Gizem Avcı 21360859071

NodeJs ile Web Programlama Hafta-9 Raporu

Bugünkü dersimizde parsel yapısını gördük. Bu yapı web sayfasında tekrar eden yapıları tek dosyadan çağırmayı sağladı. Mesela kodumuzda {name tarafından geliştirilmiştir} yazısı bulunuyordu. Bu yazıyı 3 sayfada yazdık. Herhangi bir değişiklik yapmak istersek teker teker 3 yerden de değiştirmek yerine parsel yapısı sayesinde tek dosyadan müdahale edeceğiz.

İlk olarak src klasörünün içinde bulunan app.js dosyasında hbs yi dahil ettik.

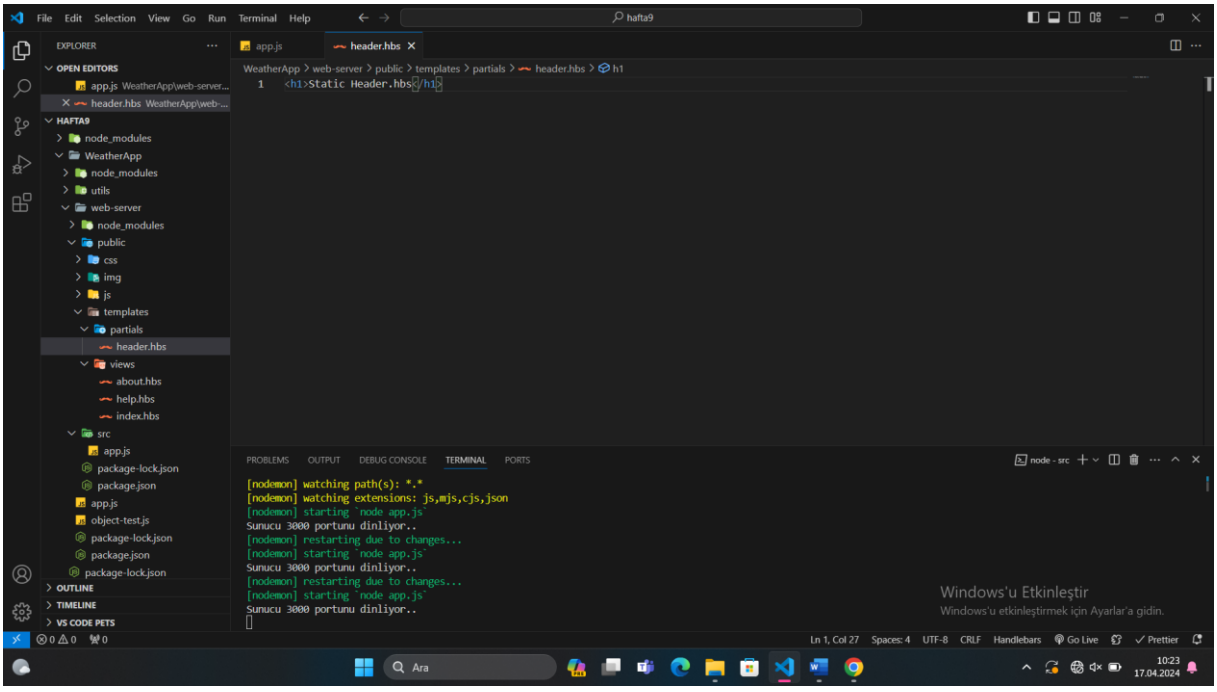
```
const hbs = require("hbs")
```

Public altına yeni bir klasör oluşturduk: templates. Daha önceden var olan views klasörünü de templates klasörünün altına ekledik. Templates klasörünün altına da ek olarak partials klasörü ekledik. {Dosya görünümünü bir sonraki ekran görüntüsünde gösterdim.}

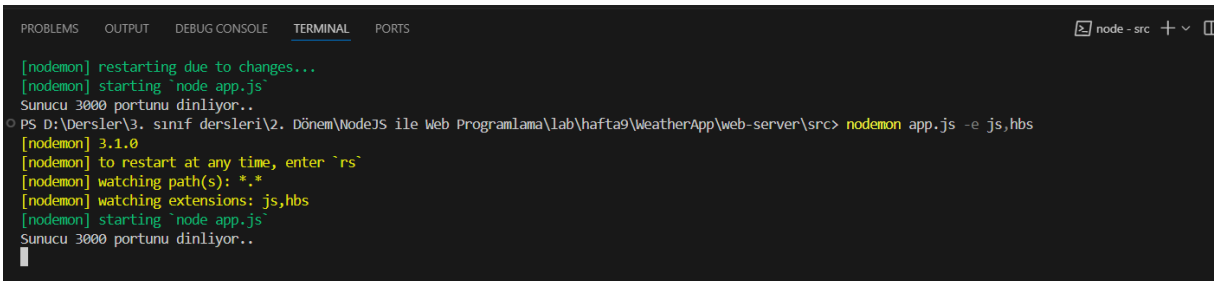
Ardından partials ve views dosyalarının path ini oluşturduk. Views önceden vardı ama yolunu değiştirdiğimizden ötürü güncelleme yaptık.

```
app.set("view engine", "hbs");
const viewsPath = path.join(__dirname, "../public/templates/views");
app.set("views", viewsPath);
const partialsPath = path.join(__dirname, "../public/templates/partials");
hbs.registerPartials(partialsPath);
```

Ardından partials içerisine header.hbs dosyası oluşturduk ve içerisine deneme amaçlı text yazdık.



Ardından {nodemon app.js -e js,hbs} formunda kodumuzu terminalde çalıştırdık. Bu yapı js ve hbs dosyalarının güncellenmesi dahilinde sunucuyu tekrar çalıştırmayı sağlar.



Dinamik hale getireceğiz. Body kısmında header klasörünü “>” işareti ile çağıracağız.

```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <title>Document</title>
  </head>
  <body>
    {{>header}}
    <p>{{helpText}}</p>
  </body>
</html>
```

Title bilgisi headerden gelecek diye tüm dosyalardan sildik yerine header yazdık.

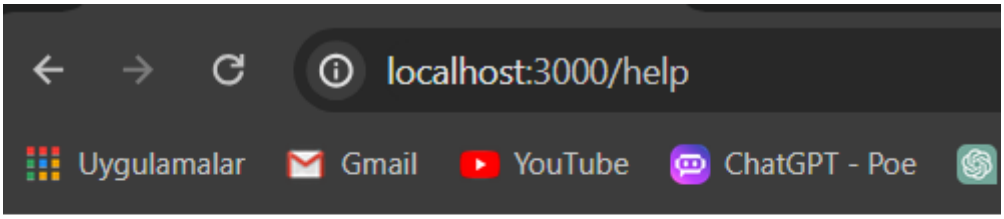
```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <script src="/js/app.js"></script>
    <title>Document</title>
  </head>
  <body>
    <h1>{{>header}}</h1>
    <p>{{name}} tarafından geliştirilmiştir.</p>
  </body>
</html>
```

Aynı şekilde index dosyasında da title ı kaldırdık ve header yazdık.

Son olarak about sayfasının body kısmına bakarsak:

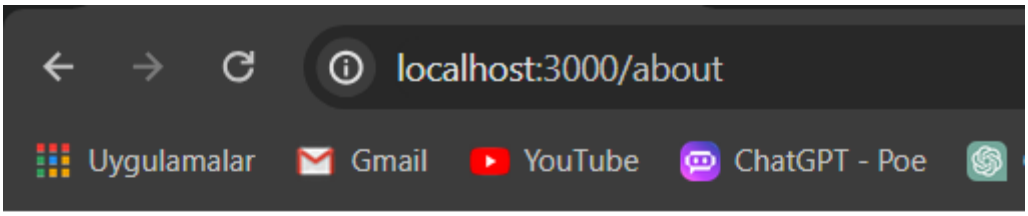
```
  <body>
    <h1>{{>header}}</h1>
    <img src="/img/resim.png" />
    <p>{{name}} tarafından geliştirilmiştir.</p>
  </body>
```

Ardından kodu kaydettik ve sayfamıza baktık:



Static Header.hbs

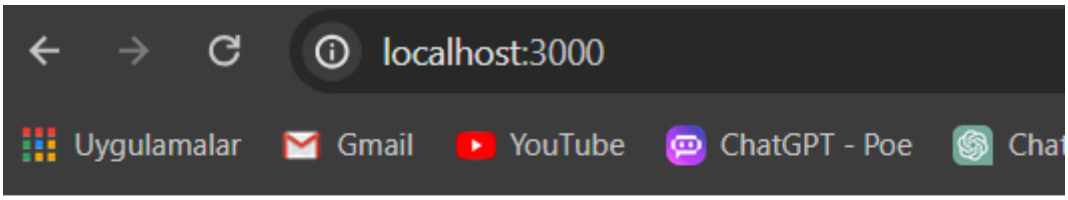
Bu bir deneme yazısıdır.



Static Header.hbs

BURSA TEKNİK ÜNİVERSİTESİ

Gizem Avci tarafından geliştirilmiştir.



Static Header.hbs

Gizem Avci tarafından geliştirilmiştir.

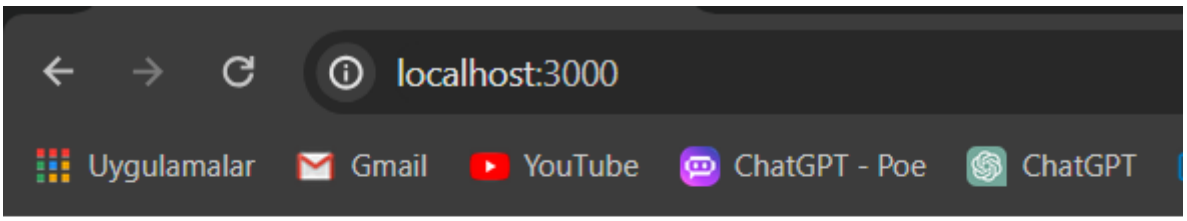
Yaptığımız kod hepsine de uygulanmış oldu. Şimdi help sayfasında name bilgisi olmadığından onu router kısmında ekledik.

```
app.get("/help", (req, res) => {
  //help
  res.render("help", {
    title: "Yardım sayfası",
    name: "Gizem Avci",
    helpText: "Bu bir deneme yazısıdır.",
  });
});
```

Header.hbs dosyasında şunu yazdık:

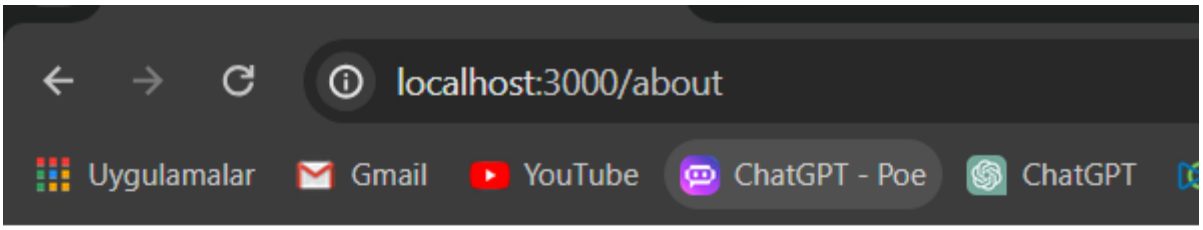
```
<h1>{{title}}</h1>
```

Sonuç olarak kodlarımızı çalıştırdığımızda :



Hava Durumu Uygulaması

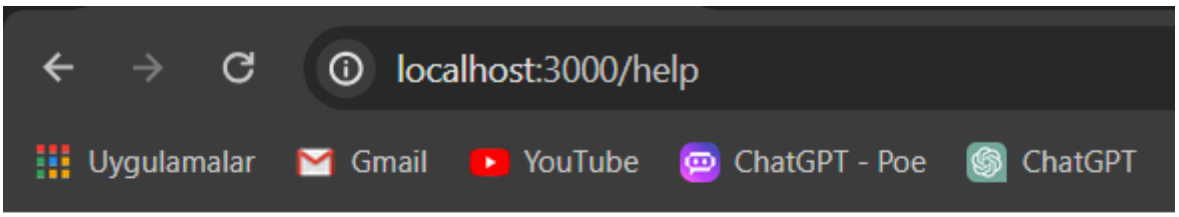
Gizem Avci tarafından geliştirilmiştir.



Hakkımızda

BURSA TEKNİK ÜNİVERSİTESİ

Gizem Avci tarafından geliştirilmiştir.



Yardım sayfası

Bu bir deneme yazısıdır.

Dosyalarımızdaki >header kısmından header.hbs dosyasına geçişi sağladık ve oradan da title bilgisini yazdırmış olduk.

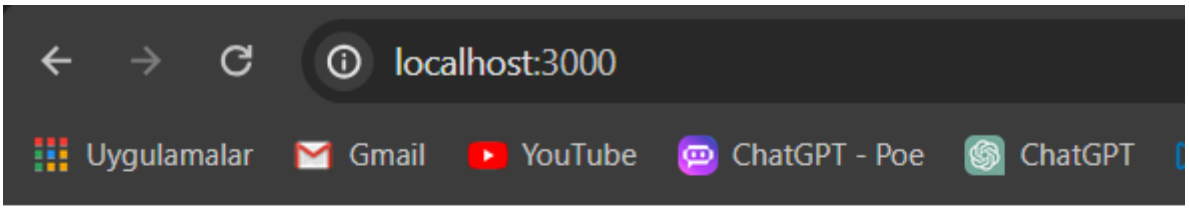
Header.hbs de bir div yapısı oluşturduk.

```
<h1>{{title}}</h1>

<div>
  <a href="/">Hava Durumu</a>
  <a href="/about">Hakkında</a>
  <a href="/help">Yardım</a>

</div>
```

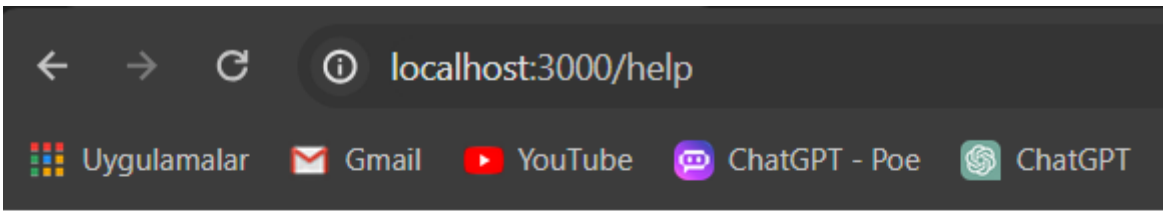
Sayfalarımızda artık link olacak ve geçiş sağlanabilecek. Görünüm olarak da şu şekilde olacak:



Hava Durumu Uygulaması

Hava Durumu Hakkında Yardım

Gizem Avci tarafından geliştirilmiştir.

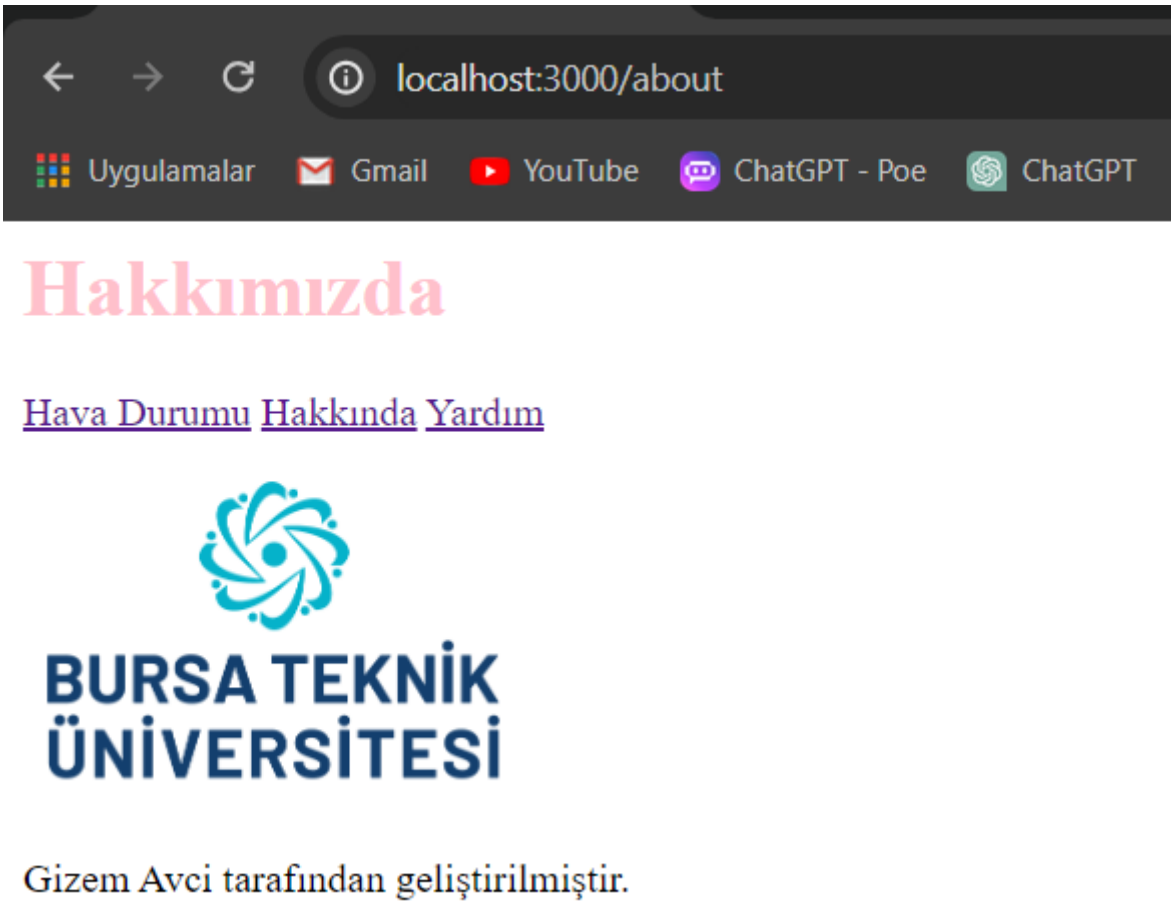


Yardım sayfası

Hava Durumu Hakkında Yardım

Bu bir deneme yazısıdır.

Gizem Avci tarafından geliştirilmiştir.



Şimdi de footer.hbs adında sayfa oluşturup bu sefer de name bilgilerini aldık. Her sayfada bulunan {name tarafından geliştirilmiştir} yazısını da parsel yapısına uyarlamış olduk.

Dosyalara yine aynı formatta çağırarak gereken eklemeyi yaptık:

```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <script src="/js/app.js"></script>
    <title>Document</title>
  </head>
  <body>
    <h1>{{>header}}</h1>
    {{>footer}}
  </body>
</html>
```

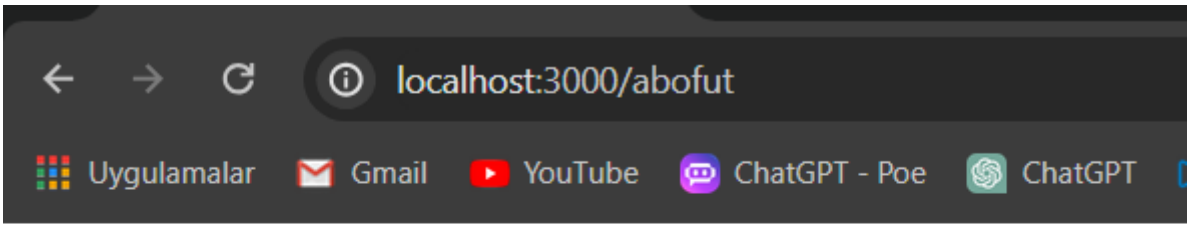
```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <title>Document</title>
  </head>
  <body>
    <h1>{{>header}}</h1>
    <img src="/img/resim.png" />
   {{>footer}}
  </body>
</html>
```

```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <title>Document</title>
  </head>
  <body>
    {{>header}}
    <p>{{helpText}}</p>
    {{>footer}}
  </body>
</html>
```

Footer ekledik ve kodu test ettik. Kodumuzun doğru çalıştığını gördük. Ardından başka bir işleme geçtik. Bu işlem sayfa gelmeme durumunda veyahut about, help veya anasayfa harici dosya ismi girme durumunda cannot get hatası almak yerine yazdığımız sayfaya yönlendirme işlemi gerçekleştirdik

```
app.get("*", (req, res) => {
  res.send("404 Sayfası");
});
```

Hiçbir sayfaya girmeme durumunda bu fonksiyona girecek. Bu anlamı da *(yıldız işareti) ile sağlarız. Ardından bir test yaparız ve olmayan bir sayfaya giriş yaparız:

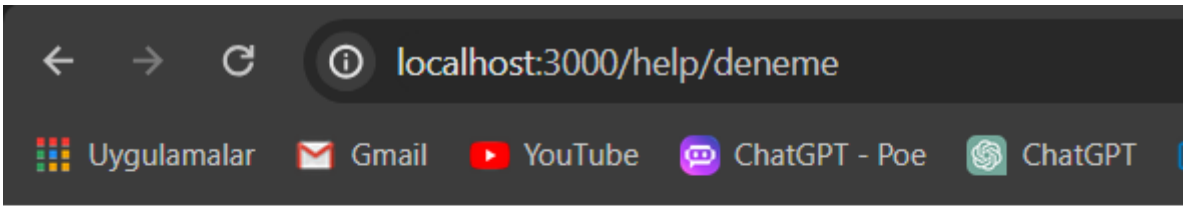


404 Sayfası

Başka bir hale bakarsak eğer help sayfasının bir alt sayfasına girmek isterse ve hata alırsa aynı hatayı değil de farklı bir hata yazdırmayı denedik. Bu kodu diğer kodun(404 sayfası) üst kısmına ekledik çünkü o kod daha genel bir yapıdır.

```
app.get("/help/*", (req, res) => {
  res.send("Aradığınız Yardım Sayfası Bulunamadı");
});
```

Yine bir test yaptık ve şu sonucu aldık:



Aradığınız Yardım Sayfası Bulunamadı

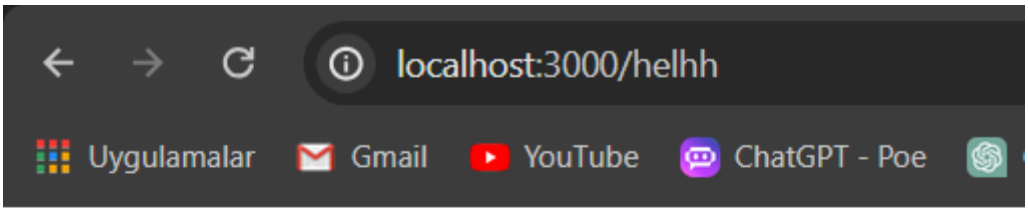
Ardından bu yapıyı da bir hsb dosyasında tuttuk. Bu yüzden partials klasörü içerisie 404.hbs sayfası açtık. Onun içerisine de html kodları ile body kısmına error mesajını yazdıkdık:

```
  <body>
    <h1>{{>header}}</h1>
    <p>{{errorMessage}}</p>
    {{>footer}}
  </body>
```

Ardından mesajı render kısmında da yazdık. Bu sayede iki farklı error mesajı olacak ama tek hbs dosyasından çağıracağız.

```
app.get("/help/*", (req, res) => {
  res.render("404", {
    title: "404 Yardım Sayfası",
    name: "Gizem Avcı",
    errorMessage: "Aradığınız Yardım Sayfası Bulunamadı",
  });
});
app.get("*", (req, res) => {
  res.render("404", {
    title: "404 Sayfası",
    name: "Gizem Avcı",
    errorMessage: "Aradığınız Sayfa Bulunamadı",
  });
});
```

Çıktı olarak da iki versiyonu da test ettik:

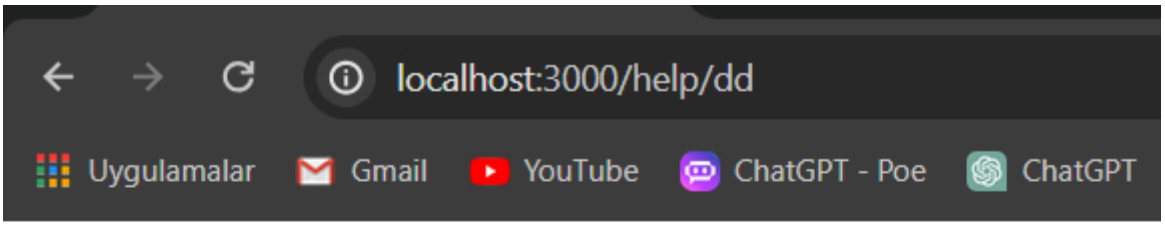


404 Sayfası

Hava Durumu Hakkında Yardım

Aradığınız Sayfa Bulunamadı

Gizem Avcı tarafından geliştirilmiştir.



404 Yardım Sayfası

Hava Durumu Hakkında Yardım

Aradığınız Yardım Sayfası Bulunamadı

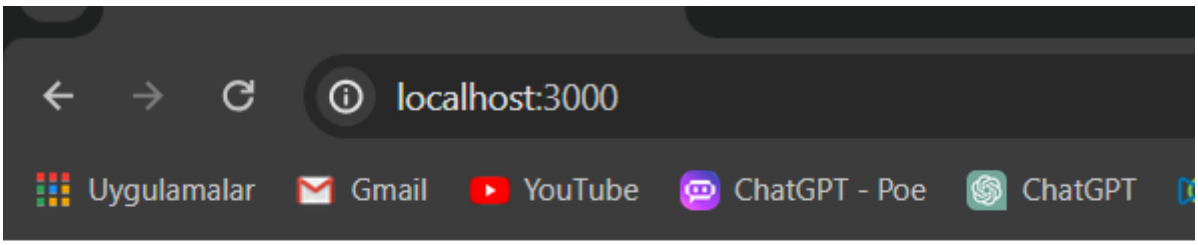
Gizem Avcı tarafından geliştirilmiştir.

.

Bundan sonra sayfa görünümü ile uğraştık ve bu yüzden css dosyası içerisindeki sytle.css dosyasını açtık ve içerisindeki yazıları silip baştan yazmaya başladık:

```
body {
  color: pink;
}
```

Deneme olarak da test ettik.



Hava Durumu Uygulaması

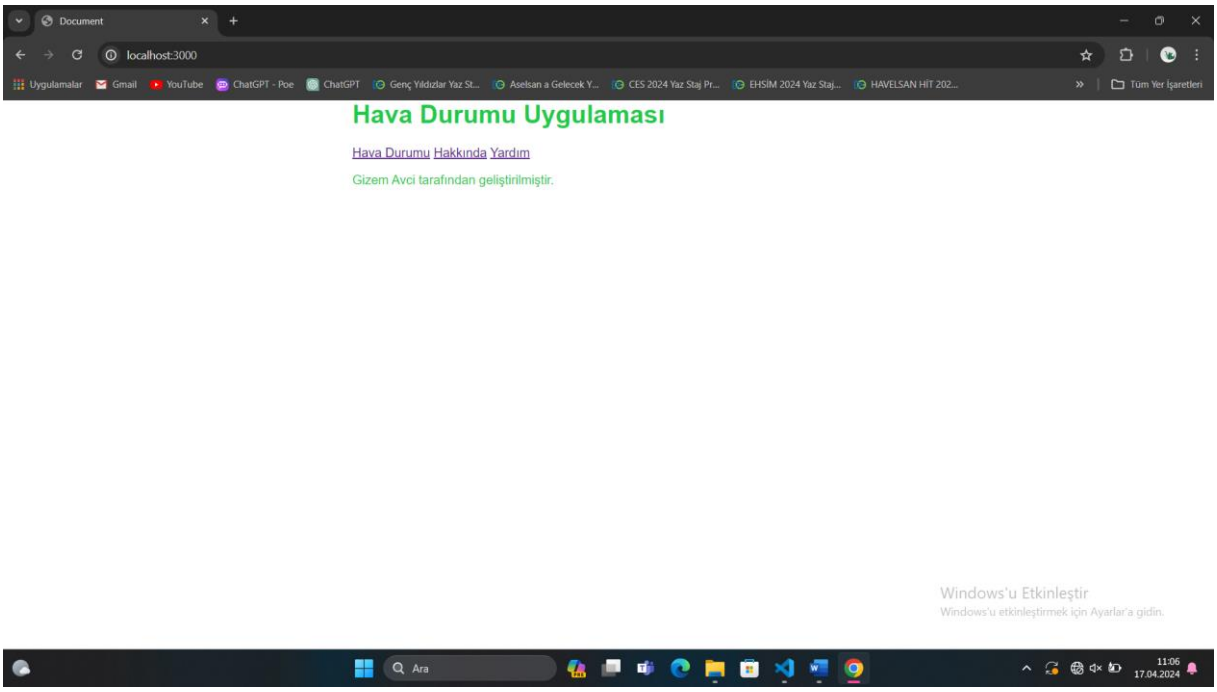
Hava Durumu Hakkında Yardım

Gizem Avcı tarafından geliştirilmiştir.

Tüm yazılar pembe oldu. Ardından font yapısı çerçeve boyutu font rengi olacak şekilde body kısmına yeni özellikler girdik.

```
body {
  color: #1ecb49;
  font-family: Arial;
  max-width: 650px;
  margin: 0 auto;
  padding: 0 16px;
}
```

Çıktı olarak şunu aldık.



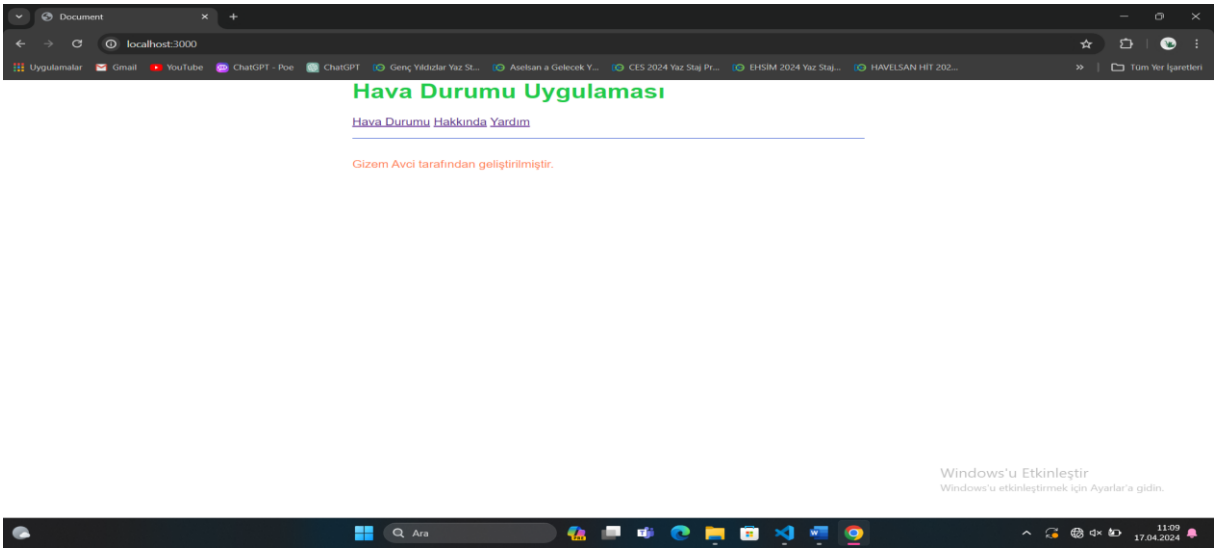
Ardından {name tarafından geliştirilmiştir} yazısı üzerinde değişiklik yapmak adında footer adında bloklara aldık.

```
<footer>
  <p> {{name}} tarafından geliştirilmiştir.</p>
</footer>
```

Ardından css kısmında şu değişikliğe gittik :

```
footer {
  color: coral;
  border-top: 1px solid #1e46cb;
  margin-top: 16px;
  padding: 16px 0;
}
```

Bir daha test işlemi yaptık ve yazımızın renk değiştiğini gördük:



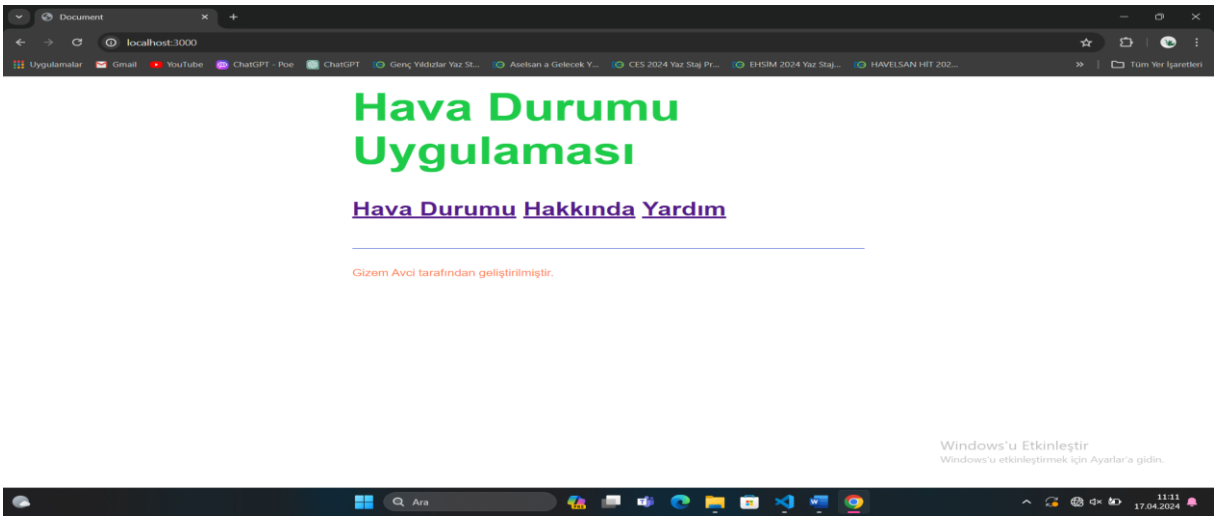
Aynı işlemi header yapısı için de yaptık:

```
<header>
  <h1>{{title}}</h1>
  <div>
    <a href="/">Hava Durumu</a>
    <a href="/about">Hakkında</a>
    <a href="/help">Yardım</a>

  </div>
</header>
```

Css dosyasında da yine özellik bilgisi girdik:

```
header {
  margin-top: 16px;
  margin-bottom: 48px;
}
```

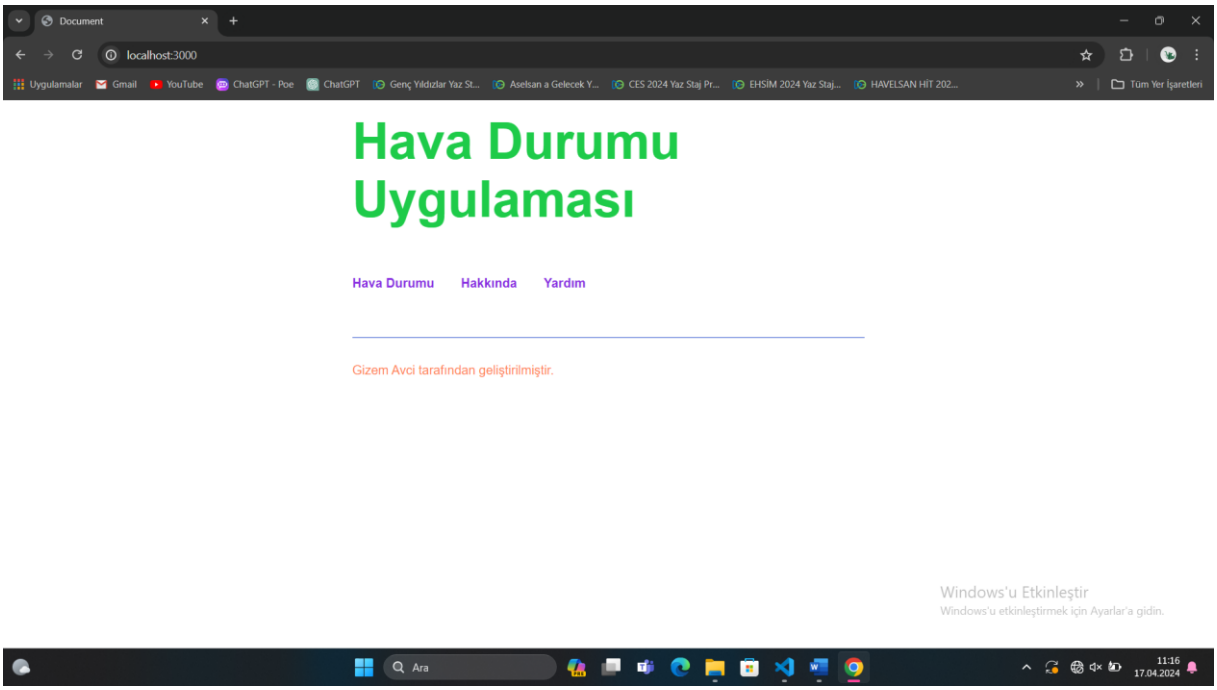


Ardından h1 başlık yazısı ve header içerisindeki link bilgine özellik atadık :

```css
h1 {
  font-size: 16px;
  margin-bottom: 16px;
}

header a {
  color: blueviolet;
  margin-right: 16px;
  text-decoration: none;
}
```

Sayfalara baktığımızda şu şekilde değişiklik olduğunu gördük:



Ardından hakkında dosyasında eklediğimiz resim yapısı için class içerisinde isim verdik ve css dosyasına geçtik.

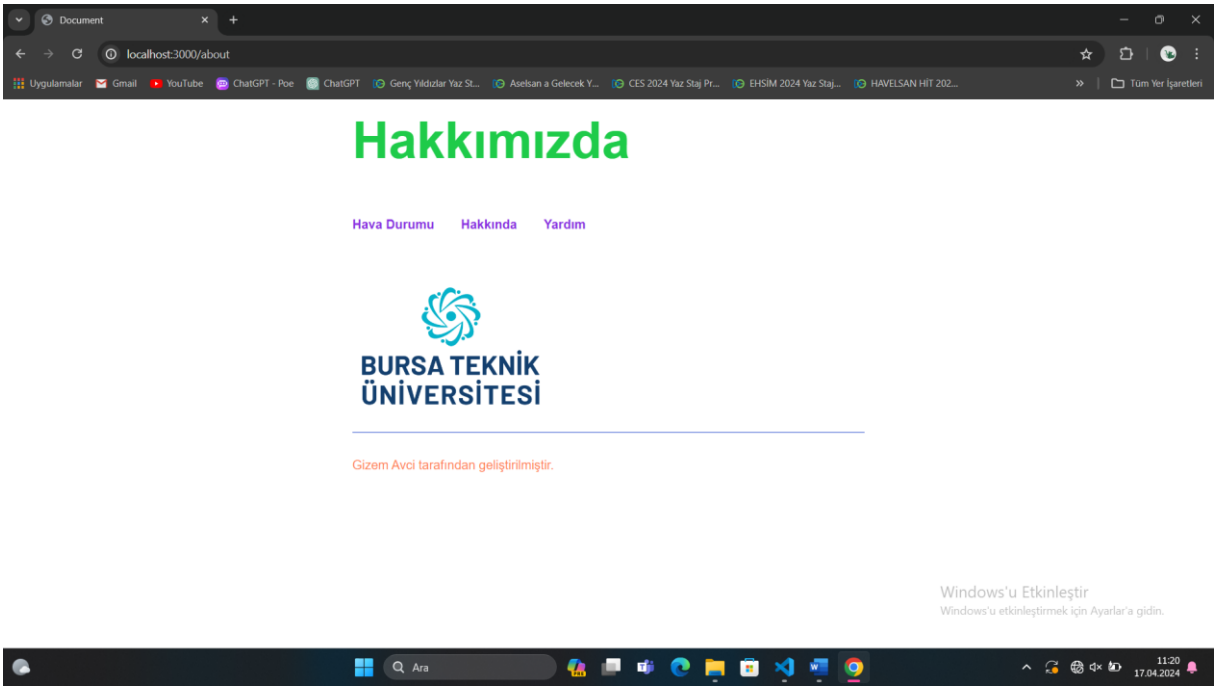
```html
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <title>Document</title>
  </head>
  <body>
    <h1>{{>header}}</h1>
   <img class="portrait" src="/img/resim.png" />

    {{>footer}}
  </body>
</html>
```

Css dosyasında class içerisinde verdiğimiz isim ile çağırma yaptık ve özellik bilgisi girdik.

```
.portrait {
  width: 250px;
}
```

Çıktı olarak da şunu aldı:



Ardından div içerisinde class a main-content ismi vererek yapıları özelleştirmek adına bloklara aldık.

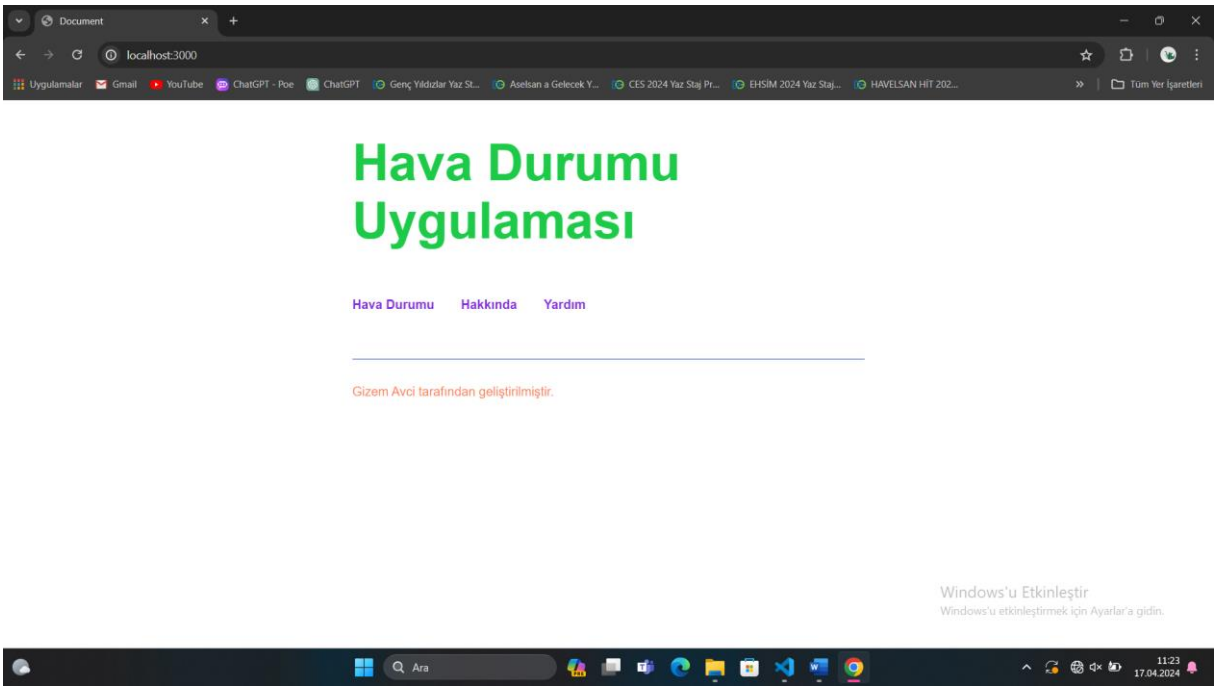
```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <script src="/js/app.js"></script>
    <title>Document</title>
  </head>
  <body>
    <div class ="main-content">
      <h1>{{>header}}</h1>
    </div>
    {{>footer}}
  </body>
</html>
```

Css dosyasında da yine özellik ekledik.

```
body {
  color: #1ecb49;
  font-family: Arial;
  max-width: 650px;
  margin: 0 auto;
  padding: 0 16px;
  display: flex;
  flex-direction: column;
  min-height: 100vh;
}

.main-content {
  flex-grow: 1;
}
```

Çıktı olarak şunu aldık:

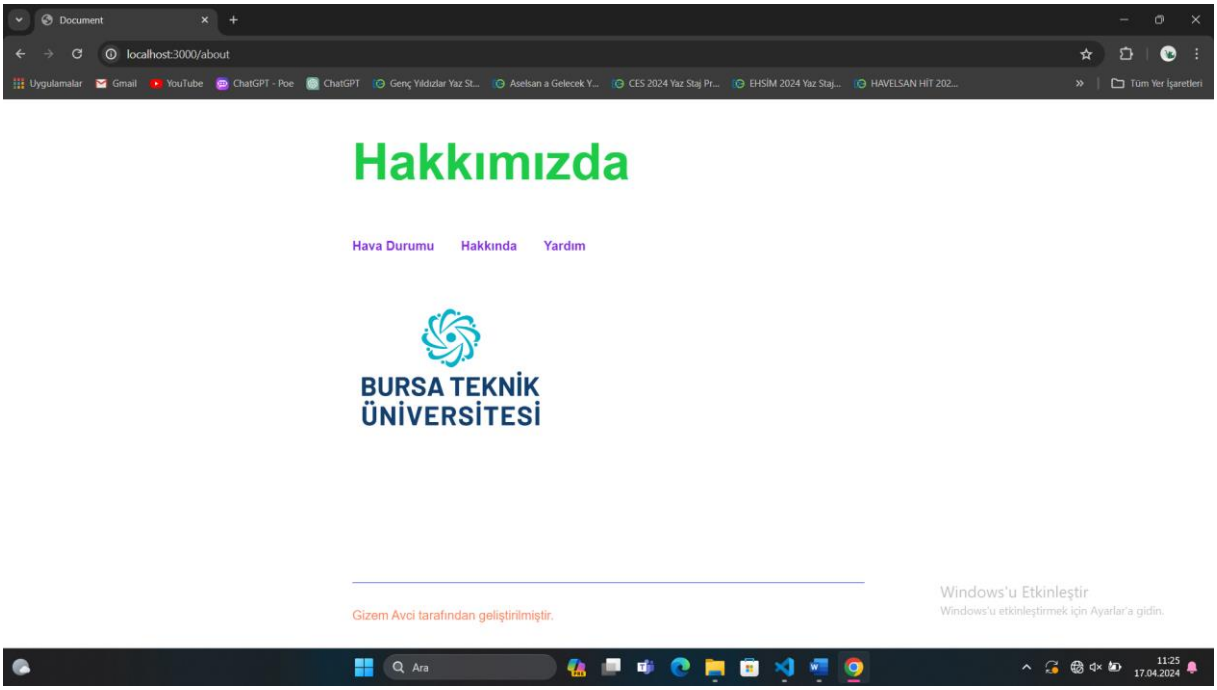


Şimdi about sayfasına için de aynı işlemi yaptık.

```
  <body>
    <div class ="main-content">
      <h1>{{>header}}</h1>
      <img class="portrait" src="/img/resim.png" />
    </div>

   {{>footer}}
  </body>
```

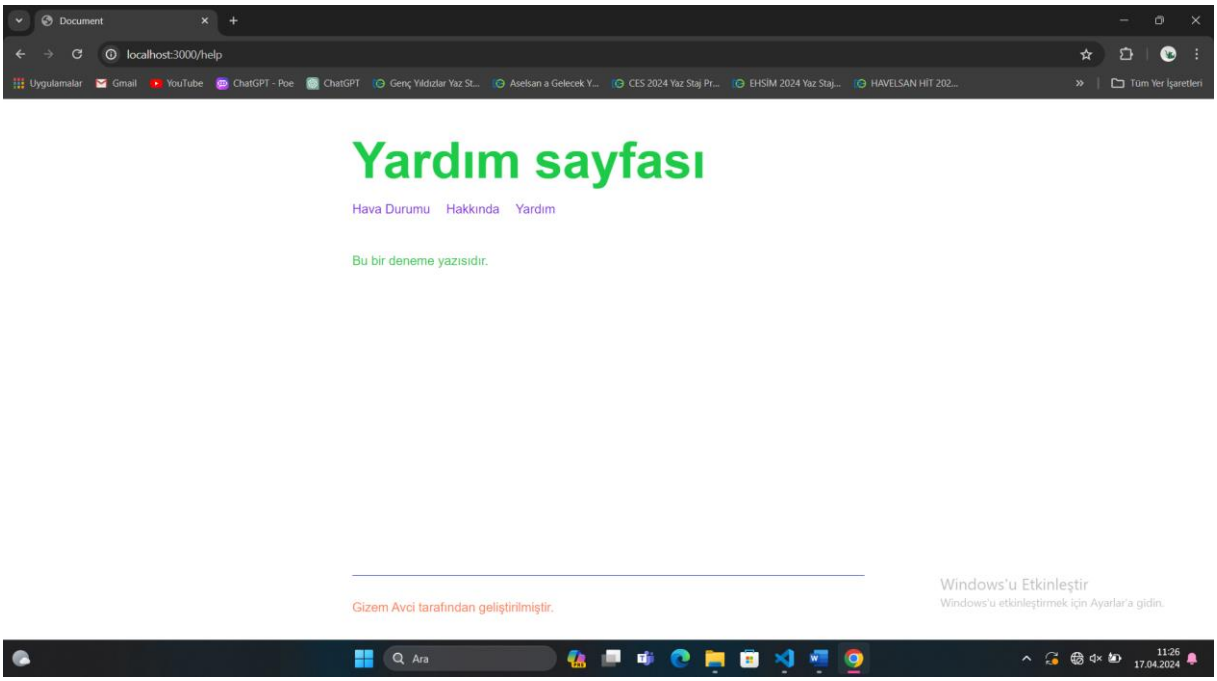
Çıktı olarak da şunu aldık:



Daha sonra help sayfasına müdahale ettik:

```
<body>
  <div class ="main-content">
  {{>header}}
  <p>{{helpText}}</p>
  </div>
  {{>footer}}
</body>
```

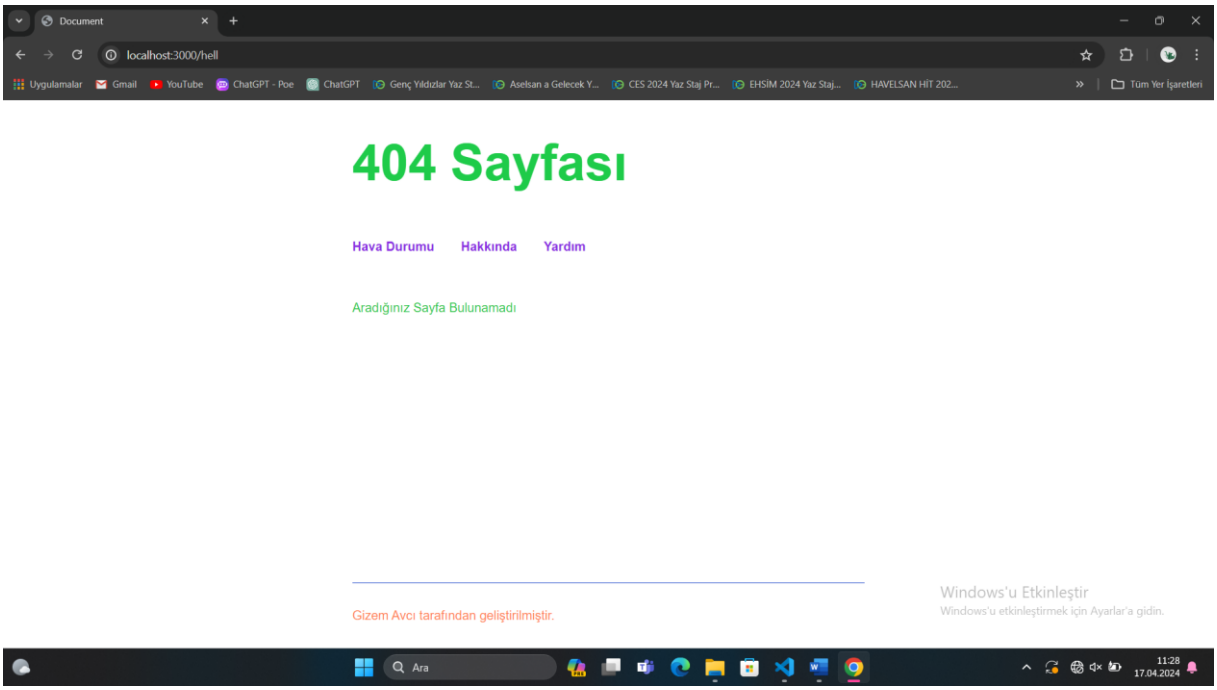
Çıktı olarak da şunu aldık:



Son olarak da 404.hbs sayfasını düzenledik:

```
<html lang="en">
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <title>Document</title>
  </head>
  <body>
    <div class="main-content">
      <h1>{{>header}}</h1>
      <p>{{errorMessage}}</p>
    </div>
    {{>footer}}
  </body>
</html>
```

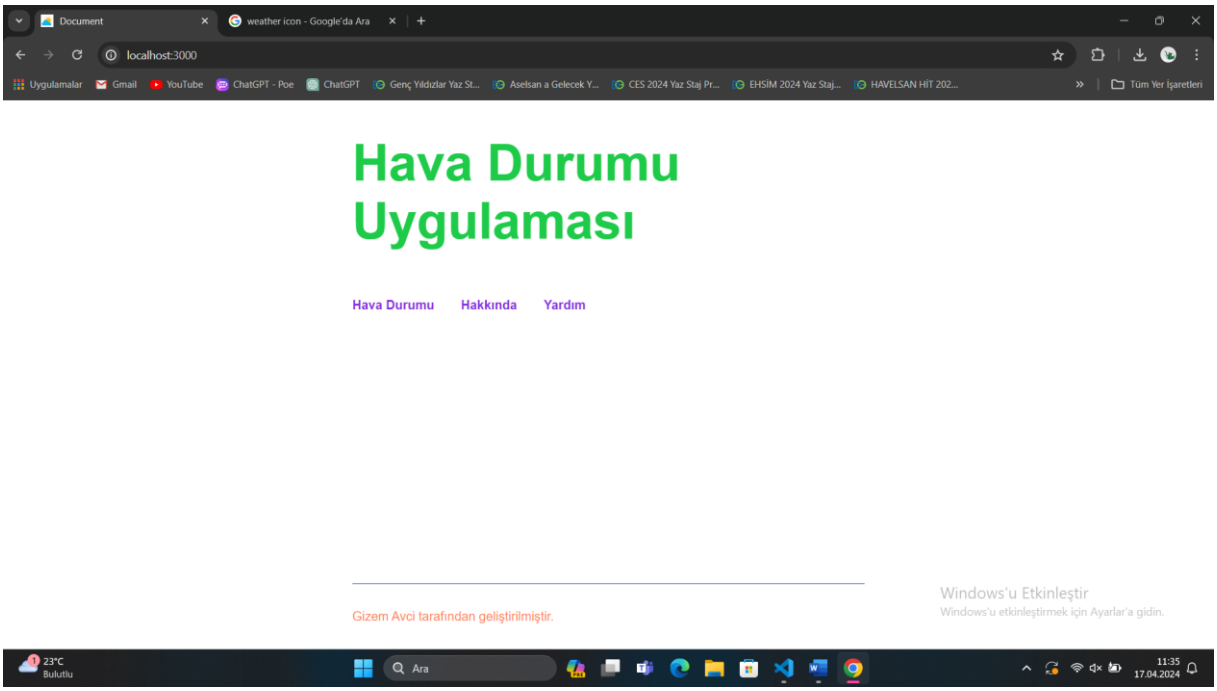
Çıktı olarak da şunu aldık:



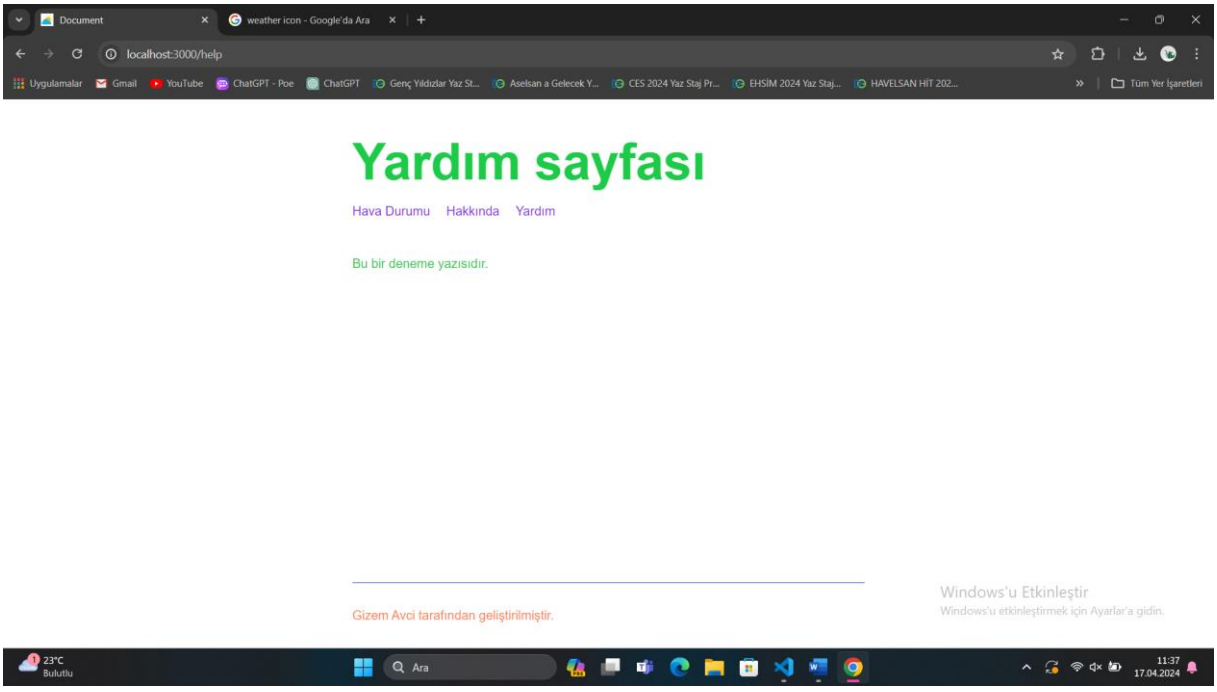
Ardından bir resim indirdik ve o resmi img klasörüne ekledik. Bunu koda dahil etmek için head içerisinde link komutu ile resmin yolunu ve türünü girdik:

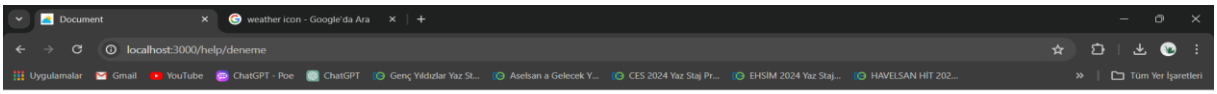
```
  <head>
    <link rel="stylesheet" href="./css/styles.css" />
    <script src="/js/app.js"></script>
    <link rel="icon" href="/img/indir.png">
    <title>Document</title>
  </head>
```

Çıktı olarak bakarsak da sayfamızın sağ üst kösesine yani pencere kısmında ikon eklenmiş oldu.



Aynı kodu diğer sayfalar da ekledik ve kontrol ettik.



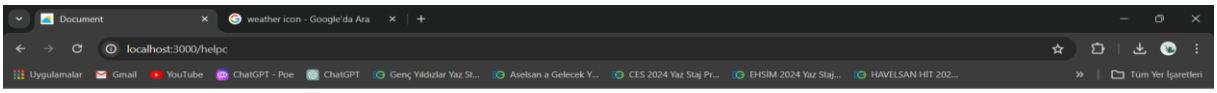


404 Yardım Sayfası

Hava Durumu Hakkında Yardım

Aradığınız Yardım Sayfası Bulunamadı

Gizem Avcı tarafından geliştirilmiştir.



404 Sayfası

Hava Durumu Hakkında Yardım

Aradığınız Sayfa Bulunamadı

Gizem Avcı tarafından geliştirilmiştir.

Hakkımızda

Hava Durumu Hakkında Yardım

BURSA TEKNİK ÜNİVERSİTESİ

Gizem Avcı tarafından geliştirilmiştir.