

Node.js Section 9-10

Merhabalar bugün node.js serimize devam ediyoruz.

Öncelikle Section5-6-7-8 i Section5-6-7-8-9-10 olarak değiştirelim.

Geçen haftada söylemeyi unuttuğum şeyden başlayayım express template view'lar için belirli bir klasör adını arar. Web-server/public/views klasörünün ismini templates yapıp çalıştırınca hata verecektir.

Web-server/src/index.js içinde viewsPath değiştirelim.

```
Section5-6-7-8-9-10 - index.js
const viewsPath = path.join(__dirname, "../public/templates")
```

Tutarlılık sağlamak amacıyla header ve footer lar yeniden kullanılabilir.Bu yüzden web-server/public/templates klasörü içinde views ve partials adında 2 tane klasör oluşturalım.

Daha sonra templates içindeki hbs dosyalarını views klasörüne taşıyalım ve index.js içindeki viewsPath'ini yeniden düzenleyelim

```
Section5-6-7-8-9-10 - index.js
const viewsPath = path.join(__dirname, "../public/templates/views")
```

partials için index.js içinde bir path oluşturalım.

```
Section5-6-7-8-9-10 - index.js
const partialsPath = path.join(__dirname, "../public/templates/partials")
```

Bunu hbs için kullanabilmemiz için hbs yi eklememiz gerekiyor.

```
Section5-6-7-8-9-10 - index.js
const hbs = require("hbs")
```

Bu kodu da başa ekleyelim.

Daha sonra app.set in altında şu kodu kullanalım.

```
Section5-6-7-8-9-10 - index.js
hbs.registerPartials(partialsPath)
```

Şimdi web-server/public/templates/partials klasörü içinde header.hbs adında bir dosya oluşturalım ve bu bizim sayfamızın header kısımlarını oluştursun.

```
Section5-6-7-8-9-10 - header.hbs
<h1>Static Header.hbs Text</h1>
```

İçide bu şekilde olsun.

Help.hbs dosyasına gidip body kısmını şu hale getirelim :

```
Section5-6-7-8-9-10 - help.hbs
<body>
  {{>header}}
  <p>{{helpText}}</p>
</body>
```

Şimdi ise terminalde nodemon ile şu şekilde çalıştırıralım :
nodemon web-server/src/index.js -e js,hbs

Bu kod hbs js kodlarının değişikliğinde kendisi otomatik olarak güncellemektedir.

Daha sonra localhost:3000/help e gidelim ve çalıştığını görelim.

Şimdi help.hbs den h1 tag'ini kaldırıldı. Diğer hbs dosyalarından da h1 kaldırıp {{>header}} ekleyelim.

Bu şekilde aslında biz header olarak bir yerde tanımladık ve bu tanımladığımız değişkeni kullandık.

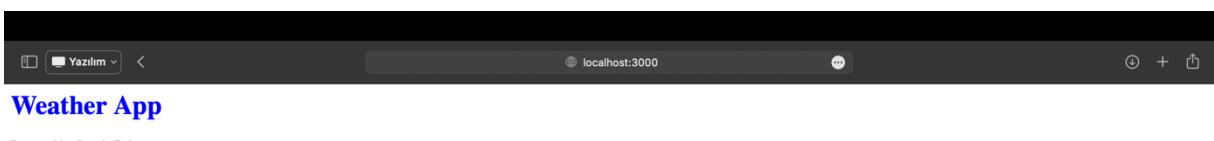
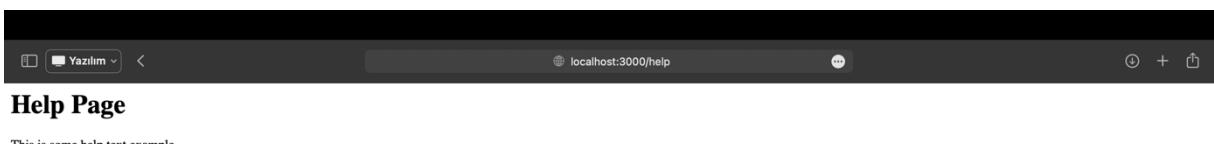
Index.js e gidip help için app.get e helpText'in yanı sıra title ve name ekleyerek şu hale getirelim :

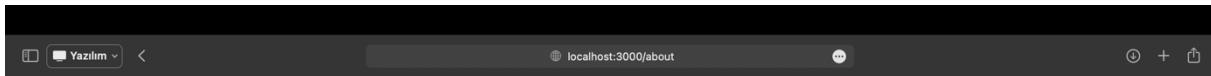
```
Section5-6-7-8-9-10 - index.js
app.get("/help", (req, res) => {
  res.render("help", {
    title: "Help Page",
    name: "Burak Gül",
    helpText: "This is some help text example"
  })
})
```

Daha sonra header.hbs deki yapımızı statikten dinamik hale getirmek için header.hbs şu halde olmalı :

```
Section5-6-7-8-9-10 - header.hbs
<h1>{{title}}</h1>
```

Kaydedip sayfalarda deneyelim.





About Me



Powered by Burak Güll

Gördüğümüz üzere app.get help vs yazınca parametre olarak ne verirsek onları görüyoruz web sitemizde.

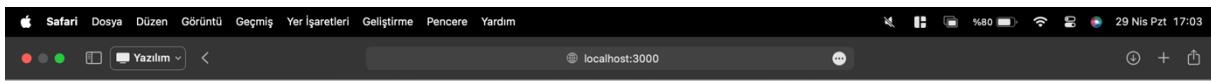
Şimdi header.hbs dosyamızda diğer sayfalara link ekleyelim.

```
Section5-6-7-8-9-10 - header.hbs

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

<div>
  <a href="/" >Weather</a>
  <a href="/about" about>About</a>
  <a href="/help" help>Help</a>
</div>
```

Bunu yapmamızın sebebi sitemizde diğer sayfalara link vererek sayfa geçişlerini yapmak.



Weather App

[Weather](#) [About](#) [Help](#)

Powered by Burak Güll

Şimdi partials klasöründe footer.hbs adında bir klasör oluşturalım ve içi şöyle olsun :

```
Section5-6-7-8-9-10 - footer.hbs
<p>Created by {{name}}</p>
```

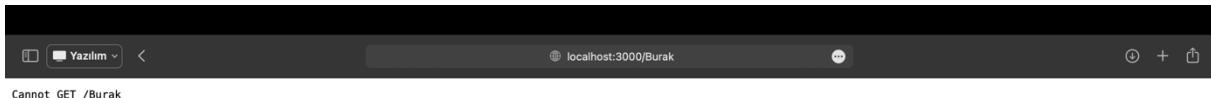
Daha sonra help.hbs ye gidelim ve aşağıdaki kodu body etiketini kapatmadan hemen önce ekleyelim.

```
Section5-6-7-8-9-10 - help.hbs
{{>footer}}
```

Aynı şeyleri index.hbs ve about.hbs için yapalım.

Daha sonra index.hbs ve about.hbs den powered by yazan kısımları silelim.

Var olmayan bir sayfaya gidelim mesela localhost:3000/Burak



Hata veriyor gördüğümüz üzere. Şimdi yeni bir router ekleyelim src/index.js altında.

```
Section5-6-7-8-9-10 - index.js
app.get("*", (req, res) => {
  res.send("My 404 Page")
})
```

Buradaki * diğer hersey anlamına gelir , express route için bir eşleşme arar bulamayınca bunun çalışması için son router bu olmalı yani sayfanın en altında olmalıdır.

Şimdi olmayan bir sayfa için çalıştırırsak ekran "My 404 Page" yazısını göreceğiz.

Localhost:3000/help/testpage e de gidersek "My 404 Page" yazısı görünecek bunu help sayfası ile ilişkili hale getirmek için

```
Section5-6-7-8-9-10 - index.js
app.get("/help/*", (req, res) => {
  res.send("Help article not found")
})
```

Bu kodu daha deminki koddan önce ekleyelim çünkü help den sonra ne yazarsak bu çalışssın diye.



Şimdi ise handlebar ile 404 sayfası hazırlayalım ve öyle gönderelelim.

Partials/views altında 404.hbs adında bir dosya oluşturalım.

```
Section5-6-7-8-9-10 ~ 404.hbs
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="/css/styles.css">
</head>
<body>
  {{>header}}
  <p>{{errorMessage}}</p>
  {{>footer}}
</body>
</html>
```

İç de böyle olsun.

Şimdi app.js e gidip

```
Section5-6-7-8-9-10 - index.js
app.get("/help/*", (req, res) => {
  res.send("Help article not found")
})

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

Bu kodları şu hale getirelim :

```
Section5-6-7-8-9-10 - index.js

app.get("/help/*", (req, res) => {
  res.render("404",{
    title: "404",
    name: "Burak Gül",
    errorMessage: "Help article not found"
  })
}

app.get("*", (req, res) => {
  res.render("404", {
    title: "404",
    name: "Burak Gül",
    errorMessage: "Page not found"
  })
})
```

Şimdi styles.css dosyasını bu hale getirelim :

```
Section5-6-7-8-9-10 - styles.css

body {
  color: #333333;
  font-family: arial;
  max-width: 650px;
  margin: 0 auto;
  padding: 0 16px;
}
```

Bu ekranı daha da iyi hale getirecektir şimdi ise footer.hbs ye gidelim footer için tag ekleyelim ki styles.css dosyasında footer için değişiklikler yapabilelim.

```
Section5-6-7-8-9-10 - footer.hbs

<footer>
  <p>Created by {{name}}</p>
</footer>
```

Şimdi de styles.css dosyasına şunu ekleyelim body den sonra :

```
Section5-6-7-8-9-10 - styles.css
```

```
footer {  
    color: #888888;  
    border-top: 1px solid #eeeeee;  
    margin-top: 16px;  
    padding: 16px 0;  
}
```

Daha sonra header.hbs yi de şu hale getirelim :

```
Section5-6-7-8-9-10 - header.hbs
```

```
<header>  
    <h1>{{title}}</h1>  
    <a href="">Weather</a>  
    <a href="" about>About</a>  
    <a href="" help>Help</a>  
</header>
```

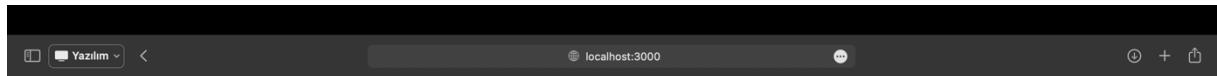
Css dosyasına da şunu da ekleyelim :

```
Section5-6-7-8-9-10 - styles.css
```

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

Şimdi biz ne yaptık bunlarla hbs dosyalarımızı tag içine alarak css de kullanılabilir hale getirdik ve bunları css içinde düzenledik.

Şimdi nasıl gidiyor bi bakalım css düzenlemelerine devam edelim.



Gayet iyi gidiyor.

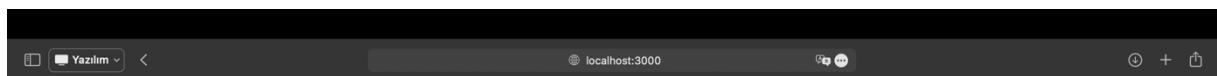
Styles.css içine şunu eklersek h1 ler büyüyecektir

```
Section5-6-7-8-9-10 - styles.css

h1 {
  font-size: 64px;
  margin-bottom: 16px;
}

header a {
  color: #888888;
  margin-right: 16px;
  text-decoration: none;
}
```

Test edersek şu hale gelecektir.



About.hbs ye gidip img ' yi değiştirelim.

```
Section5-6-7-8-9-10 - about.hbs


```

Daha sonra css dosyasında da şunu ekleyelim

```
Section5-6-7-8-9-10 ~ styles.css

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

Bu bize resmin genişliğini 250px yapacaktır.

Footer sayfanın en altına yapışmalıdır bu yüzden index.hbs ye gidelim ve body kısmını şu şekilde değiştirelim.

```
Section5-6-7-8-9-10 - index.hbs

<body>
  <div class="main-content">
    {{>header}}
    <p>Use this website to get your weather!</p>
  </div>
  {{>footer}}
</body>
```

Daha sonra css dosyamızda body kısmını şu şekilde düzenleyelim.

```
Section5-6-7-8-9-10 - styles.css

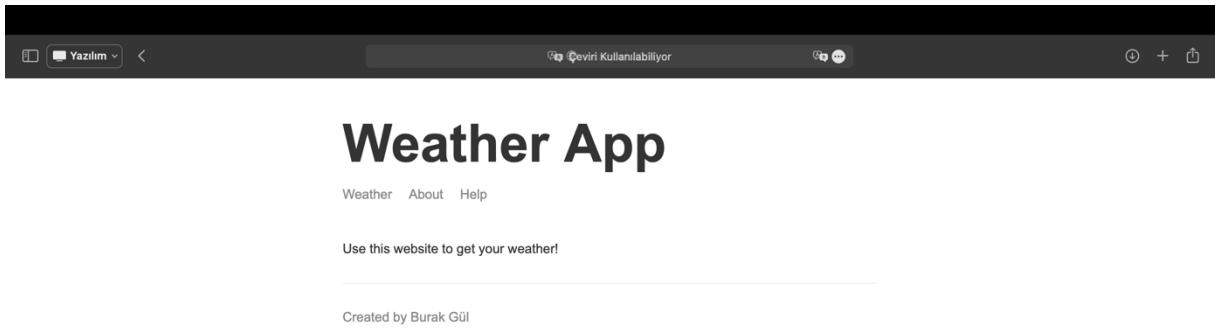
body {
  color: #333333;
  font-family: arial;
  max-width: 650px;
  margin: 0 auto;
  padding: 0 16px;

  display: flex;
  flex-direction: column;
  min-height: 100vh;
}
```

Ve şunu da ekleyelim css dosyamıza :

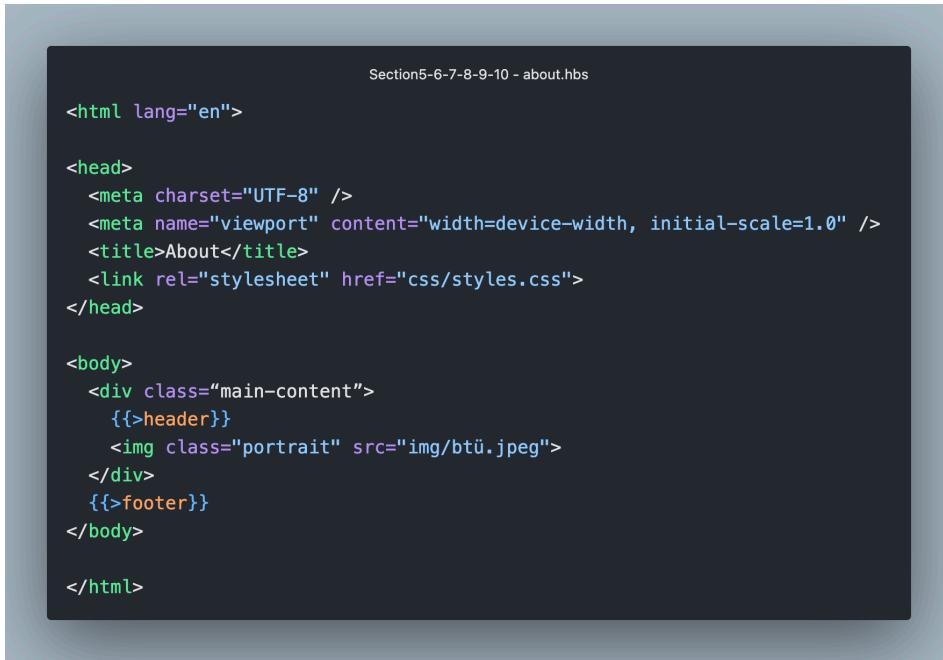


Test edelim:



Harika güzel gözüküyor ama biz bunu sadece index.hbs için yaptık.
404,about ve help için divleri değiştirelim.

About.hbs yi şu hale getirelim :



Help.hbs ise şu hale gelmeli :

```
Section5-6-7-8-9-10 - help.hbs

<html lang="en">

<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Help</title>
  <link rel="stylesheet" href="/css/styles.css">
</head>

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

</html>
```

404.hbs ise şu hale gelmeli :

```
Section5-6-7-8-9-10 - 404.hbs

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="/css/styles.css">
  <title>404</title>

</head>

<body>
  <div class="main-content">
    {{>header}}
    <p>{{errorMessage}}</p>
  </div>

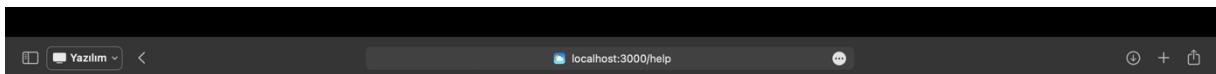
  {{>footer}}
</body>

</html>
```

Index.hbs ye de head kısmına icon ekleyebiliriz :

```
Section5-6-7-8-9-10 - index.hbs

<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="css/styles.css" />
    <script src="js/app.js"></script>
    <title>weather</title>
    <link rel="icon" href="/img/weather.png">
  </head>
  <body>
    <div class="main-content">
      {{>header}}
      <p>Use this website to get your weather!</p>
    </div>
    {{>footer}}
  </body>
</html>
```



Help Page

Weather About Help

This is some help text example

Created by Burak Güл

url kısmında gördüğümüz üzere iconumuz da eklendi.

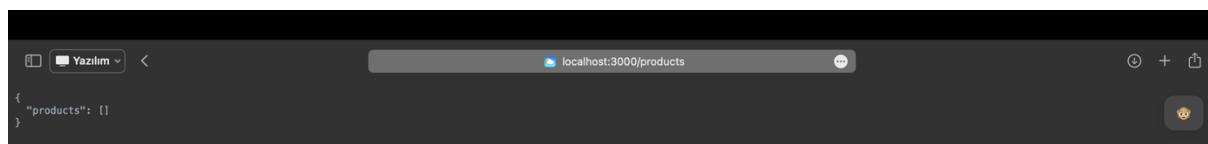
Önceki haftalarda query string ile location bilgileri eklemiştik şimdi buradan web sitemizden yapmaya çalışalım.

Index.js içinde test amaçlı bir tane router oluşturalım.

```
Section5-6-7-8-9-10 - index.js

app.get("/products", (req, res) => {
  res.send({
    products: []
  })
})
```

Bu bir boş products arrayinin JSON objectini döner



Gördüğümüz üzere bu bir statik sonuçtur değişmiyor yani. Arama yapmak ve belirli product'lar döndürmesini istiyorsak query string kullanmalıyız. Query string url'in sonunda gelmektedir.

"?search=games&rating=5" şeklinde olur ve ? sorgunun başladığını bildirirken & parametre olarak başka bir parametrenin başladığını belirtir.

Daha deminki route'ı şu hale getirelim.

```
Section5-6-7-8-9-10 - index.js

app.get("/products", (req, res) => {
  console.log(req.query);
  res.send({
    products: []
  })
})
```

url kısmına da şunu yapıştıralım : "localhost:3000/products?search=games&rating=5"

```
{ search: 'games', rating: '5' }
```

Consoleda böyle gözükmektedir. Yani vscode outuputunda biz query string'lerin listesini görmekteyiz.

```
Section5-6-7-8-9-10 - index.js

app.get("/products", (req, res) => {
  console.log(req.query.search)
  res.send({
    products: []
  })
})
```

Bu şekilde istediğimiz query stringe erişebiliriz. search'e erişmek için bu şekilde kullanabiliriz.

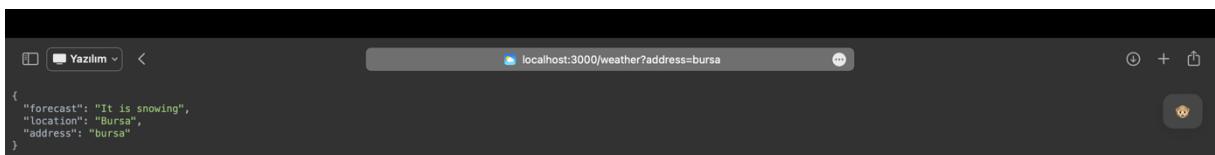
Arama yapabilmek için search parametresini zorunlu rating parametresini ise optional(isteğe bağlı) yapabilmek için route'imizi şu hale getirelim :

```
Section5-6-7-8-9-10 - index.js

app.get("/products", (req, res) => {
  if(!req.query.search){
    return res.send({
      error: "You must provide a search term"
    })
  }
  console.log(req.query.search);
  res.send({
    products: []
  })
})
```

Şimdi weather end pointini güncelleyelim. Adress alacağız address yoksa error message yollayacağız.

Dosyayı kaydedip localhost:3000/weather?address=bursa e gidelim



Gördüğümüz üzere çalışıyor.

Geocoding ve forecast için zaten biz bazı kodlara sahibiz bu kodları web-server-src içine taşıyalım.

Section5-6-7-8-9-10/utils içindeki forecast.js ve geocode.js dosyalarını utils ile beraber Section5-6-7-8-9-10/web-server/src içine taşıyalım.

Ve section5-6-7-8-9-10 içindeki node_modules,utils,index.js,package.json,package-lock.json klasör ve dosyalarını silelim. Sadece web-server klasörü kalsın.

Ve biz bazı npm modüllerine ihtiyacımız var utils içinde kullanırken utils dizinine indirmişti web-server dizinine indirmemişti onları indirelim şimdi.

Öncelikle nodemon'dan ctrl+c yaparak çıkışım.

Web-server dizinine gidelim npm i postman-request yazarak modülümüzü indirelim

Daha sonra nodemon'u çalıştırabiliriz.

Geocoding ve forecast i index.js içinde kullanabilmek için requirement'lere ekleyelim.

```
Section5-6-7-8-9-10 - index.js

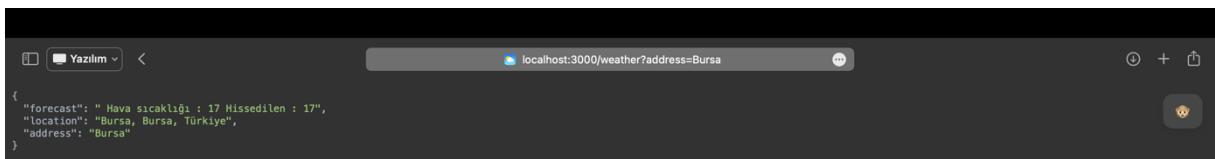
const geocode = require("./utils/geocode");
const forecast = require("./utils/forecast");
```

Şimdi weather url 'ini güncelleyelim.

```
Section5-6-7-8-9-10 - index.js

app.get("/weather", (req, res) => {
  if (!req.query.address) {
    return res.send({
      error: "You must provide a search term"
    })
  }
  geocode(
    req.query.address,
    (error, { latitude, longitude, location } = {}) => {
      if (error) {
        return res.send({ error })
      }
      forecast(latitude, longitude, (error, forecastData) => {
        if (error) {
          return res.send({ error })
        }
        res.send({
          forecast: forecastData,
          location,
          address: req.query.address
        })
      })
    }
  )
})
```

Şimdi deneyelim.



Verilerimiz geldi gayet.

Şimdi <http://localhost:3000/weather?address=>! Yazarsak tarayıcıdaki url kısmına uygulamamız çökecektir.

Problem geocode'un çağrılmamasındadır.error dönunce data dönmez ve biz data'yı aslında destructure'etmeye çalıştığımızdan dolayı crash alırız bunun çözümü default değer atamaktır.

(Normalde çalışması gerek ama çalışmıyor öğrenip burayı telafi edeceğim.)

Dataları yakalamak için fetch api kullanacağız . fetch api node js in bir kısmı değildir. Client tarafından execute edilir.

```
Section5-6-7-8-9-10 - app.js

fetch("http://localhost:3000/weather?address=bursa").then((response) => {
  response.json().then((data) => {
    if (data.error) {
      console.log(data.error)
    } else {
      console.log(data.location)
      console.log(data.forecast)
    }
  })
})
```

Promise konusu ilerleyen kısımlarda anlatılacaktır.

Public/js/app.js içinde bunu kullanalım ve homepagemizi açıp yenileyelim.

Şimdi search form oluşturalım

Index.hbs ye gidelim.

Body kısmını bu şekilde oluşturarak bir button ve input girişi eklemiş oluruz.

```
Section5-6-7-8-9-10 - index.hbs

<body>
  <div class="main-content">
    {{>header}}
    <p>Use this website to get your weather!</p>
    <form>
      <input placeholder="Location">
      <button>Search</button>
    </form>
  </div>
  {{>footer}}
</body>
```

App.js.js içinde formumuzu ekleyelim ve bir tane listener ekleyelim buton için.

```
Section5-6-7-8-9-10 - app.js
```

```
const weatherForm = document.querySelector("form")

weatherForm.addEventListener("submit", () => {
  console.log("testing")
})
```

Kaydedince hata alacağımızı önlemek için script kısmını index.hbs içinde değiştirmemiz gerekiyor çünkü js dosyası hsb dosyası render edilmeden önce geliyor bunu body kapanmadan hemen öncesine çekelim.

```
Section5-6-7-8-9-10 - index.hbs
```

```
<html lang="en">

<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <link rel="stylesheet" href="css/styles.css" />
  <title>weather</title>
  <link rel="icon" href="/img/weather.png">

</head>

<body>
  <div class="main-content">
    {{>header}}
    <p>Use this website to get your weather!</p>
    <form>
      <input placeholder="Location">
      <button>Search</button>
    </form>
  </div>
  {{>footer}}
  <script src="js/app.js"></script>
</body>

</html>
```

Şimdi butona her tıkladığımızda sayfa otomatik olarak yenilenir

```
Section5-6-7-8-9-10 - app.js
```

```
weatherForm.addEventListener("submit", (e) => {
  e.preventDefault()
  console.log("testing")
})
```

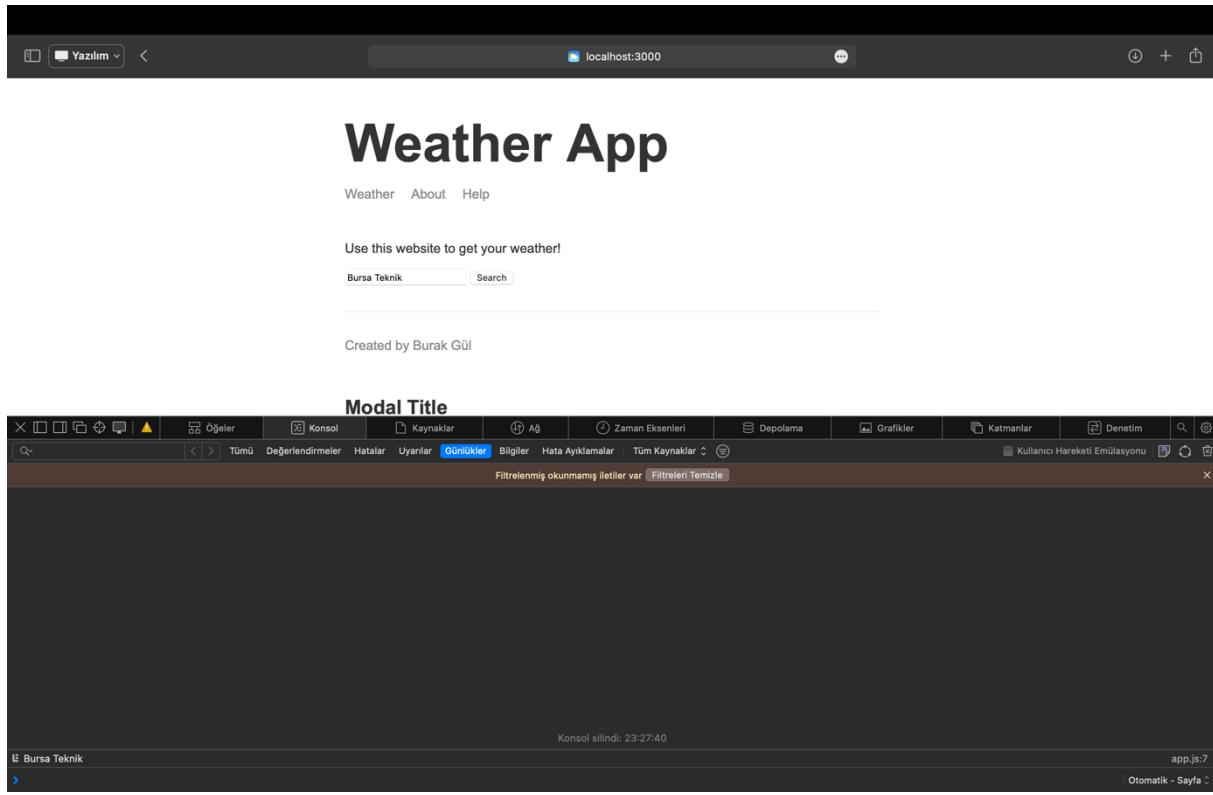
Şimdi ise yenilenmez.

Searchte yazdıklarımızı çekmeye çalışalım.

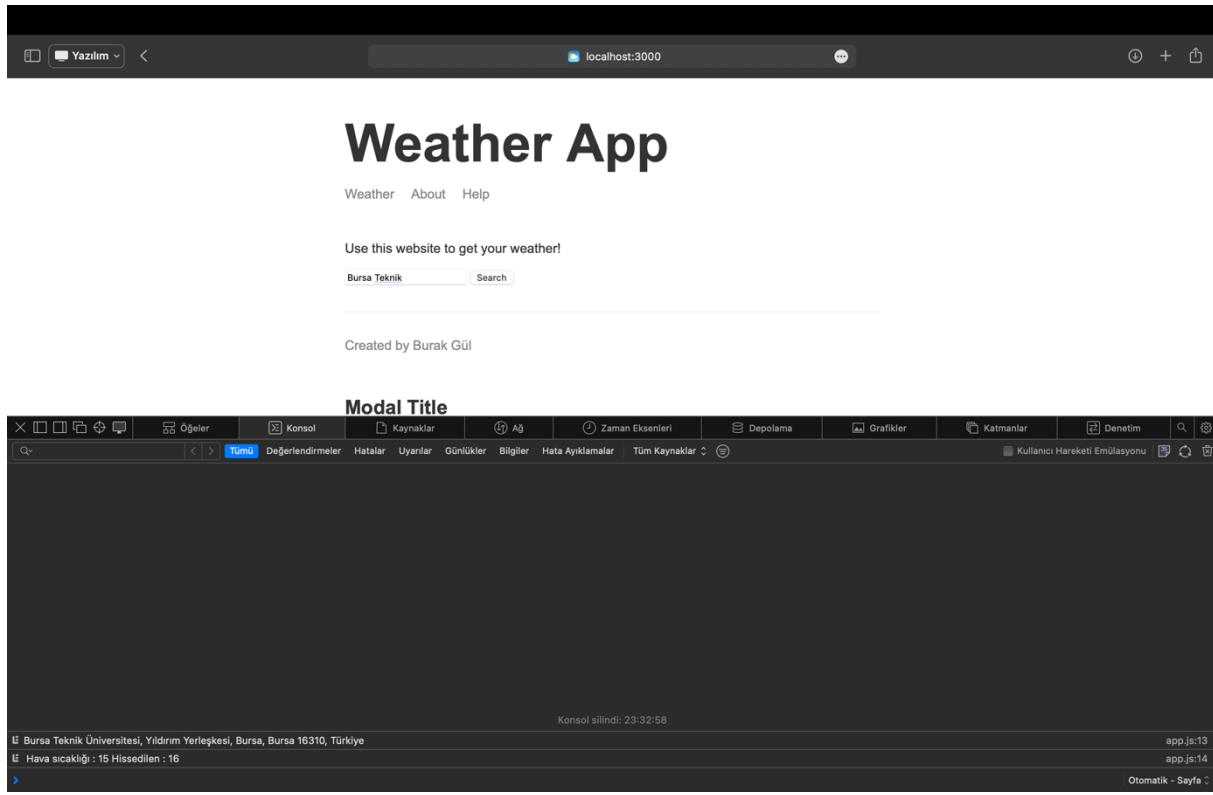
```
Section5-6-7-8-9-10 - app.js
```

```
const weatherForm = document.querySelector("form")
const search = document.querySelector("input")

weatherForm.addEventListener("submit", (e) => {
  e.preventDefault()
  const location = search.value
  console.log(location)
})
```



Gördüğümüz üzere burası çalışmaktadır şimdi fetch fonksiyonunu bu event listener içinde çağıralım.



Verimiz de geliyor süper.

Şimdi ise index.hbs de paragraflara id atayalım doğru bir şekilde kullanabilmek için.

Body kısmı şöyle olmalıdır :

```
Section5-6-7-8-9-10 - index.hbs

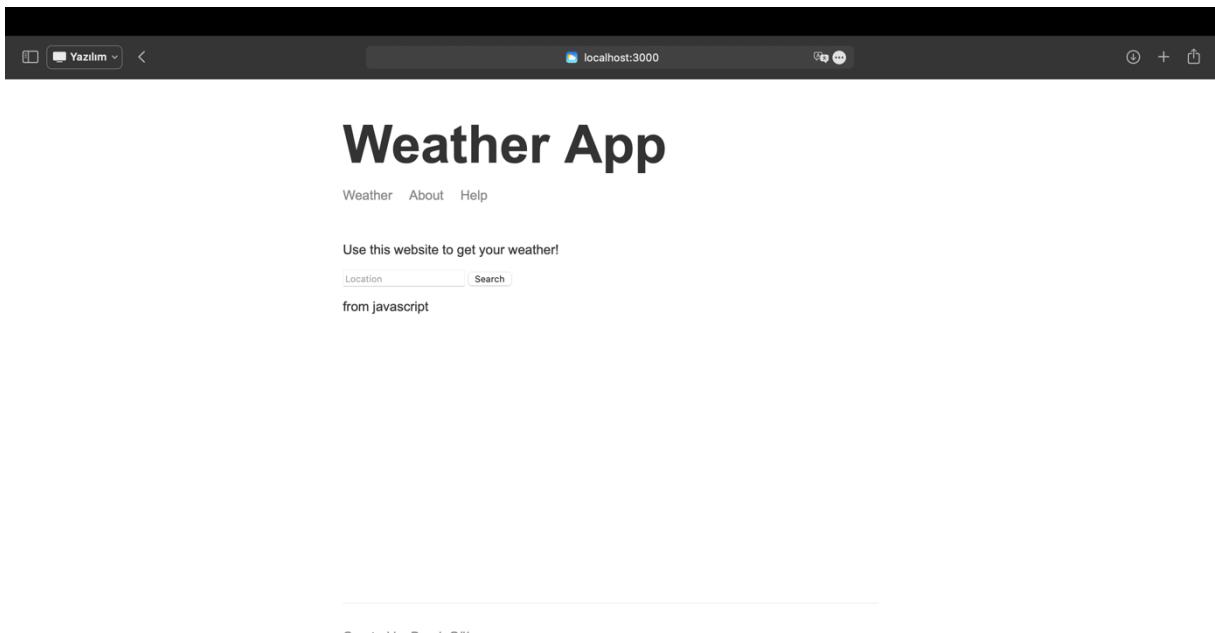
<body>
  <div class="main-content">
    {{>header}}
    <p>Use this website to get your weather!</p>
    <form>
      <input placeholder="Location">
      <button>Search</button>
      <p id="message-1"></p>
      <p id="message-2"></p>
    </form>

  </div>
  {{>footer}}
  <script src="js/app.js" defer></script>
</body>
```

Daha sonra js/app.js içine girelim.

```
Section5-6-7-8-9-10 - app.js
const messageOne = document.querySelector("#message-1")
messageOne.textContent = "from javascript"
```

Bu 2 satır kodu ekleyelim ve sayfayı yenileyelim.



From javascript geldi gördüğümüz üzere.

Second message'ı da ekleyelim ve eventlistener'ı güncelleyelim.

App.js son hali şu şekilde olmalı :

```
Section5-6-7-8-9-10 - app.js

const weatherForm = document.querySelector("form")
const search = document.querySelector("input")
const messageOne = document.querySelector("#message-1")
const messageTwo = document.querySelector("#message-2")

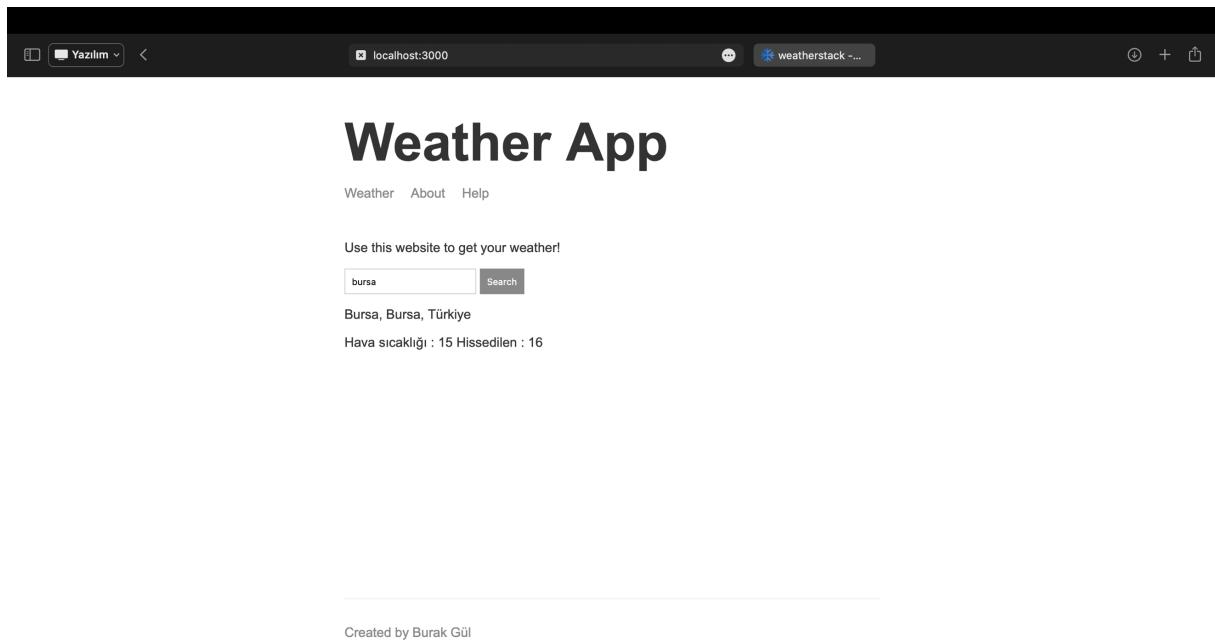
weatherForm.addEventListener("submit", (e) => {
  e.preventDefault()
  const location = search.value
  messageOne.textContent = "Loading..."
  messageOne.textContent = ""
  fetch("http://localhost:3000/weather?address=" + location).then(
    (response) => {
      response.json().then((data) => {
        if (data.error) {
          messageOne.textContent = data.error
        } else {
          messageOne.textContent = data.location
          messageTwo.textContent = data.forecast
        }
      })
    }
  )
})
```

Ve css dosyasına şunu da ekleyerek güzelleştirelim :

```
Section5-6-7-8-9-10 - styles.css

input {
  border: 1px solid #cccccc;
  padding: 8px;
}
button {
  cursor: pointer;
  border: 1px solid #888888;
  background: #888888;
  color: white;
  padding: 8px;
}
```

Şimdi de bir veri çekelim.



Eğer hata alırsanız ki almamız yüksek geocode.js içinde request fonksiyonunda body ye default olarak boş obje atayarak çözeriz o da artık bu okuyucuya ödev 😊 .