

## BURSA TEKNİK ÜNİVERSİTESİ NODE.JS ile WEB PROGLAMA DERSİ 2.HAFTA

### TEORİ DERSİ RAPORU

Bu haftanın konusu komut satırından argüman almaktır. İlk başta npm modülünü kullanmadan yapacağımız daha sonra npm modülü kullanarak bu işlemin daha kolay olduğunu gözlemleyeceğiz. Önce bir klasör (2\_week) açalım ve içinde app.js dosyamızı oluşturalım daha sonra bunu visual studio code ile açalım. Terminalde app.js dosyasına gidelim. Daha sonra terminal de npm init komutunu çalıştıralım. Default olarak gelen sorulara enter'a basarak cevap verelim.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\user\Desktop\2_week> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

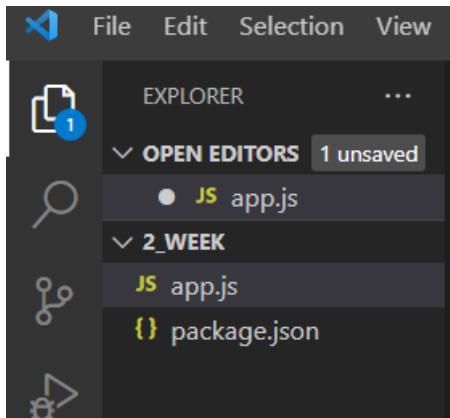
Press ^C at any time to quit.
package name: (2_week)
version: (1.0.0)
description:
git repository:
keywords:
author:
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

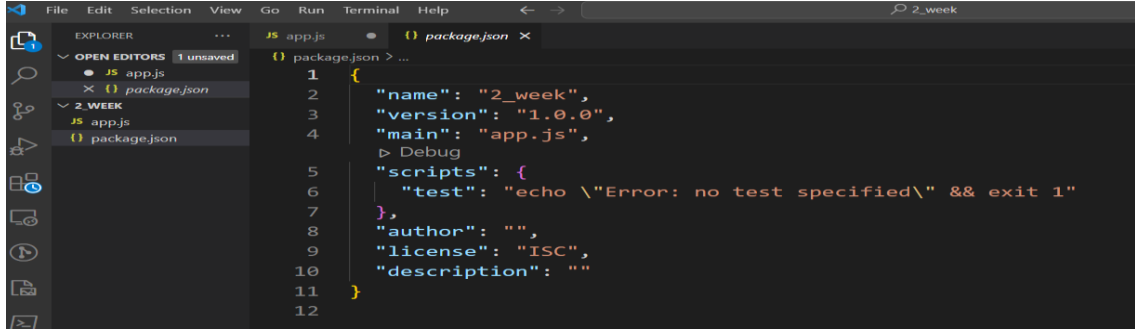
{
  "name": "2_week",
  "version": "1.0.0",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "description": ""
}

Is this OK? (yes)
PS C:\Users\user\Desktop\2_week> █
```

Yapılan bu işlem doğrultusunda package.json isimli bir dosya oluşturur.



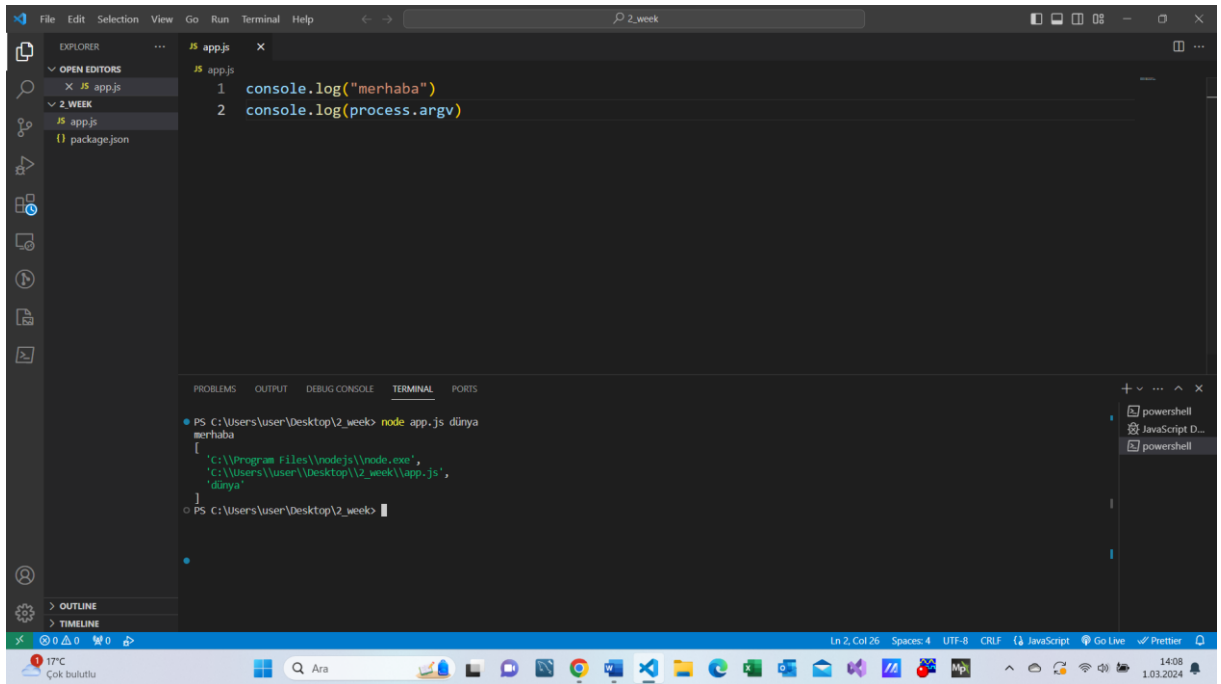
Bu dosyada az önce npm init komutu çalıştırılırken sorulara verilen cevaplar yer alır.



```
1 {
2   "name": "2_week",
3   "version": "1.0.0",
4   "main": "app.js",
5   "scripts": {
6     "test": "echo \"Error: no test specified\" && exit 1"
7   },
8   "author": "",
9   "license": "ISC",
10  "description": ""
11 }
12
```

Kaynak kodu çalıştırırken terminalde node app.js deriz. Ama bazen komut satırından argüman göndermek isteriz yani kodumuza çalışma esnasında nasıl argüman alabileceğimizi öğreneceğiz.

Argv: argüman (dışardan alınan parametreler) vektör (aslında array) demektir.

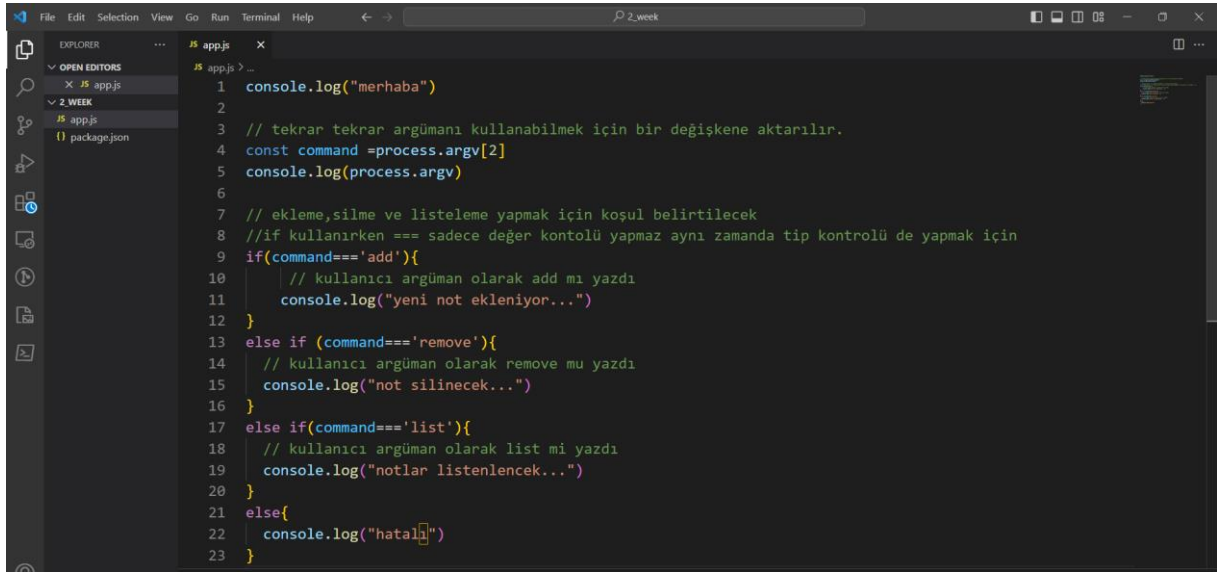


```
1 console.log("merhaba")
2 console.log(process.argv)
```

```
PS C:\Users\User\Desktop\2_week> node app.js dünya
merhaba
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\User\\Desktop\\2_week\\app.js',
  'dünya'
]
```

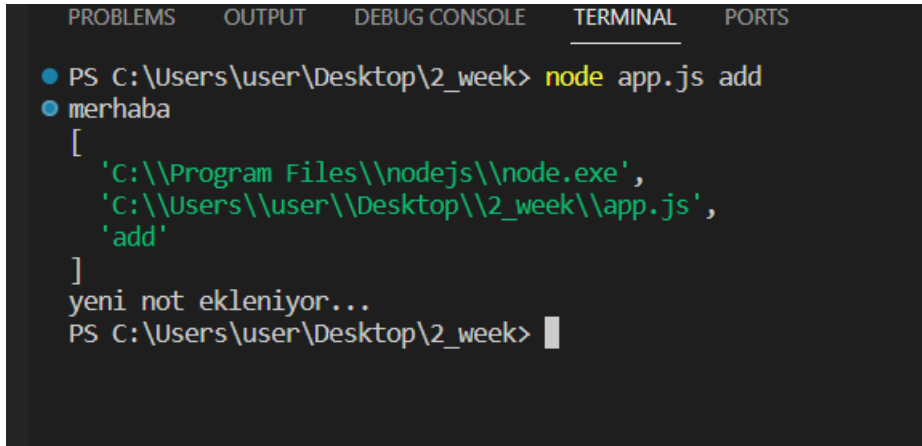
Terminalden dünya parametresi argüman olarak gönderilmiştir. Çıktısında 3 adet elemanı olan array getirir. Birincisi node un path bilgisi ikincisi ise çalıştırmak istediğimiz kaynak dosyanın path bilgisi üçüncüsü ise yazılan argümanlardır. Ama biz bir not alma uygulaması geliştiriyoruz o zaman not ekleme, silme ve listeleme özelliklerini argüman olarak dışarıdan alabilmeliyiz. Şimdi koda bazı değişiklikler yapalım vektör elemanları 0'dan başlar ama ilk 0 ve 1 bizim için önemli olmayan yukarıdaki konum bilgileridir bunun için 2 yazalım kullanıcı not

ekleme, silme ve listeleme işlemlerini yapabilmelidir.



```
1 console.log("merhaba")
2
3 // tekrar tekrar argümanı kullanabilmek için bir değişkene aktarılır.
4 const command =process.argv[2]
5 console.log(process.argv)
6
7 // ekleme,silme ve listeleme yapmak için koşul belirtilecek
8 //if kullanırken == sadece değer kontrolü yapmaz aynı zamanda tip kontrolü de yapmak için
9 if(command==='add'){
10     // kullanıcı argüman olarak add mı yazdı
11     console.log("yeni not ekleniyor...")
12 }
13 else if (command==='remove'){
14     // kullanıcı argüman olarak remove mu yazdı
15     console.log("not silinecek...")
16 }
17 else if(command==='list'){
18     // kullanıcı argüman olarak list mi yazdı
19     console.log("notlar listenlenecek...")
20 }
21 else{
22     console.log("hatalı")
23 }
```

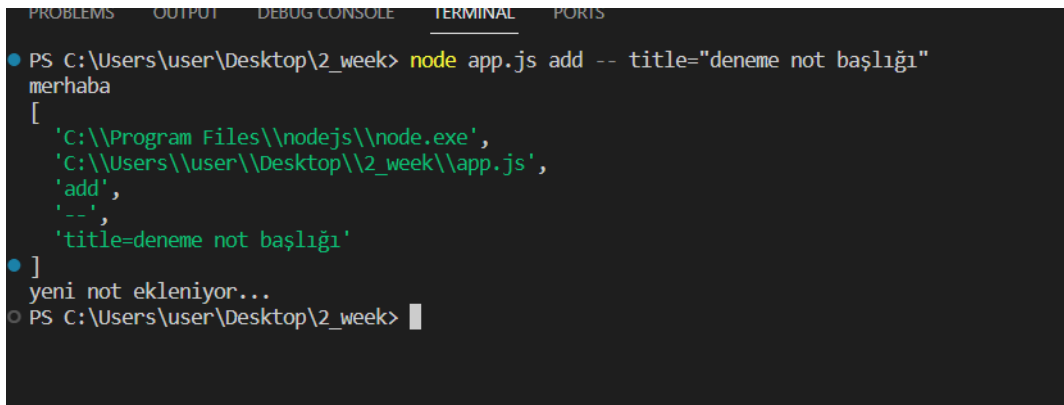
Çıktısı:



```
PS C:\Users\user\Desktop\2_week> node app.js add
merhaba
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\user\\Desktop\\2_week\\app.js',
  'add'
]
yeni not ekleniyor...
PS C:\Users\user\Desktop\2_week>
```

Notun başlık bilgisini oluşturalım. Yani yeni bir not eklerken ihtiyacımız olan veri notun başlık bilgisidir.

Terminale node app.js add -- title="deneme not başlığı" yazarsan çıktısı şu şekilde olur.



```
PS C:\Users\user\Desktop\2_week> node app.js add -- title="deneme not başlığı"
merhaba
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\user\\Desktop\\2_week\\app.js',
  'add',
  '--',
  'title=deneme not başlığı'
]
yeni not ekleniyor...
PS C:\Users\user\Desktop\2_week>
```

4 elemanlı bir array döndürür. Npm modülü kullanmadan bunu manipüle edebiliriz kodun içine aktarabiliriz ama bir parsing işlemi yapmalıyız. Fakat parsing işlemi bizim zamanımızı

oldukça alacaktır bunu önlemek için npm modülünü kullanmak daha mantıklı olacaktır. Npm “yargs” modülünü kullanalım. Terminal npm intall yargs yazalım ve bu modülü ekleyelim modülün dahil olup olmadığını öğrenmek için package.json dosyasına bakalım.

```
PS C:\Users\user\Desktop\2_week> npm install yargs
added 16 packages, and audited 17 packages in 1m

2 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\user\Desktop\2_week>
PS C:\Users\user\Desktop\2_week> 
```

```
0 package.json > {} dependencies
1 {
2   "name": "2_week",
3   "version": "1.0.0",
4   "description": "",
5   "main": "app.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "",
10  "license": "ISC",
11  "dependencies": {
12    "yargs": "^17.7.2"
13  }
```

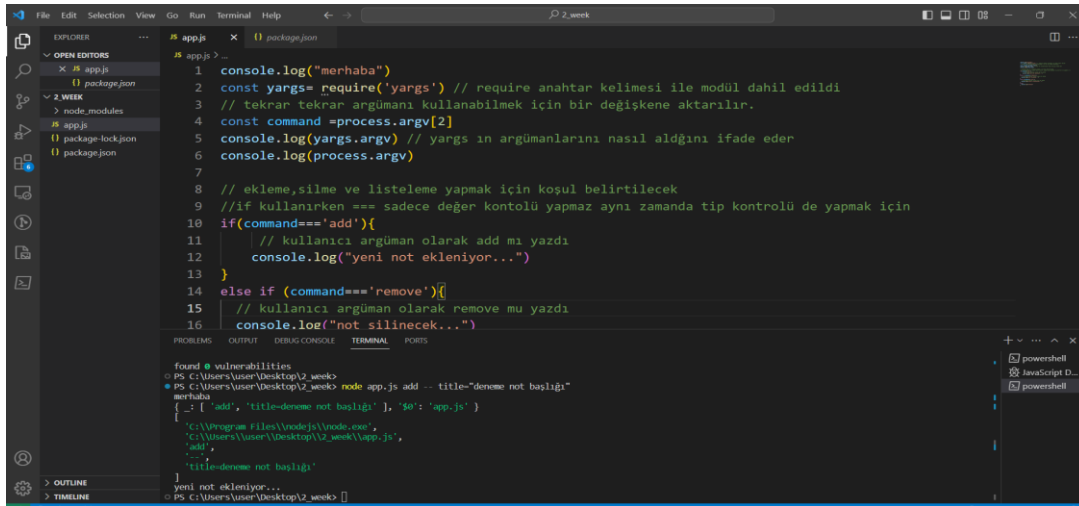
Dependencies içinde yargs modülü bulunuyor ve versiyonu 17.7.2 dir. Npm modüllerini kullanabilmek için require anahtar kelimesi ile komut yazılır. Daha detaylı bilgi için <https://www.npmjs.com/package/yargs> adresini ziyaret edebilirsiniz.

Yargs modülünü kullanabilmek için şu kodların yazılması gerekli.

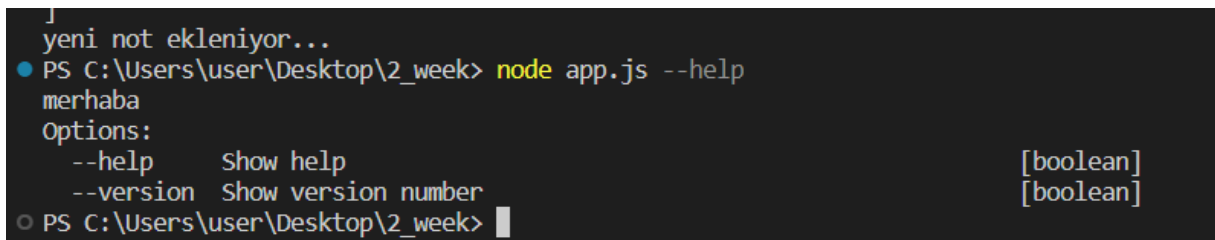
```
const yargs= require('yargs') // require anahtar
```

```
console.log(yargs.argv) // yargs ın argümanlarını nasıl aldığını ifade eder
```

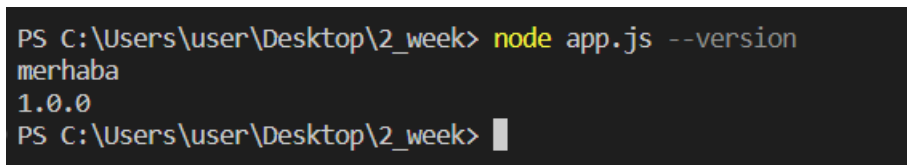
Çıktısı:



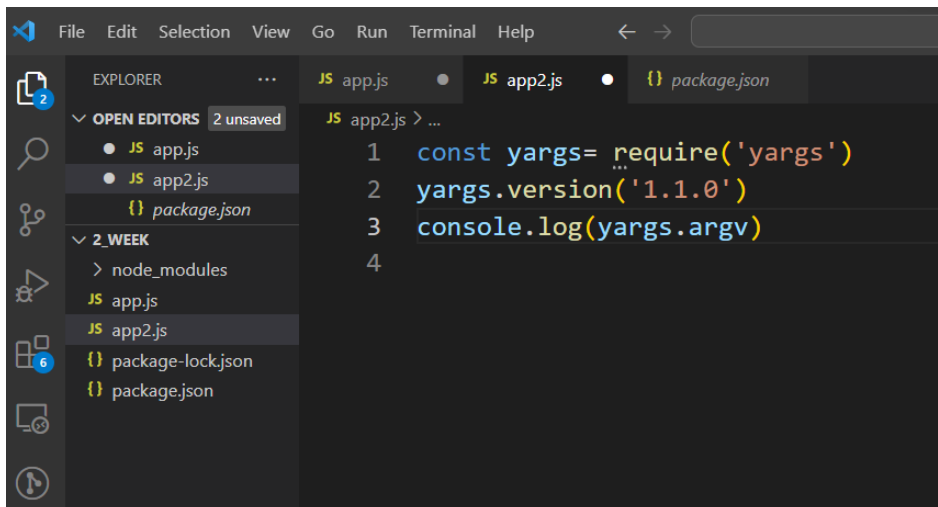
Terminale `node app.js --help` yazınca bazı bilgiler verdi bu bilgilere nasıl erişeceğiz veya nasıl değiştireceğiz bunu inceyelim.



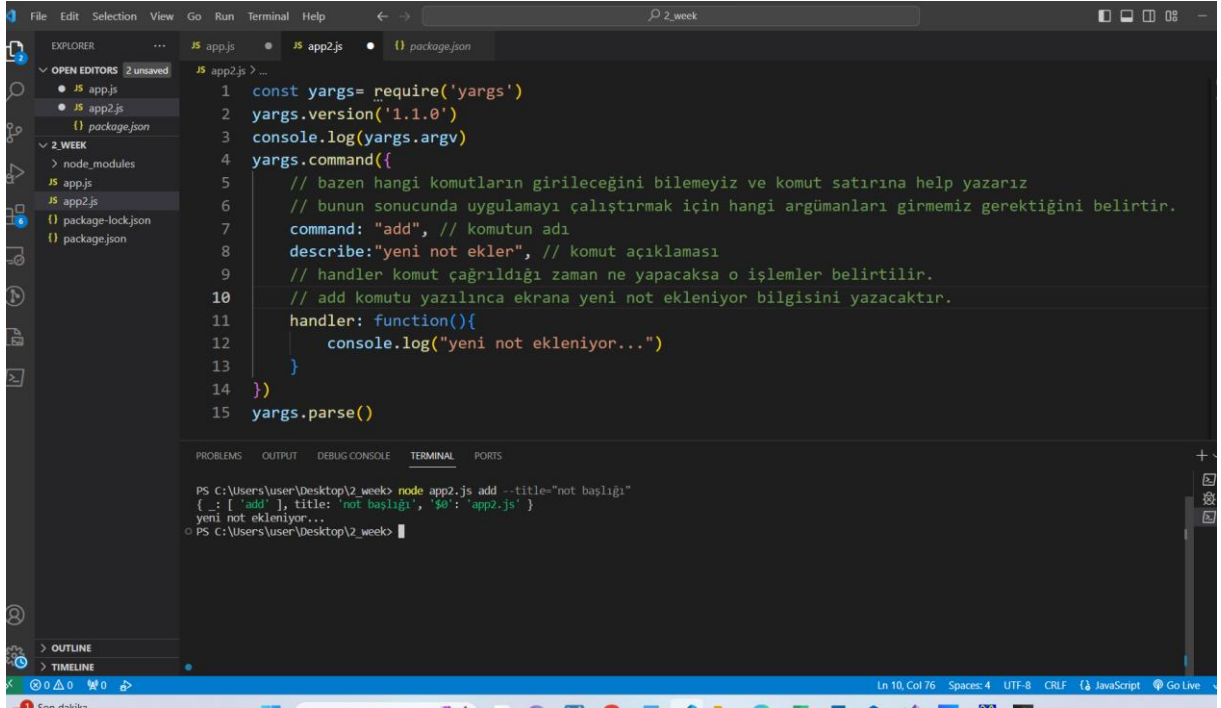
Versiyon bilimiz:



Yazılan kodların karışmaması için yeni bir dosya açalım adına `app2.js` diyelim ve içine bazı kodları ekleyelim.



Şimdi `yargs` modülünü kullanarak nasıl not ekleyebiliriz bunu inceleyelim. Birazdan yapılan işlemler her modül için geçerli değildir bu işlemler `yargs` modülü için geçerlidir.

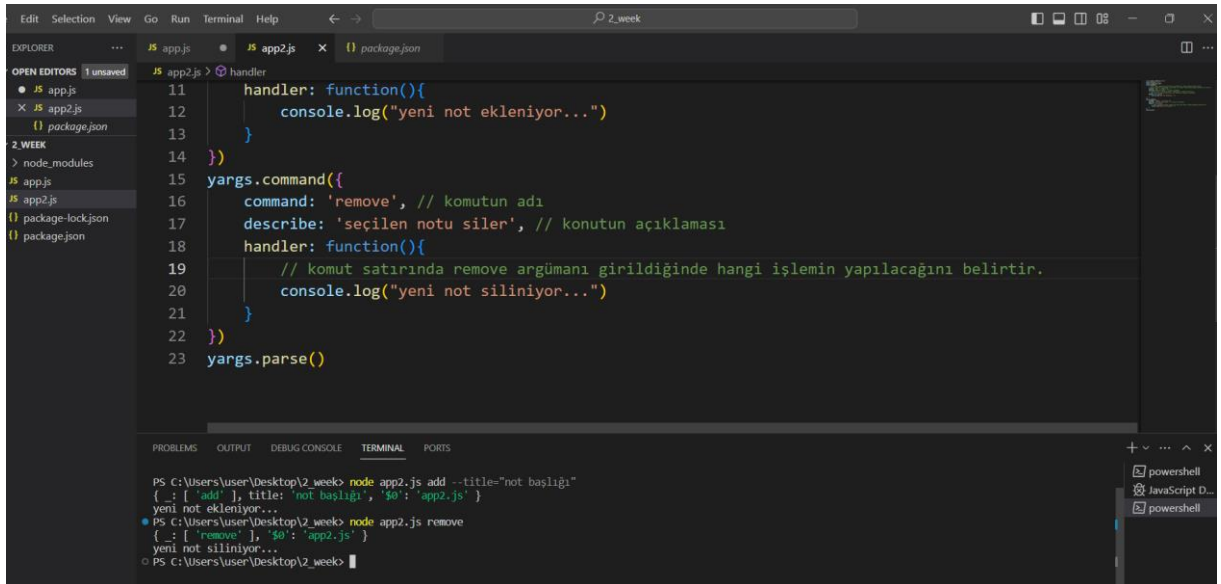


```
1 const yargs= require('yargs')
2 yargs.version('1.1.0')
3 console.log(yargs.argv)
4 yargs.command({
5   // bazen hangi komutların girileceğini bilemeyiz ve komut satırına help yazarız
6   // bunun sonucunda uygulamayı çalıştırmak için hangi argümanları girmemiz gerektiğini belirtir.
7   command: "add", // komutun adı
8   describe:"yeni not ekler", // komut açıklaması
9   // handler komut çağrıldığı zaman ne yapacaksa o işlemler belirtilir.
10  // add komutu yazılınca ekrana yeni not ekleniyor bilgisini yazacaktır.
11  handler: function(){
12    console.log("yeni not ekleniyor...")
13  }
14 })
15 yargs.parse()
```

```
PS C:\Users\User\Desktop\2_week> node app2.js add --title="not başlığı"
{ _: [ 'add' ], title: 'not başlığı', '$0': 'app2.js' }
yeni not ekleniyor...
```

Bir önceki konu da karşılaştırma yaparken birçok if else blokları kullanmak zorunda kalmıştık ama burada ise yargs.command ile if else kullanmadan sorgu yapabiliriz.

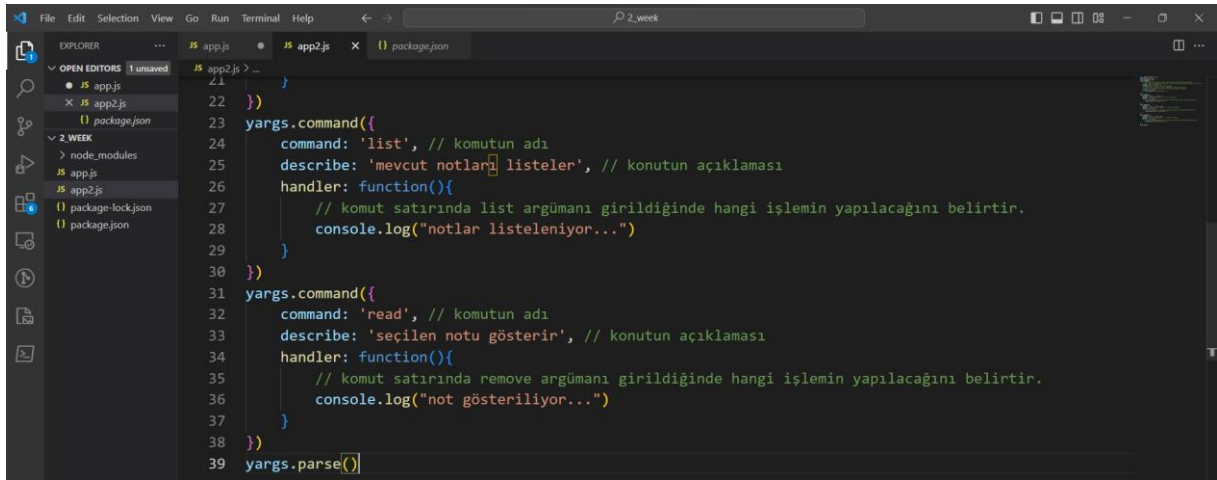
Devam edelim ve silme komutunu da ekleyelim.



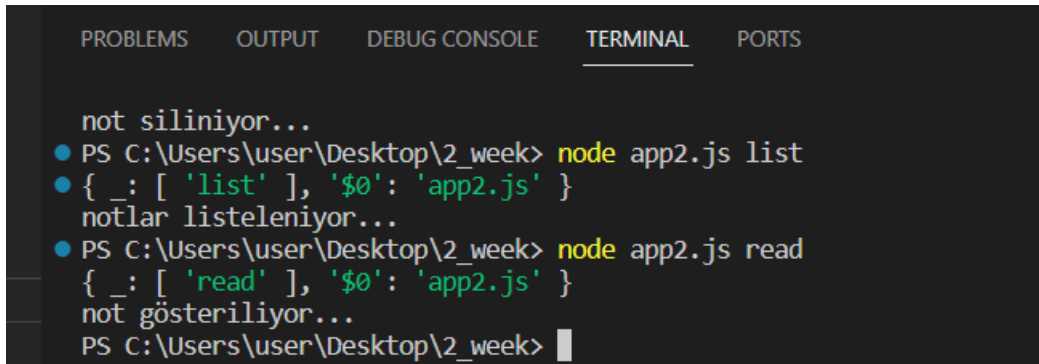
```
11 handler: function(){
12   console.log("yeni not ekleniyor...")
13 }
14 })
15 yargs.command({
16   command: 'remove', // komutun adı
17   describe: 'seçilen notu siler', // komutun açıklaması
18   handler: function(){
19     // komut satırında remove argümanı girildiğinde hangi işlemin yapılacağını belirtir.
20     console.log("yeni not siliniyor...")
21   }
22 })
23 yargs.parse()
```

```
PS C:\Users\User\Desktop\2_week> node app2.js add --title="not başlığı"
{ _: [ 'add' ], title: 'not başlığı', '$0': 'app2.js' }
yeni not ekleniyor...
PS C:\Users\User\Desktop\2_week> node app2.js remove
{ _: [ 'remove' ], '$0': 'app2.js' }
yeni not siliniyor...
```

Kısacası yargs modülü girilen argümana göre hangi kısmın çalışacağını kıyaslar ve buna bağlı olarak hangi fonksiyonun çalışacağına karar verir. Listeleme ve read komutlarını da uygulamaya dahil edelim listeleme komutu mevcut olan notları listelerken read komutu ise listeleme sırasında titlelardan birini seçerek notu okuma işlemini gerçekleştirir.



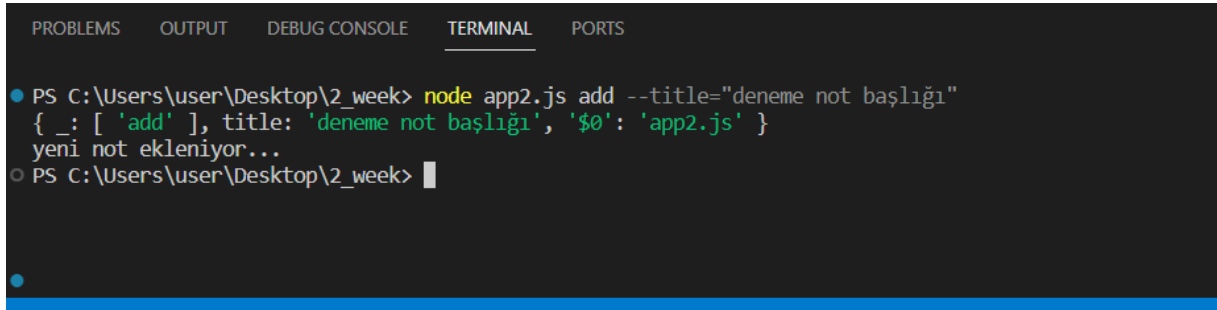
```
21 }
22 })
23 yargs.command({
24   command: 'list', // komutun adı
25   describe: 'mevcut notları listeler', // konunun açıklaması
26   handler: function(){
27     // komut satırında list argümanı girildiğinde hangi işlemin yapılacağını belirtir.
28     console.log("notlar listeleniyor...")
29   }
30 })
31 yargs.command({
32   command: 'read', // komutun adı
33   describe: 'seçilen notu gösterir', // konunun açıklaması
34   handler: function(){
35     // komut satırında remove argümanı girildiğinde hangi işlemin yapılacağını belirtir.
36     console.log("not gösteriliyor...")
37   }
38 })
39 yargs.parse()
```



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

not siliniyor...
● PS C:\Users\user\Desktop\2_week> node app2.js list
● { _: [ 'list' ], '$0': 'app2.js' }
notlar listeleniyor...
● PS C:\Users\user\Desktop\2_week> node app2.js read
● { _: [ 'read' ], '$0': 'app2.js' }
not gösteriliyor...
PS C:\Users\user\Desktop\2_week> 
```

Şimdiki konumuz ise title ile işlemler yapabilmektir.

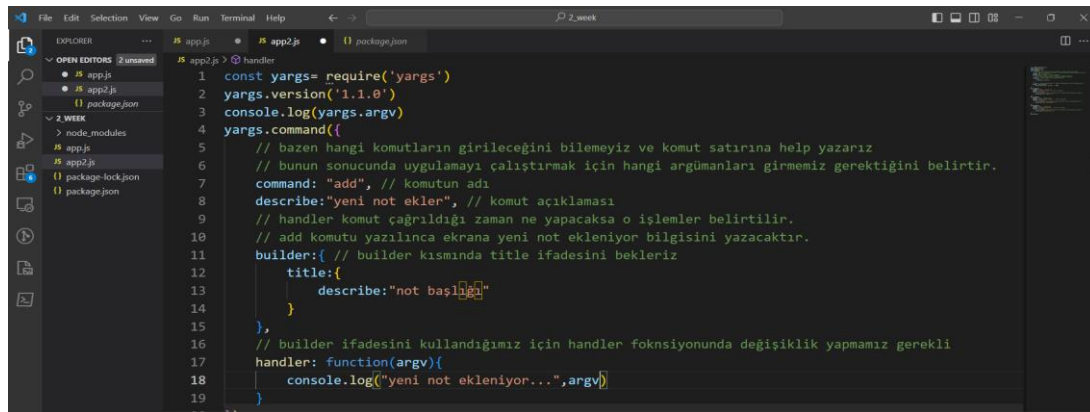


```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS C:\Users\user\Desktop\2_week> node app2.js add --title="deneme not başlığı"
● { _: [ 'add' ], title: 'deneme not başlığı', '$0': 'app2.js' }
yeni not ekleniyor...
○ PS C:\Users\user\Desktop\2_week> 
```

Görüldüğü gibi --title=" deneme not başlığı" ifadesinin çıktıya hiçbir etkisi yoktu olsa da çalışır olmasa da o zaman bu bilgiyi anlamlı kılabilmek için düzenleme yapalım. Builder anahtar kelimesi ile bazı değişiklikler yapalım ve build builder ifadesini kullandığımız için handler fonksiyonunda değişiklik yapmamız gerekli der ifadesini kullandığımız için handler fonksiyonunda değişiklik yapmamız gerekli.



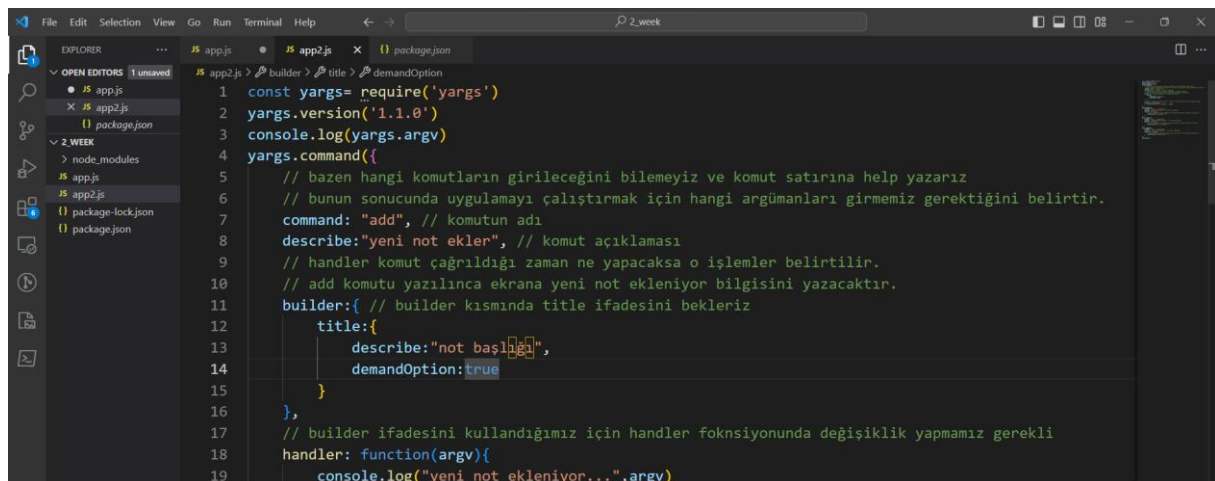


Çıktısı:

```
PS C:\Users\user\Desktop\2_week> node app2.js add --title="Eklenen Not Başlığı"
{ _: [ 'add' ], title: 'Eklenen Not Başlığı', '$0': 'app2.js' }
yeni not ekleniyor...
PS C:\Users\user\Desktop\2_week>
```

```
PS C:\Users\user\Desktop\2_week> node app2.js add
{ _: [ 'add' ], '$0': 'app2.js' }
yeni not ekleniyor...
```

Ama şu anda title bilgisi olmadan da komut çalışıyor bizim istediğimiz olay komutun çalışabilmesi için title verisinin girilmiş olmasının gerekmesidir. Builder kısmında bazı komutlar eklenmeli demandOption:true eklenir.



Terminale title olmadan node app2.js add yazarsan hata alırsın.

```
PS C:\Users\user\Desktop\2_week> node app2.js add
{ _: [ 'add' ], '$0': 'app2.js' }
app2.js add
```

• yeni not ekler

Options:

--help	Show help	[boolean]
--------	-----------	-----------



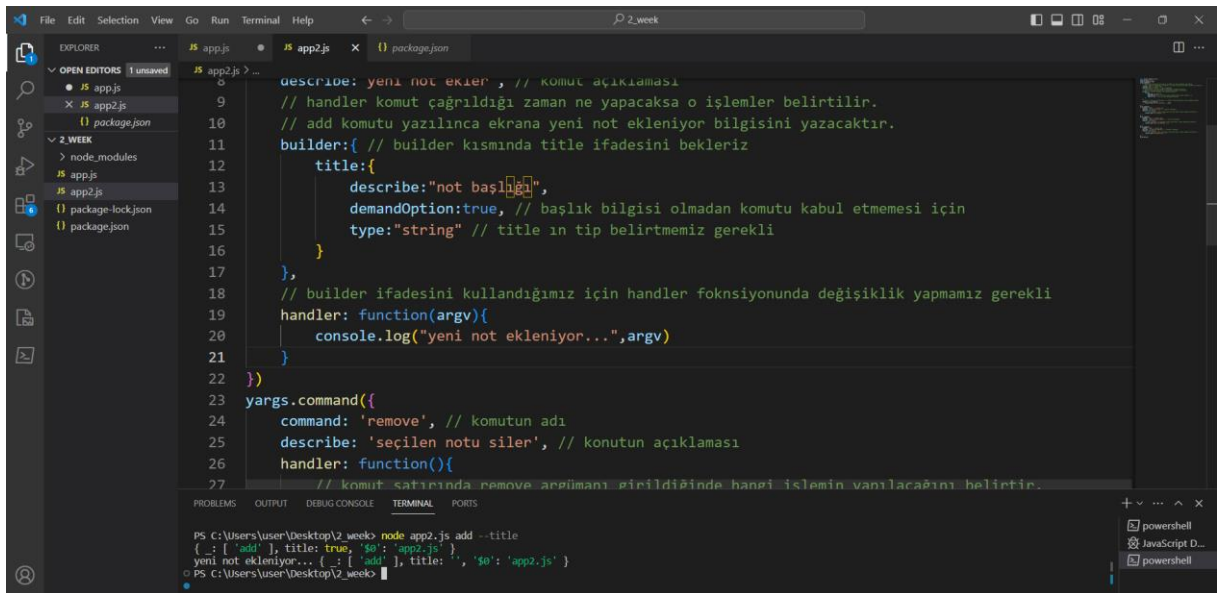
```
Options:
  --help      Show help                                [boolean]
  --version   Show version number                       [boolean]
  --title     not başlığı                               [required]

Missing required argument: title
PS C:\Users\user\Desktop\2_week>
```

Fakat terminale node app2.js add --title= veya node app2.js title yazınca hata vermez bu ifadeleri kabul ediyor.

```
PS C:\Users\user\Desktop\2_week> node app2.js add --title=
{ _: [ 'add' ], title: '', '$0': 'app2.js' }
yeni not ekleniyor... { _: [ 'add' ], title: '', '$0': 'app2.js' }
PS C:\Users\user\Desktop\2_week> node app2.js add --title
{ _: [ 'add' ], title: true, '$0': 'app2.js' }
yeni not ekleniyor... { _: [ 'add' ], title: true, '$0': 'app2.js' }
PS C:\Users\user\Desktop\2_week>
```

Kabul etmesinin sebebi title için herhangi bir değer ataması yapılmadığında js tipinin ne olduğunu bilemez ve bu yüzden default olarak boolean true olarak değer ataması yapar. Type: "string" kodunu eklememiz lazım.



The screenshot shows the VS Code editor with the `app2.js` file open. The file contains a Yargs command definition for `add`. The `title` option is defined with `type: "string"`. The terminal output shows the command `node app2.js add --title` being executed, resulting in the title being set to `true` instead of a string.

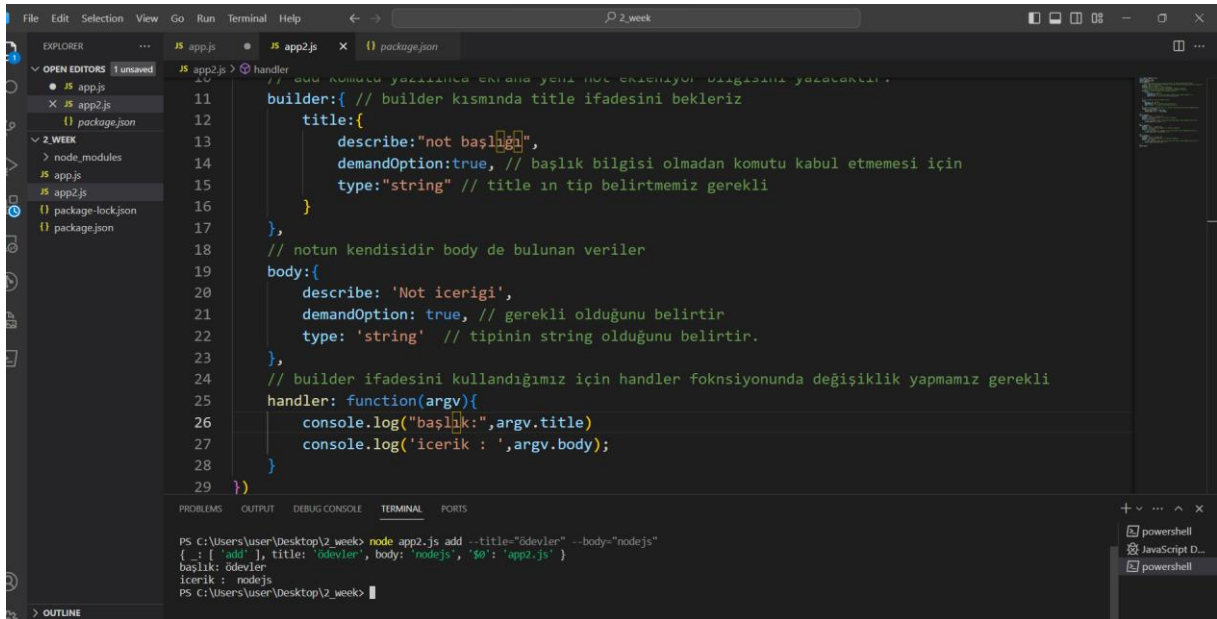
```
describe: yeni not ekler, // komut açıklaması
// handler komut çağrıldığı zaman ne yapacaksa o işlemler belirtilir.
// add komutu yazılınca ekrana yeni not ekleniyor bilgisini yazacaktır.
builder: { // builder kısmında title ifadesini bekleriz
  title: {
    describe: "not başlığı",
    demandOption: true, // başlık bilgisi olmadan komutu kabul etmemesi için
    type: "string" // title ın tip belirtmemiz gerekli
  },
},
// builder ifadesini kullandığımız için handler fonksiyonunda değişiklik yapmamız gerekli
handler: function(argv) {
  console.log("yeni not ekleniyor...", argv)
}

yargs.command({
  command: 'remove', // komutun adı
  describe: 'seçilen notu siler', // konunun açıklaması
  handler: function() {
    // komut satırında remove argümanı girildiğinde hangi işlemin yapılacağını belirtir.
  }
})
```

Terminal Output:

```
PS C:\Users\user\Desktop\2_week> node app2.js add --title
{ _: [ 'add' ], title: true, '$0': 'app2.js' }
yeni not ekleniyor... { _: [ 'add' ], title: true, '$0': 'app2.js' }
PS C:\Users\user\Desktop\2_week>
```

Ama hala title değer ataması yapılmadan argüman verildiğinde title kısmını artık true olarak değil de boş olarak gösteriyor biz bunu da istemiyoruz. Başlık ve gövde kısmını ekleyelim.

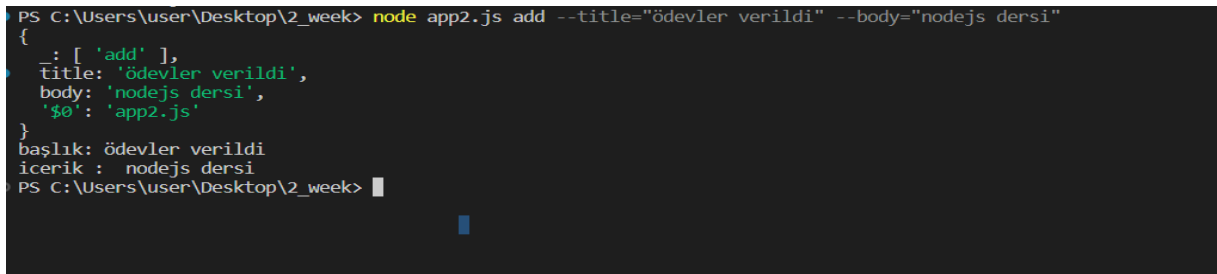


The screenshot shows the VS Code editor with the file explorer on the left. The '2\_WEEK' folder is expanded, showing 'node\_modules', 'app.js', 'app2.js', 'package-lock.json', and 'package.json'. The 'app2.js' file is open in the editor, showing a JavaScript script that uses the 'builder' and 'handler' functions. The terminal at the bottom shows the command 'node app2.js add --title="ödevler" --body="nodejs"' and its output, which is a JSON object with 'title' and 'body' properties.

```
10 // bu komutu yazdıktan en alta yeni not ekleniyor başlığını yazdıkta.
11 builder: { // builder kısmında title ifadesini bekleriz
12   title: {
13     describe: "not başlığı",
14     demandOption: true, // başlık bilgisi olmadan komutu kabul etmemesi için
15     type: "string" // title in tip belirtmemiz gerekli
16   },
17 },
18 // notun kendisidir body de bulunan veriler
19 body: {
20   describe: 'Not icerigi',
21   demandOption: true, // gerekli olduğunu belirtir
22   type: 'string' // tipinin string olduğunu belirtir.
23 },
24 // builder ifadesini kullandığımız için handler fonksiyonunda değişiklik yapmamız gerekli
25 handler: function(argv){
26   console.log("başlık:", argv.title)
27   console.log('icerik : ', argv.body);
28 }
29 })
```

```
PS C:\Users\user\Desktop\2_week> node app2.js add --title="ödevler" --body="nodejs"
{ _: [ 'add' ], title: 'ödevler', body: 'nodejs', '$0': 'app2.js' }
başlık: ödevler
icerik : nodejs
PS C:\Users\user\Desktop\2_week>
```

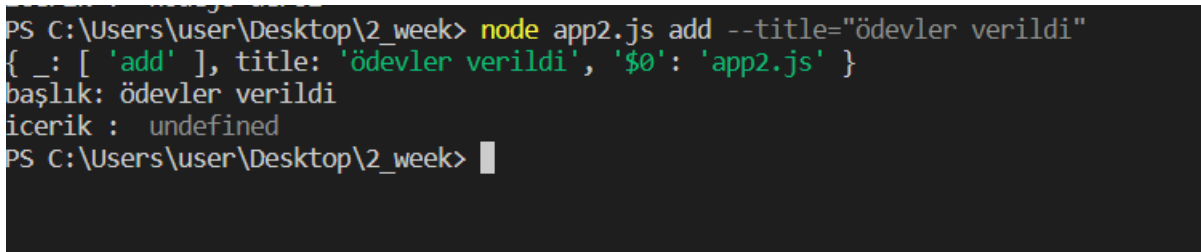
Body bloğu eklenir ve bu şekilde nota erişilmiş olunur terminaldeki çıktısı da başlık: ödevler ve içerik: nodejs şeklindedir.



The screenshot shows a terminal window with the command 'node app2.js add --title="ödevler verildi" --body="nodejs dersi"' and its output, which is a JSON object with 'title' and 'body' properties.

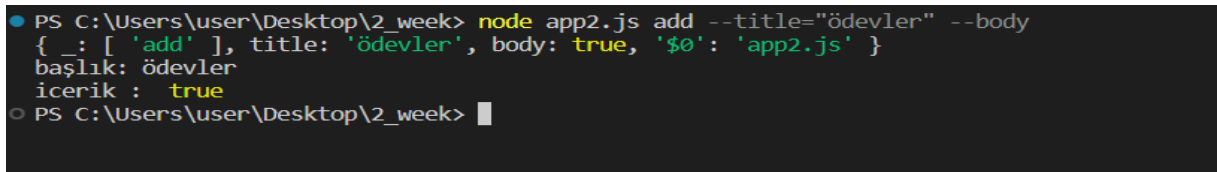
```
PS C:\Users\user\Desktop\2_week> node app2.js add --title="ödevler verildi" --body="nodejs dersi"
{ _: [ 'add' ], title: 'ödevler verildi', body: 'nodejs dersi', '$0': 'app2.js' }
başlık: ödevler verildi
icerik : nodejs dersi
PS C:\Users\user\Desktop\2_week>
```

Body yazmadan terminalde çalıştırsan şu şekilde çıktı verir.



The screenshot shows a terminal window with the command 'node app2.js add --title="ödevler verildi"' and its output, which is a JSON object with 'title' and '\$0' properties, but the 'body' property is undefined.

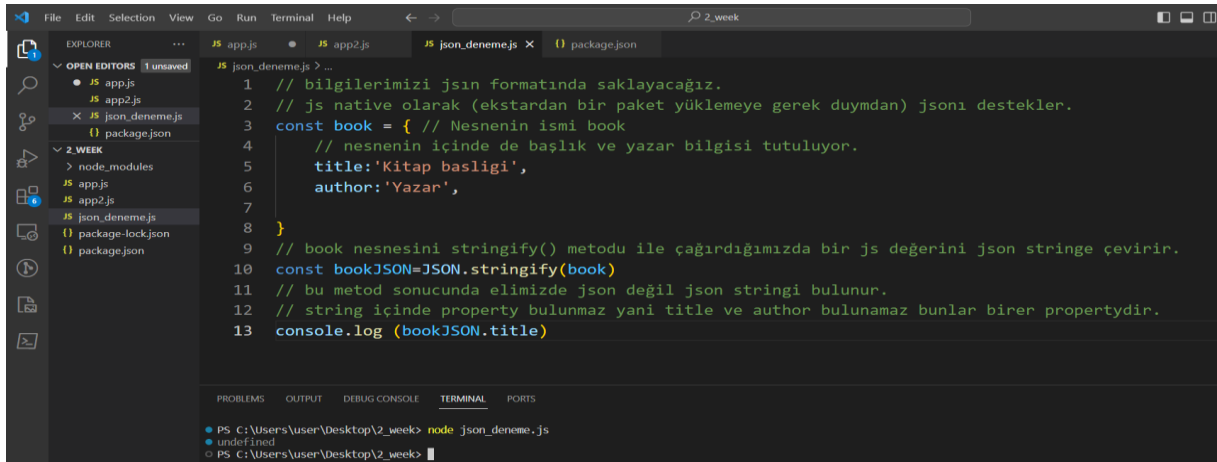
```
PS C:\Users\user\Desktop\2_week> node app2.js add --title="ödevler verildi"
{ _: [ 'add' ], title: 'ödevler verildi', '$0': 'app2.js' }
başlık: ödevler verildi
icerik : undefined
PS C:\Users\user\Desktop\2_week>
```



The screenshot shows a terminal window with the command 'node app2.js add --title="ödevler" --body true' and its output, which is a JSON object with 'title' and 'body' properties, where 'body' is true.

```
PS C:\Users\user\Desktop\2_week> node app2.js add --title="ödevler" --body true
{ _: [ 'add' ], title: 'ödevler', body: true, '$0': 'app2.js' }
başlık: ödevler
icerik : true
PS C:\Users\user\Desktop\2_week>
```

Bu kısımda json\_deneme.js dosyasını açalım. Json nesneleri oluşturalım ve her json nesnesi bir tane not tutulacaktır. Aşağıdaki örnekte bir json nesnesi oluşturuldu ve stringify() metodu ile de json stringine çevrildi.



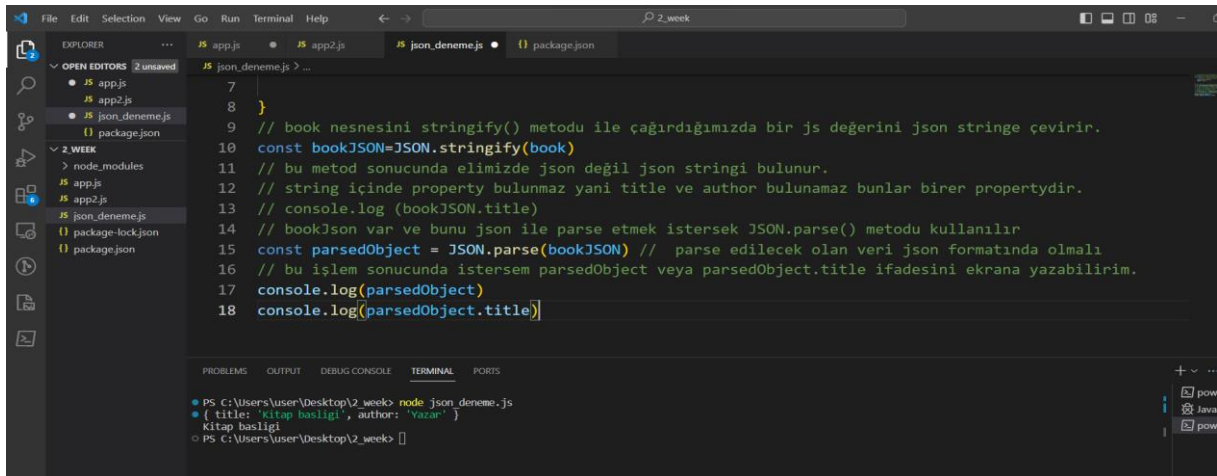
```
1 // bilgilerimizi jsın formatında saklayacağız.
2 // js native olarak (ekstardan bir paket yüklemeye gerek duymadan) jsonı destekler.
3 const book = { // Nesnenin ismi book
4   // nesnenin içinde de başlık ve yazar bilgisi tutuluyor.
5   title: 'Kitap basligi',
6   author: 'Yazar',
7 }
8
9 // book nesnesini stringify() metodu ile çağırdığımızda bir js değerini json stringe çevirir.
10 const bookJSON=JSON.stringify(book)
11 // bu metod sonucunda elimizde json değil json stringi bulunur.
12 // string içinde property bulunmaz yani title ve author bulunamaz bunlar birer propertydir.
13 console.log (bookJSON.title)
```

PS C:\Users\User\Desktop\2\_week> node json\_deneme.js

undefined

PS C:\Users\User\Desktop\2\_week>

Stringify() metodu sonucunda elimizde bir json stringi oluşacağı için console.log() üzerinden o nesnenin hiçbir özelliğine erişemem terminal çıktısında da bu ifadenin tanımsız olduğunu belirtmiş. Şimdi bu işlemin tam tersine bakalım. Elimizde bookJson var ve bunu json ile parse etmek istersek aşağıdaki işlemler yapılmalıdır. Parse etme işlemini JSON.parse(parse edilecek veri) metodu kullanılır. Parse edilecek veri json formatında olmalıdır.



```
7
8 }
9 // book nesnesini stringify() metodu ile çağırdığımızda bir js değerini json stringe çevirir.
10 const bookJSON=JSON.stringify(book)
11 // bu metod sonucunda elimizde json değil json stringi bulunur.
12 // string içinde property bulunmaz yani title ve author bulunamaz bunlar birer propertydir.
13 // console.log (bookJSON.title)
14 // bookJson var ve bunu json ile parse etmek istersek JSON.parse() metodu kullanılır
15 const parsedObject = JSON.parse(bookJSON) // parse edilecek olan veri json formatında olmalı
16 // bu işlem sonucunda istersem parsedObject veya parsedObject.title ifadesini ekrana yazabilirim.
17 console.log(parsedObject)
18 console.log(parsedObject.title)
```

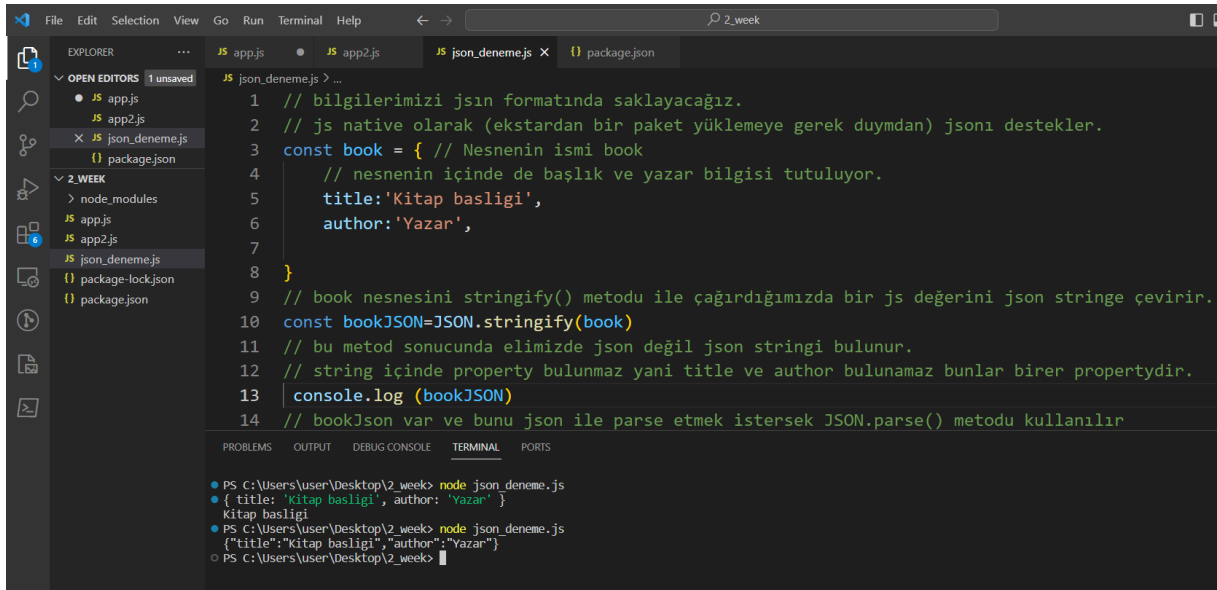
PS C:\Users\User\Desktop\2\_week> node json\_deneme.js

```
{ title: 'Kitap basligi', author: 'Yazar' }
```

Kitap basligi

PS C:\Users\User\Desktop\2\_week>

Ekrana yazdırma işleminin tam çalışabilmesi için on üçüncü satırı yorum satırı haline getirdim ve terminalde kodu çalıştırabilmek için node json\_deneme.js yazdık bu sayede ekran çıktısı şeklindeki gibidir. Ekran çıktısında süslü parantezler içinde verileri yazdırmış ve yeşil renkte yazdırmıştır tam bir json formatındadır ama birkaç bölüm öncesinde yani yeni yazdığımız kodları yorum satırına alalım ve on üçüncü satırı düzenleyelim ekran çıktısına bakalım.



```
1 // bilgilerimizi jsın formatında saklayacağız.
2 // js native olarak (ekstardan bir paket yüklemeye gerek duymdan) jsonı destekler.
3 const book = { // Nesnenin ismi book
4   // nesnenin içinde de başlık ve yazar bilgisi tutuluyor.
5   title:'Kitap basligi',
6   author:'Yazar',
7 }
8
9 // book nesnesini stringify() metodu ile çağırdığımızda bir js değerini json stringe çevirir.
10 const bookJSON=JSON.stringify(book)
11 // bu metod sonucunda elimizde json değil json stringi bulunur.
12 // string içinde property bulunmaz yani title ve author bulunamaz bunlar birer propertydir.
13 console.log (bookJSON)
14 // bookJSON var ve bunu json ile parse etmek istersek JSON.parse() metodu kullanılır
```

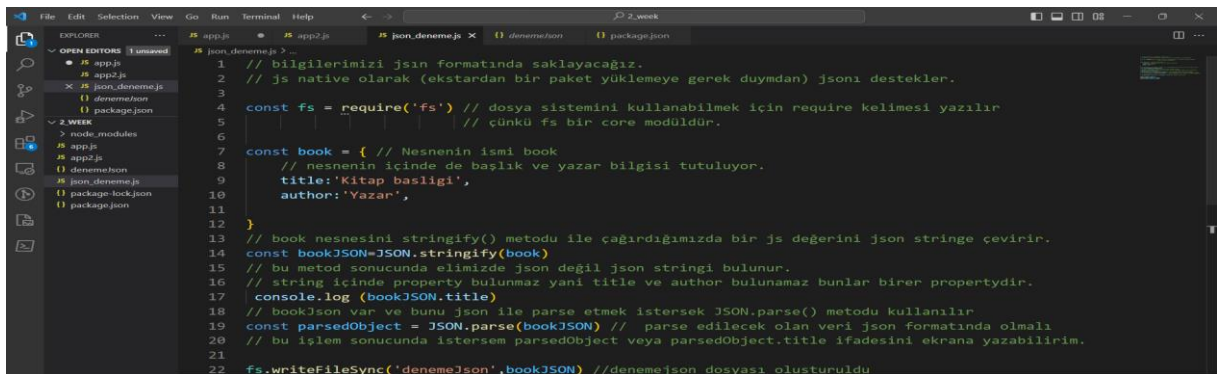
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS C:\Users\user\Desktop\2\_week> node json\_deneme.js
- { title: 'Kitap basligi', author: 'Yazar' }
- Kitap basligi
- PS C:\Users\user\Desktop\2\_week> node json\_deneme.js
- {"title":"Kitap basligi","author":"Yazar"}
- PS C:\Users\user\Desktop\2\_week>

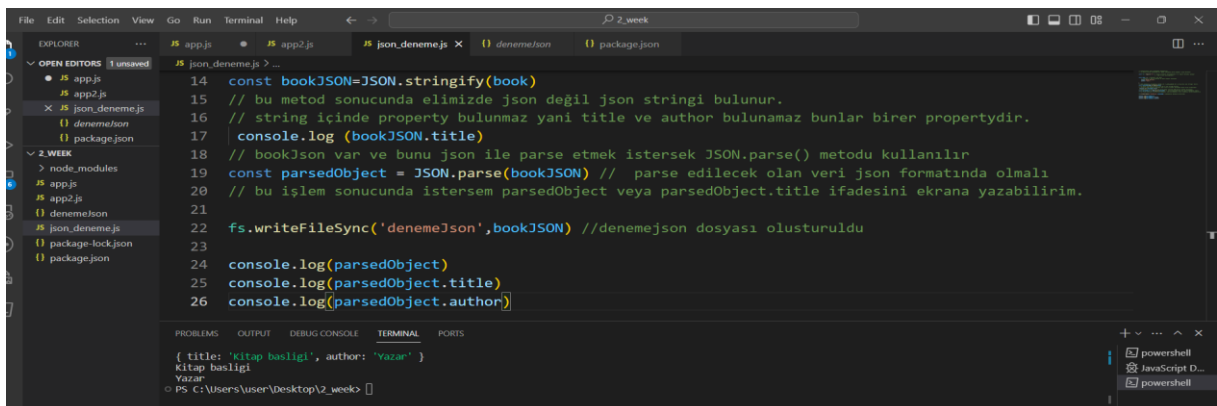
Burada ekran çıktısı yine süslü parantezler içinde fakat renkleri beyaz yani tam olarak bir json formatında değil. Json stringi halindedir.

Bu bilgileri dosya sistemine kaydedelim. Bazı kodları ekleyelim.

const fs = require('fs') ve fs.writeFileSync('denemeJson',bookJSON)



```
1 // bilgilerimizi jsın formatında saklayacağız.
2 // js native olarak (ekstardan bir paket yüklemeye gerek duymdan) jsonı destekler.
3
4 const fs = require('fs') // dosya sistemini kullanabilmek için require kelimesi yazılır
5 // çünkü fs bir core modüldür.
6
7 const book = { // Nesnenin ismi book
8   // nesnenin içinde de başlık ve yazar bilgisi tutuluyor.
9   title:'Kitap basligi',
10  author:'Yazar',
11 }
12
13 // book nesnesini stringify() metodu ile çağırdığımızda bir js değerini json stringe çevirir.
14 const bookJSON=JSON.stringify(book)
15 // bu metod sonucunda elimizde json değil json stringi bulunur.
16 // string içinde property bulunmaz yani title ve author bulunamaz bunlar birer propertydir.
17 console.log (bookJSON.title)
18 // bookJSON var ve bunu json ile parse etmek istersek JSON.parse() metodu kullanılır
19 const parsedObject = JSON.parse(bookJSON) // parse edilecek olan veri json formatında olmalı
20 // bu işlem sonucunda istersem parsedObject veya parsedObject.title ifadesini ekrana yazabilirim.
21 fs.writeFileSync('denemeJson',bookJSON) //denemejson dosyası olusturuldu
22
```



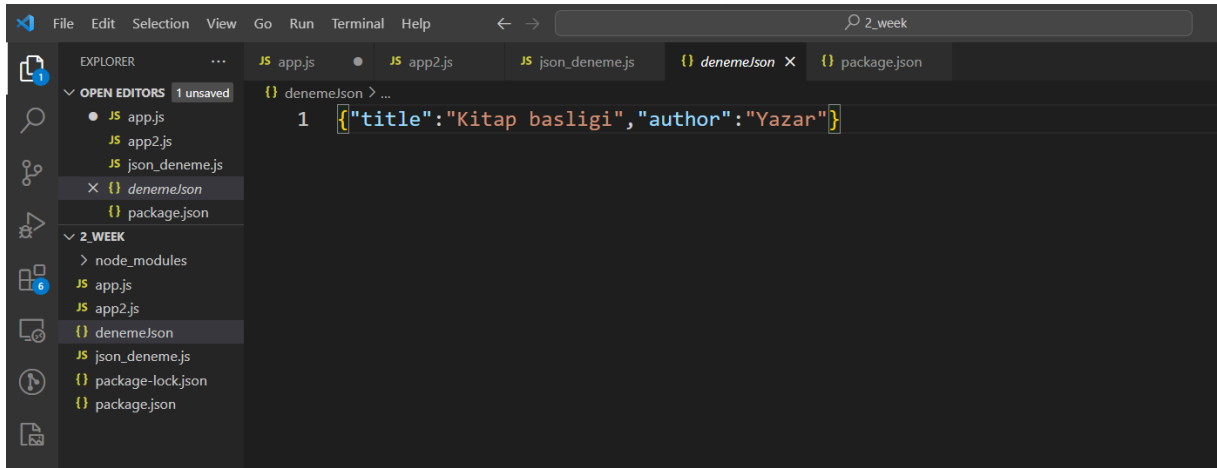
```
14 const bookJSON=JSON.stringify(book)
15 // bu metod sonucunda elimizde json değil json stringi bulunur.
16 // string içinde property bulunmaz yani title ve author bulunamaz bunlar birer propertydir.
17 console.log (bookJSON.title)
18 // bookJSON var ve bunu json ile parse etmek istersek JSON.parse() metodu kullanılır
19 const parsedObject = JSON.parse(bookJSON) // parse edilecek olan veri json formatında olmalı
20 // bu işlem sonucunda istersem parsedObject veya parsedObject.title ifadesini ekrana yazabilirim.
21 fs.writeFileSync('denemeJson',bookJSON) //denemejson dosyası olusturuldu
22
23 console.log(parsedObject)
24 console.log(parsedObject.title)
25 console.log(parsedObject.author)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
{ title: 'Kitap basligi', author: 'Yazar' }
Kitap basligi
Yazar
```

PS C:\Users\user\Desktop\2\_week>

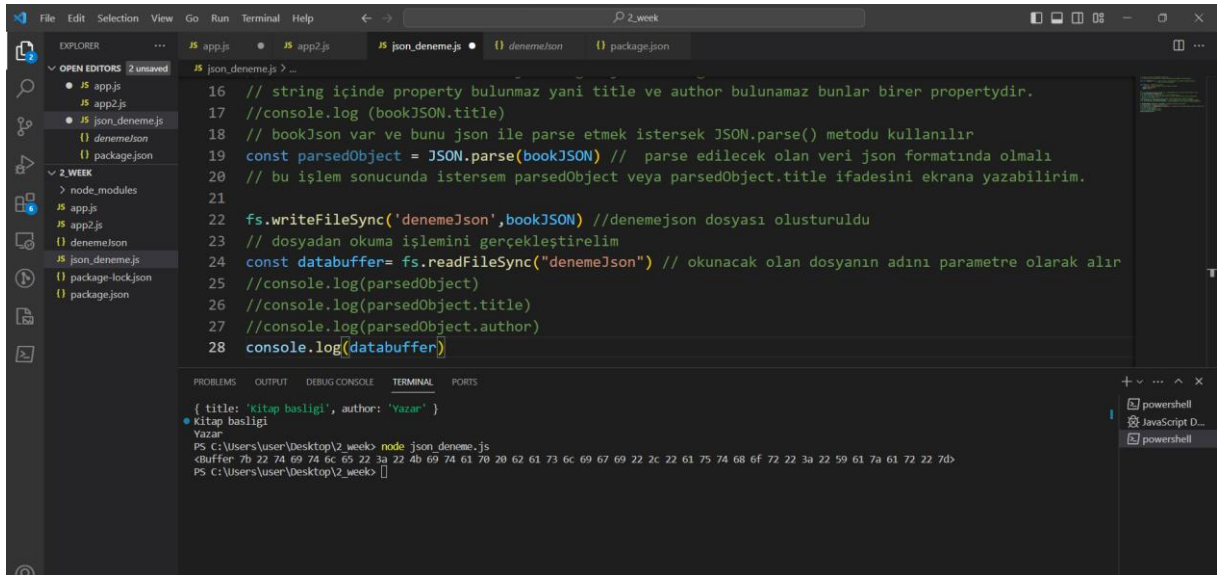
Bu işlem sonucunda sol tarafta bir denemeJson dosyası oluşturur ve içeriği şu şekildedir.



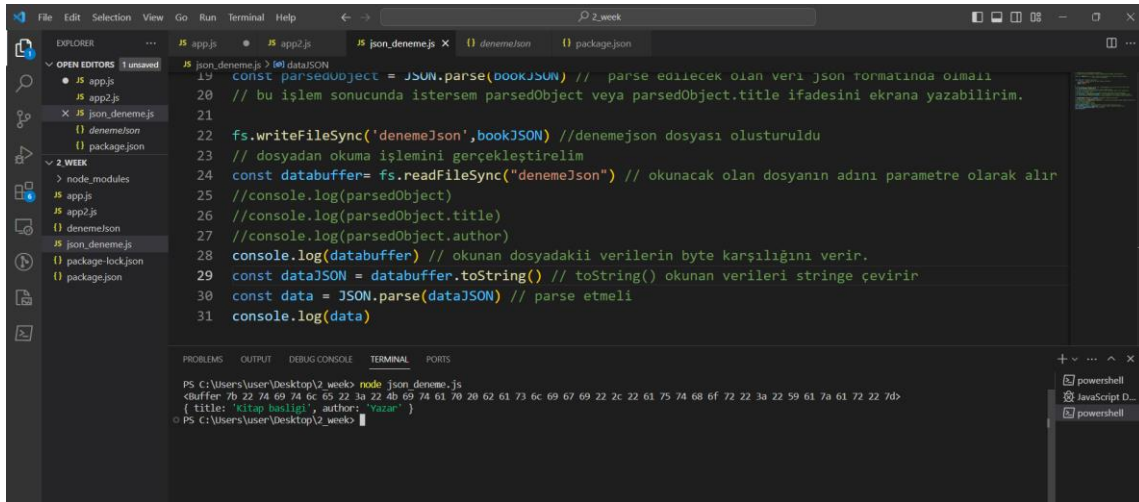
Az önce dosyaya yazma işlemini yaptık bu bölümde ise dosyada okuma işlemini yapalım.

`const databuffer= fs.readFileSync(\"denemeJson\")` ve `console.log(databuffer)`

Kodlarını ekleyelim



Çıktısında dosyadaki bulunan verileri okur ama byte olarak karşılığını verir bizim bu ifadeleri stringe çevirmemiz gerekli. `toString()` metodu ile çevirme işlemi gerçekleşir ve daha sonra bu veriyi parse etmemiz lazım.



```
19 const parsedObject = JSON.parse(bookJSON) // parse edilecek olan veri json formatında olmalı
20 // bu işlem sonucunda istersem parsedObject veya parsedObject.title ifadesini ekrana yazabilirim.
21
22 fs.writeFileSync('denemeJson', bookJSON) //denemejson dosyası oluşturuldu
23 // dosyadan okuma işlemini gerçekleştirelim
24 const databuffer= fs.readFileSync("denemeJson") // okunacak olan dosyanın adını parametre olarak alır
25 //console.log(parsedObject)
26 //console.log(parsedObject.title)
27 //console.log(parsedObject.author)
28 console.log(databuffer) // okunan dosyadaki verilerin byte karşılığını verir.
29 const dataJSON = databuffer.toString() // toString() okunan verileri stringe çevirir
30 const data = JSON.parse(dataJSON) // parse etmeli
31 console.log(data)
```

PS C:\Users\User\Desktop\2\_week> node json\_deneme.js

<Buffer 7b 22 74 69 74 6c 65 22 3a 22 4b 69 74 61 70 20 62 61 73 6c 69 67 69 22 2c 22 61 75 74 68 6f 72 22 3a 22 59 61 7a 61 72 22 7d>

{ title: 'Kitap Hasiği', authors: 'Yazar' }

Kısacası elimizdeki java nesnesini stringify ile json formatında bir stringe çevirip dosyaya yazıp daha sonra dosyadan okuma işlemi gerçekleştirildi.