  
  </mat-form-field>  
  
  <button  
    (click)="login(emailInput.value,  
passwordInput.value)"  
    mat-raised-button  
    color="primary"  
  >  
    giriş  
  </button>  
</div>
```

src\app\pages\admin-login\admin-  
login.component.ts

```
import { Component, OnInit } from
"@angular/core";
import { AuthService } from
"src/app/services/auth.service";
import { stringify } from
"@angular/compiler/src/util";
import { Router } from "@angular/router";

@Component({
  selector: "app-admin-login",
  templateUrl: "../admin-
login.component.html",
  styleUrls: ["../admin-login.component.css"]
})
export class AdminLoginComponent implements
OnInit {
  constructor(private authService:
AuthService, private router: Router) {}

  ngOnInit() {}
  login(email: string, password: string) {
    this.authService.IsAuthenticated(email,
password).subscribe(result => {
      if (result.status == true) {
        localStorage.setItem("email", email);
        localStorage.setItem("password",
password);
        this.router.navigate(["/admin"]);
      } else {
        alert("Email veya şıfe yanlışı");
      }
    });
  }
}
```

src\app\pages\archive\archive.component.html

```
<app-articles
  [totalCount]="totalCount"
  [articles]="articles"
  [page]="page"
  [pageSize]="pageSize"
  [loadingItem]="loadingItem"
  [typeList]="'archive'"
></app-articles>
```

src\app\pages\archive\archive.component.ts

```
import { Component, OnInit } from
"@angular/core";
import { Article } from
"src/app/models/article";
import { ActivatedRoute } from
"@angular/router";
import { ArticleService } from
"src/app/services/article.service";

@Component({
  selector: "app-archive",
  templateUrl: "./archive.component.html",
  styleUrls: ["./archive.component.css"]
})
export class ArchiveComponent implements
OnInit {
  page: number = 1;
  articles: Article[] = [];
  totalCount: number;
  pageSize: number = 5;
  loadingItem: number = 5;
  ajax;
  categoryId: number;
  constructor(
    private route: ActivatedRoute,
    private articleService: ArticleService
  ) {}

  ngOnInit() {
    this.route.paramMap.subscribe(params => {
      if (this.ajax != null)
this.ajax.unsubscribe();

      this.articleService.loading = true;
      this.articles = [];
      this.totalCount = 0;

      if (params.get("page")) {
        this.page =
Number(params.get("page"));
      }

      let year = Number(params.get("year"));
      let month =
Number(params.get("month"));

      this.ajax = this.articleService
        .getArticleArchiveList(year, month,
this.page, this.pageSize)
        .subscribe(data => {
          this.articles = data.articles;
          this.totalCount = data.totalCount;
        });
    });
  }
}
```

	<pre>}); } }</pre>
--	----------------------------



src\app\pages\article\article.component.html

```
<div class="card bg-light" style="height:
500px" *ngIf="articleService.loading">
  <div class="card-body text-center">
    <h3>Makale yükleniyor....</h3>
    <div class="spinner-grow text-primary"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-secondary"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-success"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-danger"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-warning"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-info"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-light"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
    <div class="spinner-grow text-dark"
role="status">
      <span class="sr-only">Loading...</span>
    </div>
  </div>
</div>
<div id="articleWrapper"
*ngIf="!articleService.loading">
  <div class="card border-primary">
    <div class="card-header">
      <h3>{{ article.title }}</h3>

      <div>
        <span class="badge badge-pill badge-
dark">
          <i class="fa fa-calendar"></i>
          {{ article.publishDate | date:
'dd.MM.yyyy' }}
        </span>
        <span class="mr-1"></span>
        <span class="badge badge-pill badge-
```

	<pre>success"&gt;     &lt;i class="fa fa-folder"&gt;&lt;/i&gt; {{ category.name }} &lt;/span&gt; &lt;span class="mr-1"&gt;&lt;/span&gt; &lt;span class="badge badge-pill badge- success"&gt;     &lt;i class="fa fa-eye"&gt;&lt;/i&gt; {{ article.viewCount }} &lt;/span&gt; &lt;span class="mr-1"&gt;&lt;/span&gt;      &lt;span class="badge badge-pill badge- primary"&gt;         &lt;i class="fa fa-comment"&gt;&lt;/i&gt; {{ article.commentCount }}     &lt;/span&gt; &lt;/div&gt; &lt;/div&gt; &lt;div class="card-body"&gt;     &lt;div class="ck ck-content" [innerHTML]="article.contentMain"&gt;&lt;/div&gt;     &lt;/div&gt; &lt;/div&gt; &lt;app-add-comment (Reload)="ReloadCommnetList()"&gt;&lt;/app-add- comment&gt;     &lt;app-list-comments&gt;&lt;/app-list-comments&gt; &lt;/div&gt;</pre>
--	---

src\app\pages\article\article.component.ts

```
import { Component, OnInit, ViewChild } from
 '@angular/core';
import { ArticleService } from
 'src/app/services/article.service';
import { ActivatedRoute } from '@angular/router';
import { Article } from 'src/app/models/article';
import { Category } from 'src/app/models/category';
import { ListCommentsComponent } from
 'src/app/components/list-comments/list-
 comments.component';

@Component({
  selector: 'app-article',
  templateUrl: './article.component.html',
  styleUrls: ['./article.component.css']
})
export class ArticleComponent implements OnInit {
  article: Article;
  category: Category;
  @ViewChild(ListCommentsComponent, { static: false })
  listComponent: ListCommentsComponent;
  constructor(
    private articleService: ArticleService,
    private route: ActivatedRoute
  ) {}

  ngOnInit() {
    this.route.paramMap.subscribe(params => {
      this.articleService.loading = true;

      // www.mysite.com/makale/asp.net core'gelen
      yenilikler/4
      let id =
Number(this.route.snapshot.paramMap.get('id'));

      this.articleService.getArticle(id).subscribe(data
=> {
        this.article = data;
        this.category = data.category;

this.articleService.ArticleViewCountUp(this.article.id).
subscribe();
      });
    });
  }

  ReloadCommnetList() {
    this.listComponent.reLoad();
  }
}
```

--	--

src\app\pages\category-articles\category-articles.component.html

```
<app-articles
  [totalCount]="totalCount"
  [articles]="articles"
  [page]="page"
  [pageSize]="pageSize"
  [loadingItem]="loadingItem"
  [typeList]="'category'"
></app-articles>
```

src\app\pages\category-articles\category-articles.component.ts

```
import { Component, OnInit } from
"@angular/core";
import { Article } from
"src/app/models/article";
import { ActivatedRoute } from
"@angular/router";
import { ArticleService } from
"src/app/services/article.service";

@Component({
  selector: "app-category-articles",
  templateUrl: "./category-
articles.component.html",
  styleUrls: ["./category-
articles.component.css"]
})
export class CategoryArticlesComponent
implements OnInit {
  page: number = 1;
  articles: Article[] = [];
  totalCount: number;
  pageSize: number = 5;
  loadingItem: number = 5;
  ajax;
  categoryId: number;
  constructor(
    private route: ActivatedRoute,
    private articleService: ArticleService
  ) {}

  ngOnInit() {
    this.route.paramMap.subscribe(params => {
      if (this.ajax != null)
this.ajax.unsubscribe();

      this.articleService.loading = true;
      this.articles = [];
      this.totalCount = 0;
      if (params.get("id")) {
        this.categoryId =
Number(params.get("id"));
      }
      if (params.get("page")) {
        this.page =
Number(params.get("page"));
      }

      this.ajax = this.articleService
.getArticlesWithCategory(this.categoryId,
this.page, this.pageSize)
.subscribe(data => {
    this.articles = data.articles;
```

	<pre>        this.totalCount = data.totalCount;     }); } }</pre>
--	---

src\app\pages\home\home.component.html

```
<app-articles
  [totalCount]="totalCount"
  [articles]="articles"
  [page]="page"
  [pageSize]="pageSize"
  [loadingItem]="loadingItem"
  [typeList]="'home' "
></app-articles>
```



src\app\pages\home\home.component.ts

```
import { Component, OnInit, OnDestroy }
from "@angular/core";
import { Article } from
"src/app/models/article";
import { ArticleService } from
"src/app/services/article.service";
import { Router, ActivatedRoute } from
"@angular/router";
@Component({
  selector: "app-home",
  templateUrl: "../home.component.html",
  styleUrls: ["../home.component.css"]
})
export class HomeComponent implements OnInit,
OnDestroy {
  ngOnDestroy(): void {
    if (this.ajax != null)
this.ajax.unsubscribe();
  }
  page: number = 1;
  articles: Article[];
  totalCount: number;
  pageSize: number = 5;
  loadingItem: number = 5;
  ajax;
  constructor(
    private articleService: ArticleService,
    private router: Router,
    private route: ActivatedRoute
  ) {}

  ngOnInit() {
    this.route.paramMap.subscribe(params => {
      if (params.get("page")) {
        this.page =
Number(params.get("page"));
      }

      if (this.totalCount > 0) {
        if (this.totalCount >= this.page *
this.pageSize) {
          this.loadingItem = 5;
        } else {
          this.loadingItem = this.totalCount
- (this.page - 1) * this.pageSize;
        }
      }

      this.articles = [];
      this.totalCount = 0;

      this.ajax = this.articleService
        .getArticles(this.page,
```

	<pre>this.pageSize)     .subscribe(data =&gt; {       this.articles = data.articles;       this.totalCount = data.totalCount;     });   }); }</pre>
--	---

src\app\pages\search\search.component.html

```
<app-articles
  [totalCount]="totalCount"
  [articles]="articles"
  [page]="page"
  [pageSize]="pageSize"
  [loadingItem]="loadingItem"
  [typeList]="'search'"
></app-articles>
```

src\app\pages\search\search.component.ts

```
import { Component, OnInit } from
"@angular/core";
import { Article } from
"src/app/models/article";
import { ActivatedRoute } from
"@angular/router";
import { ArticleService } from
"src/app/services/article.service";
import { ThrowStmt } from "@angular/compiler";

@Component({
  selector: "app-search",
  templateUrl: "./search.component.html",
  styleUrls: ["./search.component.css"]
})
export class SearchComponent implements OnInit {
  page: number = 1;
  articles: Article[] = [];
  totalCount: number;
  pageSize: number = 5;
  loadingItem: number = 5;
  ajax;
  searchText: string;

  constructor(
    private route: ActivatedRoute,
    private articleService: ArticleService
  ) {}

  ngOnInit() {
    this.route.url.subscribe(params => {
      if (this.ajax != null)
this.ajax.unsubscribe();

      this.articles = [];
      this.totalCount = 0;
      this.articleService.loading = true;

      if
(this.route.snapshot.paramMap.get("page")) {
        this.page =
Number(this.route.snapshot.paramMap.get("page"))
;
      }

      this.searchText =
this.route.snapshot.queryParamMap.get("s");

      this.ajax = this.articleService
        .getSearchArticles(this.searchText,
this.page, this.pageSize)
        .subscribe(data => {
          this.articles = data.articles;
```

```
        this.totalCount = data.totalCount;
    });
});
}
}
```

src\app\pages\main.module.ts

```
import { NgModule } from "@angular/core";
import { CommonModule } from
"@angular/common";

import { BrowserModule } from
"@angular/platform-browser";
import { HttpClientModule } from
"@angular/common/http";
import { AppRoutingModule } from "../app-
routing.module";
import { ComponentsModule } from
"../components/components.module";
import { MaterialModule } from
"../modules/material.module";

import { HomeComponent } from
"./home/home.component";
import { AboutMeComponent } from "./about-
me/about-me.component";
import { ContactComponent } from
"./contact/contact.component";
import { MainLayoutComponent } from
"../layout/main-layout/main-
layout.component";
import { MainNavComponent } from
"../nav/main-nav/main-nav.component";
import { ArticleComponent } from
"./article/article.component";
import { CategoryArticlesComponent } from
"./category-articles/category-
articles.component";
import { SearchComponent } from
"./search/search.component";
import { ArchiveComponent } from
"./archive/archive.component";
import { AdminLoginComponent } from "./admin-
login/admin-login.component";
import { FooterNavComponent } from
"../nav/footer-nav/footer-nav.component";
@NgModule({
  declarations: [
    MainLayoutComponent,
    MainNavComponent,
    HomeComponent,
    AboutMeComponent,
    ContactComponent,
    ArticleComponent,
    CategoryArticlesComponent,
    SearchComponent,
    ArchiveComponent,
    AdminLoginComponent,
    FooterNavComponent
  ],
```

	<pre>imports: [   CommonModule,   BrowserModule,   HttpClientModule,   AppRoutingModule,   ComponentsModule,   MaterialModule ] }) export class MainModule {}</pre>
--	---

src\app\Pipes\urlformat.pipe.ts

```
import { Pipe, PipeTransform } from
"@angular/core";

@Pipe({
  name: "urlformat"
})
export class UrlformatPipe implements
PipeTransform {
  transform(value: any, args?: any): any {
    value = value.toLowerCase();
    value =
value.replace(/[".*+?^${}()|[\]\\"/g, "-");
    value = value.replace(/\\s/g, "-");
    value = value.replace(/[ç]/g, "c");
    value = value.replace(/[ğ]/g, "g");
    value = value.replace(/[ı]/g, "i");
    value = value.replace(/[ö]/g, "o");
    value = value.replace(/[ş]/g, "s");
    value = value.replace(/[ü]/g, "ü");

    return value;
  }
}
```



# MODELS

src\app\models\archive.ts

```
export class Archive {  
  year: number;  
  month: number;  
  count: number;  
  monthName: string;  
}
```

src\app\models\article-pg.ts

```
import { Article } from "../article";

export class ArticlePg {
  totalCount: number;
  articles: Article[];
}
```

src\app\models\article.ts

```
import { Category } from "../category";

export class Article {
  id: number;
  title: string;
  contentMain: string;
  contentSummary: string;
  publishDate: Date;
  picture: string;
  viewCount: number;
  commentCount: number;
  category: Category;
}
```

src\app\models\category.ts

```
export class Category {  
  id: number;  
  name: string;  
}
```

src\app\models\comment.ts

```
export class Comment {  
  id: number;  
  articleId: number;  
  name: string;  
  contentMain: string;  
  publishDate: Date;  
}
```

