T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİTİRME PROJESİ

PROJE KONUSU: NodeJs ve WebRTC kullanarak Görüntülü Görüşme yapma

Projeyi Alanlar:	**Salih Yesir**	**G130910050**
	Şafak Batuhan Güleryüz	**G130910055**
	Emre Sakız	**G130910043**

İÇİNDEKİLER

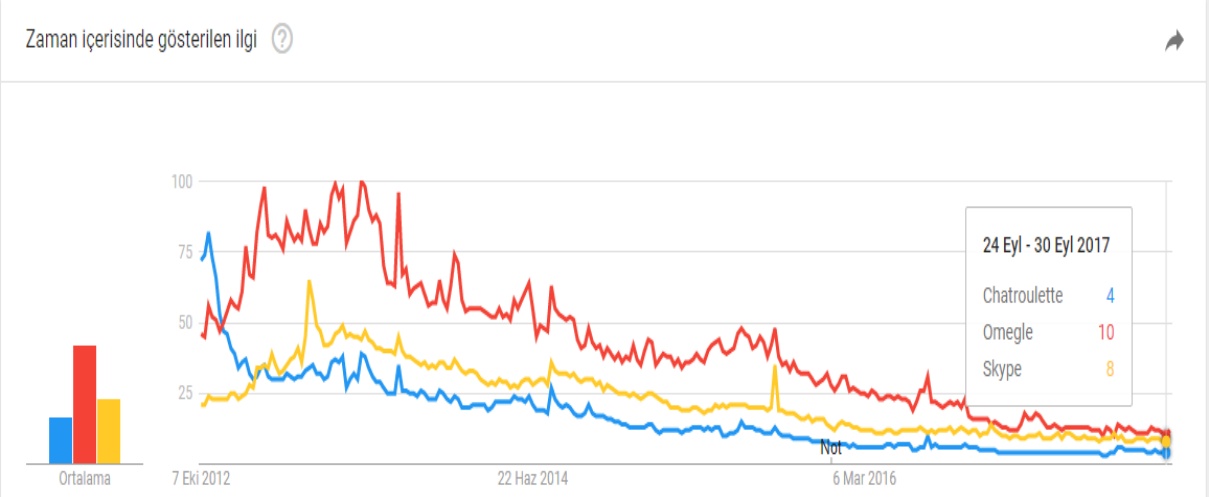
BÖLÜM 3.

BÖLÜM 4.

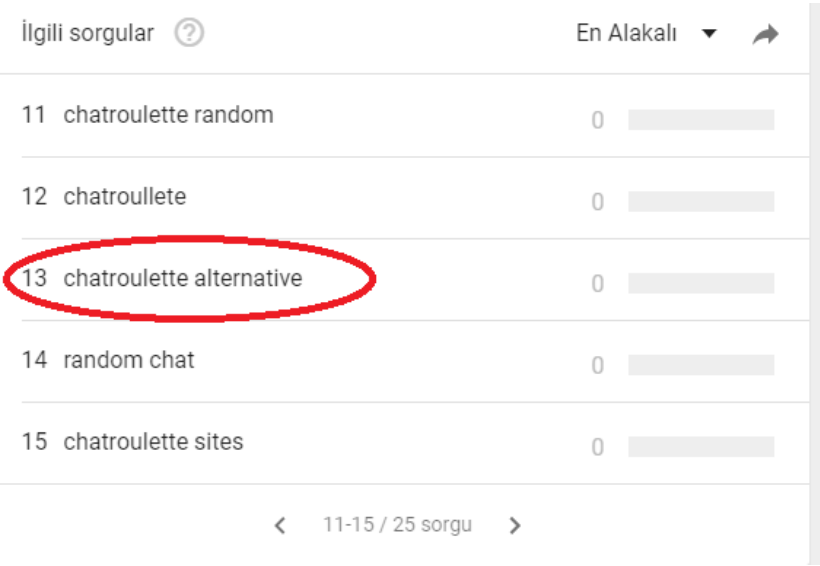
Görüntülü Görüşme

İletişim gün geçtikçe farklılaşıyor. İnsanlar gün geçtikçe birbirleriyle farklı şekilde iletişim kurma ihtiyacı duyuyorlar. Günümüzde bunun en verimli yolu görüntülü görüşme, bu sayede insanlar hem birbirlerini görüyor hem de birbirlerinin seslerini duyuyorlar yani neredeyse yüz yüze konuşuyorlar bu da onlara baya kolaylık sağlıyor.

Biz de yaptığımız araştırmalar sonucunda bir zamanlar popüler olan görüntülü görüşmelerin son zamanlarda gün geçtikçe kullanımının azalmış olduğunu gördük.



Hatta insanların bunlara alternatif arama ihtiyacı duyduklarını da gördük.



Daha sonra da böyle bir bitirme projesi yapmaya karar verdik ve

araştırmalarımıza başladık.

WebRTC nedir?

WebRTC, **web tarayıcılarda gerçek zamanlı iletişim(Real-Time Communications yani -RTC-) özelliklerini Javascript API'leri ile hayata geçirebilmemizi sağlayan açık kaynak projesidir.**

Biraz daha derine inecek olursak…

WebRTC'nin Bize Sundukları:

- **Video+ses konferans uygulamaları**
- **Ekran paylaşımı(full hd)**
- **Dosya paylaşımı (sunum dosyaları, resimler ve diğer dosyalar)**
- **Anlık gerçekleşen oyunlar(ör: satranç) için WebRTC Data Channel'ları kullanılmakta**

- **Ve tüm bunlar için plugin gerekmeyişi. Evet, oldukça cezbedici**

Bu işin arkasındaki kim? Sadece Javascript API?

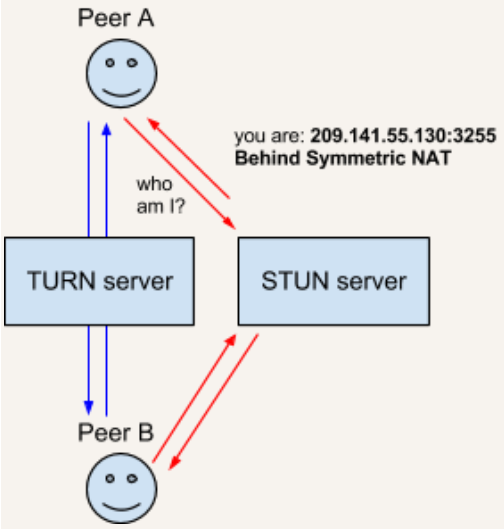
Tüm bu özellikler IETF, W3C grubu ve diğer destek verenler (Google,Mozilla,Opera) tarafından ortaya çıktı ve Javascript API'leri ile yapılabiliyor. Bu durumda Javascript ve *WebRTC*'nin nasıl çalıştığını bilmemiz yeterli olacaktır. WebRTC'nin nasıl çalıştığı konusu yeni başlayanlar için kafa karıştırıcı gibi gözükse de aslında gayet kolay anlaşılabilir ve zekice olduğunu kavrayacaktır. Bu yüzden bazı protokolleri, standartları ve API'leri bilmemiz gerekiyor. Bunları anlatalım öyleyse…

WebRTC Protokolleri

WebRTC mimarisini anlamak için bilgi sahibi olmamız gereken bazı protokoller vardır. Çok fazla detaya inmeden, bu protokolleri açıklamaya çalıştım.

- **ICE(Interactive Connectivity Establishment) : Web tarayıcı ile eş(peer) arası bağlantı kuran bir çatıdır. ICE sunucuları önemlidir çünkü kullanıcıların güvenlik duvarları UDP veya TCP portlarını bloke etmemesi gerekmektedir.**
- **STUN(Session Traversal Utilities for NAT) : Eş(peer) ile direkt bağlantıyı sağlamak için public adresimizi ve herhangi kısıtlama varsa bunları tespit eden bir protokoldür kendisi. Örnek şekil ise şöyle oluyor:**

- **<u>TURN</u>(Traversal Using Relays around NAT) : Simetrik NAT türünde çaprazlama bir yapı varsa bu durumda TURN işimizi karşılayacaktır. Bu tür sistemler public ip adreslerini dış ağlardan gizliyor(STUN burada çaresiz kalıyor). Dolayısıyla TURN sunucusu devreye giriyor ve rastgele public ip adresleri oluşturup stream veri akışını özel portlar üzerinden sağlıyor.**



Ana WebRTC Bileşenleri

- **getUserMedia : Tarayıcı üzerinden kamera ve mikrofona erişebiliyoruz**
- **RTCPeerConnection : Ses ve video çağrılarını ayarlar. Yani iki eş(peer) arasındaki veri bağlantısı ve yönetimi sağlanıyor.**
- **RTCDataChannels : Tarayıcıların peer-to-peer yani eşten-eşe (eşlerden kastımız kullanıcıların kendisi oluyor bir bakıma) veri paylaşımı yapmaktadır.**

Hangi Tarayıcılar Destekliyor?

PC ve Android sistemlerde ;

- **Google Chrome**
- **Mozilla Firefox**
- **Opera**

tarafından destekleniyor şu an. Bu tarayıcıların hangi özellikleri ne oranda, hangi kalitede desteklediğini öğrenmek için ise http://iswebrtcreadyyet.com üzerinden bakabilirsiniz.

iOS sistemlerde:

- **Bowser**

MONGODB

NoSql (Not Only SQL) kavramı ile hayatımıza girmiş olan mongodb, C++ ile yazılmış açık kaynaklı ilişkisel olmayan ve önde gelen NoSQL veritabanıdır. Veriler, JSON döküman yapısında ve hiyerarşik olarak saklanır. Büyük ölçekli uygulamalar için tasarlanmıştır.

Hemen nosql nedir ondan da bahsedelim eksik kalmasın; ilişkisel veritabanları ve modern yazılım geliştirmenin eksiklerini gidermek amacıyla gelmiştir. NoSQL ile çalışırken alışmış olduğumuz ilişkisel veritabanlarımızdaki gibi verilerimizi satır satır saklamaya ve diğer tablolarla ilişkilendirip tanımlamalarımızı yapmıyoruz. Veriler JSON ya da XML formatında saklanıyor. Ancak burada dikkat edilmesi gereken nokta NoSQL, Fire and Forget prensibi ile çalıştığı için bankacılık vb. kritik uygulamalarda kullanılmamalıdır. Aksine verinin 100% önemli olmadığı durumlarda kullanılabilir.

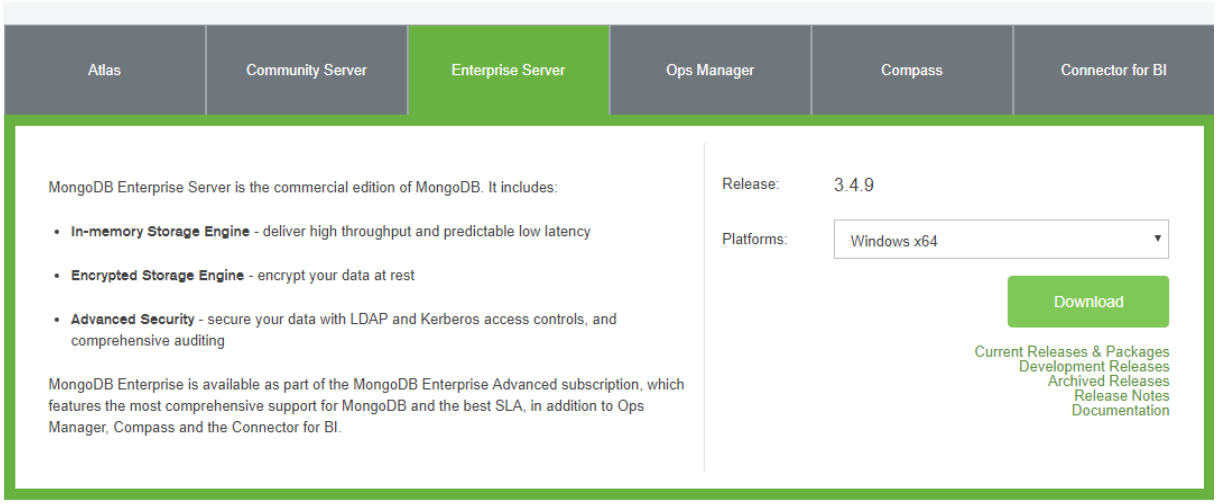
Neden MongoDB?

MongoDB, kendisini diğer NoSQL çözümlerinden ayıran ve öne çıkaran önemli artılara sahiptir. Bunlar arasında

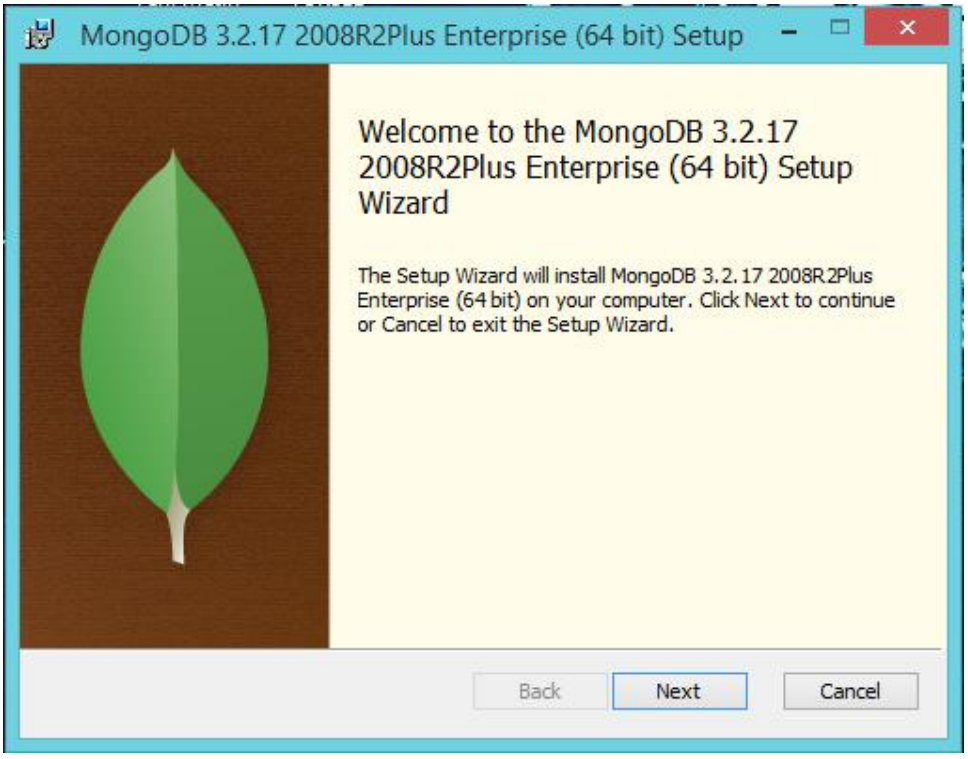
- **Sorgu(*query*) desteği. Pekçok NoSQL çözümü veriye sadece anahtarlar(*key*) üzerinden erişme olanağı sağlarken, MongoDB istenilen alanlar ve belirli aralıklara(*range query*) göre, ayrıca düzenli ifadelerle(*regular expression*) de sorgulama imkanı sunuyor.**
- **İkincil(secondary) index desteği. İstenilen alanlara göre sorgulama yanı sıra, bu alanları secondary index olarak tanımlayabilmek, veriye daha performanslı erişim imkanı sağlıyor.**
- **Master-Slave Replication desteği. Yazma ve okuma işlemlerini ayrı sunuculara yönlendirebilme, master sunucu erişilemez olduğunda bir slave sunucuyu master sunucu olarak çalıştırabilme kuşkusuz çok önemli bir artı değer.**
- **Sharding desteği. Büyük ölçekli verilerin sunucular arasında paylaştırılması özelliği de MongoDB'yi benzerlerinden ayıran, artı değerlerden birisi.**
- **MapReduce desteği.**
- **Pekçok yazılım dili için sürücü(*driver*) desteği en başta sayılabilir.**

MONGODB KURULUM

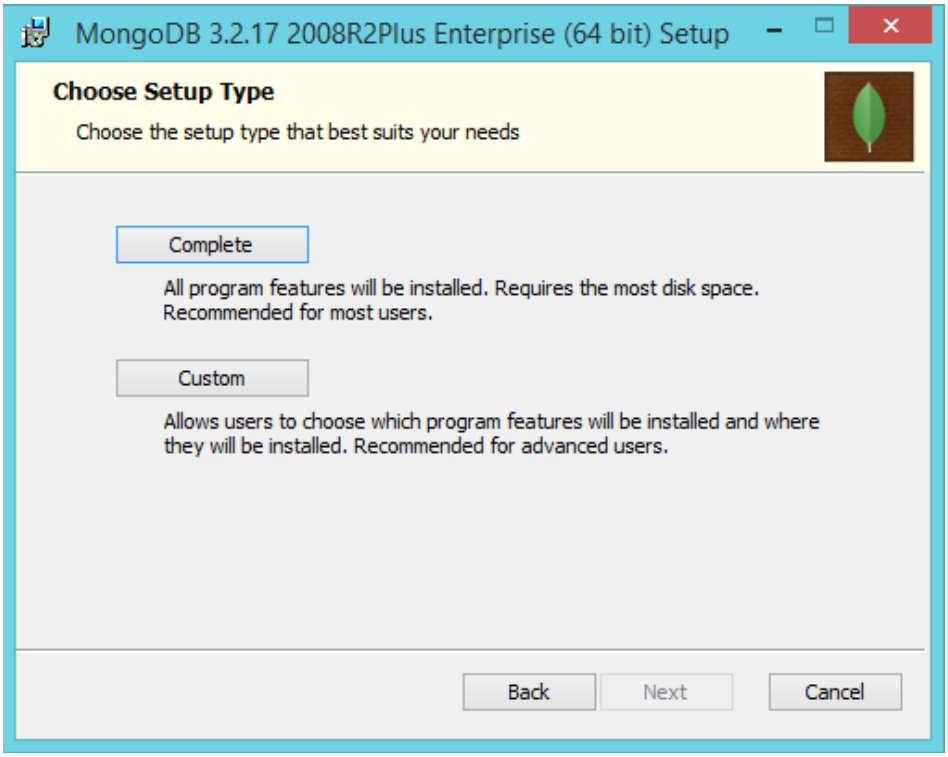
1) İlk olarak www.mongodb.com sitesine giderek MongoDb'yi indiriyoruz.



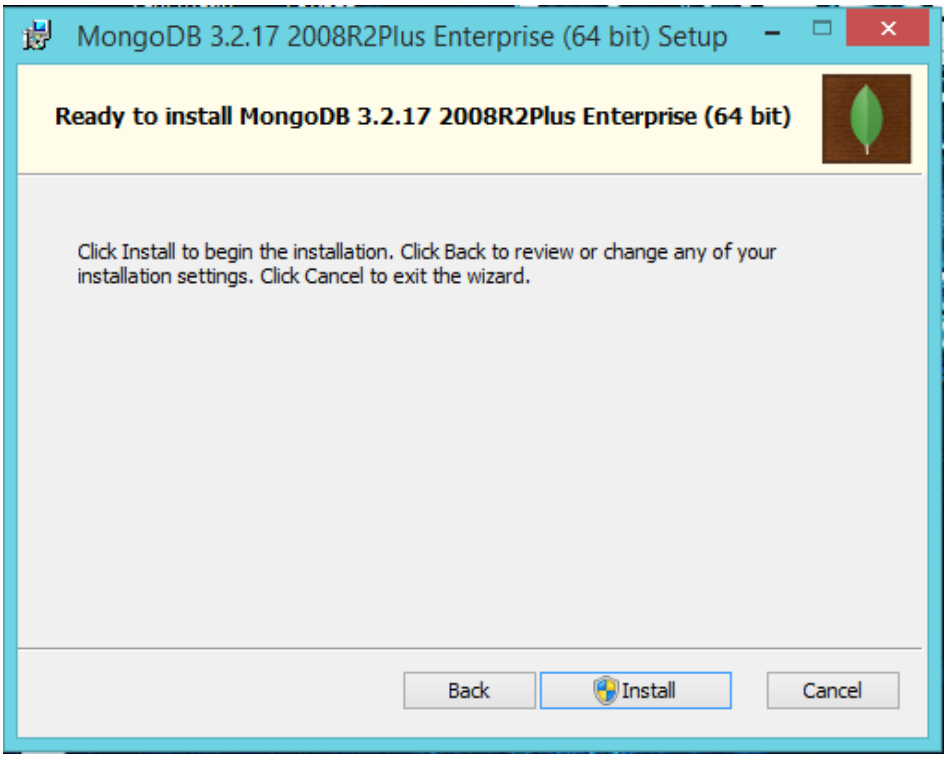
2) İndirdiğimiz Setup dosyasının kurulumuna başlıyoruz.

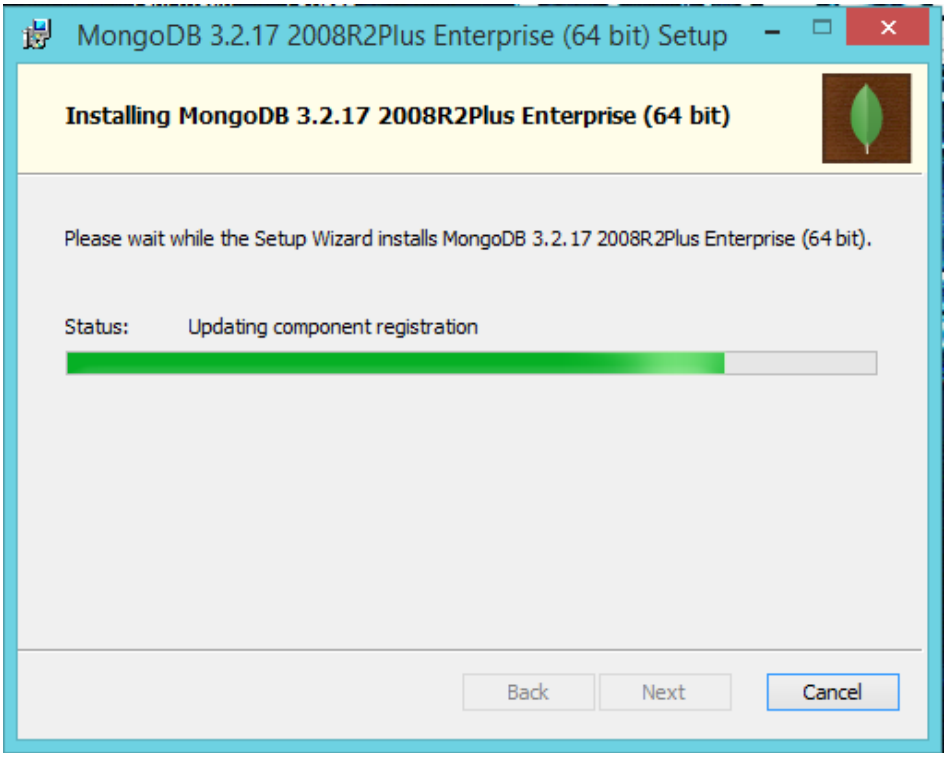


3) Direk kurulumu gerçekleştirmek istiyorsak "Complete" butonuna basarak kurulumu gerçekleştiriyoruz. Böylece direk Program Files'ın içine kurulum gerçekleşiyor. "Custom" butonuna basarsak istediğimiz yeri seçip oraya kurulumu gerçekleştirebiliyoruz.

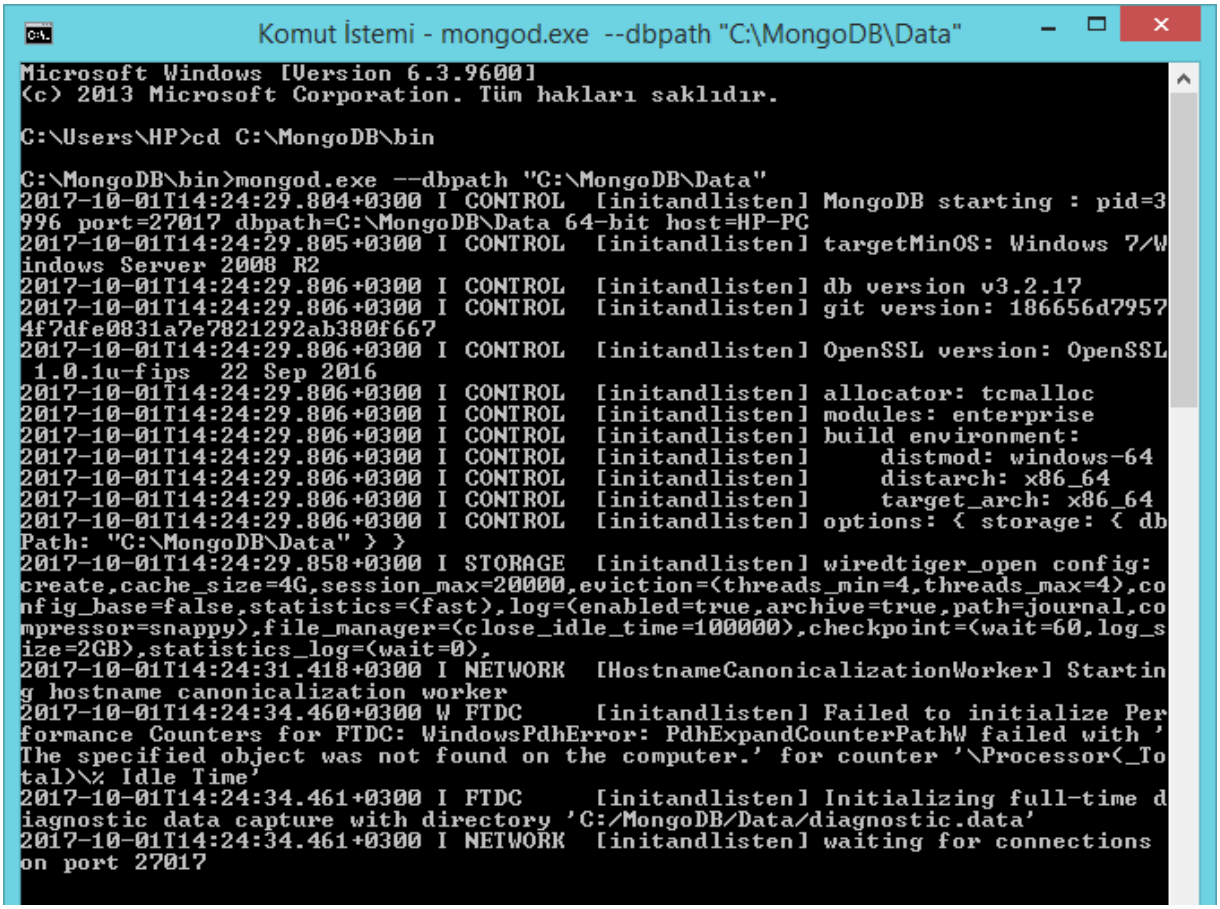


4) ”Install” diyerek kurulumu gerçekleştiriyoruz.





5) Son olarak kurulumun gerçekleştiği yere bir Data klasörü oluşturuyoruz. Komut İstemcisine girerek bu data içine gerekli config ayarları gerçekleştiriyoruz böylece kullanılacak olan port ayarlanmış oluyor.



NODEJS Nedir ?

NodeJS, Chrome web tarayıcısının da üzerinde çalıştığı gibi, V8 javascript motoru üzerinde çalışan, event-driven, nonblocking I/O modeli kullanan, ölçeklenebilir uygulamalar geliştirmek için dizayn edilmiş bir platformdur. Açık kaynak bir proje olan NodeJS 2009 yılında geliştirilmeye başlanmış olup Joyent şirketinin kayıtlı bir markasıdır.

Öncelikle V8 javascript motorundan biraz söz edelim. V8 daha öncede belirttiğimiz gibi chrome web tarayıcılarınında üzerinde çalıştığı google tarafından geliştirilen , C/C++ ve javascript dilleriyle geliştirilen esasında yaptığı iş javascript kodunu makine koduna (native machine code) çevirmek olan bir javascript motoru. NodeJS de bu yüksek performanslı javascript motorunun üzerine inşa edilmiş bir platform.

NodeJS platformunda iş parçacığı mantığının yerine her bir olayın sıraya atılarak işlenmesi söz konusu. Non-blocking I/O yapısı ile birleşince bu yapı yüksek performanslı uygulamalar çıkarmamıza, düşük özellikli sunucularla bile milyonlarca web istemini (web request) karşılamamıza imkan sağlıyor. Javascript'in doğasında olan asenkron programlama yapısıda bu modele bire bir uyum sağlıyor.

- **NodeJS 'in çalışma yapısının bir örnekle açıklamaya çalışalım. Bir fast-food restorantında tek bir personelin çalıştığını ve sipariş verdiğinizde, çalışanın ödemenizin alınmasından, yiyeceğinizin hazırlanmasına kadar yani sizin işiniz tamamen bitene kadar başka bir sipariş almadığını varsayalım. Bu durumda oluşacak sırayı gözünüzde canlandırın. Böyle bir modelde daha fazla sipariş almanın tek yolu çalışan sayısını arttırmaktır. Hatta ve hatta bir noktada çalışan sayısının artması bile bizi istediğimiz çözüme ulaştırmayacaktır. Burada sipariş alma bloklama gerektirmeyen bir işlemken yiyeceğin hazırlanma süreci bloklama gerektiren bir iştir. Bloklama gerektiren bir işten dolayı bu yapıda tüm işlemler durmakta, bloklama yapan işlemin bitmesini beklemektedir. Bloklama yapmayan (non-blocking) bir yapıda ise bloklama yapmayan işlemler işlenir bloklama yapan işlemler başlatılır ve sonuçları callback adı verilen geri dönüş metodları aracılığıyla elde edilir. Yiyeceğiniz hazırlandıktan sonra çağrılmanız callback'ler için verilebilecek güzel bir örnektir.**

  NodeJS'den bahsedip de NPM (node package manager)' den bahsetmeden olmaz. NPM NodeJS kurulumu ile birlikte gelen paket yönetim aracıdır. NPM üzerinde komünite tarafından geliştirilen açık kaynaklı NodeJS modülleri yer almaktadır.Her gün daha fazla büyüyen ve geliştireceğiniz uygulamayla ilgili aklınıza gelen hemen hemen her türlü kütüphanenin bulunduğu bu canlı ve dinamik yapı çok daha hızlı geliştirme yapma imkanı sağlamaktadır.

  Popülerliği giderek artan bu platform milyonlara hitap eden web uygulamalarının da dikkatinden kaçmadı. Burada Linkedin[1] ve PayPal[2] örneklerinden bahsetmek istiyorum. Linkedin mobil sunucu tarafında NodeJS 'e geçmeden önce Rails kullanıyordu.

NodeJS avantajları:

1)30 sunucudan 3 sunucuya düşen server maliyeti.
2)Bazı işlemlerde 20 kata kadar varan hız artışı
3)Geliştiricilerin tek bir dil üzerinde anlaşabiliyor olması ve backend geliştiricilerinin javascript geliştirme yeteneklerinin artması.

Paypal firmasının NodeJS kullanımı:

PayPal örneği ise daha yakın tarihli. Ürün ortamını riske etmemek adına direkt NodeJS'e geçişi göze alamayan PayPal Java ve NodeJS olarak 2 plaftormda paralel olarak geliştirmeye başlıyor. Java takımında 5 kişilik bir ekip varken NodeJS tarafında 2 kişilik bir ekip çalışıyor. Java tarafında hazır olan geliştirme ve çalışma ortamını kurmak NodeJS ekibinin 2 ayını alıyor. Yani 2 ay boyunca java takımı geliştirme yaparken NodeJS takımı sadece gerekli alt yapıyı kurmakla uğraşıyor. 6 ayın sonunda ise NodeJS ekibi java ekibini yakalıyor aynı fonksiyonaliteye sahip uygulamayı geliştirmeyi başarıyorlar. Daha az kişilik bir ekiple 2 aylık gecikmeyle ve aşağıda sıralayacağım farklarla.

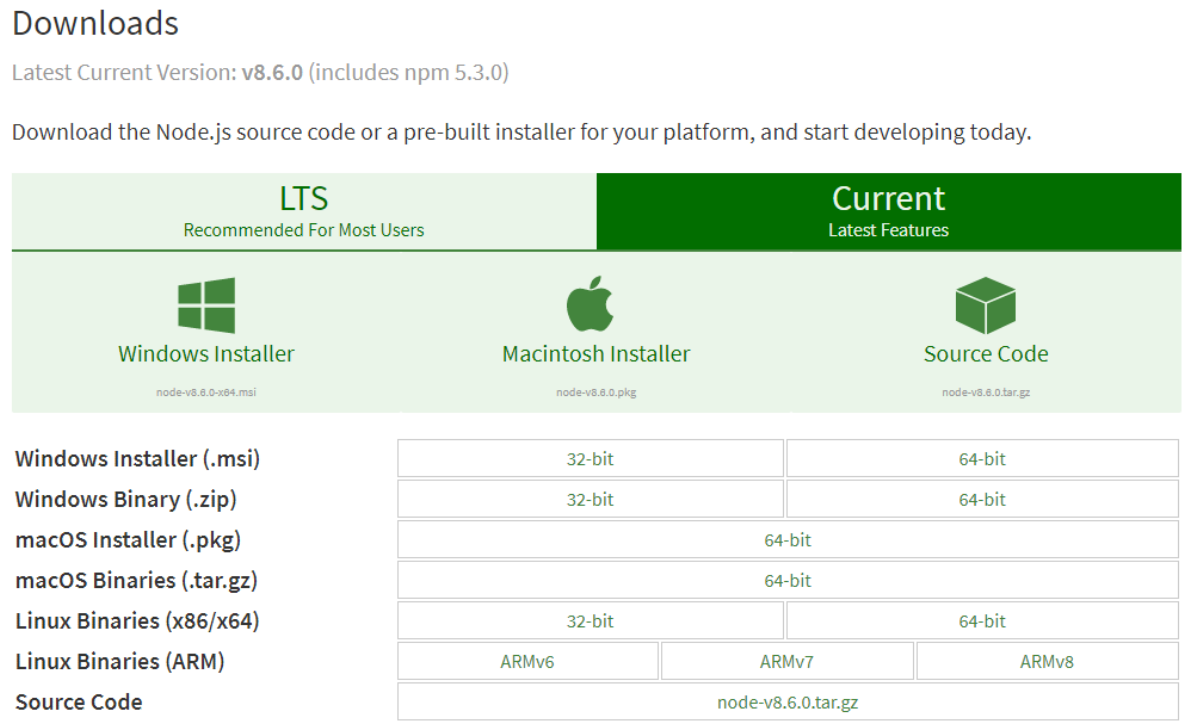
- **Neredeyse 2 katı bulan geliştirme hızı**
- **%33 oranında daha az satır kodla yazılması**
- **%40 oranında daha az dosya ile geliştirilmesi.**

Gelelim performans karşılaştırmasına.Yaptıkları testler sonucunda NodeJS uygulamasının java uygulamasına göre 2 kat daha fazla saniyelik istemi(request per sec) karşılayabildiğini görmüşler. Üstelik NodeJS uygulaması tek çekidekli bir işlemci üzerinden çalışırken java uygulamasının beş çekirdekli işlemci üzerinde çalışmasına rağmen.

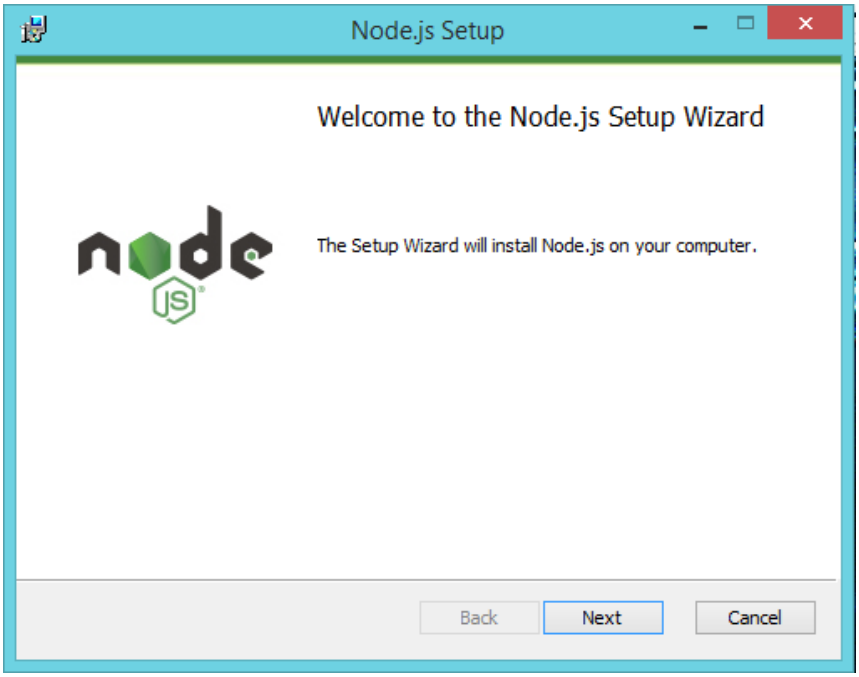
İkinci olarak cevap süresinde (response time) %35 oranında artış tespit etmişler.Bu da her bir cevabın yaklaşık 200 ms daha hızlı karşılanmasına karşılık geliyor.Özetlemek gerekirse NodeJS özellikle web dünyasında hızla yerini almış bence yeterli olgunluğa ulaşmış, ölçeklenebilirlik açısından nosql veri tabanlarıyla da son derece uyumlu çalışabilmesi sayesinde tercih edilebilecek en uygun geliştirme platformu. NodeJS 'in bir programlama dili olmadığını anlamak , sunduğu çözümleri ihtiyaçlarımızı karşılamasına göre değerlendirerek kullanmak gerektiğini de belirtelim.

NodeJS Kurulum

1)NodeJS'in sitesine girerek kendi işletim sistemimize göre indirme işlemini gerçekleştiriyoruz.

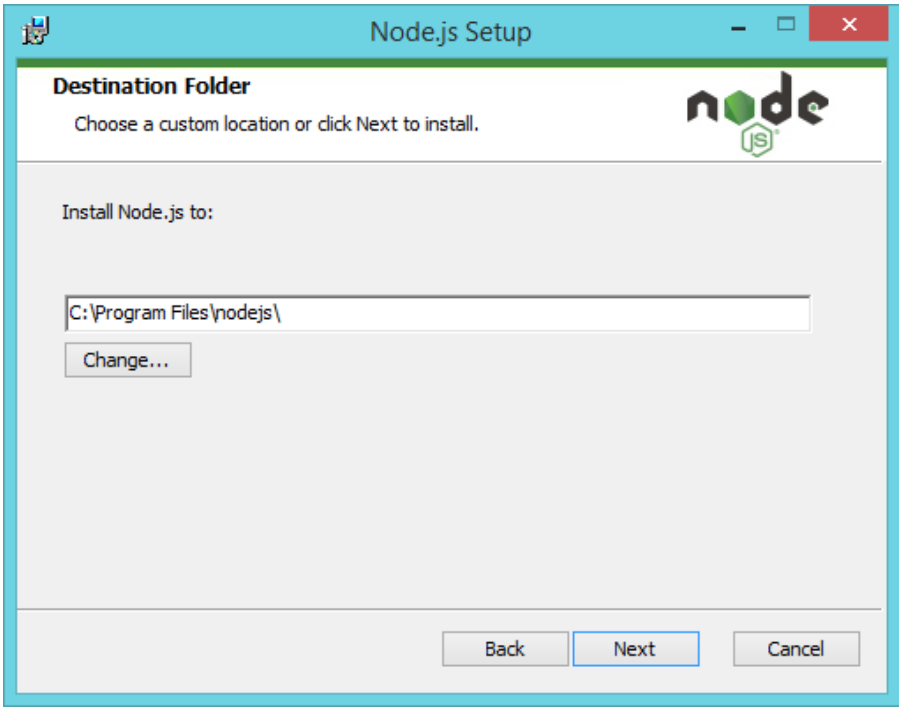


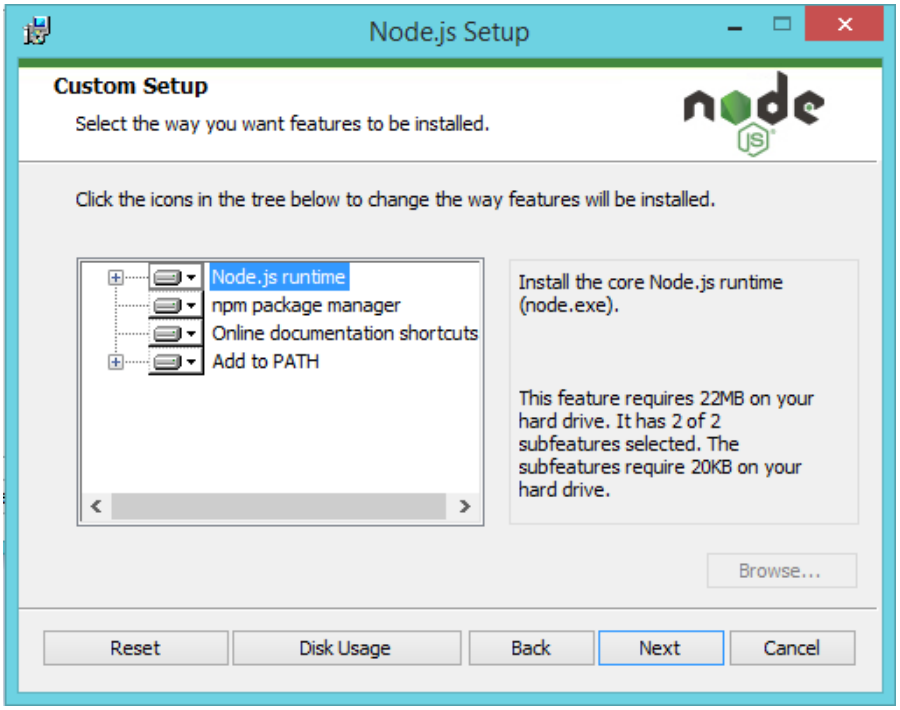
2)İndirdiğimiz Setup dosyasından kuruluma başlıyoruz.

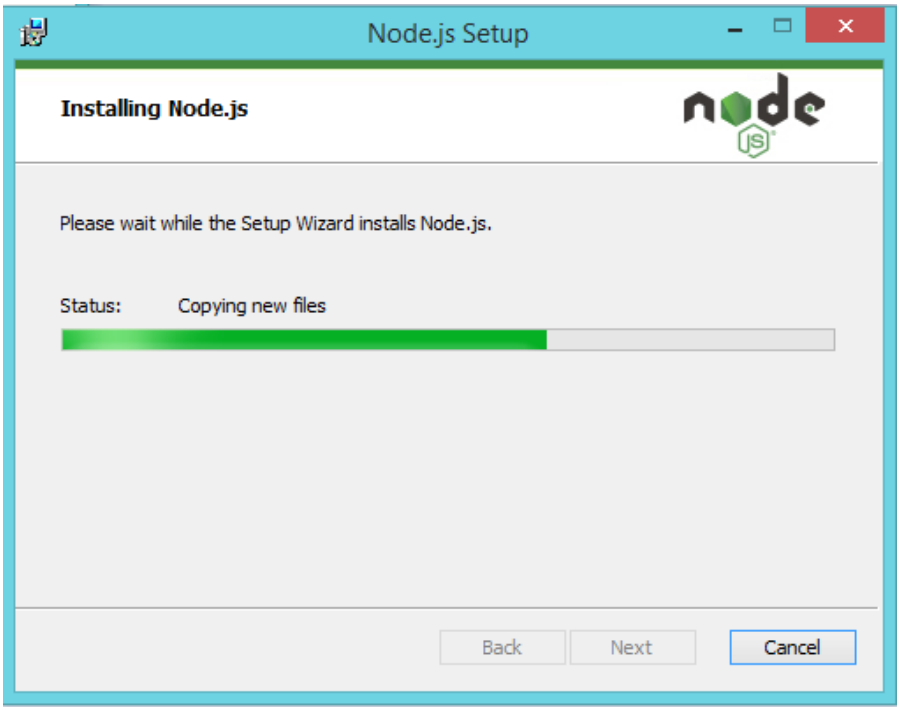


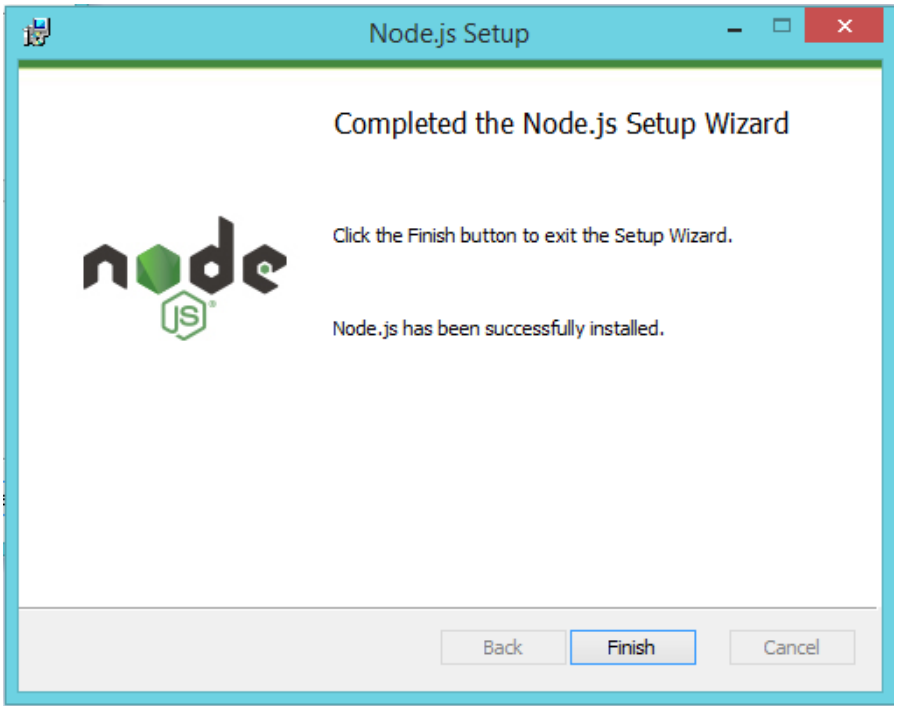
3)Yükleyeceğimiz yeri seçiyoruz ve nasıl bir yükleme olacağını seçip yüklemeye

başlıyoruz.


Node.js Setup
Destination Folder
Choose a custom location or click Next to install.
Install Node.js to:
C:\Program Files\nodejs\
Change...
Back
Next
Cancel


Node.js Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
Node.js runtime
npm package manager
Online documentation shortcuts
Add to PATH
Install the core Node.js runtime (node.exe).
This feature requires 22MB on your hard drive. It has 2 of 2 subfeatures selected. The subfeatures require 20KB on your hard drive.
Browse...
Reset
Disk Usage
Back
Next
Cancel





4)Yükleme bittikten sonra da versiyonu aşağıda ki şekilde kontrol edip yüklemenin gerçekleştiğini görüyoruz.

Express Web Framework

Express modül Node.js için yazılmış bir templete çatısıdır. Yapısı ve kullanışlı methotları gereği işlerimizi oldukça kolaylaştırmaktadır. npm üzerinden paket olarak yükleyebileceğimiz Express sayesinde Url parse işlemlerini(routing) daha kolay bir şekilde yapabilir, statik dosya yönetimi işlerimizi daha kolay bir şekilde yapabiliriz. Kısaca nodejs ile web sitesi/uygulaması yapmak için gerekli tüm alt yapıyı Expressjs bize sağlamaktadır. Express'i yüklemek için konsola npm install express --save yazmamız yeterli.

```
var express = require('express');
var app = express();
```

MVC nedir?

MVC (Model-View-Controller), yazdığımız uygulamanın iş mantığı ile (business logic) kullanıcı arayüzünü birbirinden ayrıştıran, uygulamanın farklı amaçlara hizmet eden kısımlarının birbirine girmesini engelleyen yazılım mimarisidir.

Kodun farklı amaçlara hizmet eden yapılarını birbirinden ayırarak, kodu daha rahat geliştirilebilir ve test edilebilir (yani daha az hata çıkartma potansiyeline sahip) hale getirmiş oluruz.

Model

- **Uygulamada kullanılan verileri temsil eder ve verilerin işlenme** mantığının **saklandığı kısımdır. (Verilerin validasyonu burada yapılır)**

- **Genelde verilerin veritabanı (veya XML gibi benzer bir yere) kaydedilmesi ve kayıtlı yerden alınması işlemleri yine burada olabilir. (Genelde dedim, şu an detaya girmeyeceğim. Daha sonraki yazılarda bu konuyu detaylıca inceleyeceğim.)**

View

- **Basitçe, uygulamanızın kullanıcılarınızın gözüyle gördüğü kısmıdır, arayüzdür.**

Controller

- **Model ve View arasında getir götür işlemlerini gerçekleştirir.**
- **Kullanıcıların View üzerinden gerçekleştirdiği işlemlerle alınan veriyi Model'e taşır, Model'den aldığı veriyi View üzerinden kullanıcıya gösterir.**

- **MVC yapısında ana mantık Model ve View yapısının ayrılmasıdır. Bu iki yapı arasındaki haberleşmeyi sağlayan köprüye Controller diyoruz.**

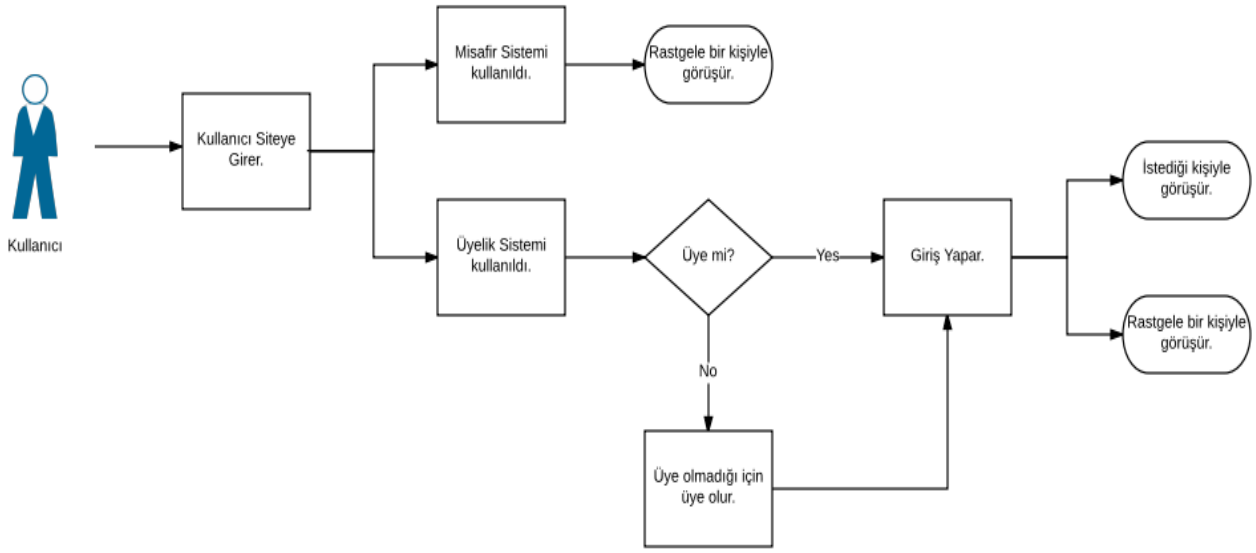
Ejs Layout

Html şablonlarıyla, servis tarafımızdan gelen verileri birleştirebilmemizi sağlayan javascript kütüphanesidir.

```
var ejsLayouts = require('express-ejs-layouts');
```

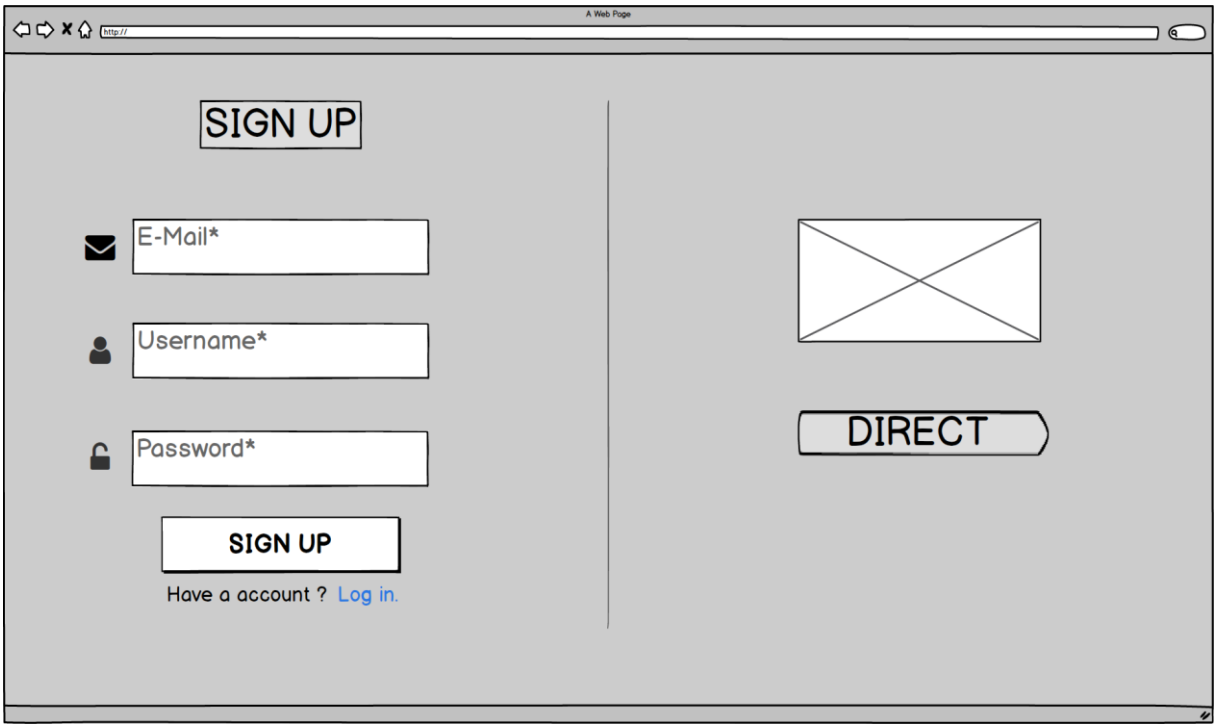
```
app.set('view engine','ejs');
app.use(ejsLayouts);
```

Projenin Use Case Diyagramı

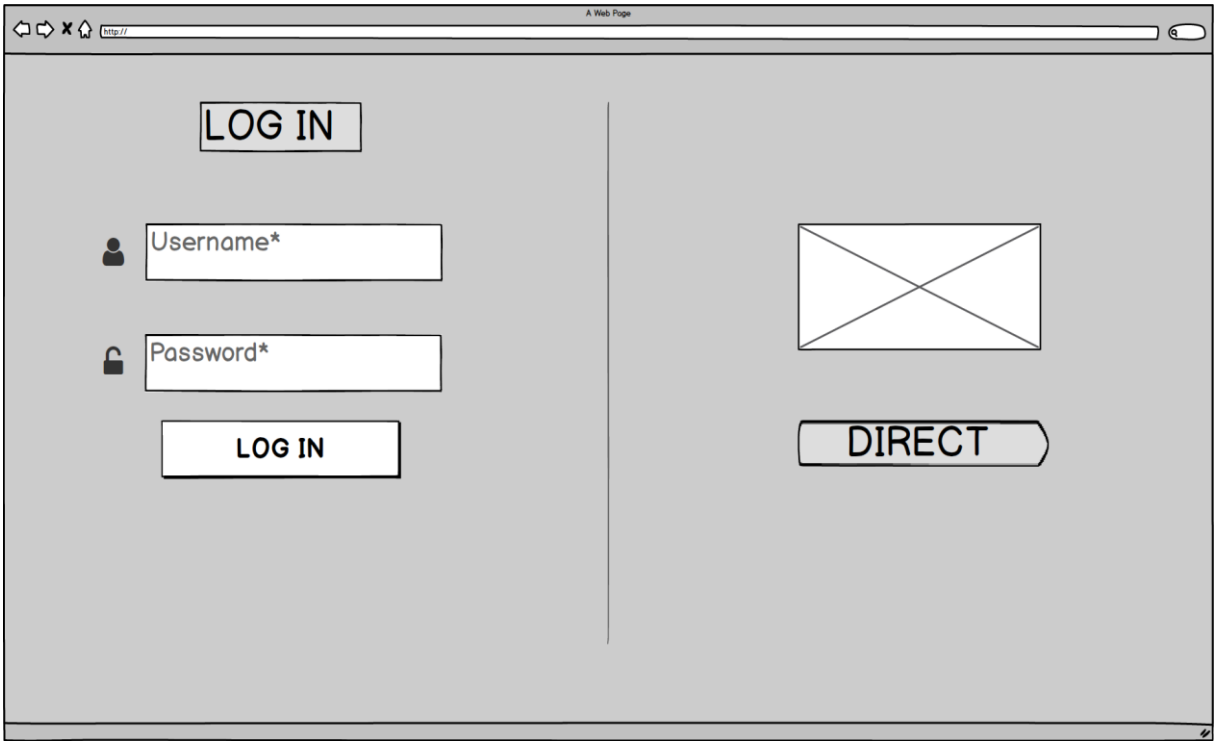


Proje Eskiz Tasarımları

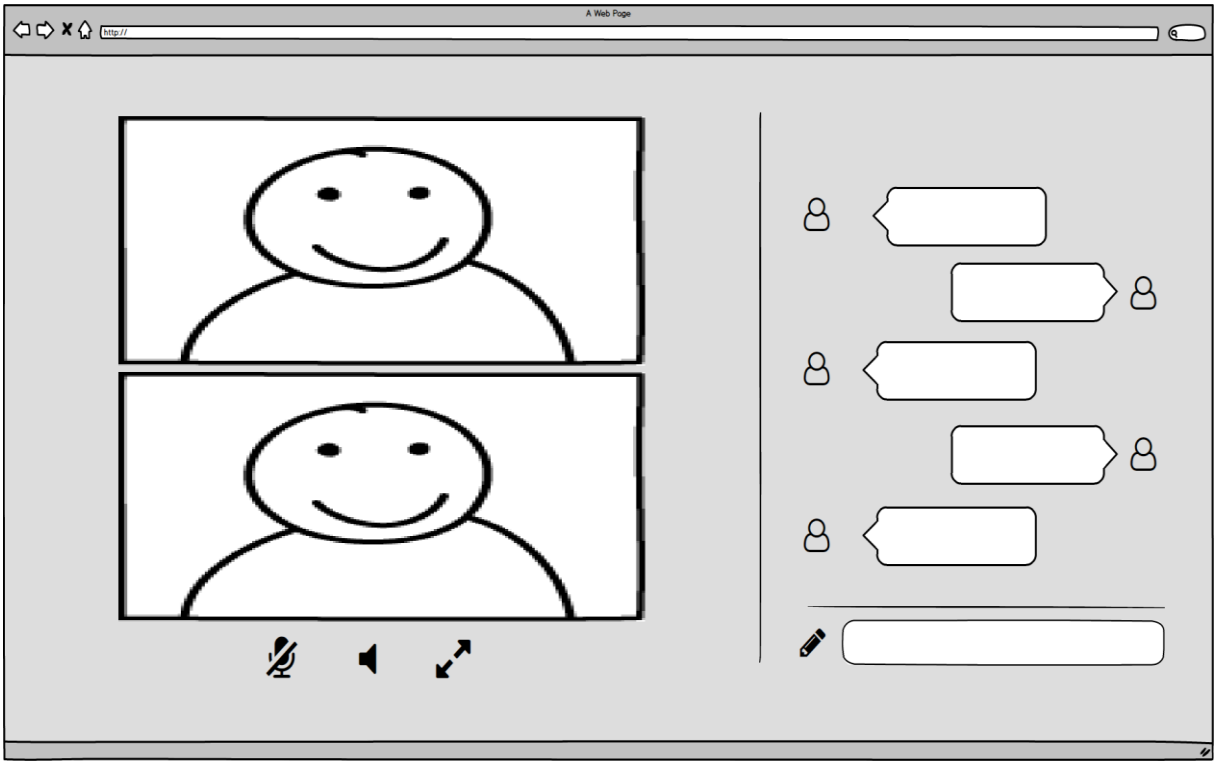
1)Giriş Ekranı



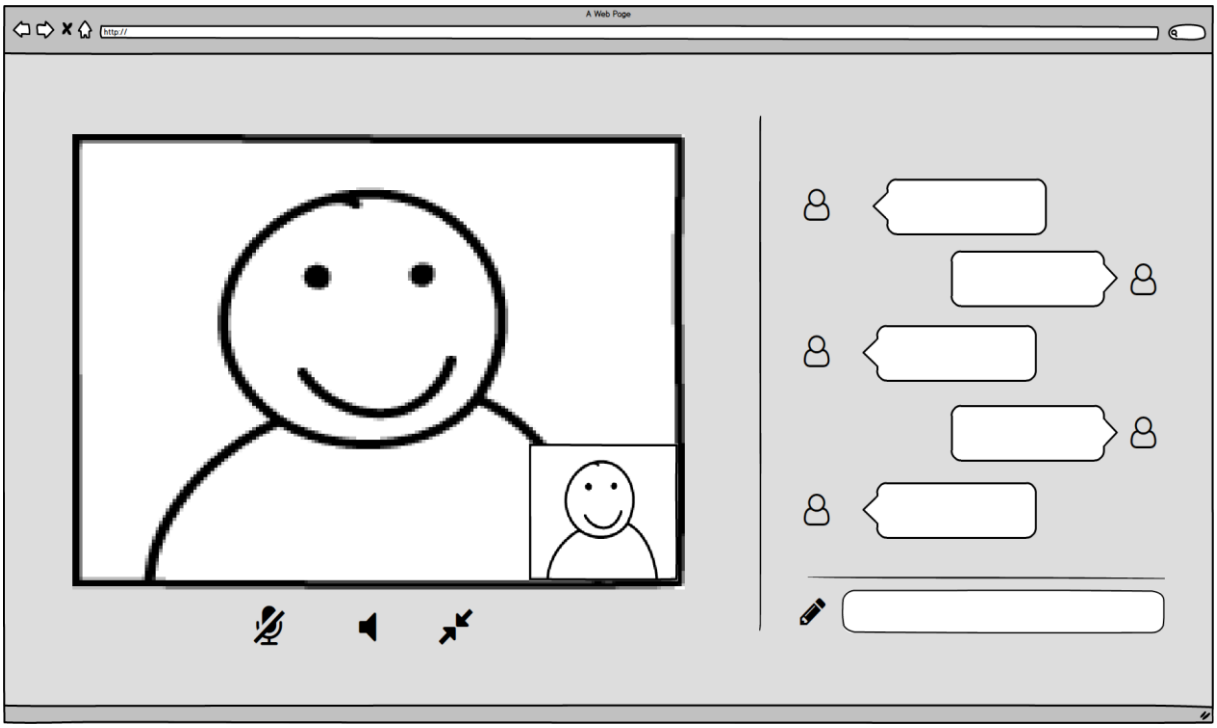
2)Üye Girişi Ekranı



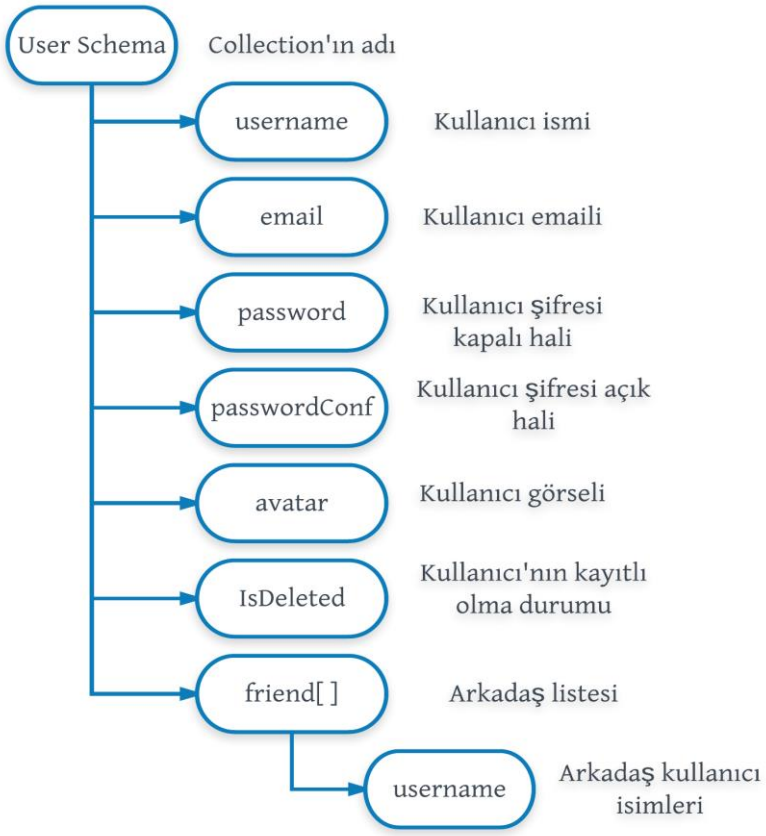
3)Küçük Kameralı Görüşme Modu



4)Büyük Kameralı Görüşme Modu



Veritabanı Diagramı



Veritabanı Kod ve Tasarım

```
var UserSchema = new mongoose.Schema({
    userName:{ type: String, required:true, unique:true, trim: true},
    email: { type: String, unique: true, required: true, trim: true},
    password: {
      type: String,
      required: true,
    },
    passwordConf: {
      type: String,
      required: true,
    },
    avatar: { type:Buffer, contentType:String,default:null},
    isDeleted: { type:Boolean, required:true , default:false},
    friend:[String]
});
```

Veritabanımız da sırasıyla username, email, password, passwordConf, avatar, isDeleted, friend

yer almakta. Username kısmında kullanıcının adını, mail kısmında kullanıcı kayıt olduğunda kullandığı maili, password da kullanıcını üyelik şifresini, avatar da üye olduktan sonra kullanıcağı avatar resmini, friend de kullanıcının arkadaşlarını, isDeleted da ise kullanıcı üyeliğini sildiğinde veritabanından kullanıcıyı silmeyerek burada pasif hale getiriyoruz.

-Username tip olarak string tipinde olmalı, gerekli olmalı ve unique olmalı ki aynı isimde başka kullanıcı olmasın.

-mail tip olarak string olmalı.

-password tip olarak string olmalı ve gerekli olmalı.

-passwordConf tip olarak string olmalı ve gerekli olmalı ayrıca şifrenin açık halini göstermektedir.

-avatar tip olarak buffer olmalı, content tip olarak string olarak tutulmalı ve istenirse null olarak bırakılabilinir.

-friend string tipinde olmalı ancak birden fazla olacağı için dizi şeklinde tutulmalı.

-isDeleted ise type olarak boolean olmalı, üyelik açıldığında boolean değer olarak 0 tutulacak eğer kullanıcı silmek isterse 1 olarak değiştirilip pasif hale getirilecek. Gerekli olmalı ve default yani varsayılan olarak false yani 0 olmalı.

Mongoose

Mongoose, MongoDB üzerinde çalışan esnek, şema bazlı, zengin özellikleriyle geliştiriciye kullanım kolaylığı sağlayan bir ODM(Object Document Mapper) yapısı. Mongoose ise ORM çatılarının sağladığı kolaylıkları bize mongoDB tarafında sağlayan bir araç.

JSON nesneleri ile çalışmamızı, bu nesneler üzerinden CRUD işlemleri gerçekleştirmemize olanak sağlayan, kısıtlamaları(constraints) ve varsayılan değerleri belirlememizi kolaylaştıran, hesaplatılmış alanlar(calculated fields) tanımlamamıza imkan veren Mongoose, MongoDB ile uygulama geliştirmeyi çok daha akıcı ve kolay bir hale getiriyor.

```
var mongoose = require ('mongoose');
var mongoDB= 'mongodb://localhost/NodeProje';
var promise = mongoose.connect(mongoDB,{
```

Express-Session Library

Yazılımınızın authentication(kullanıcı, login işlemleri) gerçekleştirmenizi sağlayan ExpressJs kütüphanesidir.

Ejs Library

Javascript kodlarınızın servis tarafından gelen verilerinizin, html içine yerleştirdiğinizde çalışmasını sağlayan kütüphanedir.

EasyRTC

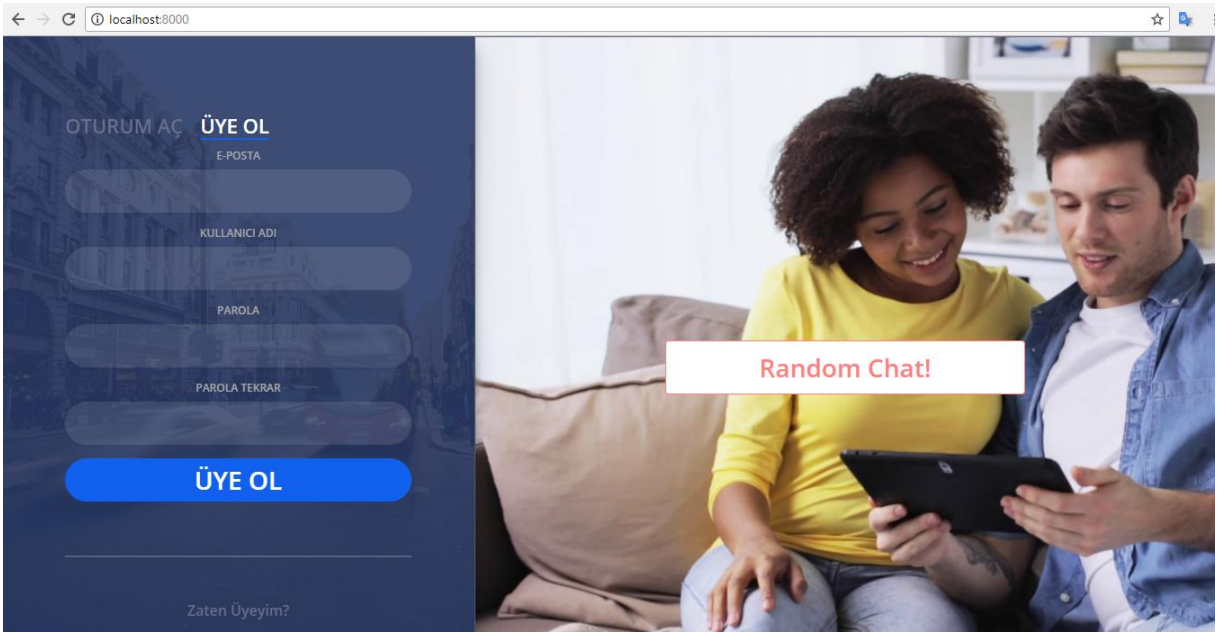
EasyRTC, web tarayıcıları arasında doğrudan ses, video ve verilerin gerçek zamanlı iletişimi için ortaya çıkan bir W3C / IETF standardı olan WebRTC'nin üzerine kurulmuş bir frameworkdür. WebRTC, desteklenen sunuculara çok az yük bindirerek eşler arası olarak ses, video ve verilerin transferini destekler.

Socket.io

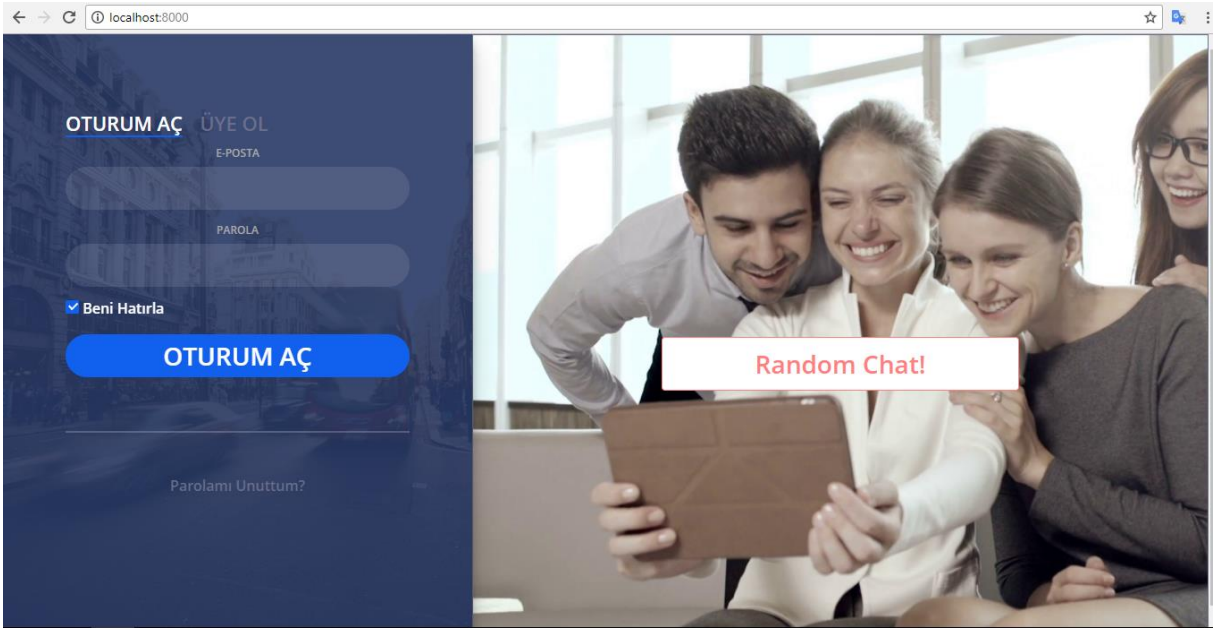
Socket.io, NodeJS ile anlık uygulamalar yapabilmemizi sağlayan, sunucuyu ve istemciyi yormayan, gecikme oranı düşük, performans oranı yüksek, kullanımı kolay bir kütüphanedir. En güzel yanlarından birisi ise; mobil, masaüstü cihaz ayırt etmeksizin hemen hemen bütün tarayıcıları desteklemesi.

Proje Tasarımları

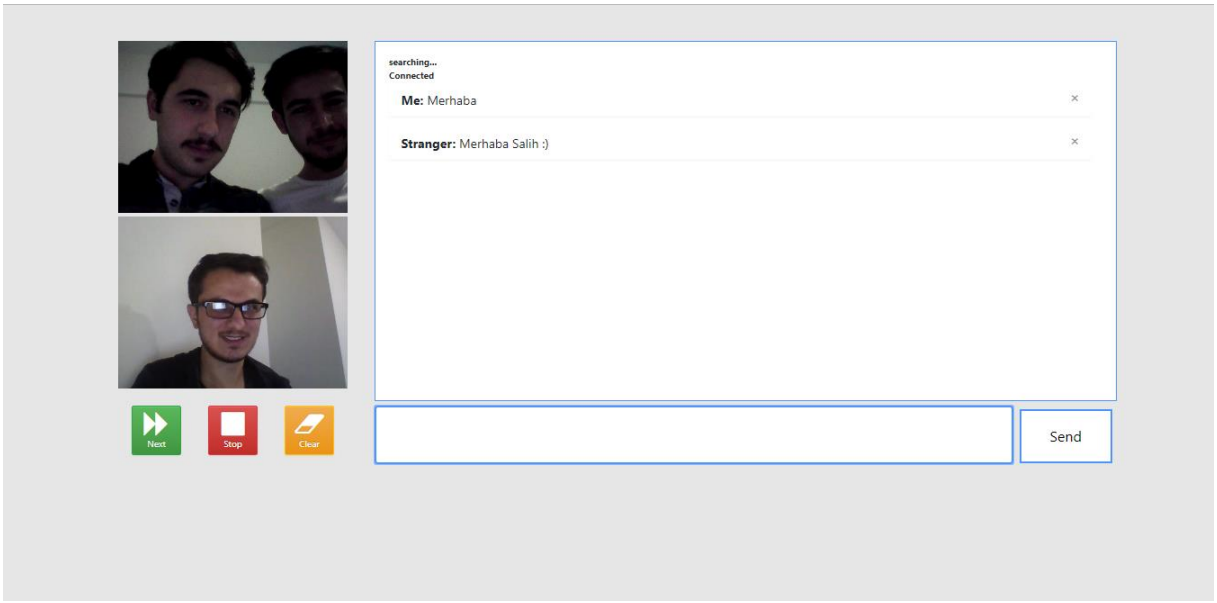
1)Giriş Ekranı



2)Üye Girişi Ekranı



3) Rastgele Görüntülü Konuşma Ekranı



KAYNAKLAR

[1] https://codelabs.developers.google.com/codelabs/webrtc-web/#0
[2] https://easyrtc.com/docs/
[3] https://medium.com/of-all-things-tech-progress/starting-with-authentication-a-tutorial-with-node-js-and-mongodb-25d524ca0359
[4] https://github.com/socketio/socket.io
[5] http://highscalability.com/blog/2012/10/4/linkedin-moved-from-rails-to-node-27-servers-cut-and-up-to-2.html
[6] https://www.paypal-engineering.com/2013/11/22/node-js-at-paypal/
[7] https://candelibas.wordpress.com/2014/11/16/webrtc-nedir-yenir-mi/
[8] https://trends.google.com.tr/trends
[9] https://scotch.io/tutorials/using-mongoosejs-in-node-js-and-mongodb-applications
[10] http://koddit.com/yazilim/mvc-nedir-gercek-orneklerle-mvc-nedir-anlayalim/