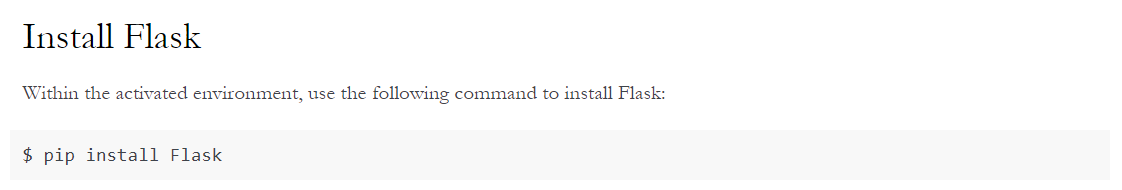
# Bölüm 19: Flask Frameworkle Web Geliştirme Temelleri

Flask, Python ile yazılmış bir mikro web çerçevesidir. Belirli araçlar veya kitaplıklar gerektirmediğinden mikro çerçeve olarak sınıflandırılır. Veritabanı soyutlama katmanı, form doğrulama veya önceden var olan üçüncü taraf kitaplıklarının ortak işlevler sağladığı diğer bileşenlere sahip değildir. (wiki)

<https://medium.com/kodlayan-nesil/flask-nedir-9364c1bb5f41>

<https://flask.palletsprojects.com/en/2.2.x/installation/#python-version>

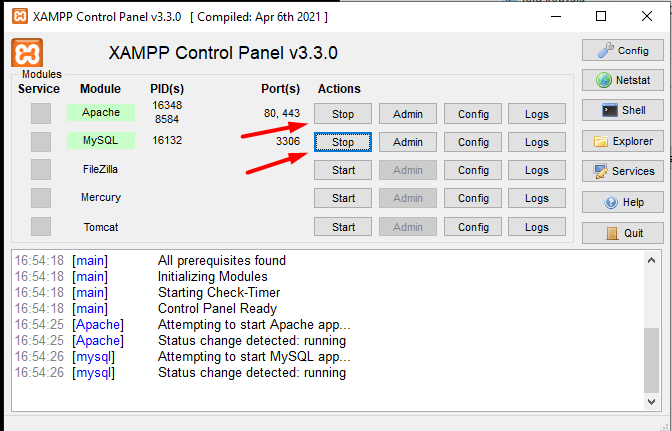
komut sistemine” $ pip install Flask“ yazıyoruz.

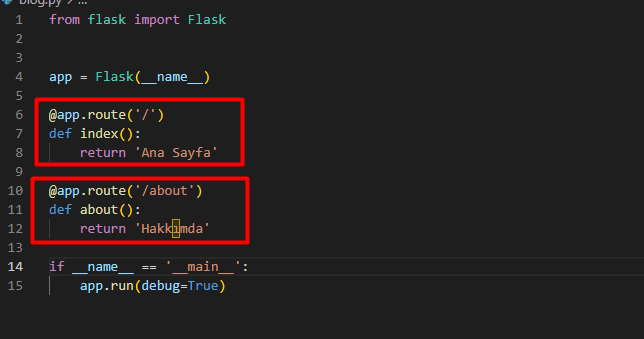


**XAMPP** bir web sunucusu yazılımıdır. Xampp server ile bilgisayara PHP, MariaDB, Perl ve Apache yanında FileZilla ve MercuryMail gibi sistemler kurularak hazır bir web sunucusu oluşturulabilmektedir. XAMPP serverda phpMyAdmin de kurulu olarak gelmektedir.

<https://www.apachefriends.org/tr/index.html>

Yükledikten sonra aramadan xampp control panel’i yazarak açıyoruz.





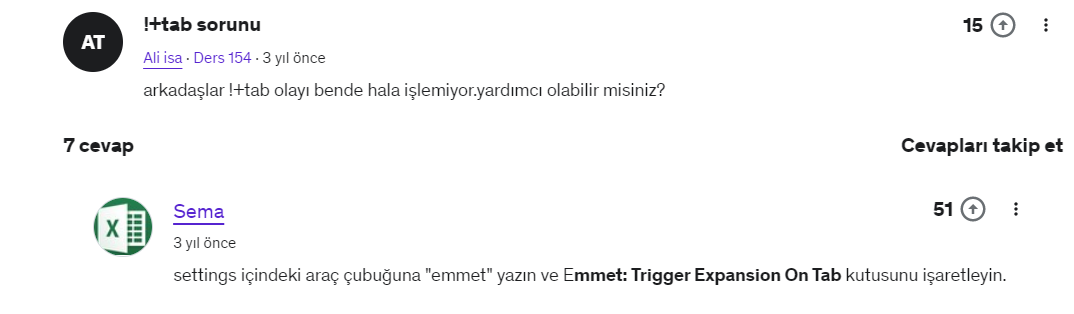
* Flask’ta url yapısının çalışması.

**Jinja,** Python programlama dili için oluşturulmuş bir tasarım motorudur. Django şablon motoruna benzer ancak, Python'a benzer ifadeler kullanır ve şablon dosyaları bir Sandbox içerisine konur. Diğer tasarım motorlarında olduğu gibi; Tasarım ve kodun birbirinden ayrıştırılması

from flask import Flask, render\_template

render\_template ‘i ekledik. Ardından templates diye bir klasör açtık. Render\_template otomatik olarak templates klasörüne bakıyor. Templates klasörüne index.html dosyası oluşturduk.

İndex.html’de !+tab ilk başta çalışmadı. Sorulardan cevabı bulundu.



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

</body>

</html>

Yukardakiler otomatik olarak dolduruldu.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Anasayfa</title>

</head>

<body>

    <h3>Anasayfa</h3>

</body>

</html>

Title ve paragraf başlıklarında “anasayfa” oluşturduk.

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

Ardından render\_template fonksiyonunu kullanıyoruz.

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    sayi=10

    sayi2=20

    return render\_template('index.html',number = sayi,number2=sayi2)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

python’da sayi ve sayi2 diye iki değişkeni oluşturup render template’te number ve number2 olarak tanımlıyoruz.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Anasayfa</title>

</head>

<body>

    <h3>Anasayfa</h3>

    <p>Burası anasayfadır.</p>

    <h5>Sayı</h5>

    <p>{{number}}</p>

    <p>{{number2}}</p>

</body>

</html>

Sayı başlığı oluşturarak n ve n2’yi çağırıyoruz. Python işlemi çağırmak için iki adet süslü parantez kullanıyoruz. {{ }}

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    article = dict()

    article['title'] = 'Deneme'

    article['body'] = 'Deneme 123'

    article['author'] = 'Yasin Burak Aydın'

    return render\_template('index.html',article=article)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

sözlük dizini oluşturduk.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Anasayfa</title>

</head>

<body>

    <h3>Anasayfa</h3>

    <p>Burası anasayfadır.</p>

    <p>{{article['title']}}</p>

    <p>{{article['body']}}</p>

    <p>{{article['author']}}</p>

</body>

</html>

Sözlükleri html’de çağırdık.

    <p>{{article.title}}</p>

    <p>{{article.body}}</p>

    <p>{{article.author}}</p>

Bu yöntemle de sözlükleri çağırabiliriz ve bu yöntem daha kolay.

**TEMPLATE INHERITANCE VE BLOKLAR**

Templates klasörüne gelip ‘layout.html’ oluşturuyoruz.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Layout</title>

</head>

<body>

    <h3>Layout</h3>

</body>

</html>

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('layout.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

Şimdi index.html ‘de yapacağımız işlem, layout.html’den miras (inheritance) almak olacak.

{% extends "layout.html" %}

Bu sayede index, layout’tan miras almış olacak ve

def index():

    return render\_template('index.html')

yazsak bile layout’u kullandığını göreceğiz.

## BLOK

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Layout</title>

</head>

<body>

    <h3>Layout</h3>

{% block yasbu %}

<p>Burası layouttur...</p>

{% endblock  %}

</body>

</html>

Yukarıda layout.html’de bir blok oluşturduk. İndex, layout’tan miras alacağı için ‘burası layouttur’ yazısını göreceğiz. Ancak index’te blokları değiştirip indexte yazan değeri görebiliriz.

{% extends "layout.html" %}

{% block yasbu  %}

<p>Burası indextir...</p>

{% endblock  %}

Artık sayfayı yenilediğimizde burası indextir diye gözükecektir.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>

        {% block title %}

        Layout

        {% endblock  %}

    </title>

</head>

<body>

    <h3>Layout</h3>

{% block yasbu %}

<p>Burası layouttur...</p>

{% endblock  %}

</body>

</html>

Yukarıda title bölümünde bir blok oluşturarak diğer bölümlerde overwrite yapabiliriz.

{% extends "layout.html" %}

{% block yasbu  %}

<p>Burası indextir...</p>

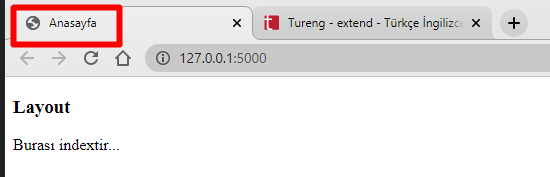
{% endblock  %}

{% block title %}

Anasayfa

{% endblock  %}

Yukarda index.html’ de title overwrite yaparak ‘Anasayfa’ yazmasını sağlamış oluyoruz.



## Bootstrap Kullanımı ve Navigation Bar

Templates’e yeni bir layout.html oluşturuyoruz.

<https://getbootstrap.com/docs/4.0/getting-started/introduction/> sitesine girerek bootstrapları yüklüyoruz.

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

Link’i metanın altına script’leri body nin içine yazıyoruz.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

    <title>YSB Blog</title>

</head>

<body>

    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

</body>

</html>

<https://getbootstrap.com/docs/4.0/examples/navbar-fixed/> sitesine girip sayfa kaynağını görüntüle dedikten sonra navbar kısmını alarak includes klasörünün içine oluşturuduğumuz navbar.html içine yazıyoruz.



<nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">

      <a class="navbar-brand" href="#">Fixed navbar</a>

      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">

        <span class="navbar-toggler-icon"></span>

      </button>

      <div class="collapse navbar-collapse" id="navbarCollapse">

        <ul class="navbar-nav mr-auto">

          <li class="nav-item active">

            <a class="nav-link" href="#">Home <span class="sr-only">(current)</span></a>

          </li>

          <li class="nav-item">

            <a class="nav-link" href="#">Link</a>

          </li>

          <li class="nav-item">

            <a class="nav-link disabled" href="#">Disabled</a>

          </li>

        </ul>

        <form class="form-inline mt-2 mt-md-0">

          <input class="form-control mr-sm-2" type="text" placeholder="Search" aria-label="Search">

          <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>

        </form>

      </div>

</nav>

Navbar.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

    <title>YSB Blog</title>

</head>

<body>

    {% include "includes/navbar.html" %}

    <br><br><br>

    <div class='container'>

        {% block body %}

        {% endblock body %}

    </div>

    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

</body>

</html>

Layout’ta {% include "includes/navbar.html" %} yazarak dahil ediyoruz.

EN SON HALİYLE

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

blog.py

{% include "layout.html" %}

{% block body %}

<h3>Ana Sayfa</h3>

{% endblock body %}

İndex.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

    <title>YSB Blog</title>

</head>

<body>

    {% include "includes/navbar.html" %}

    <br><br><br>

    <div class='container'>

        {% block body %}

        {% endblock body %}

    </div>

    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

</body>

</html>

Layout.html

## NAVİGATİON BAR ÖZELLEŞTİRME

<nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">

      <a class="navbar-brand" href="/">YSB BLOG</a>

      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">

        <span class="navbar-toggler-icon"></span>

      </button>

      <div class="collapse navbar-collapse" id="navbarCollapse">

        <ul class="navbar-nav mr-auto">

          <li class="nav-item active">

            <a class="nav-link" href="/about">Hakkımızda</a>

          </li>

          <li class="nav-item active">

            <a class="nav-link" href="/articles">Makaleler</a>

          </li>

        </ul>

        <ul class="navbar-nav ml-auto">

          <li class="nav-item active">

            <a class="nav-link" href="/register">Kayıt Olun</a>

          </li>

          <li class="nav-item active">

            <a class="nav-link" href="/login">Giriş</a>

          </li>

        </ul>

      </div>

</nav>

## HAKKIMDA SAYFASI OLUŞTURMA

Templates dosyasına about.html oluşturuyoruz. About’a layout.html’i extend edeceğiz.

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html')

@app.route('/about')

def about():

    return render\_template('about.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

blog.py

{% extends "layout.html" %}

{% block body %}

<h3>Hakkımda</h3>

<hr>

Sorunlar, sıkıntılar ve çalkantılarla dolu bir yılın daha sonuna geldik. Ama hayatın tüm curcunası ve bilinmezliği bir yana, son 400 yıldır değişmeyen bir gerçek, hayatımızın bir sabiti olarak yoluna devam etti: Bilim. 2022, birbirinden harika bilimsel gelişmelere sahne oldu ve tahmin ediyorum önemli bir bölümü, Türkiye'nin vıcık vıcık gündemi içinde eriyip kayboldu. O nedenle gelin bugün sizlerle bilim camiasında bu yıl yaşanan önemli gelişmelere bir göz atalım ve 2022'de bilim sayesinde karanlığın içinde nereleri fethetmişiz bir gözden geçirelim.

<hr>

{% endblock body %}

about.html

## TANITIM SAYFASI OLUŞTURMA

{% extends "layout.html" %}

{% block body %}

<div class='jumbotron text-center'>

    <h3>Veri Bilimi Bloğuna Hoşgeldiniz</h3>

    <p>Bu blokta veri bilimci olarak yazılarınızı paylaşabilirsiniz.</p>

</div>

{% endblock body %}

İndex.html

## TEMPLATELARDA KOŞUL KULLANIMI

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html',answer = 'evet')

@app.route('/about')

def about():

    return render\_template('about.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

yukarda def index():’te answer=’evet’ olarak koşul tanımladık.

{% extends "layout.html" %}

{% block body %}

<div class='jumbotron text-center'>

    <h3>Veri Bilimi Bloğuna Hoşgeldiniz</h3>

    <p>Bu blokta veri bilimci olarak yazılarınızı paylaşabilirsiniz.</p>

</div>

{% if answer == 'evet' %}

<p>Cevabımız Evet...</p>

{% else %}

<p>Cevabımız Evet değil...</p>

{% endif %}

{% endblock body %}

İndex.html içinde bir if koşulu yarattık. Answer’ı evet eşit durumunu ve else durumunu girdik. Def index’te answer koşulu evet olduğundan ‘Cevabımız Evet…’ yazısını göreceğiz. Eğer başka bir cevap girilseydi ‘Cevabımız Evet değil… i görecektik.

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html',islem = 1)

@app.route('/about')

def about():

    return render\_template('about.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

{% extends "layout.html" %}

{% block body %}

<div class='jumbotron text-center'>

    <h3>Veri Bilimi Bloğuna Hoşgeldiniz</h3>

    <p>Bu blokta veri bilimci olarak yazılarınızı paylaşabilirsiniz.</p>

</div>

{% if islem == 1 %}

<p>1. işlem</p>

{% elif islem == 2 %}

<p>2. işlem</p>

{% else %}

<p>Geçersiz İşlem</p>

{% endif %}

{% endblock body %}

Koşullu durumlar için bu sefer işlemlere elif’i ekliyoruz.

## TEMPLATELARDA DÖNGÜ KULLANIMI

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    numbers = [1,2,3,4,5]

    return render\_template('index.html',numbers = numbers)

@app.route('/about')

def about():

    return render\_template('about.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

{% extends "layout.html" %}

{% block body %}

<div class='jumbotron text-center'>

    <h3>Veri Bilimi Bloğuna Hoşgeldiniz</h3>

    <p>Bu blokta veri bilimci olarak yazılarınızı paylaşabilirsiniz.</p>

</div>

<ul>

    {% for number in numbers %}

        <li>{{number}}</li>

    {% endfor %}

</ul>

{% endblock body %}

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    articles = [

        {'id':1, 'title':'Deneme1','content':'Deneme1 içerik'},

        {'id':2, 'title':'Deneme2','content':'Deneme2 içerik'},

        {'id':3, 'title':'Deneme3','content':'Deneme3 içerik'},

    ]

    return render\_template('index.html',articles = articles)

@app.route('/about')

def about():

    return render\_template('about.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

{% extends "layout.html" %}

{% block body %}

<div class='jumbotron text-center'>

    <h3>Veri Bilimi Bloğuna Hoşgeldiniz</h3>

    <p>Bu blokta veri bilimci olarak yazılarınızı paylaşabilirsiniz.</p>

</div>

<ul>

    {% for article in articles %}

        <li>Article ID: {{article.id}} - Title: {{article.title}} - Content: {{article.content}} </li>

    {% endfor %}

</ul>

{% endblock body %}

## FLASK DİNAMİK URL YAPISI

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html',)

@app.route('/about')

def about():

    return render\_template('about.html')

@app.route('/article/<string:id>')

def detail(id):

    return 'Article Id:' + id

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

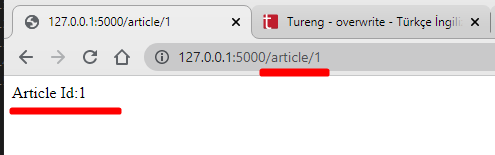
“

@app.route('/article/<string:id>')

def detail(id):

    return 'Article Id:' + id

“ def detail(id): yazarak dinamikleştirmiş olduk.





Makalelerin id sini yakalayarak url leri atayabiliriz.

## GEREKLİ MODÜLLER , MYSQL VERİTABANI OLUŞTURMA

pip install flask-mysqldb

pip install Flask-WTF

pip install passlib

from flask import Flask, render\_template,flash,redirect,url\_for,session,logging,request

from flask\_mysqldb import MySQL

from wtforms import Form,StringField,TextAreaField,PasswordField,validators

from passlib.hash import sha256\_crypt

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return render\_template('index.html',)

@app.route('/about')

def about():

    return render\_template('about.html')

@app.route('/article/<string:id>')

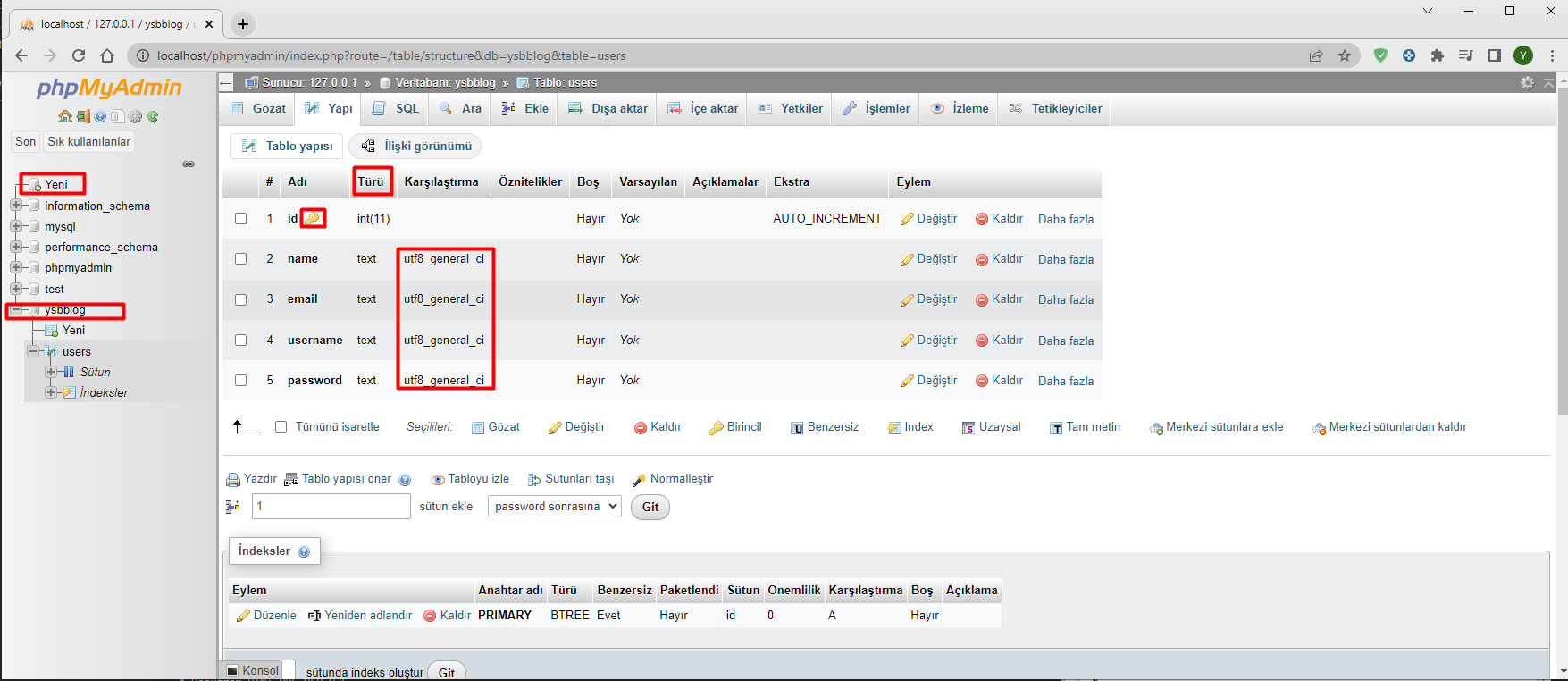
def detail(id):

    return 'Article Id:' + id

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

XAMPP control panelinden mysql admini açıyoruz ve veritabanı oluşturuyoruz.



## FLASK İLE MYSQL KONFİGURASYONU

<https://flask-mysqldb.readthedocs.io/en/latest/>

from flask import Flask, render\_template,flash,redirect,url\_for,session,logging,request

from flask\_mysqldb import MySQL

from wtforms import Form,StringField,TextAreaField,PasswordField,validators

from passlib.hash import sha256\_crypt

app = Flask(\_\_name\_\_)

app.config['MYSQL\_HOST'] = 'localhost'

app.config['MYSQL\_USER'] = 'root'

app.config['MYSQL\_PASSWORD'] = ''

app.config['MYSQL\_DB'] = 'ysbblog'

app.config['MYSQL\_CURSORCLASS'] = 'DictCursor'

mysql = MySQL(app)

@app.route('/')

def index():

    return render\_template('index.html',)

@app.route('/about')

def about():

    return render\_template('about.html')

@app.route('/article/<string:id>')

def detail(id):

    return 'Article Id:' + id

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

## WTFORMS İLE KULLANICI KAYIT FORMU

Kullanıcı kayıt formunu html’den değil python class sayesinde yapacağız.

<https://wtforms.readthedocs.io/en/3.0.x/>

<https://flask-wtf.readthedocs.io/en/1.0.x/quickstart/#creating-forms>

from flask import Flask, render\_template,flash,redirect,url\_for,session,logging,request

from flask\_mysqldb import MySQL

from wtforms import Form,StringField,TextAreaField,PasswordField,validators

from passlib.hash import sha256\_crypt

#Kullanıcı kayıt formu

class RegisterForm(Form):

    name = StringField('İsim Soyisim', validators=[validators.Length(min=4,max=25)])

    username = StringField('Kullanıcı Adı', validators=[validators.Length(min=5,max=35)])

    email = StringField('Email Adresi', validators=[validators.Email(message='Lütfen Geçerli Bir Email Adresi Girin...')])

    password = PasswordField('Parola:', validators=[

        validators.DataRequired(message='Lütfen Parola Belirleyin'),

        validators.EqualTo(fieldname='confirm',message='Parolanız Uyuşmuyor...')

    ])

    confirm = PasswordField('Parola Doğrula')

app = Flask(\_\_name\_\_)

app.config['MYSQL\_HOST'] = 'localhost'

app.config['MYSQL\_USER'] = 'root'

app.config['MYSQL\_PASSWORD'] = ''

app.config['MYSQL\_DB'] = 'ysbblog'

app.config['MYSQL\_CURSORCLASS'] = 'DictCursor'

mysql = MySQL(app)

@app.route('/')

def index():

    return render\_template('index.html',)

@app.route('/about')

def about():

    return render\_template('about.html')

@app.route('/article/<string:id>')

def detail(id):

    return 'Article Id:' + id

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

email = StringField('Email Adresi', validators=[validators.Email(message='Lütfen Geçerli Bir Email Adresi Girin...')])

çalışmado onun yerine aşağıdaki kullanıldı

<https://wtforms.readthedocs.io/en/2.3.x/crash_course/>

<https://flask.palletsprojects.com/en/2.2.x/patterns/wtforms/>

<https://wtforms.readthedocs.io/en/2.3.x/validators/>

email = StringField('Email Address', [validators.Length(min=6, max=35)])

**REGİSTER SAYFASI OLUŞTURMA**

from flask import Flask, render\_template,flash,redirect,url\_for,session,logging,request

request sayesinde postrequest mi getrequest mi olduğunu öğrenebiliriz. Bunu flask’a belirtmemiz lazım.

<https://flask.palletsprojects.com/en/2.2.x/patterns/wtforms/>

<https://getbootstrap.com/docs/4.0/components/buttons/>

{% macro render\_field(field) %}

  <dt>{{ field.label }}

  <dd>{{ field(\*\*kwargs)|safe }}

  {% if field.errors %}

    <ul class=errors>

    {% for error in field.errors %}

      <li>{{ error }}</li>

    {% endfor %}

    </ul>

  {% endif %}

  </dd>

{% endmacro %}

includes/formhelpers.html

{% extends "layout.html" %}

{% block body %}

{% from "includes/formhelpers.html" import render\_field %}

<h3>Kayıt Olun</h3>

<hr>

<form method = 'POST'>

    <dl>

        {{ render\_field(form.name,class ='form-control')}}

        {{ render\_field(form.username,class ='form-control') }}

        {{ render\_field(form.email,class ='form-control') }}

        {{ render\_field(form.password,class ='form-control') }}

        {{ render\_field(form.confirm,class ='form-control') }}

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

    </dl>

</form>

{% endblock body %}

Register.html

@app.route('/')

def index():

    return render\_template('index.html',)

@app.route('/about')

def about():

    return render\_template('about.html')

@app.route('/article/<string:id>')

def detail(id):

    return 'Article Id:' + id

#Kayıt olma

@app.route('/register',methods=['GET','POST'])

def register():

    form = RegistrationForm(request.form)

    if request.method == 'POST':

        return redirect(url\_for('index'))

    else:

        return render\_template('register.html',form=form)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

**\*\*\*\*ÇOK KARIŞIK BİR KONU. TEKRAR TEKRAR DİNLEMEK VE İZLEMEK GEREKİYOR.**

## KULLANICININ VERİTABANINA KAYIT EDİLMESİ

from flask import Flask, render\_template,flash,redirect,url\_for,session,logging,request

from flask\_mysqldb import MySQL

from wtforms import Form,StringField,TextAreaField,PasswordField,validators, EmailField

from wtforms.validators import InputRequired, Email

from passlib.hash import sha256\_crypt

#Kullanıcı kayıt formu

class RegistrationForm(Form):

    name = StringField('İsim Soyisim', validators=[validators.Length(min=4,max=25)])

    username = StringField('Kullanıcı Adı', validators=[validators.Length(min=5,max=35)])

    email = StringField('Email Address', [validators.Length(min=6, max=35)])

    password = PasswordField('Parola:', validators=[

        validators.DataRequired(message='Lütfen Parola Belirleyin'),

        validators.EqualTo(fieldname='confirm',message='Parolanız Uyuşmuyor...')

    ])

    confirm = PasswordField('Parola Doğrula')

app = Flask(\_\_name\_\_)

app.config['MYSQL\_HOST'] = 'localhost'

app.config['MYSQL\_USER'] = 'root'

app.config['MYSQL\_PASSWORD'] = ''

app.config['MYSQL\_DB'] = 'ysbblog'

app.config['MYSQL\_CURSORCLASS'] = 'DictCursor'

mysql = MySQL(app)

@app.route('/')

def index():

    return render\_template('index.html',)

@app.route('/about')

def about():

    return render\_template('about.html')

@app.route('/article/<string:id>')

def detail(id):

    return 'Article Id:' + id

#Kayıt olma

@app.route('/register',methods=['GET','POST'])

def register():

    form = RegistrationForm(request.form)

    if request.method == 'POST' and form.validate(): #form.validate() ile formları denetlemeyi sağladık

        name = form.name.data           #formdaki bilgileri aldık ve name'a eşitledik

        username = form.username.data

        email = form.email.data

        password = sha256\_crypt.encrypt(form.password.data) #password'u şifrledik

        cursor = mysql.connection.cursor() #phpmyadmin üzerinde işlem yapmak cursor'u oluşturduk

        sorgu = 'INSERT into users(name,email,username,password) VALUES(%s,%s,%s,%s)' #name,user,..,..  %s'lerin yerine geçecek

        cursor.execute(sorgu,(name,username,email,password))

        mysql.connection.commit() #veri tabanında güncelleme yaptığımız için commit yapılması gerekiyor.

        cursor.close() #cursor'ı kapatarak boşa çalışmasından kaçınıyoruz

        return redirect(url\_for('index'))

    else:

        return render\_template('register.html',form=form)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

blog.py

if request.method == 'POST' and form.validate(): #form.validate() ile formları denetlemeyi sağladık

        name = form.name.data           #formdaki bilgileri aldık ve name'a eşitledik

        username = form.username.data

        email = form.email.data

        password = sha256\_crypt.encrypt(form.password.data) #password'u şifrledik

        cursor = mysql.connection.cursor() #phpmyadmin üzerinde işlem yapmak cursor'u oluşturduk

        sorgu = 'INSERT into users(name,email,username,password) VALUES(%s,%s,%s,%s)' #name,user,..,..  %s'lerin yerine geçecek

        cursor.execute(sorgu,(name,username,email,password))

        mysql.connection.commit() #veri tabanında güncelleme yaptığımız için commit yapılması gerekiyor.

        cursor.close() #cursor'ı kapatarak boşa çalışmasından kaçınıyoruz

yapılan işlemler

## FLASH MESAJLARINI KULLANMA

<https://flask.palletsprojects.com/en/0.12.x/patterns/flashing/>

Kullanıcıya belli bir feedback vermek için kullanacağız.

{% with messages = get\_flashed\_messages(with\_categories=true) %}

  {% if messages %}

    <ul class=flashes>

    {% for category, message in messages %}

      <li class="{{ category }}">{{ message }}</li>

    {% endfor %}

    </ul>

  {% endif %}

{% endwith %}

Yukardaki siteden kodları includes klasörüne messages.html dosyasına kaydettik.

<div class='container'>

        {% include "includes/messages.html" %}

            {% block body %}

            {% endblock body %}

    </div>

(layout.html) ardından layout.html dosyasının container ın altına messeges.html’i include ettik.

app = Flask(\_\_name\_\_)

app.secret\_key='ysbblog'

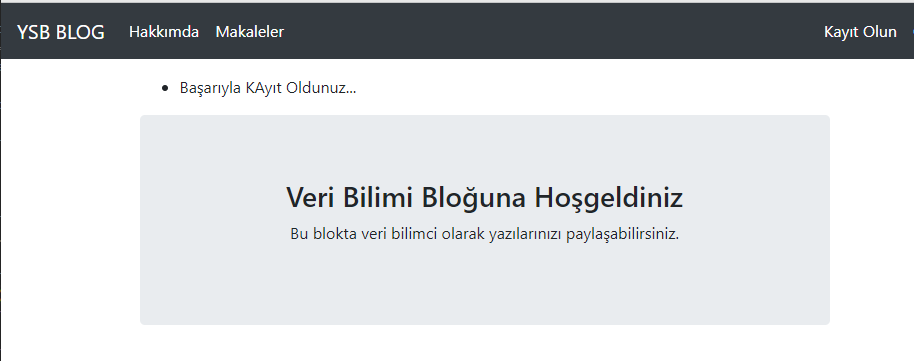
(blog.py) yukardaki secret key oluşturarak hata almamızı engelliyoruz.

cursor.close() #cursor'ı kapatarak boşa çalışmasından kaçınıyoruz

        flash('Başarıyla Kayıt Oldunuz...',category='success')

        return redirect(url\_for('index'))

(blog.py)



Çıkan mesajı daha güzel bir hale getirelim.

<https://getbootstrap.com/docs/4.0/components/alerts/>

<div class="alert alert-success" role="alert">

This is a success alert—check it out!

</div>

Yukardaki siteden aldık

{% with messages = get\_flashed\_messages(with\_categories=true) %}

  {% if messages %}

    {% for category, message in messages %}

    <div class="alert alert-{{category}}" role="alert">

    {{message}}

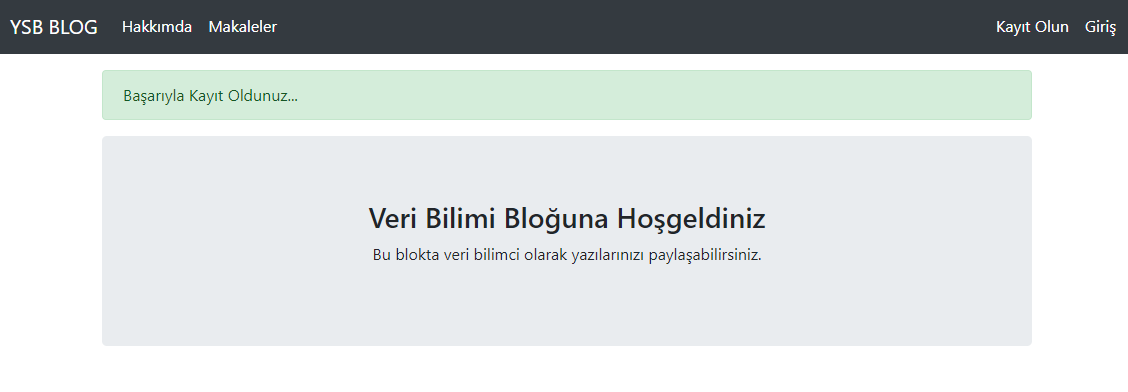
    </div>

    {% endfor %}

  {% endif %}

{% endwith %}

(messages.html) Gerekli değişiklikler yapıldı.



Sayfayı yenileyince mesaj gider. Çünkü tek seferlik mesajlardır.

## LOGİN URL VE LOGİN FORMUN TEMPLATEDA GÖSTERİLMESİ

{% extends "layout.html" %}

{% block body %}

<h3>Giriş Yap</h3>

<hr>

{% from "includes/formhelpers.html" import render\_field %}

<form method = 'POST'>

    <dl>

        {{ render\_field(form.username) }}

        {{ render\_field(form.password) }}

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

    </dl>

</form>

{% endblock body %}

Login.html

#Login İşlemi

@app.route('/login',methods=['GET','POST'])

def login():

    form=LoginForm(request.form)

    return render\_template('login.html',form=form)

blog.py

return redirect(url\_for('login'))

kayıt olmada get url’yi indexten login sayfası olarak değiştirdik.

## KULLANICI GİRİŞİ YAPMA

#Login İşlemi

@app.route('/login',methods=['GET','POST'])

def login():

    form=LoginForm(request.form)

    if request.method == 'POST':

        username=form.username.data

        password\_entered=form.password.data

        cursor = mysql.connection.cursor()

        sorgu = 'SELECT \* FROM users where username = %s'

        result = cursor.execute(sorgu,(username,))

        if result > 0:

            data = cursor.fetchone()

            real\_password = data['password']

            if sha256\_crypt.verify(password\_entered,real\_password):

                flash('Başarıyla Giriş Yaptınız...','success')

                return redirect(url\_for('index'))

            else:

                flash('Parolanızı Yanlış Girdiniz...','danger')

                return redirect(url\_for('login'))

        else:

            flash('Kullanıcı Adı Bulunamadı...','danger') #dangerla bağladık

            return redirect(url\_for('login')) #tekrar login ekranına döndük

    return render\_template('login.html',form=form)

blog.py

## SESSİON KONTROLÜ (Oturum Kontorlü)

Session kontrolü dict kullanımına benziyor.

            if sha256\_crypt.verify(password\_entered,real\_password):

                flash('Başarıyla Giriş Yaptınız...','success')

                session['logged\_in'] = True

                session['username'] = username

                return redirect(url\_for('index'))

blog.py

        <ul class="navbar-nav ml-auto">

          <li class="nav-item active">

            <a class="nav-link" href="/logout">Çıkış Yap</a>

          </li>

Navbar.html Çıkış yap barını ekliyoruz.

Şimdi yapmak istediğimiz, çıkış yapı sadece giriş yaptığımızda göstermek ve giriş yapı sadece çıkış yaptığımızda göstermek.

       <ul class="navbar-nav ml-auto">

          {% if session['logged\_in'] %}

            <li class="nav-item active">

              <a class="nav-link" href="/logout">Çıkış Yap</a>

            </li>

          {% else %}

            <li class="nav-item active">

              <a class="nav-link" href="/login">Giriş Yap</a>

            </li>

          {% endif %}

          <li class="nav-item active">

            <a class="nav-link" href="/register">Kayıt Olun</a>

          </li>

        </ul>

Navbar.html. if else kontrolünü sessionda sağladık. Giriş ve çıkış yapları duruma göre göreceğiz artık.

Şimdi bir logout işlemi gerçekleştireceğiz.

@app.route('/logout')

def logout():

    session.clear() #session'ı sonlandıracak

    return redirect(url\_for('index'))

blog.py

## MAKALELER İÇİN KONTROL PANELİ OLUŞTURMA

          {% if session['logged\_in'] %}

            <li class="nav-item active">

              <a class="nav-link" href="/dashboard">Kontrol Paneli</a>

            </li>

Navbar.html, log in durumunda dashboard ekliyoruz.

@app.route('/dashboard')

def dashboard():

    return render\_template('dashboard.html')

blog.py

{% extends "layout.html" %}

{% block body %}

<h3>Kontrol Paneli</h3>

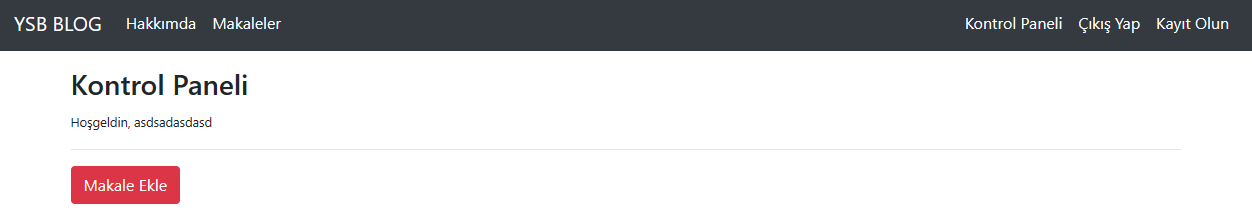
<small>Hoşgeldin, {{session['username']}}</small>

<hr>

<a href="/addarticle" class='btn btn-danger'>Makale Ekle</a>

{% endblock body %}

Dashboard.html, makale ekle diye button yaptık



## DECORATORLAR İLE KULLANICI GİRİŞ KONTROLÜ

Giriş yapılmadığı halde url’ye /dashboard yazarak girebiliyoruz. Bunları decoratorlar ile engelleyeceğiz.

[View Decorators — Flask Documentation (2.2.x) (palletsprojects.com)](https://flask.palletsprojects.com/en/2.2.x/patterns/viewdecorators/)

from functools import wraps

#Kullanıcı Giriş Dacorator'ı

def login\_required(f):

    @wraps(f)

    def decorated\_function(\*args, \*\*kwargs):

        if 'logged\_in' in session:

            return f(\*args, \*\*kwargs)

        else:

            flash('Bu sayfayı görüntülemek için lütfen giriş yapın...', 'danger')

            return redirect(url\_for('login'))

    return decorated\_function

blog.py

@app.route('/dashboard')

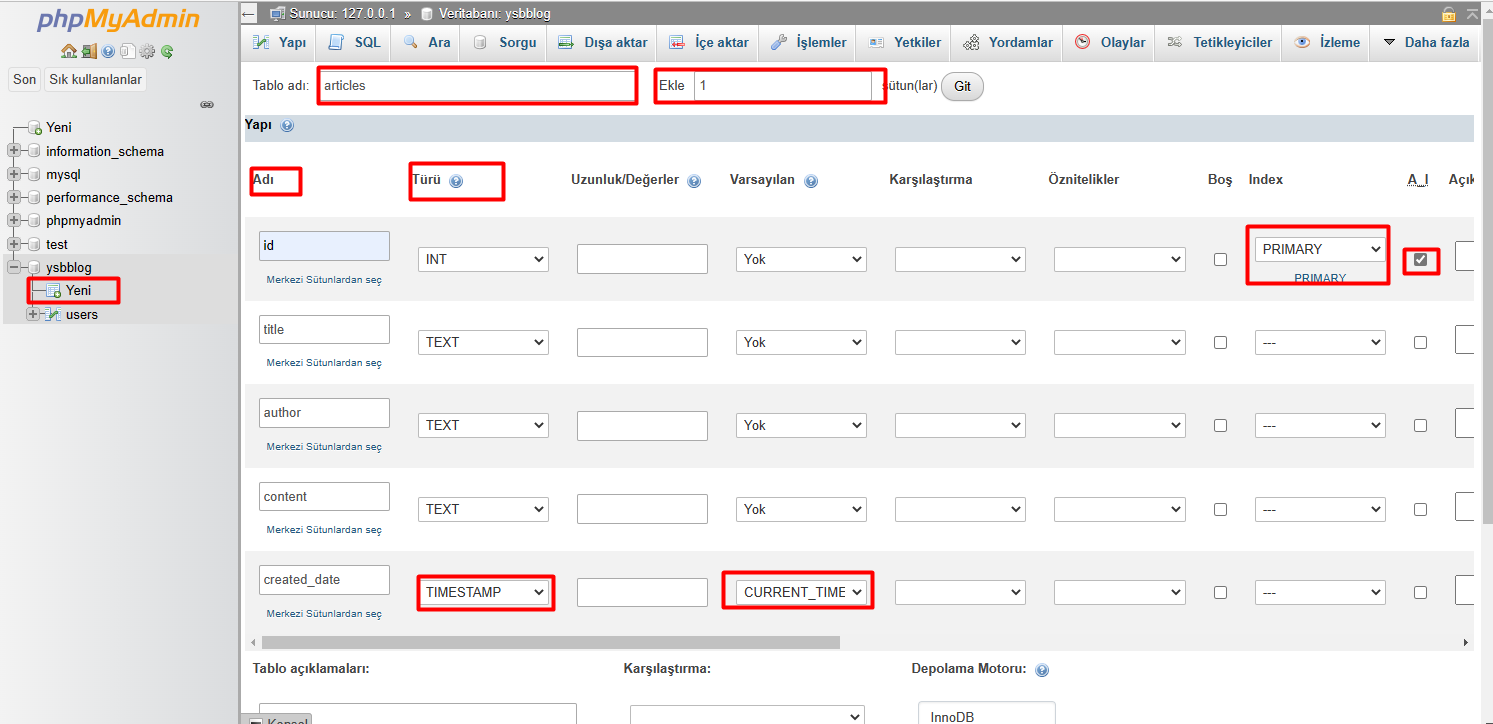
@login\_required

def dashboard():

    return render\_template('dashboard.html')

blog.py

## MAKALE TABLOSUNUN OLUŞTURULMASI



## MAKALE EKLEME SAYFASI VE MAKALE FORMU OLUŞTURMA

{% extends "layout.html" %}

{% block body %}

<h3>Makale Oluştur</h3>

<hr>

{% endblock body %}

addarticle.html oluşturuldu

#Makale Ekleme

@app.route('/addarticle', methods = ['GET','POST'])

def addarticle():

    return render\_template('addarticle.html')

#Makele Form

class AricleForm(Form):

    title = StringField('Makale Başlığı',validators=[validators.length(min=5,max=100)])

    content = TextAreaField('Makale İçeriği',validators=[validators.length(min=10)]) #Uzun bir yazı olacağından textareafield

blog.py

#Makale Ekleme

@app.route('/addarticle', methods = ['GET','POST'])

def addarticle():

    form = ArticleForm(request.form)

    return render\_template('addarticle.html',form=form)

yapılanları addarticle.html’de göstermek için yukardaki ekleme yapıldı.

{% from "includes/formhelpers.html" import render\_field %}

<form method = 'POST'>

    <dl>

        {{ render\_field(form.title, class='form-control') }}

        {{ render\_field(form.content, class='form-control') }}

        <button type="submit" class="btn btn-danger">Makale Ekle</button>

    </dl>

</form>

addarticle.html

## MAKALEYİ VERİTABANINA KAYDETME

İsteğin get mi post mu olduğunu anlayacak kontrolü yapacağız.

#Makale Ekleme

@app.route('/addarticle', methods = ['GET','POST'])

def addarticle():

    form = ArticleForm(request.form)

    if request.method == 'POST' and form.validate():

        title = form.title.data

        content = form.content.data

        cursor = mysql.connection.cursor()

        sorgu ='Insert into articles(title,author,content) VALUES(%s,%s,%s)'

        cursor.execute(sorgu,(title,session['username'],content))

        mysql.connection.commit()

        cursor.close()

        flash('Makale Başarıyla Eklendi','succes')

        return redirect(url\_for('dashboard'))

    return render\_template('addarticle.html',form=form)

blog.py

## CKEDİTOR KULLANIMI

[CKEditor CDN](https://cdn.ckeditor.com/)

#Makele Form

class ArticleForm(Form):

    title = StringField('Makale Başlığı',validators=[validators.length(min=5,max=100)])

    content = TextAreaField('Makale İçeriği',validators=[validators.length(min=10)]) #Uzun bir yazı olacağından textareafield

textarea content isminde yer almakta. Blog.py

<script src="https://cdn.ckeditor.com/ckeditor5/35.4.0/classic/ckeditor.js"></script>

    <script>

                        ClassicEditor

                                .create( document.querySelector( '#content' ) )

                                .then( editor => {

                                        console.log( editor );

                                } )

                                .catch( error => {

                                        console.error( error );

                                } );

        </script>

Layout.html. yukardaki linkte yer alan scriptleri kopyalayıp ilgili ismi content olarak değiştidik.

## MAKALE SAYFASINDA MAKALELERİ GÖRÜNTÜLEME

[List group · Bootstrap (getbootstrap.com)](https://getbootstrap.com/docs/4.0/components/list-group/)

{% extends "layout.html" %}

{% block body %}

<h3>Makaleler</h3>

<hr>

{% if articles %}

<ul class="list-group">

  {% for article in articles %}

  <li class="list-group-item">{{article.title}}</li>

  {% endfor %}

</ul>

{% else %}

<div class='alert alert-danger'>Bu blokta henüz makale bulunmuyor</div>

{% endif %}

{% endblock body %}

Articles.html

#Makale Sayfası

@app.route('/articles')

def articles():

    cursor = mysql.connection.cursor()

    sorgu = 'SELECT \* FROM articles'

    result = cursor.execute(sorgu)

    if result > 0:

        articles = cursor.fetchall()

        return render\_template('articles.html',articles=articles)

    else:

        return render\_template('articles.html')

blog.py

## MAKALELERİN KONTROL PANELİNDE GÖSTERİLMESİ

[Tables · Bootstrap (getbootstrap.com)](https://getbootstrap.com/docs/4.1/content/tables/)

@app.route('/dashboard')

@login\_required

def dashboard():

    cursor = mysql.connection.cursor()

    sorgu = 'SELECT \* FROM articles where author = %s'

    result = cursor.execute(sorgu,(session['username'],))

    if result > 0:

        articles = cursor.fetchall()

        return render\_template('dashboard.html',articles=articles)

    else:

        return render\_template('dashboard.html')

blog.py

{% if articles %}

<table class="table table-hover">

  <thead>

    <tr>

      <th scope="col">id</th>

      <th scope="col">Başlık</th>

      <th scope="col">Yazar</th>

      <th scope="col">Tarih</th>

      <th scope="col"></th>

      <th scope="col"></th>

    </tr>

  </thead>

  <tbody>

    {% for article in articles %}

    <tr>

      <th scope="row">{{article.id}}</th>

      <td><a href='/article/{{article.id}}'>{{article.title}}</a></td>

      <td>{{article.author}}</td>

      <td>{{article.created\_date}}</td>

      <td><a href='/edit/{{article.id}}' class ='btn btn-danger'>Güncelle</a></td>

      <td><a href='/delete/{{article.id}}' class ='btn btn-danger'>Sil</a></td>

    </tr>

    {% endfor %}

  </tbody>

</table>

{% else %}

<div class='alert alert-danger'>Henüz makaleniz bulunmuyor</div>

{% endif %}

Dashboard.html

## MAKALE DETAY SAYFASI

#Detay Sayfası

@app.route('/article/<string:id>') #dinamik id

def article(id):

    cursor = mysql.connection.cursor()

    sorgu = 'SELECT \* FROM articles where id = %s'

    result = cursor.execute(sorgu,(id),)

    if result > 0:

        article = cursor.fetchone()

        return render\_template('article.html',article=article)

    else:

        return render\_template('article.html')

blog.py

{% extends "layout.html" %}

{% block body %}

{% if article %}

<h4>{{article.title}}</h4>

<hr>

Yazar: {{article.author}}

<hr>

Tarih: {{article.created\_date}}

<hr>

{{article.content | safe}} <!--safe yazarak html içeriğini düzeltiyor -->

{% else %}

<div class='alert alert-warning'>

    Böyle bir makale bulunmuyor...

</div>

{% endif %}

{% endblock body %}

Article.html (articles.html değil) !!safe önemli

## MAKALE SİLME

Giriş yapmadan delete ‘e gitmememiz gerekiyor. Başkasının makalesini silmememiz gerekiyor.

#Makale Silme

@app.route('/delete/<string:id>') #dinamik id

@login\_required

def delete(id):

    cursor = mysql.connection.cursor()

    sorgu = 'SELECT \* FROM articles WHERE author = %s and id = %s'

    result= cursor.execute(sorgu,(session['username'],id))

    if result>0:

        sorgu2 = 'DELETE from articles WHERE id = %s'

        cursor.execute(sorgu2,(id,))

        mysql.connection.commit()

        return redirect(url\_for('dashboard'))

    else:

        flash('Bu makaleyi silme yetkiniz yok veya böyle bir makale yok','danger')

        return redirect(url\_for('index'))

blog.py

## MAKALE GÜNCELLEME

#Makale Güncelleme

@app.route('/edit/<string:id>',methods =['GET','POST'])

@login\_required

def update(id): #dinamik url'den dolayı id geliyor

    if request.method == 'GET': #get mi post mu kontrolünü yapacağız

        cursor = mysql.connection.cursor()

        sorgu = 'SELECT \* FROM articles WHERE author = %s and id = %s'

        result= cursor.execute(sorgu,(session['username'],id))

        if result==0:

            flash('Böyle bir makale yok veya işleme yetkiniz yok','danger')

            return redirect(url\_for('index'))

        else:

            article = cursor.fetchone()

            form = ArticleForm()

            form.title.data = article['title']

            form.content.data = article['content']

            return render\_template('update.html',form=form)

    else: #POST REQUEST

        form = ArticleForm(request.form)

        newTitle = form.title.data

        newContent = form.content.data

        sorgu2 = 'UPDATE articles SET title = %s, content=%s WHERE id = %s'

        cursor=mysql.connection.cursor()

        cursor.execute(sorgu2,(newTitle,newContent,id))

        mysql.connection.commit()

        flash('Makale Başarıyla Güncellendir','success')

        return redirect(url\_for('dashboard'))

blog.py

{% extends "layout.html" %}

{% block body %}

<h3>Makale Güncelle</h3>

<hr>

{% from "includes/formhelpers.html" import render\_field %}

<form method = 'POST'>

    <dl>

        {{ render\_field(form.title, class='form-control') }}

        {{ render\_field(form.content, class='form-control') }}

        <button type="submit" class="btn btn-danger">Güncelle</button>

    </dl>

</form>

{% endblock body %}

Update.html

## MAKALE ARAMA

[SQL LIKE Operator (w3schools.com)](https://www.w3schools.com/sql/sql_like.asp)

[Bootstrap Search Examples (bootsnipp.com)](https://bootsnipp.com/tags/search)

{% if articles %}

<form action="/search" method='post'>

  <input type="text" class="input-sm" maxlength="64" placeholder="Ara" />

  <button type="submit" class="btn btn-danger">Ara</button>

</form>

Articles.html

{% extends "layout.html" %}

{% block body %}

<h3>Makaleler</h3>

<hr>

{% if articles %}

<form action="/search" method='post'>

  <input type="text" name = 'keyword' class="input-sm" maxlength="64" placeholder="Ara" />

  <button type="submit" class="btn btn-danger">Ara</button>

</form>

<hr>

<ul class="list-group">

  {% for article in articles %}

  <li class="list-group-item"><a href='/article/{{article.id}}'>{{article.title}}</a></li>

  {% endfor %}

</ul>

{% else %}

<div class='alert alert-danger'>Bu blokta henüz makale bulunmuyor</div>

{% endif %}

{% endblock body %}

Articles.html

#Arama URL

@app.route('/search', methods = ['GET','POST']) #sadece post requestte çalışması lazım.

def search():

    if request.method == 'GET':

        return redirect(url\_for('index'))

    else:

        keyword = request.form.get('keyword')

        cursor = mysql.connection.cursor()

        sorgu = "SELECT \* FROM articles WHERE title like '%" + keyword +"%'"

        result =cursor.execute(sorgu)

        if result==0:

            flash('Aranan Kelimeye Uygun Makale Bulunamadı','warning')

            return redirect(url\_for('articles'))

        else:

            articles=cursor.fetchall()

            return render\_template('articles.html',articles=articles)

blog.py

## BLOGA CODE SNİPPET ÖZELLİĞİ EKLEME

[googlearchive/code-prettify: An embeddable script that makes source-code snippets in HTML prettier. (github.com)](https://github.com/googlearchive/code-prettify)

<script src="https://cdn.jsdelivr.net/gh/google/code-prettify@master/loader/run\_prettify.js"></script>

Bizim kullandığımız ck editörü farklı olduğundan kod yazma kısmı yoktu.

## BUNDAN SONRA NASIL İLERLEYEBİLİRSİNİZ ?

Arayüzü geliştirmek için flask’ı araştırın ve bloğu geliştirin.