

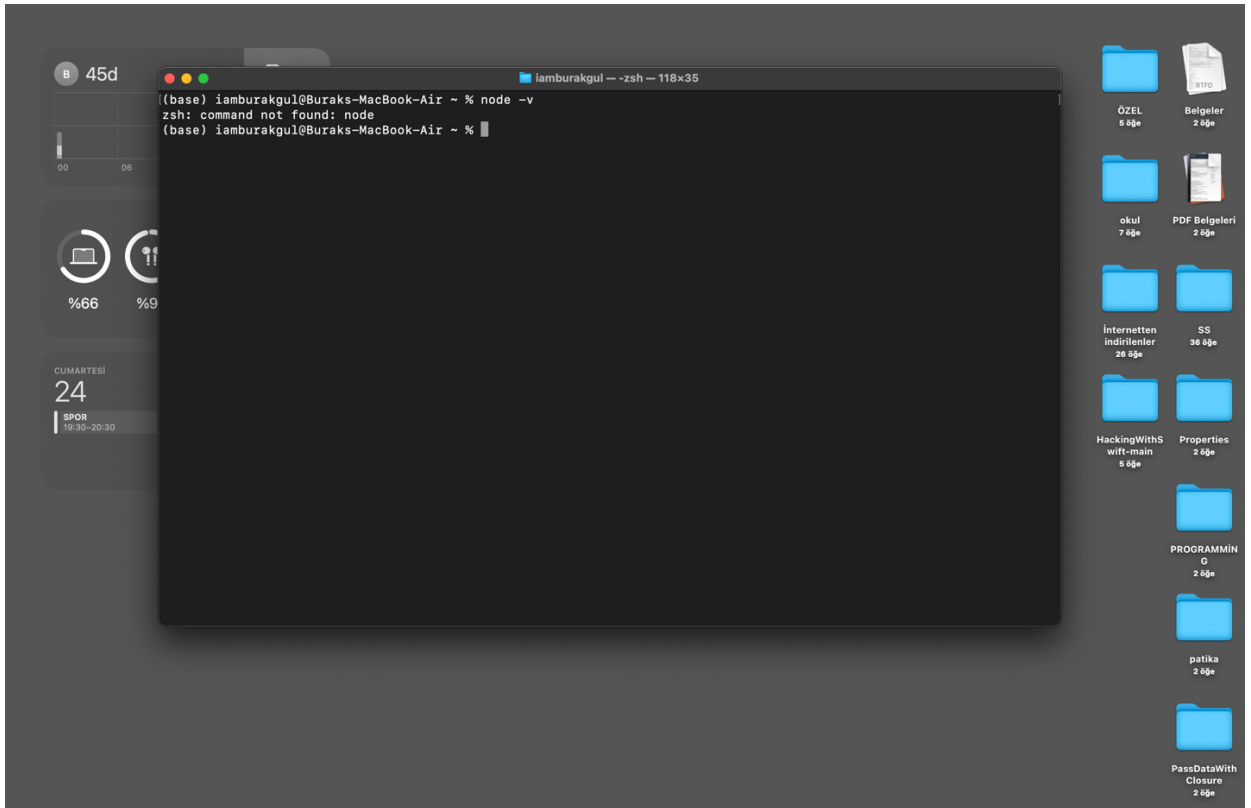
# Node.js Section 1

Merhaba bugün Node.js nasıl indirilir onunla başlayarak neler yapabileceğimize bakacağız.

Node.js Client-Server mimarisinde Server kısmını ayarlamak ,yazmak için kullanacağımız teknolojidir.

Ben macOS işletim sistemine sahip bir cihaz kullandığım için bunun üzerinden anlatacağım.

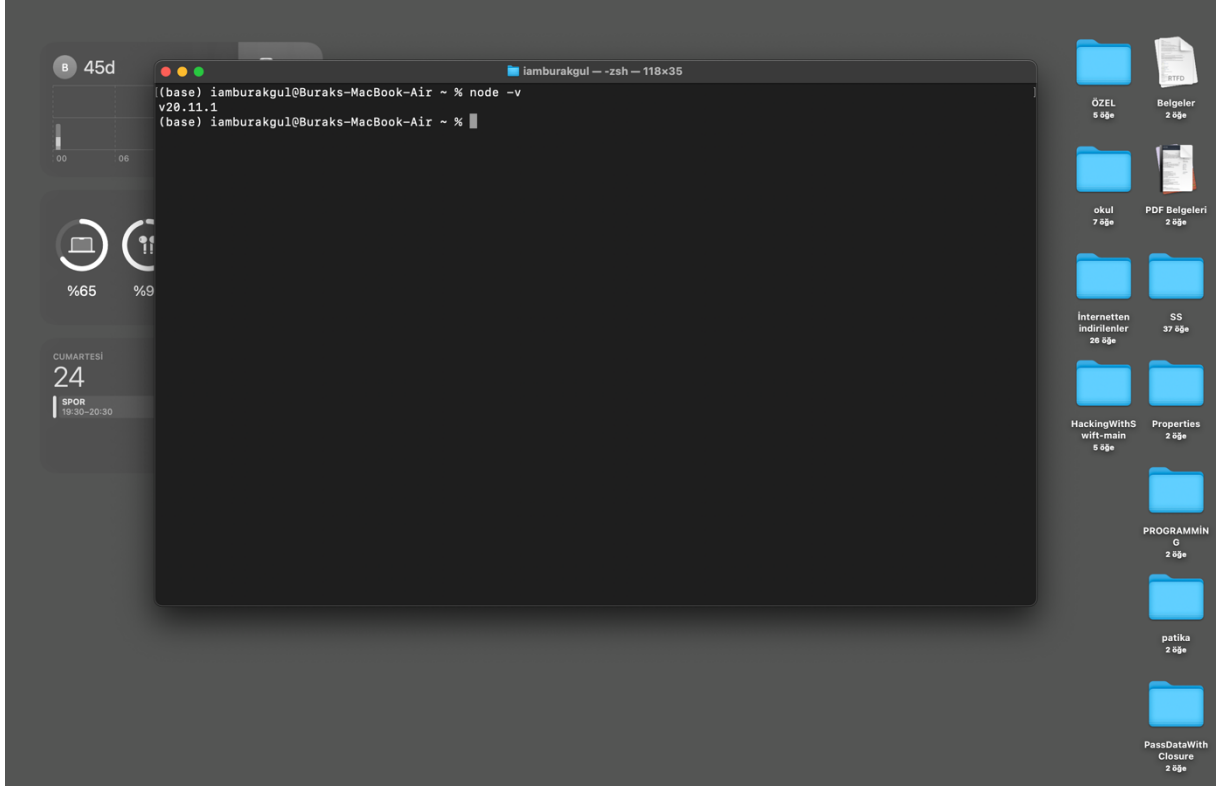
Node.js sistemimizde var mı yok mu diye Terminal i açıyoruz ve “node -v” yazıyoruz.



node ' u bulamadı şimdi indirmek için

<https://nodejs.org/en/download> bu siteye gidiyoruz ve LTS (Long Term Support ) olan kısımdan indiriyoruz.

Kurulumları yaptıktan sonra tekrardan Terminalden “node -v”yaparak indiğini kontrol ediyoruz.



Şimdi Node.js indirildi.

## Peki Node.js nedir ?

JavaScript yalnızca tarayıcıda çalışır ve kendi makinemizde bağımsız bir işlem(process) olarak çalışmaktadır.

Tarayıcının yapmasına izin verdiği şeyler sınırlıdır , daha çok genel amaçlı bir programlama diline benzer.

Node.js, bir programlama dili değildir; JavaScript'e özel fonksiyonlar ve nesneler ekleyerek özel işlevler sunar. Chrome ve Node.js, JavaScript kodunu V8'e gönderir ve sonuçları alır. Bu süreçte, her ikisi de JavaScript dilini özelleştirilmiş fonksiyonlar ve nesnelerle genişletir.

Node.js, JavaScript'in tarayıcı sınırlamalarının ötesine geçmesini sağlar. Normalde JavaScript yalnızca tarayıcı içinde çalışırken, Node.js sayesinde JavaScript kodunu sunucu tarafında çalıştırabiliriz. Bu, JavaScript'in dosya sistemine erişim, veritabanlarıyla bağlantı kurma gibi genel amaçlı programlama dillerinin yapabildiği işlemleri yapabilmesini sağlar.

Node.js, JavaScript'in işlevselliğini C++ bağlantıları sağlayarak genişletir. Bu, JavaScript ile C++'ın yapabildiği her şeyi, örneğin dosya sistemine erişim gibi işlemleri yapabilmesini sağlar.

Tarayıcı ortamında (örneğin Chrome'da) window ve document gibi global nesneler kullanılırken, Node.js ortamında global ve process gibi global nesneler kullanılır. Her ikisi de JavaScript kodunu çalıştırabilir; örneğin, basit bir matematik işlemi yapmak (2+3) veya bir metin üzerinde işlem yapmak ('Can'.toUpperCase()) gibi.

Sonu olarak, Node.js, JavaScript'i sadece tarayıcıda alıřan bir dil olmaktan ıkarıp, dosya sistemine erişim veya veritabanıyla etkileřim gibi işlemleri yapabilen genel amaçlı bir programlama diline dönüřtürür. Bu sayede, JavaScript kodunu sunucu tarafında alıřtırarak web uygulamaları geliřtirmemize olanak tanır.

## **Peki Neden Node.js kullanmalıyız ?**

Node.js, event driven bir non-blocking (engellemeyen) I/O modeli kullanır, bu da onu hafif ve verimli kılar. I/O, giriş/ıkış anlamına gelir ve uygulamamız, alıřtığı makineyle (dosya sistemi gibi) veya uzak bir sunucuyla iletişim kurmak için I/O kullanır.

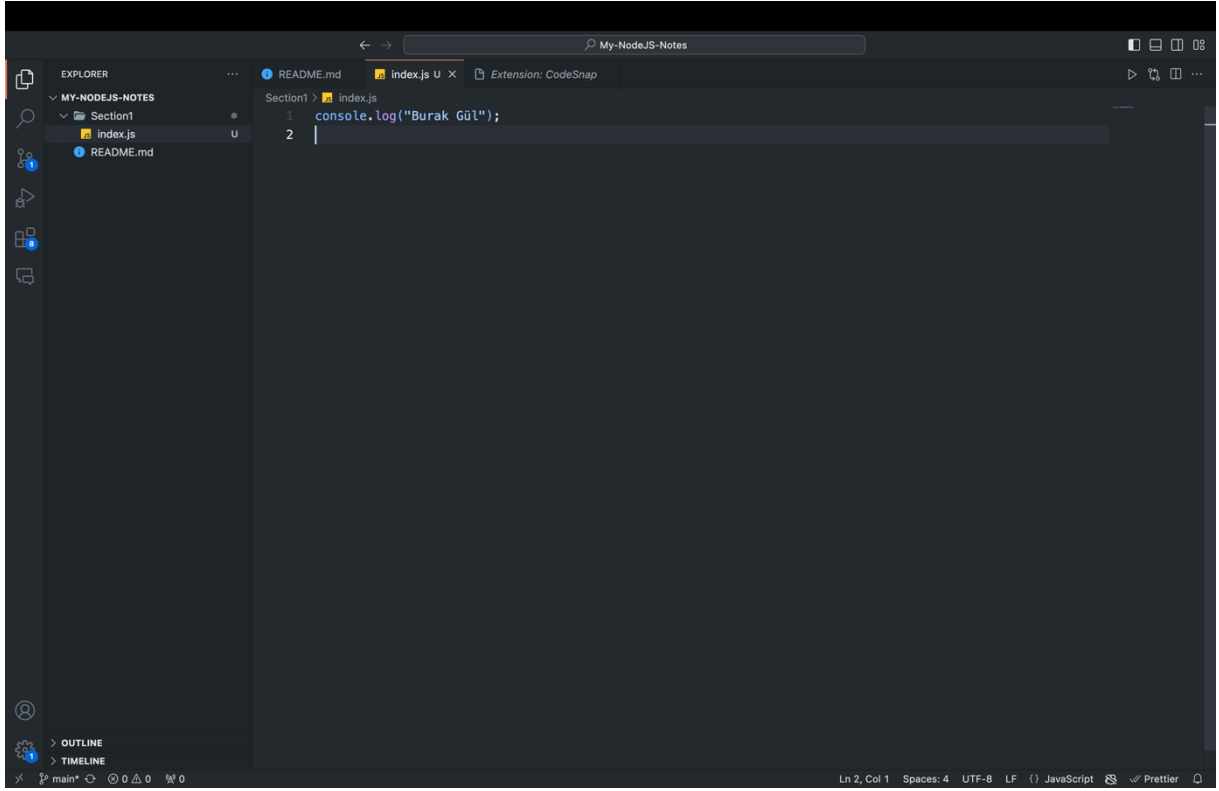
"Non-blocking" (engellemeyen) terimi, uygulamanın bir response beklerken başka işlemler yapabilmesini (diğer kodları işleme, başka isteklerde bulunma vs.) ifade eder. Bu, uygulamanın veri beklerken kullanıcı deneyiminin kötü olmasını önler; örneğin, veriler alınırken kullanıcı bağlantıları veya butonlara tıklayamazsa, bu kötü bir deneyim olurdu. Bu tür operasyonlar, hem Node.js hem de tarayıcıda arka planda alıřır.

"Etkinlik tabanlı" (event driven) olmak, geri ağırma fonksiyonlarının (callbacks) kaydedilmesi ve I/O işlemi tamamlandığında bu fonksiyonların ağırılmasını ifade eder. Bu model, Node.js'in verimli alıřmasını sağlar ünkü uygulama, bir işlemi tamamlamak için beklemek yerine, diğer işlemlere devam edebilir.

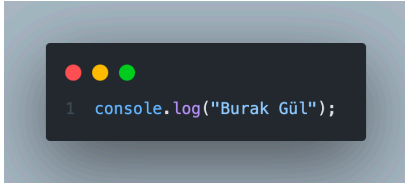
Özetle, Node.js'in bu özellikleri, özellikle I/O ağırlıklı uygulamalar için ideal bir platform olmasını sağlar. Etkinlik tabanlı ve non-blocking modeli sayesinde, aynı anda ok sayıda bağlantıyı veya işlemi verimli bir şekilde yönetebilir, bu da onu gerek zamanlı uygulamalar, web sunucuları ve mikro hizmet mimarileri için popüler bir seenek yapar.

řimdi bunları alıřtırabilmek için bir file oluřturalım ve içinde bir JavaScript kodu yazalım.

Ben řimdi bir tane index.js adında file oluřturuyorum



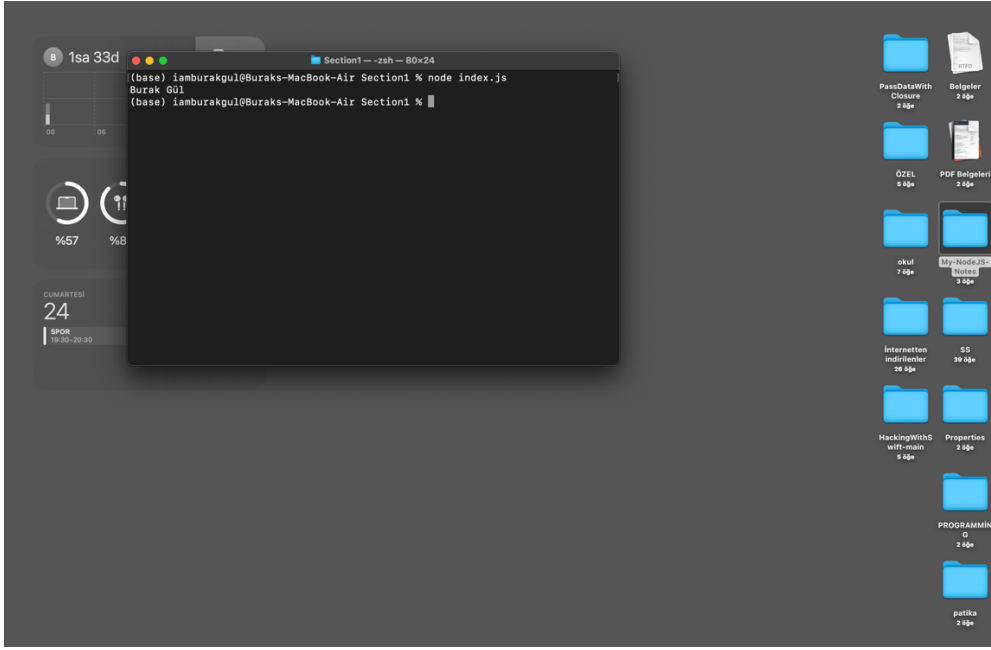
ve içine



Kodunu yazıyorum.

Şimdi bu index.js file 'ın olduğu folder a gidiyorum . (cd pathOfFolder)

Ve içinde node index.js yazdığımda bana index.js file ını çalıştıracaktır.

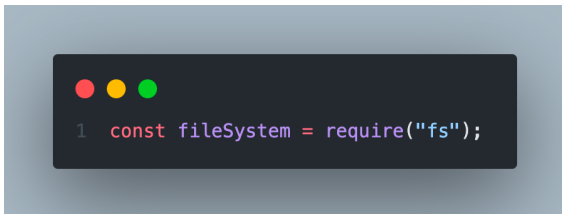


Terminal işlemlerine VS Code içinden de devam edebiliriz oradan devam edeceğim daha rahat olması açısından.

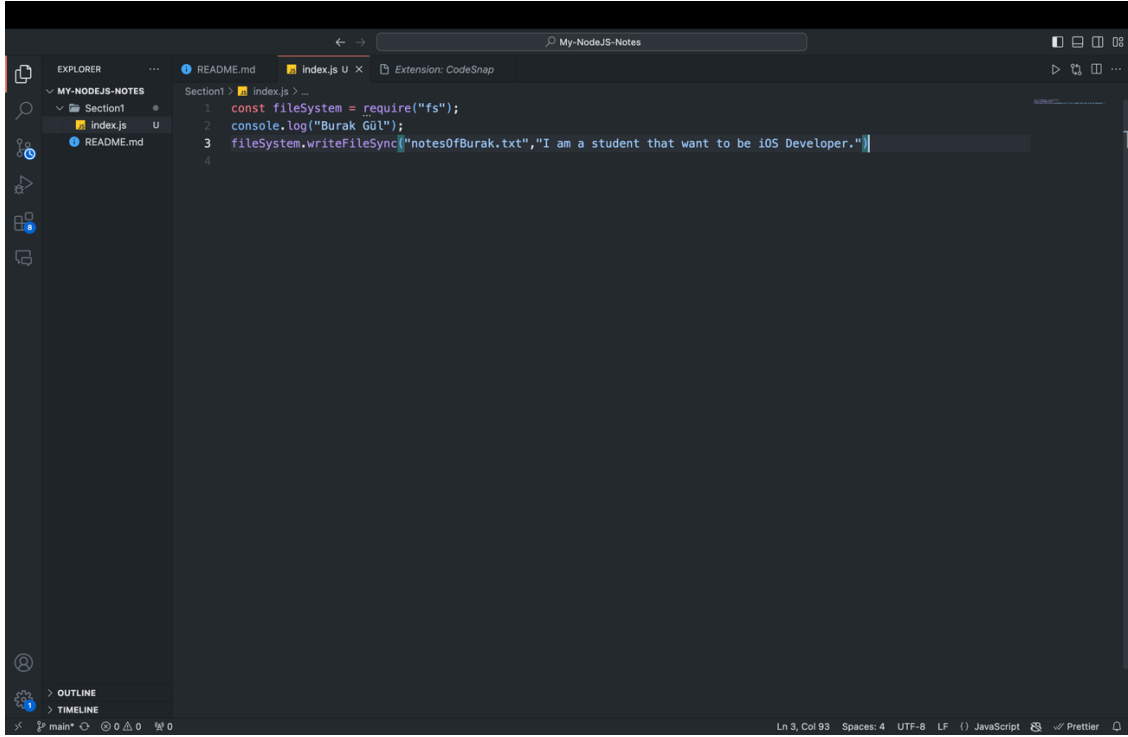
Şimdi biz bu şekilde yaparak Node.js e şu dosyayı çalıştır demiş olduk.

Peki diyelim ki ben modülleri kullanarak bir şeyler yapmak istiyorum. O zaman modülleri file içinde import etmem gerek.

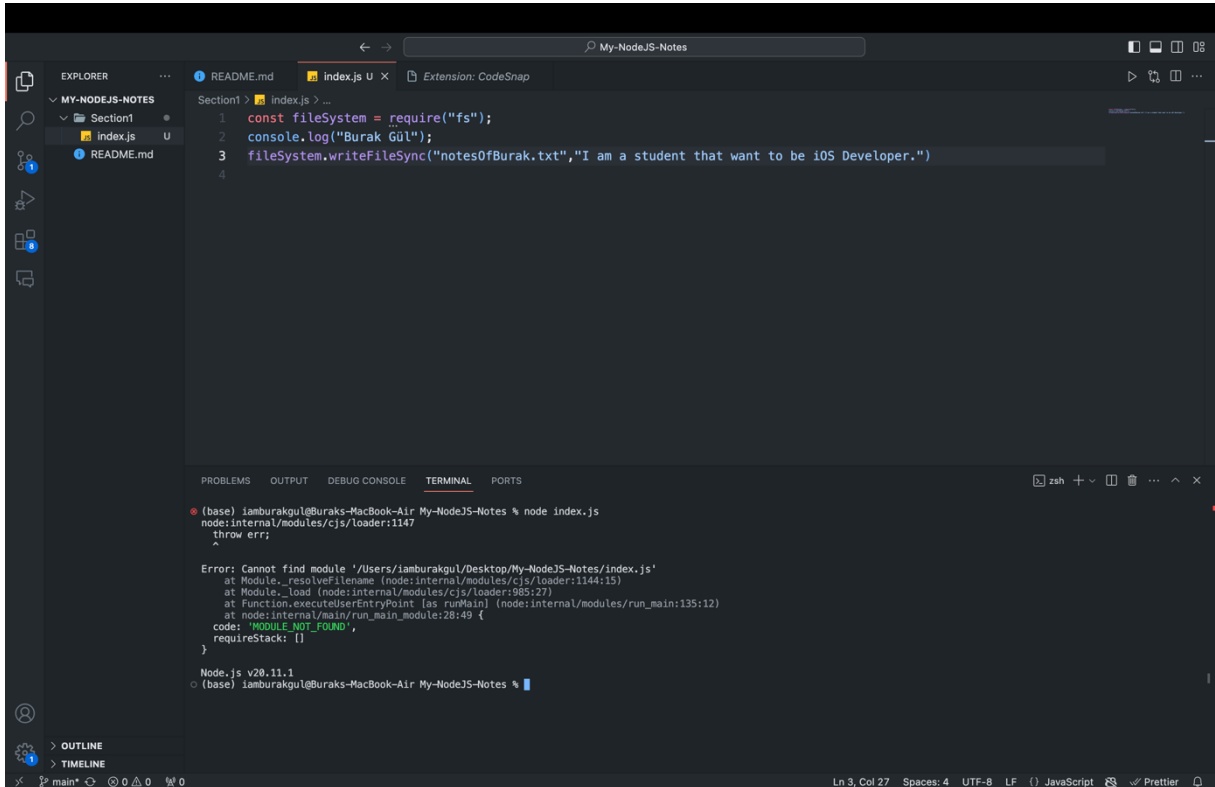
File System modülünü import edelim şu kod ile



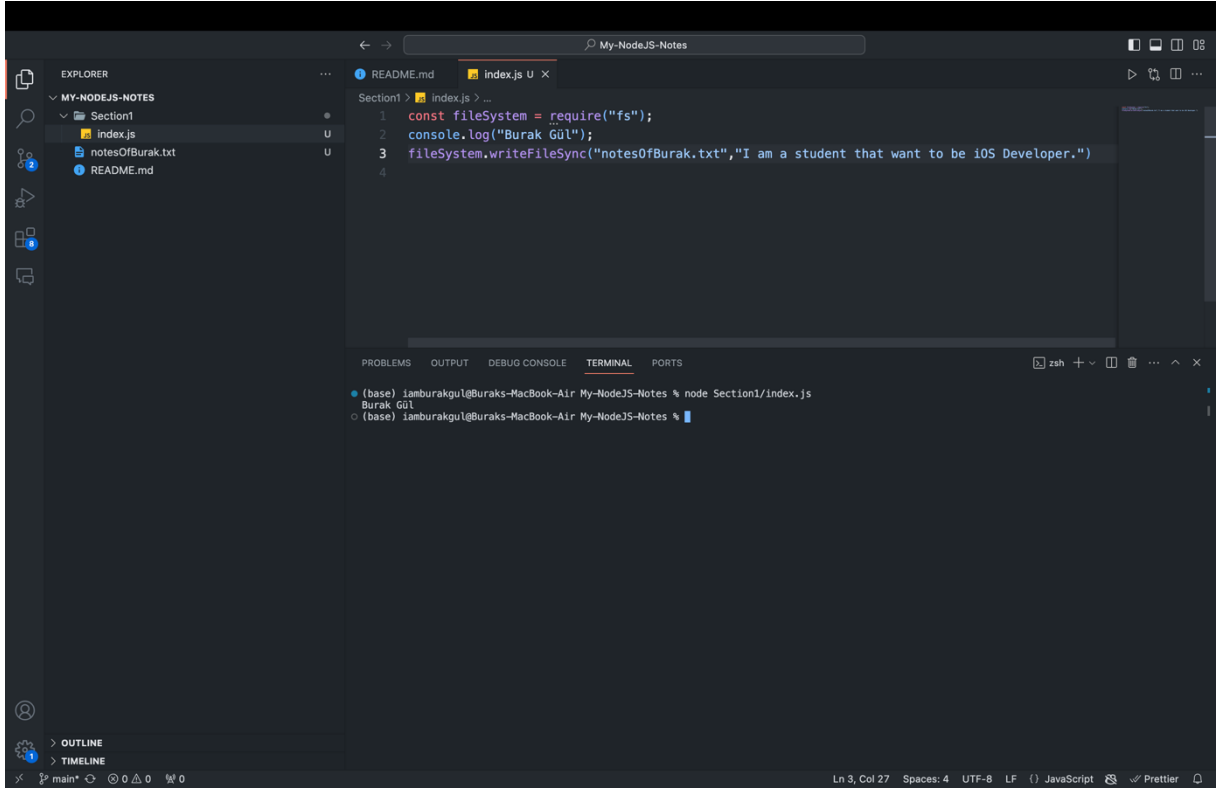
Buradaki “fileSystem” benim oluşturduğum bir değişken adı aslında bu değişken adını kullanarak File System modülü ile yapabildiğim şeyleri yapmaya çalışacağım.



Şimdi bu index.js dosyasını çalıştıralım VS Code 'un terminalinden.



Hata aldık sebebi ise Section1 folder'ında olmam lazım ya da o yolda araması gerektiğini söylemem lazım.



A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project named 'MY-NODEJS-NOTES' with a folder 'Section1' containing 'index.js', 'notesOfBurak.txt', and 'README.md'. The main editor window displays 'index.js' with the following code:

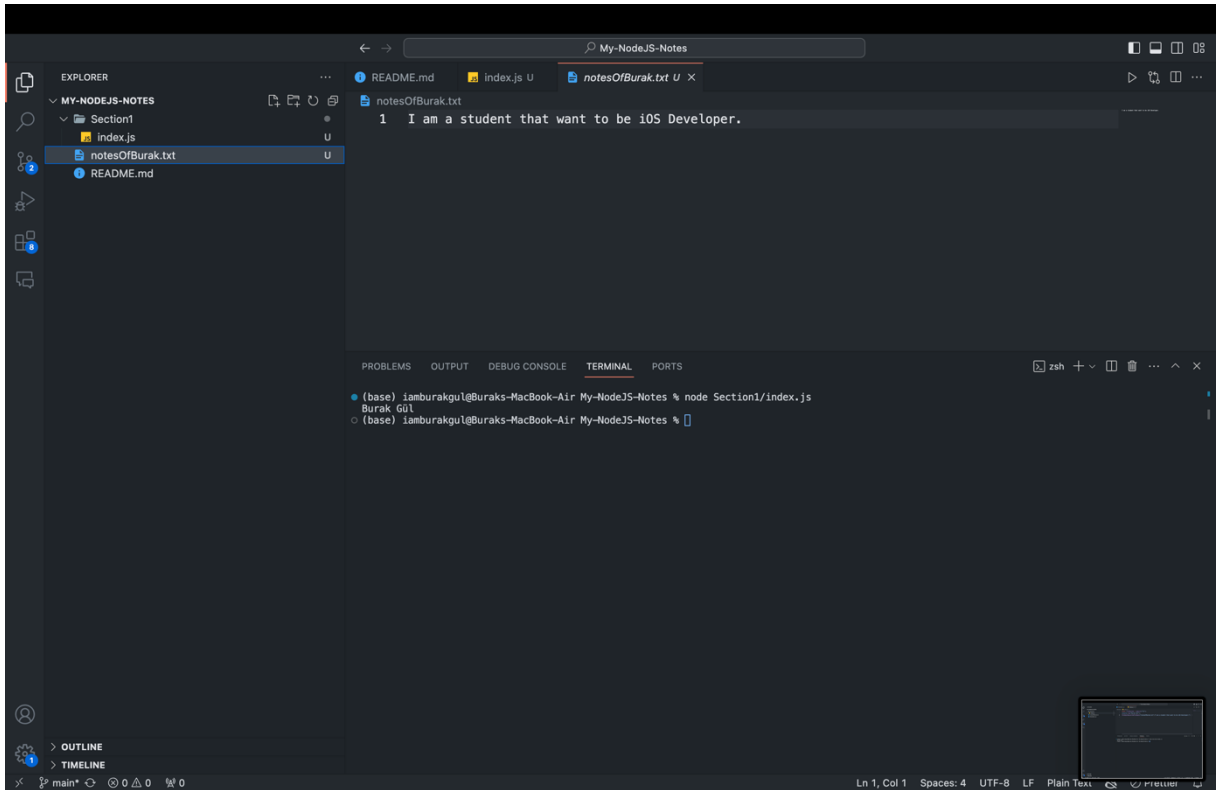
```
1 const fileSystem = require("fs");
2 console.log("Burak Gül");
3 fileSystem.writeFileSync("notesOfBurak.txt","I am a student that want to be iOS Developer.");
4
```

The bottom panel shows the TERMINAL with the following output:

```
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes % node Section1/index.js
Burak Gül
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes %
```

Önce Console.log ile stringi yazdırdı daha sonra fileSystem ile bir txt file ı oluşturduk.

Şimdi içine girip bakalım o txt nin.



A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows the same project structure. The main editor window displays 'notesOfBurak.txt' with the following content:

```
1 I am a student that want to be iOS Developer.
```

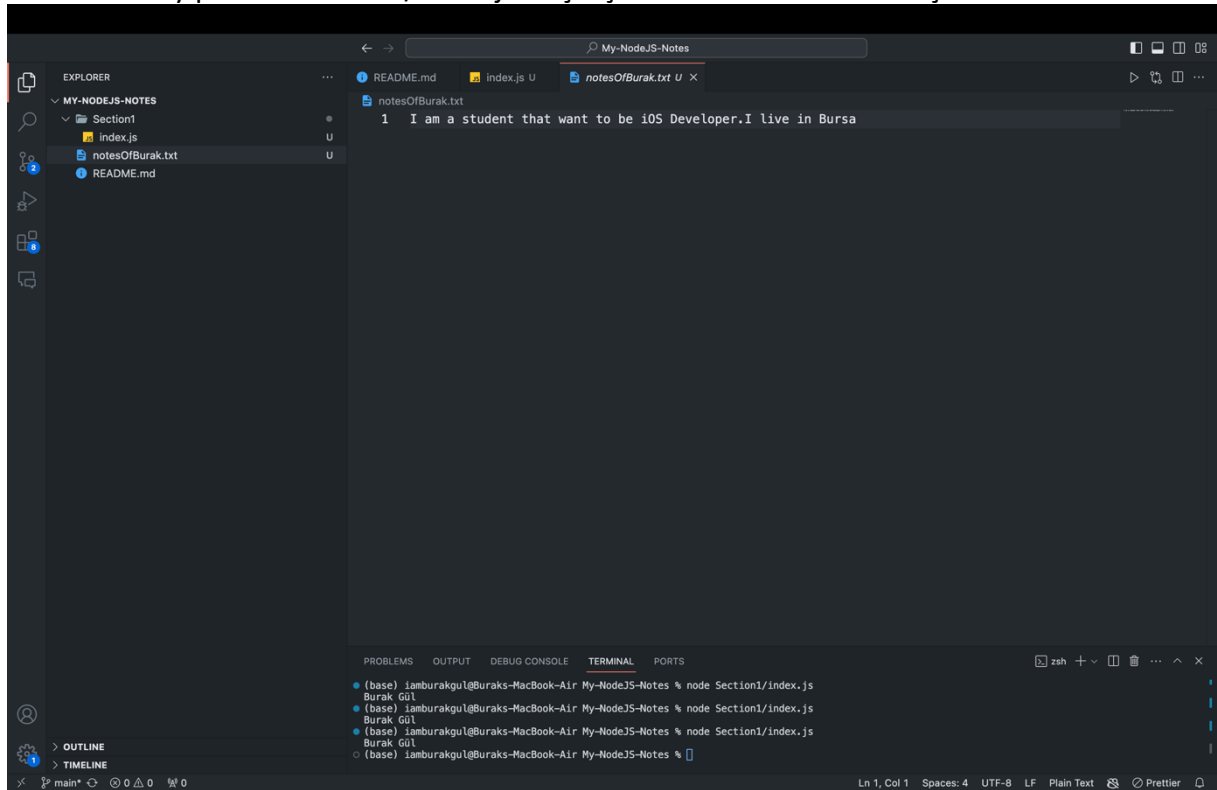
The bottom panel shows the TERMINAL with the same output as the previous screenshot:

```
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes % node Section1/index.js
Burak Gül
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes %
```

Txt içine ver eklemek için ise

```
1  fileSystem.appendFileSync("notesOfBurak.txt", "I live in Bursa");
```

Bu kodu ekleyip node Section1/index.js ile çalıştıralım tekrardan ve txt içine bakalım.



Verimizi de ekledik.

Projede büyüdükçe projeyi yönetebilmek açısından başka file oluşturalım.

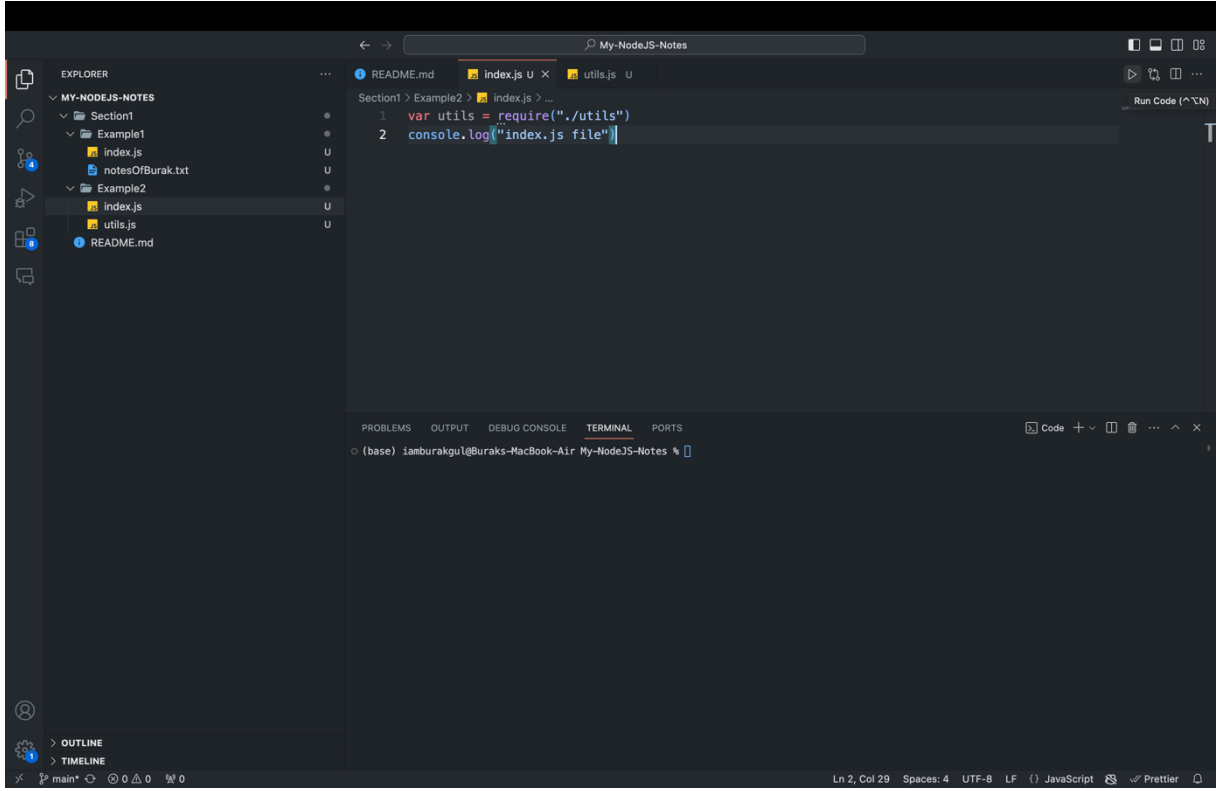
Şimdiye kadar olan kısmı example 1 diye klasörleyeceğim.

Şimdi example2 folderında utils.js ve index.js adında 2 file oluşturalım. utils.js file nın içinde

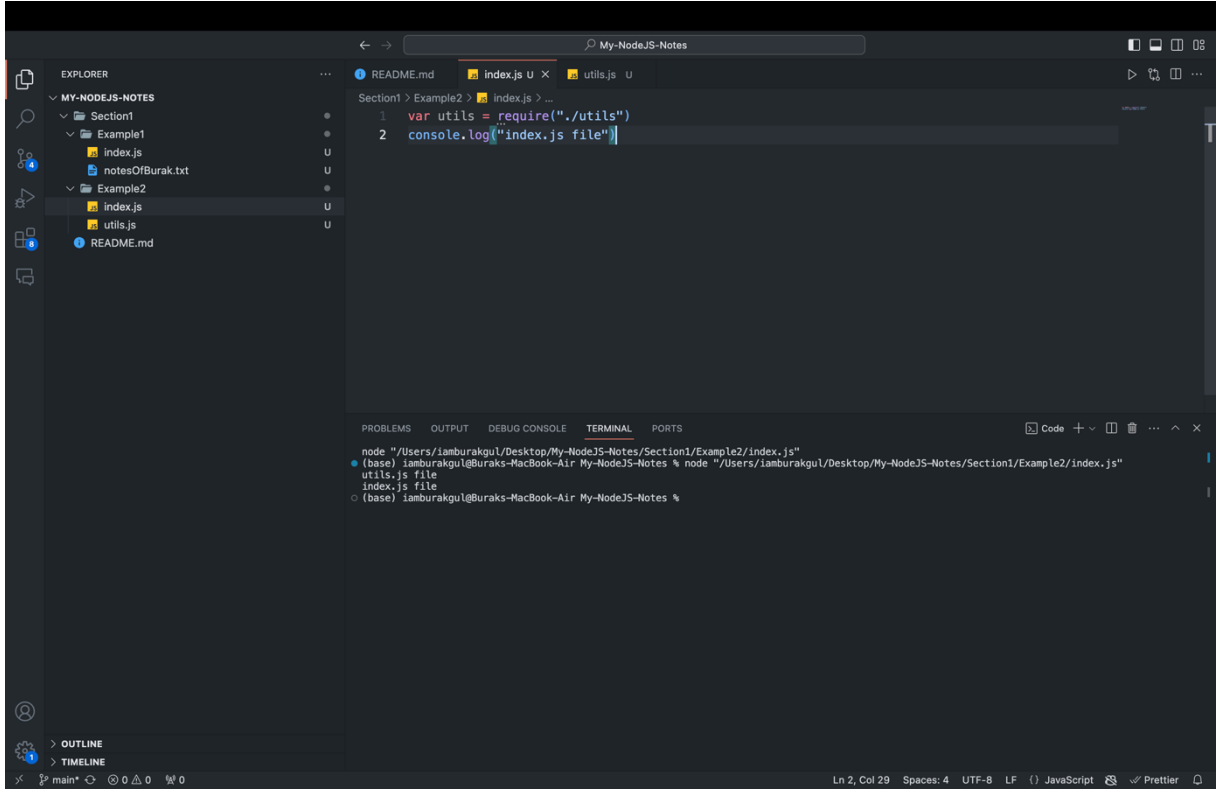
```
1  console.log("utils.js file");
```

Kodunu yazalım. Ve bu utils.js file nı index.js içinden çağırmaya çalışalım.





Buradaki run tuşundan da çalıştırabilirsiniz.  
Çalıştıralım ve bakalım.



Önce utils.js file ı çalıştı sonra console.log u çalıştırdı index.js miz.

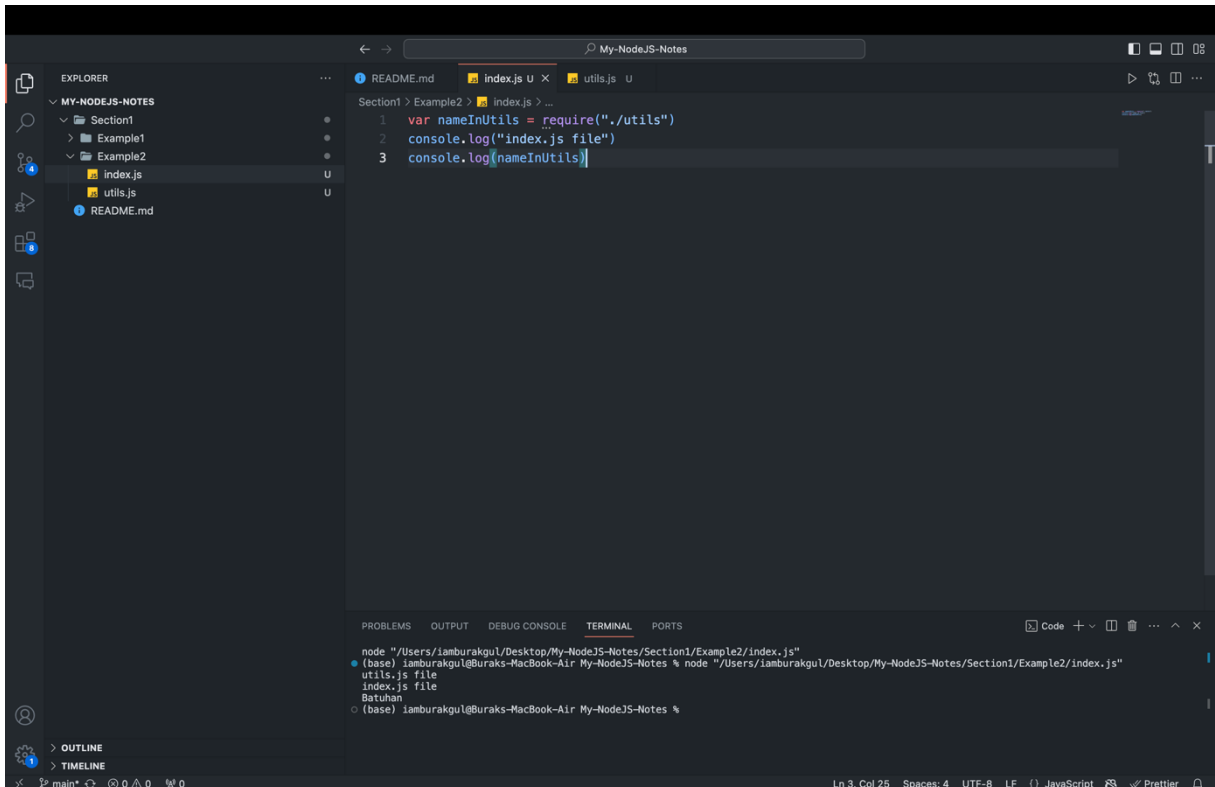
utils.js file içinde bir değişken tanımlayalım ve index.js içinden erişelim.

utils.js içi şöyle olacaktır.

Module.exports diyerek bir bu file i import eden yere gidecek veriyi demiş oluyorum aslında.



index.js içinden erişmeye çalışalım bu değişkene.

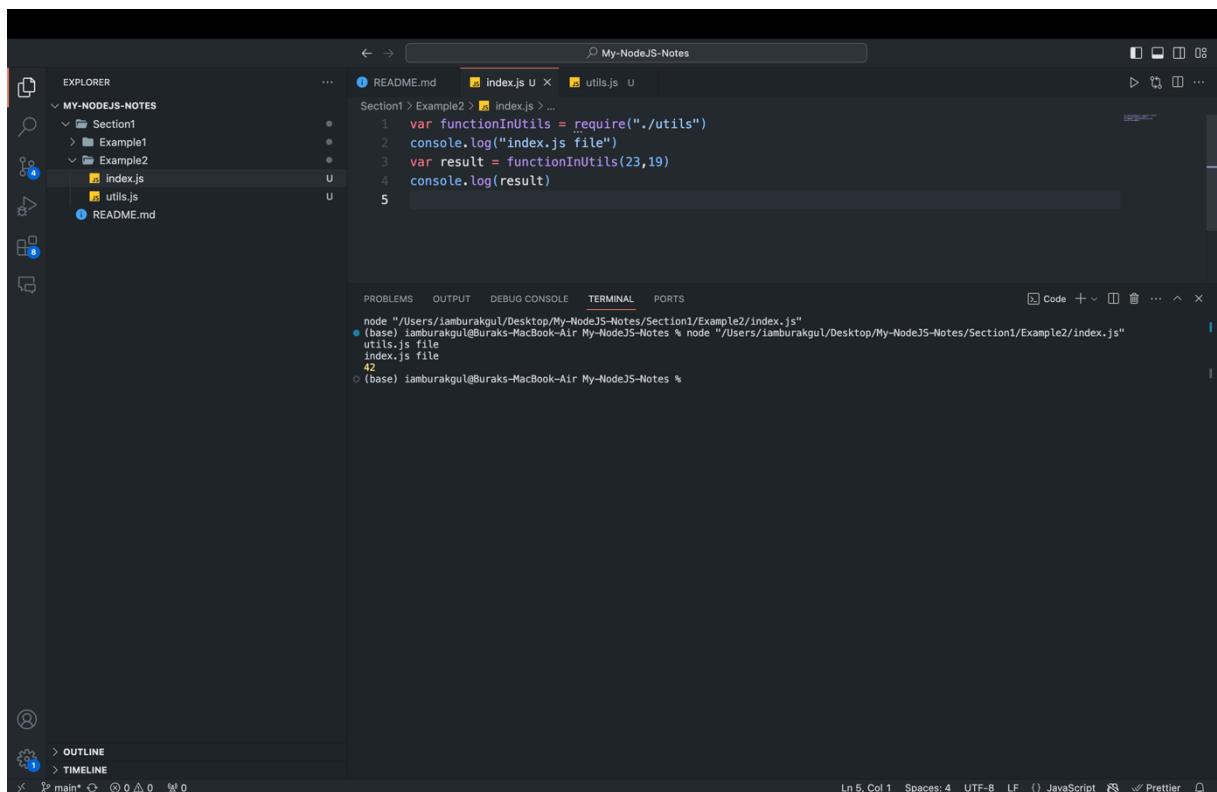


Gördüğümüz üzere utils.js file ındaki değişkene eriştik. Şimdi ise bir fonksiyon tanımlayalım ve ona erişelim.

utils.js içinde şöyle yaptık

```
1 console.log("utils.js file");
2 const name = "Batuhan";
3 function add(number1,number2) {
4     return number1 + number2
5 }
6 module.exports = add
```

add fonksiyonunu dışarı aktarıyoruz bu sefer.



Burada ise şimdi utils.js file ındaki fonksiyonu kullandık ve sonucu yazdırdık.

Şimdi npm leri ekleyelim modüllerimize.

# NPM Nedir ?

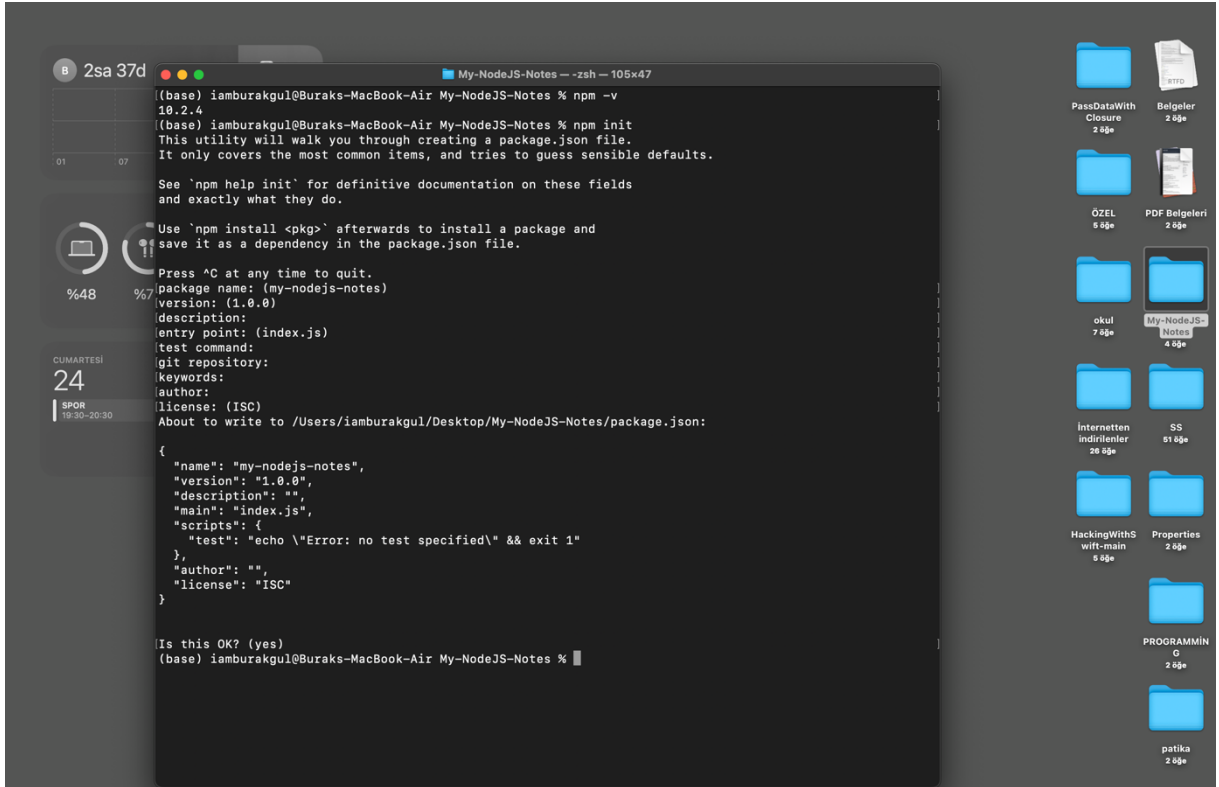
Node.js Package Manager . Kısaca olayın özeti şu ben bir şeyi baştan icat etmeyeceğim icat edilenleri kullanacağım.

SPM (Swift Package Manager ) ile aynı mantık gibi düşünebiliriz.

Mesela insanlar bir stringin url email gibi formatlara uyduğunu kontrol etmek için yazmışlar kodu ben de onları indirerek kullanacağım.

Root levelde oluşturmamız lazım npm i o yüzden terminal üzerinden yapmak istiyorum. Ve oluştururken default value'ları kabul edeceğim.

npm init diyerek npm oluşturuyoruz



```
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes % npm -v
10.2.4
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes % 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: (my-nodejs-notes)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /Users/iamburakgul/Desktop/My-NodeJS-Notes/package.json:

{
  "name": "my-nodejs-notes",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes)
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes %
```

Ve bize bir package.json file ı oluşturuyor. Bu dosya bizim nelere bağımlı olduğumuzu vs söyleyen bir file aslında.

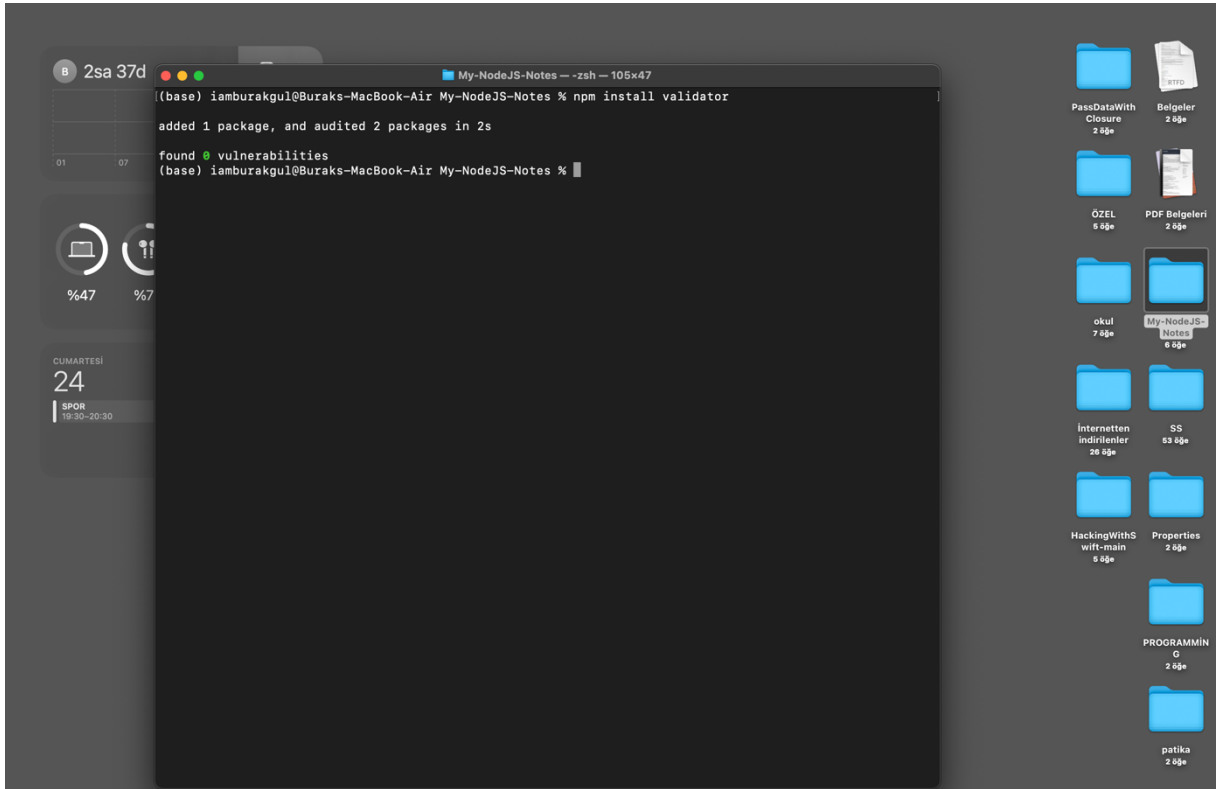
<https://www.npmjs.com> bu siteden ihtiyacımız olan şeyleri bulmakta kullanabiliriz.

Validator diye aratalım ve bakalım.

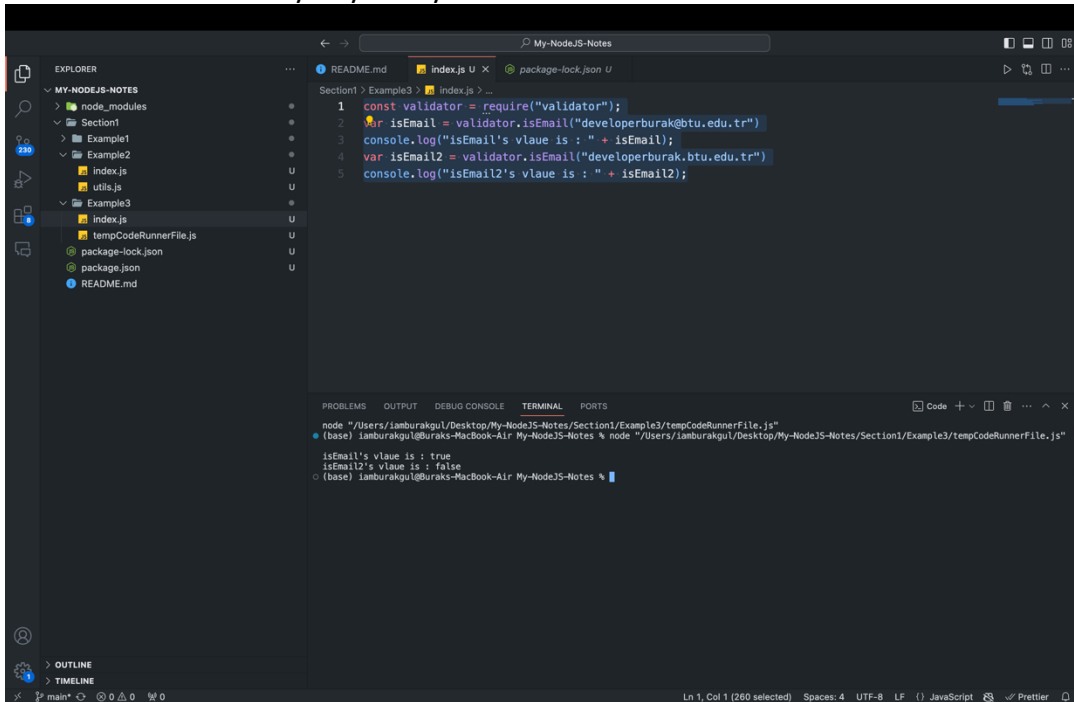
<https://www.npmjs.com/package/validator>

bunu kullanacağım. Okuyarak nasıl indireceğimizi ve kullanacağımızı anlayabiliriz.

Npm install validator diyerek indirdik



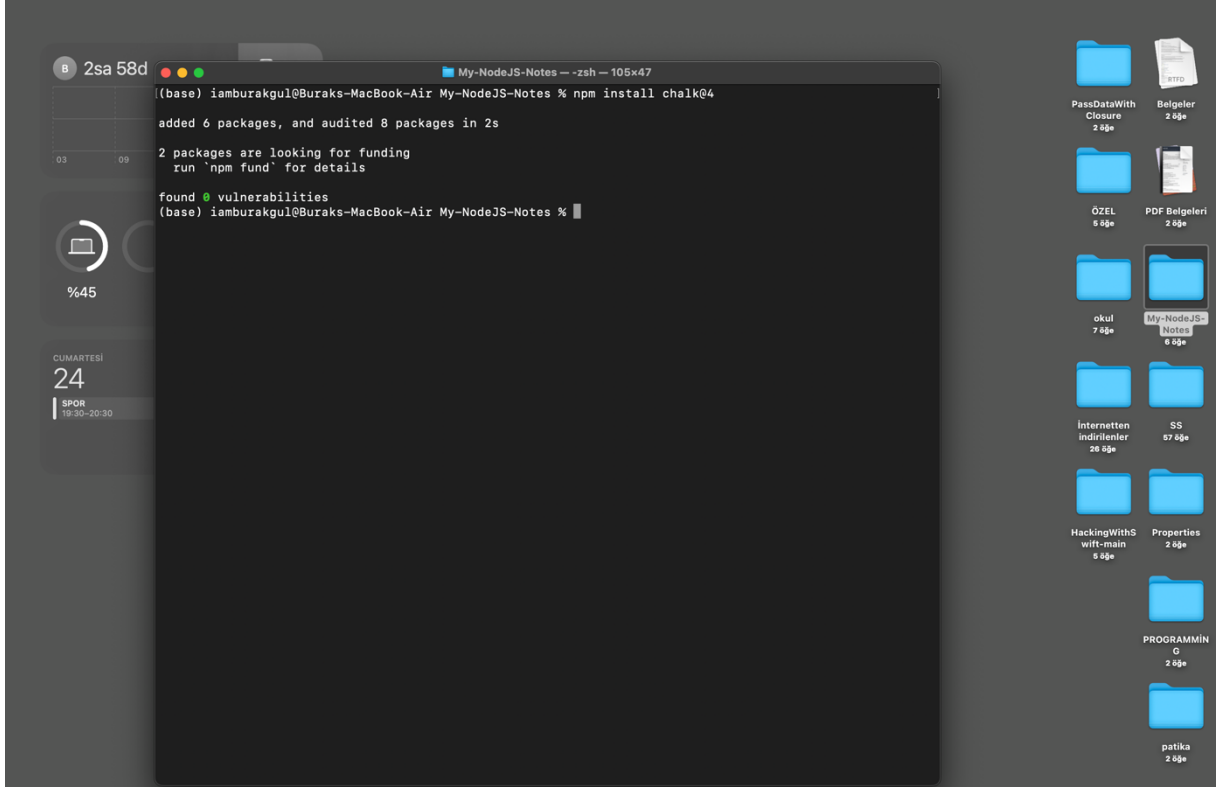
Bu örnekleri de example3 adlı klasörde toplayacağım.  
Ve index.js adlı file oluşturalım ve içindeki kodları çalıştıralım.  
Ve göreceğimiz gibi email mi değil mi sonuçları gösteriyor.  
Bir daha tek tek fonksiyon yazmaya zaman harcamadık.



Burada olan başka fonksiyonlara da bakabiliriz Ama uğraşmayacağım bunun için.

Şimdi başka bir npm indirelim. Console da yazıları renkli görmek için chalk indirelim.

<https://www.npmjs.com/package/chalk>



Şimdi projemizde dosyamıza require ile ekleyelim ve çalıştıralım.

```
1 const chalk = require("chalk");
2 console.log(chalk.green.bold("Burak"));
3 console.log(chalk.bold.inverse.red("Success!"));
4
```

```
node "/Users/iamburakgul/Desktop/My-NodeJS-Notes/Section1/example4/index.js"
(base) iamburakgul@Buraks-MacBook-Air My-NodeJS-Notes % node "/Users/iamburakgul/Desktop/My-NodeJS-Notes/Section1/example4/index.js"
Burak
Success!
```

Gördüğümüz üzere çalışıyor.

## Global ve Local NPM Modülleri

Local NPM modülleri projeye özgüdür, projenin “node\_modules” klasöründe saklanır ve projenin bağımlılıkları olarak yönetilir.

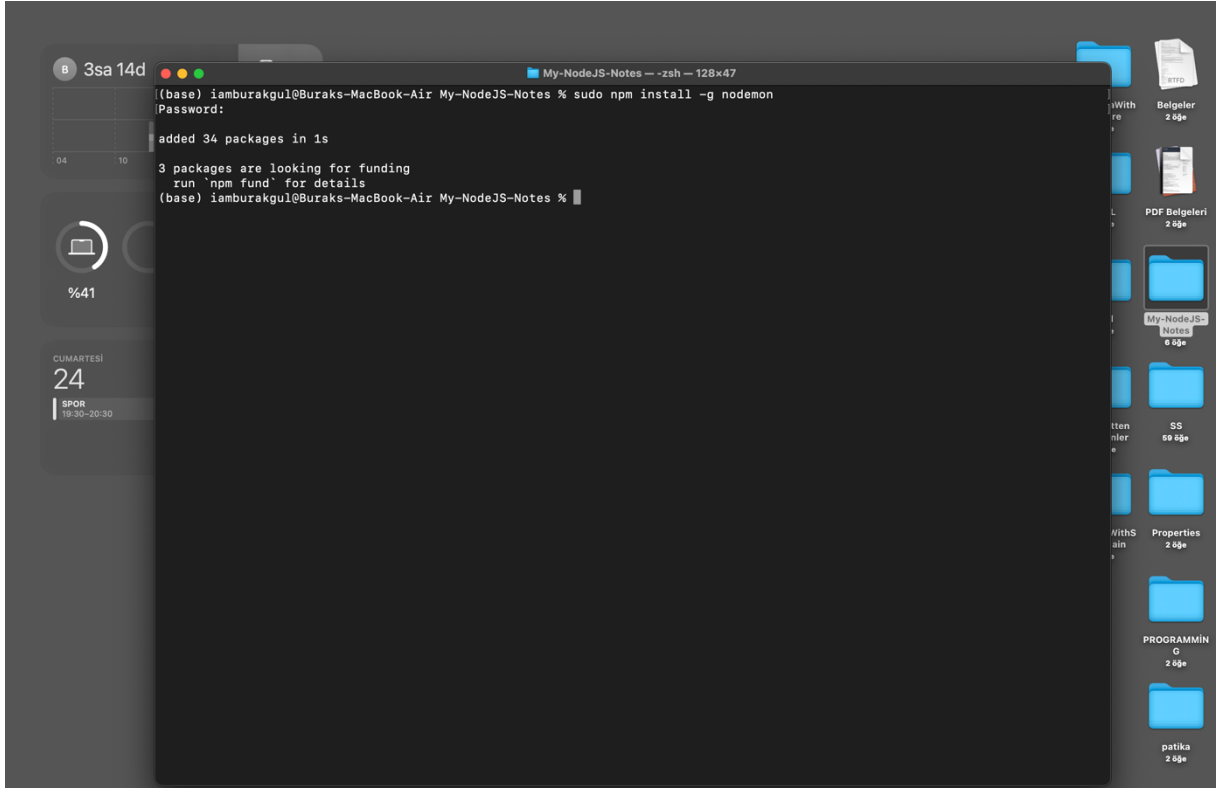
Global NPM modülleri sistem genelinde yüklenir, her yerden erişilebilir ve genellikle genel amaçlı komut satırı araçları veya yardımcı programlardır.

Şimdi ise her değişiklikte tek tek run etmek yerine bir modül daha indirelim ve her değişiklikte kendisi otomatik olarak çalışsın. Haliyle bu global bir modül olacaktır.

Modülümüz ise nodemon dur.

Sudo npm install -g nodemon yazarak indireceğiz.

<https://www.npmjs.com/package/nodemon>



Sizden şifrenizi isteyektir ve girdiğinizde herşey inecektir.

Şimdi ise nerede geçerli olmasını istiyorsak o pathe gidelim ve nodemon başlamasını istediğimiz file adını yazalım.

Nodemon index.js yazalım bulunduğumuz konumda.



```
Section1 > example4 > index.js > ...
1  const chalk = require("chalk");
2  console.log(chalk.green.bold("Burak"));
3  console.log(chalk.bold.inverse.red("Successsss!"));
4
```

```
(base) iamburakgul@Buraks-MacBook-Air Example4 % nodemon index.js
[nodemon] 3.1.0
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting node index.js
Burak
Successsss!
[nodemon] clean exit - waiting for changes before restart
```

Şimdi ise değişiklikleri tek tek gözlemlemek için kodlarda değişiklik yapın ve anında değişecek.