

**T.C.**  
**SAKARYA ÜNİVERSİTESİ**  
**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI**

**NODE.JS İLE GERÇEK ZAMANLI GÖRÜNTÜLÜ**  
**GÖRÜŞME UYGULAMASI**

**B130910040 – Yasin YALÇIN**  
**B140910031 – Ali Rıza OYAN**  
**B141210070 – Utku AKGÜNGÖR**

**Bölüm** : **BİLGİSAYAR MÜHENDİSLİĞİ**  
**Danışman** : **Prof. Dr. Celal ÇEKEN**

**2017-2018 Güz Dönemi**

## ÖNSÖZ

İnsanlık var olduğu andan beri iletişime büyük önem vermiş, haberleşme alanında sürekli geliştirmeler yapmıştır. Haberleşme ağı, iletişim standartları ne kadar gelişirse bir toplumun da o kadar geliştiği, o kadar üstün olduğu gözlemlenmiştir. Bunu göz önüne alan toplumlar her geçen gün haberleşmenin daha yeni, daha kolay ve daha hızlı bir yolunu bulmuşlardır.

Elektriğin hayatımıza girmesi ile sesli ve yazılı, internet teknolojilerinin gelişmesiyle de hem sesli, hem yazılı hem de görüntülü haberleşmeler gittikçe yaygınlaşmıştır. Öyle ki artık klasik GSM servisleri yerini yavaş yavaş internet teknolojilerine bırakmaktadır. Bu bağlamda hayata geçirmek istediğimiz bu projede de öncelikle yazılı, ardından görüntülü görüşme uygulaması yapmak için harekete geçtik. Bu çalışmamızda günümüzün popüler teknolojilerinden olan server tabanlı çalışan Node.js dilini kullanarak yazılı ve görüntülü haberleşme uygulaması yapacağız. Çalışmamızda birçok farklı kaynaktan yararlanıp bu bilgileri harmanladık ve kullandık.

**Node.js**, açık kaynaklı, sunucu tarafında çalışan ve ağ bağlantılı uygulamalar için geliştirilmiş bir çalışma ortamıdır. Node.js uygulamaları genelde istemci tarafı betik dili olan JavaScript kullanılarak geliştirilir.

En önemli avantajı JavaScript'in sağladığı bloklamayan imkânıyla yüksek ölçeklenebilirliği ve yüksek veri aktarabilmesidir. Bu teknolojiler sık sık gerçek zamanlı Web uygulamalarında tercih edilmekle beraber kullanım alanı popülaritesiyle orantılı olarak genişlemiştir.

Node.js, Google V8 JavaScript motorunu kullanarak betik dilini yorumlar ve içerisinde standart olarak dağıtılan kütüphaneler sayesinde ek bir sunucu yazılımına (Apache HTTP Sunucusu, Nginx, IIS v.s.) gerek kalmadan uygulamanın Web sunucusu görevini görür.

Projemizde önemli bir yer edindiği için Socket.io'dan da biraz bahsetmemiz gerekir. Socket.io'nun amacı, gerçek zamanlı olarak hemen hemen her tarayıcı ile (mobil cihazlar dahil) farklı iletişim protokollerine rağmen iletişim kurabilmelerini sağlamaktır. Web Socket ile ilgili protokollerin tamamını standartlaştıran ve sabit bir kullanım sunan web socket emülatörüdür. Node.js ile birlikte kusursuz bir kullanımı vardır. Fakat sadece Node.js ile kullanılmaz. Python, PHP ya da Ruby on Rails ile entegre olarak da kullanılabilen bir modüldür. Gerçek zamanlı uygulamalar için kullanılır. Teşekkür eder, saygılarımızı sunarız.

## İÇİNDEKİLER

ÖNSÖZ.....	2
İÇİNDEKİLER.....	4
SİMGELER VE KISALTMALAR LİSTESİ.....	5
ŞEKİLLER LİSTESİ.....	6
BÖLÜM 1.	
ÖZET.....	7
BÖLÜM 2.	
GİRİŞ.....	8
2.1. Proje Tanımı.....	8
2.2. Proje Amacı ve Kapsamı.....	8
2.3 Projenin Hedefleri.....	8
2.4 Literatür Taraması.....	9
BÖLÜM 3.	
YÖNTEMLER VE KULLANILAN TEKNOLOJİLER.....	10
3.1. Node.js.....	10
3.2. MongoDB.....	13
3.3. WebRTC.....	17
BÖLÜM 4.	
GELİŞTİRİLEN SİSTEMİN ANLATIMI.....	20
BÖLÜM 5.	
SONUÇLAR VE TARTIŞMA.....	21
KAYNAKÇA.....	22
ÖZGEÇMİŞ.....	23
BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI	
DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI.....	24

## SİMGELER VE KISALTMALAR LİSTESİ

Js : JavaScript  
Db :Database

## ŞEKİLLER LİSTESİ

Şekil 3.1.	login.ejs.....	10
Şekil 3.2.	server.js router kullanımı.....	11
Şekil 3.3.	anasayfaRouter.js.....	11
Şekil 3.4.	loginController.js.....	12
Şekil 3.5	kayitPost.js.....	12
Şekil 3.6	indexPost.js.....	13
Şekil 3.7.	Veri tabanı şeması.....	14
Şekil 3.8.	mongod ve mongo terminal ekranları.....	14
Şekil 3.9.	db.js.....	15
Şekil 3.10.	chat.js.....	16
Şekil 3.11.	server.js mongodb kullanımı.....	16
Şekil 3.12.	Robomongo.....	17
Şekil 3.13.	index.js WebRTC.....	18
Şekil 3.14.	main.js WebRTC.....	18

## **BÖLÜM 1. ÖZET**

Anahtar kelimeler: Node.js, MongoDB, Socket.io, WebRTC, Robomongo, Mongoose

Çağımızın gelişen teknolojileriyle birlikte gerçek zamanlı görüntülü haberleşme uygulamalarının önemi gittikçe artmaktadır. Bu nedenle gerçek zamanlı uygulamalar için geliştirilen diller ve teknolojiler de gün geçtikçe çoğalmaktadır. Biz de bunların arasında önemli bir yere sahip olan Node.js ile gerçek zamanlı görüntülü haberleşme uygulaması yapmaya karar verdik. Gerçek zamanlı olarak hemen hemen her tarayıcıda iletişim kurulmasını Node.js kütüphanesi olan Socket.io, veri tabanı yönetim sistemi olarak MongoDB, gerçek zamanlı uygulama olması için de WebRTC kullanılacaktır. Veri tabanına bağlantı için Mongoose kütüphanesi kullanılacaktır. Sayfa tasarımları için bootstrap, görüntü motoru olarak EJS(Embedded Java Script) kullanılacaktır. Veri tabanı ara yüzü olarak da Robomongo kullanılacaktır

## **BÖLÜM 2. GİRİŞ**

### **2.1. Proje Tanımı**

Socket.io ve WebRTC kullanılarak gerçek zamanlı görüntülü ve yazılı haberleşme uygulaması.

### **2.2. Proje Amacı ve Kapsamı**

Bu proje aynı anda iki ayrı kullanıcının birbirleriyle yazışmasını ve video görüşmesi yapabilmesini amaçlamaktadır. Bu kapsamda Node.js'in Socket.io ve WebRTC modülleri kullanılacaktır. Kullanıcı kaydı ve mesajlaşmaların kaydedilmesi için veri tabanı kullanılacaktır. MongoDB veri tabanında bu veriler tutulacaktır.

Bu projede yazılı ve görüntülü haberleşme sağlanacaktır. Kullanıcı kaydı gerçekleştirildikten ve kullanıcı, oturum açtıktan sonra kendi arkadaş listesindeki kişilerle isteğe bağlı olarak yazılı ya da görüntülü haberleşme yapabilir. Arkadaş listesine yeni kişiler ekleyip çıkarabilir. Kullanıcı istediği an oturumu kapatabilecektir.

### **2.3. Projenin Hedefleri**

Uygulama web tarayıcılarda çalışması amacıyla programlanacaktır. Mobil uygulaması olmayacaktır. Web tabanlı olup sadece tarayıcıda çalışacaktır.

Diğer platformlardan daha hızlı, daha esnek ve kullanılabilirliği daha yüksek bir uygulama geliştirilmek temel hedeftir. Bu bağlamda diğer dillerden daha kullanışlı olan ve geliştirilme süreci daha hızlı, yeni bir teknoloji olan JavaScript framework'ü, sunucu tabanlı çalışan Node.js framework'ü kullanıldı. Gerçek zamanlı görüşme istenildiği için de WebRTC modülleri kullanıldı.



## 2.4. Literatür Taraması

Node.js yeni bir teknoloji olduğu için basılı fazla kaynak bulamadık. Bu yüzden internet araştırmalarına yoğunlaştık. Yazılı ve görsel kaynakları takip ettik. Takip ettiğimiz kaynaklar aşağıda kaynakçada belirtilmektedir.

Bu konuda daha önce yapılmış çalışmalara örnek olarak OnSIP, Bistri, GoInstant gibi web siteleri örnek verilebilir.

### OnSIP

- %100 web tabanlı
- Sesli, görüntü ve yazılı haberleşme uygulaması
- Slack ve Zendesk ile entegre
- Google Chrome eklentisi

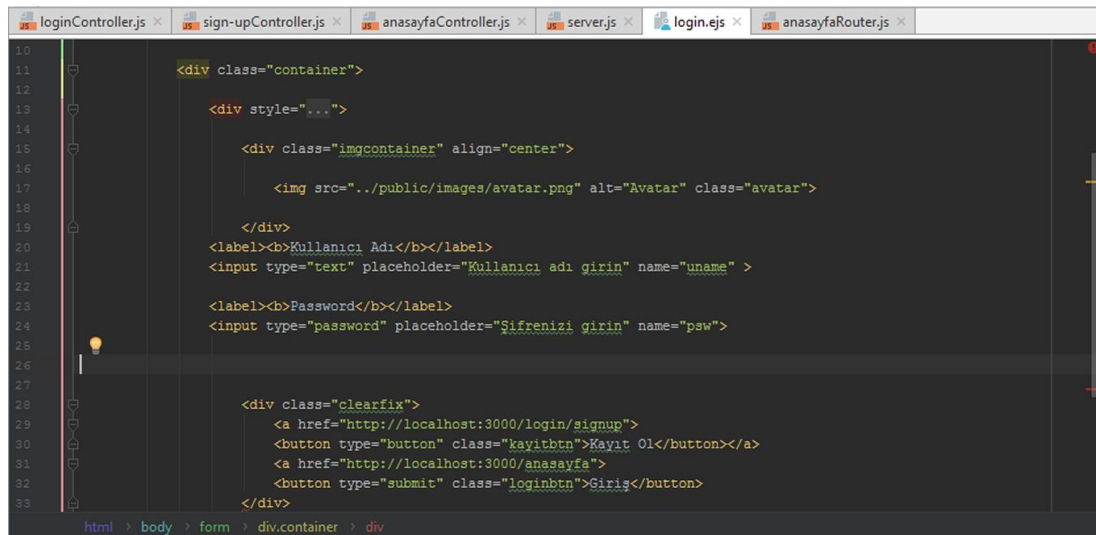
### Skype

- Bütün tarayıcılarla uyumlu çalışmamaktadır.
- Arkadaş listesi oluşturulabilir.
- Sesli, görüntü ve yazılı haberleşme uygulaması
- Linux platformunda WebRTC kullanılmasına karşın Windows, MacOS, web ve mobil platformlarda WebRTC kullanmamaktadır.

## BÖLÜM 3. YÖNTEMLER VE KULLANILAN TEKNOLOJİLER

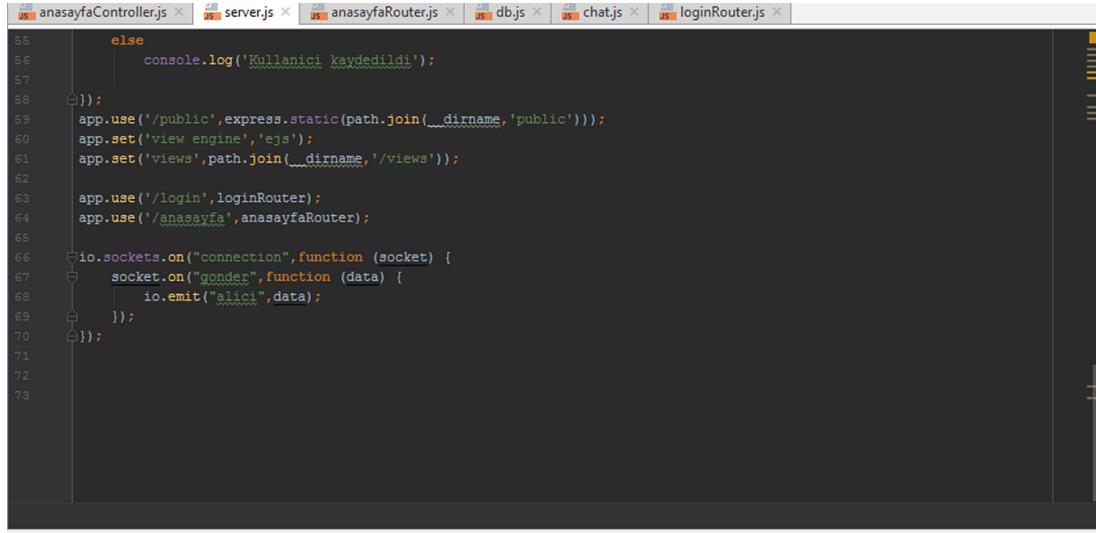
### 3.1. Node.js

Node.js ile sayfaların tasarımını ve sayfa bağlantılarını oluşturduk. Üç adet sayfamız vardır. Bunlar, login, sign-up ve ChatPage sayfalarıdır. Projemizin içindeki bütün sayfaların tasarımını EJS görüntü motoruyla yaptık. Yine bu sayfaların tasarımında yoğunlukla Bootstrap kütüphanelerini kullandık.



Şekil 3.1 : login.ejs

Başlangıç sayfamız server.js'dir. server.js içinde express, path, socket-io kütüphanelerini kullandık. express kütüphanesi sayesinde, kullanıcılara public klasörünün altındaki klasörlere(css, js vb.) erişim izni verildi. EJS görüntü motorunun kullanılması sağlandı. path kütüphanesi, public klasörüne erişebilmek için gerekli yolu gösterir. Kullanıcılar arasında bağlantı kurmak için socket.io kütüphanesi kullanıldı.



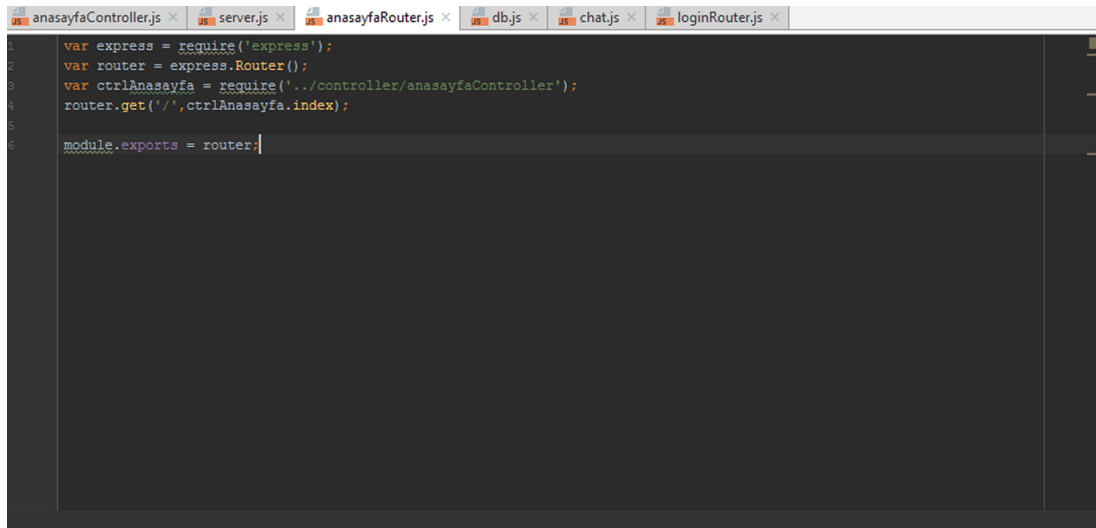
```

55     else
56         console.log('Kullanıcı kaydedildi');
57     });
58 });
59 app.use('/public', express.static(path.join(__dirname, 'public')));
60 app.set('view engine', 'ejs');
61 app.set('views', path.join(__dirname, 'views'));
62
63 app.use('/login', loginRouter);
64 app.use('/anasayfa', anasayfaRouter);
65
66 io.sockets.on("connection", function (socket) {
67     socket.on("gonder", function (data) {
68         io.emit("alici", data);
69     });
70 });
71
72
73

```

Şekil 3.2 : server.ejs router kullanımı

URL'den gelen parametrelere göre oluşturduğumuz router js dosyaları gerekli controller'ları çağırıp ejs dosyalarını ekrana çıktı olarak verir. Örneğin anasayfaRouter.js içinde anasayfaController çağrılır. anasayfaController içinde ise ChatPage.ejs render edilmektedir. Son olarak anasayfaRouter.js içinde modül export edilerek oluşturulan router dışarıya ulaşma imkanı kazanır.

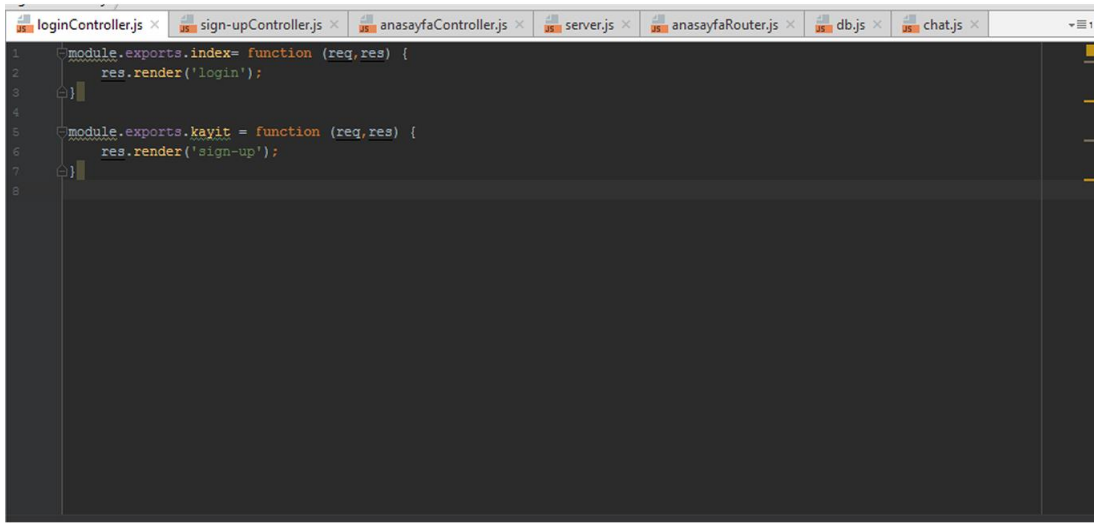


```

1  var express = require('express');
2  var router = express.Router();
3  var ctrlAnasayfa = require('../controller/anasayfaController');
4  router.get('/', ctrlAnasayfa.index);
5
6  module.exports = router;

```

Şekil 3.3 : anasayfaRouter.js



Şekil 3.4 : loginController.js

Uygulamamızın bu aşamasında server.js içinde kendi elimizle eklediğimiz kayıtları sildik ve verileri formlardan almaya başladık. loginController.js içinde indexPost ve kayitPost modüllerini ekledik. kayitPost sign-up.ejs'deki formdan alınan verileri(kullanıcı adı ve şifre) veri tabanına kayıt eder. indexPost modülü ise login.ejs'de forma girilen değerleri veri tabanındaki verilerle karşılaştırmak için kullanılır. Bu aşamada MongoDB'nin findOne( ) metodunu kullanarak eşleştirme işlemlerini yaptık.



Şekil 3.5 : kayitPost.js

```

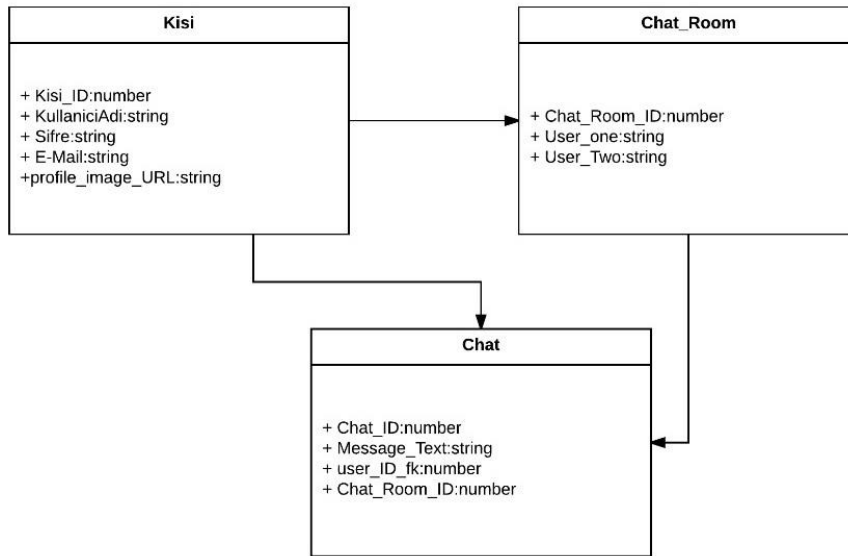
9  module.exports.indexPost= function (req,res) {
10
11
12
13      Kullanici.find({"kullaniciAdi": req.body.kullaniciAdi})
14
15
16      Kullanici.find({"kullaniciAdi": req.body.kullaniciAdi},function (err,docs) {
17          for(var i =0; i<docs.length;i++) {
18
19              if(req.body.sifre=== docs[i].sifre)
20              {
21                  res.render('ChatPage');
22
23              }
24              else
25              {
26                  //res.render('login');
27                  res.send(500,"Kullanıcı Adı ya da parola hatalı");
28              }
29          }
30      });
31  }
32
33  });
34
35

```

Şekil 3.6 : indexPost.js

### 3.2. MongoDB

Projemizde kullanıcıların, chat odalarında hangi kullanıcıların kayıtlı olduğunun ve mesajların tutulması için MongoDB veri tabanı sistemi kullanılmaktadır. Veri tabanımızın collection'ları ve ilişkileri mantıksal açıdan şu şekilde olacaktır.



Şekil 3.7 : Veri tabanı şeması

Veri tabanına ulaşım için öncelikle komut satırı açılır. `cd C:\Program Files\MongoDB\Server\3.4\bin` yazılır ve bin klasörünün altına gelinir. Burada `mongod` yazılır. Burada veri tabanımız bizden komut beklemektedir. Ardından yeni bir komut satırı açılır. Aynı dizinin altına gelinerek bu sefer `mongo` yazılır. Terminal ekranında js komutları yazılarak veri tabanı oluşturulur.

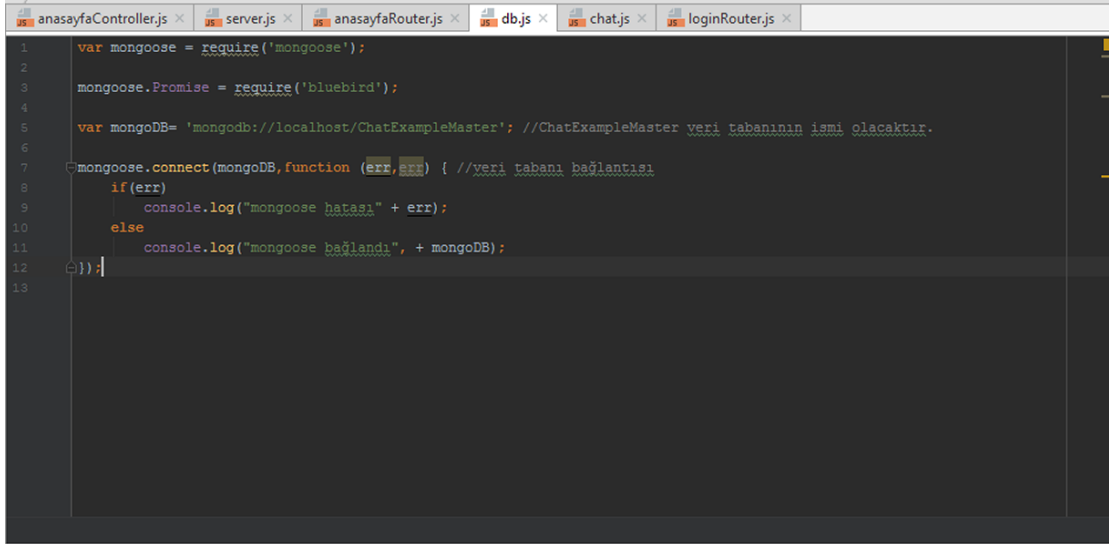
```

Komit Istemi - mongod
name: "Microsoft Windows 8", architecture: "x86_64", version: "6.2 (build 9280
2017-11-10T16:32:43.570+0300 I - [conn31] end connection 127.0.0.1:13616
2017-11-10T16:32:43.571+0300 I - [conn32] end connection 127.0.0.1:13619
2017-11-10T16:32:43.572+0300 I - [conn33] end connection 127.0.0.1:13620
2017-11-10T16:32:43.573+0300 I - [conn34] end connection 127.0.0.1:13621
2017-11-10T16:32:51.298+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13859 #37 (8 connections now open)
2017-11-10T16:32:51.398+0300 I NETWORK [conn37] received client metadata from 1
27.0.0.1:13859 conn37: {< driver: {< name: "nodejs", version: "2.2.33" }, os: { ty
pe: "Windows_NT", name: "win32", architecture: "x64", version: "6.3.9600" }, pla
tform: "Node.js v6.11.3, LE, mongodb-core: 2.1.17" } }
2017-11-10T16:32:51.503+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13862 #38 (9 connections now open)
2017-11-10T16:32:51.558+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13863 #39 (10 connections now open)
2017-11-10T16:32:51.602+0300 I NETWORK [conn37] connection accepted from 127.0
0.1:13864 #40 (11 connections now open)
2017-11-10T16:36:13.074+0300 I - [conn37] end connection 127.0.0.1:13859
2017-11-10T16:36:13.075+0300 I - [conn40] end connection 127.0.0.1:13864
2017-11-10T16:36:13.075+0300 I - [conn38] end connection 127.0.0.1:13862
2017-11-10T16:36:13.075+0300 I - [conn39] end connection 127.0.0.1:13863
2017-11-10T16:36:17.438+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13807 #41 (8 connections now open)
2017-11-10T16:36:17.548+0300 I NETWORK [conn41] received client metadata from 1
27.0.0.1:13807 conn41: {< driver: {< name: "nodejs", version: "2.2.33" }, os: { ty
pe: "Windows_NT", name: "win32", architecture: "x64", version: "6.3.9600" }, pla
tform: "Node.js v6.11.3, LE, mongodb-core: 2.1.17" } }
2017-11-10T16:36:17.670+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13809 #42 (9 connections now open)
2017-11-10T16:36:17.738+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13808 #43 (10 connections now open)
2017-11-10T16:36:17.793+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:13891 #44 (11 connections now open)
2017-11-10T16:37:04.432+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:14129 #45 (12 connections now open)
2017-11-10T16:37:04.433+0300 I NETWORK [conn45] received client metadata from 1
27.0.0.1:14129 conn45: {< application: {< name: "Robomongo", driver: {< name: "Mo
ngodB Internal Client", version: "3.4.3-7-gf394f8c" }, os: { type: "Windows", na
me: "Microsoft Windows 8", architecture: "x86_64", version: "6.2 (build 9280
" } } }
2017-11-10T16:37:04.562+0300 I NETWORK [thread1] connection accepted from 127.0
0.1:14130 #46 (13 connections now open)
2017-11-10T16:37:04.564+0300 I NETWORK [conn46] received client metadata from 1
27.0.0.1:14130 conn46: {< application: {< name: "MongoDB Shell", driver: {< name:
"MongoDB Internal Client", version: "3.4.3-7-gf394f8c" }, os: { type: "Windows",
name: "Microsoft Windows 8", architecture: "x86_64", version: "6.2 (build 9280
" } } }

C:\Windows\system32\cmd.exe - mongo
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. Tüm hakları saklıdır.
C:\Users\yasu>cd ..
C:\Users>cd ..
C:\Program Files\MongoDB\Server\3.4\bin>mongo
connecting to: mongod://127.0.0.1:27017
MongoDB server version: 3.4.9
Server has startup warnings:
2017-11-10T15:10:19.300+0300 I CONTROL [initandlisten] ** WARNING: Access contr
ol is not enabled for the database.
2017-11-10T15:10:19.301+0300 I CONTROL [initandlisten] ** Read and writ
e access to data and configuration is unrestricted.
2017-11-10T15:10:19.303+0300 I CONTROL [initandlisten]
  
```

Şekil 3.8 : mongod ve mongo terminal ekranları

Diğer bir yöntem ise bunu PHPStorm içinde js dosyaları içinde yapmaktır. Biz ikinci yöntemi tercih ettik. Projemizin altında models adında bir klasör oluşturduk. Bu klasörün altında db.js, user.js, chatRoom.js ve chat.js adında dört adet js dosyası oluşturduk.



```

1  var mongoose = require('mongoose');
2
3  mongoose.Promise = require('bluebird');
4
5  var mongoDB= 'mongodb://localhost/ChatExampleMaster'; //ChatExampleMaster veri tabanının ismi olacaktır.
6
7  mongoose.connect(mongoDB,function (err,err) { //veri tabanı bağlantısı
8    if(err)
9      console.log("mongoose hatası" + err);
10   else
11     console.log("mongoose bağlandı", + mongoDB);
12  });
13

```

Şekil 3.9 : db.js

db.js içinde veri tabanına bağlantı için mongoose kütüphanesini kullandık. Veri tabanı için ayrı bir server'ımız olmayıp localhost'ta çalışmaktayız. Veri tabanına bağlantı yaparken asenkron şekilde çalışmak için bluebird kütüphanesini kullandık.

Diğer üç js dosyası içinde veri tabanı collection'larını oluşturduk. Bu dosyalarda yine mongoose ve id alanlarının otomatik olarak artabilmesi için mongoose-auto-increment kütüphanelerini kullandık. Her birinde veri tiplerinin belli olması için bir schema oluşturduk. Bu schema'ları nesnelere atadık ve son olarak bu nesneleri dışarıya aktarmak için modülleri export ettik.

```

1  var mongoose = require('mongoose');
2  var autoIncrement = require('mongoose-auto-increment');
3  var Schema = mongoose.Schema; //Kullanımı kolaylaştırmak için Schema adlı değişkene aktarıyoruz.
4
5  autoIncrement.initialize(mongoose.connection);
6  //Kullanacağımız Schema db'ye aktarılacak olan modelin veri tipini belirler.
7  var chatSchema = new Schema({
8    messageText : String,
9    userIDFK : Number,
10   chatRoomIDFK : Number
11 }, {collection: 'chat'}); //Normalde chats diye oluşturur tabloyu. chat olarak oluşturmasını istersek sonuna bunu ekleriz.
12
13 chatSchema.plugin(autoIncrement.plugin, 'Chat');
14 var Chat = mongoose.model('Chat', chatSchema); //Yeni bir model oluşturduk.
15 module.exports = Chat; //Modülü export ettik. ChatRoom'ı server.js dosyamızın içinde kullanacağız.

```

Şekil 3.10: chat.js

Ana js dosyamız olan server.js içinde ise models klasörünün altındaki js dosyalarını kütüphane olarak alan User, ChatRoom ve Chat nesnelerini oluşturduk. Bu nesnelerden yeni nesneler türettik. Yeni nesnelerin içine veri girdik ve kaydettik.

```

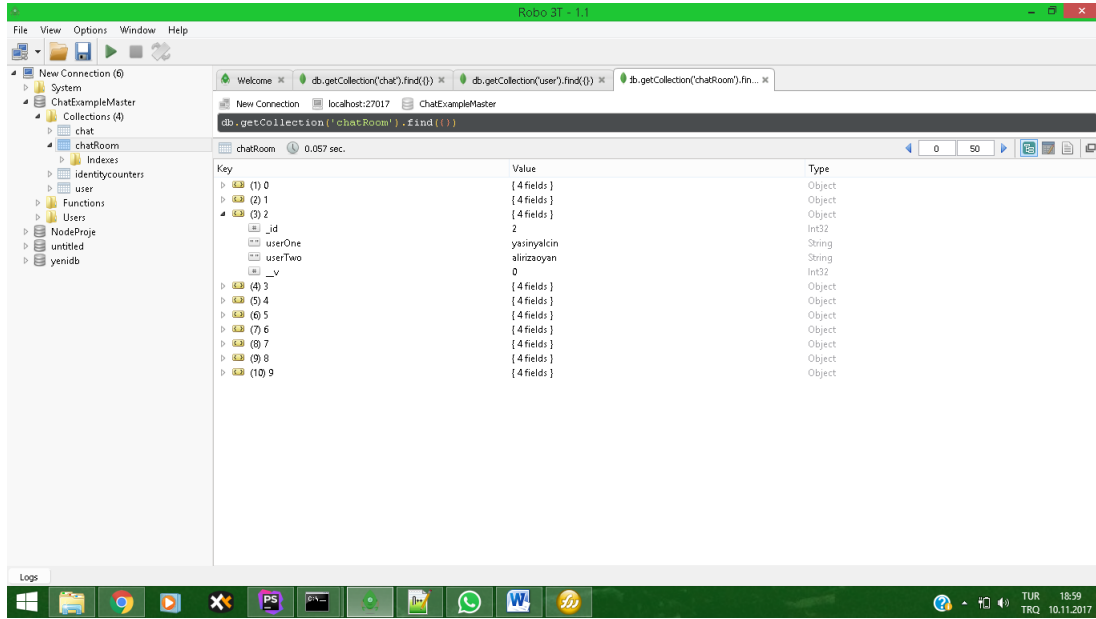
40  var newChatRoom = new ChatRoom({
41    userOne : 'yasinvalcin',
42    userTwo : 'alirizaogyan'
43  });
44  var newUser = new User({ //Yeni kayıt giriliyor. Schema'ya göre girilir.
45    kullanıcıAdi : 'ahmetdervis',
46    password : '1459876',
47    email : 'yasin@gmail.com',
48    profileImageUrl : 'profilResmiYasin'
49  });
50
51
52
53  newChat.save(function (err) {
54    if(err)
55    {
56      console.log('Chat kaydedilmesinde hata olustu ' + err);
57    }
58    else
59      console.log('Chat oluşturuldu ve kaydedildi');
60  });
61
62  newChatRoom.save(function (err) {
63    if(err)
64      console.log('ChatRoom kaydedilmesinde hata olustu ' + err);
65    else
66      console.log('ChatRoom oluşturuldu ve kaydedildi');
67  });
68  //callback for save()

```

Şekil 3.11 : server.js mongodb kullanımı

Veriler kaydedildikten sonra bu verilerin kayıtlarının başarılı olup olmadığını görmek için iki yol vardır. Ya komut satırı açılıp görmek için gerekli js kodları yazılır ya da Robomongo IDE'si ile collection'lar görülür. Biz ikinci yolu tercih ettik ve Robomongo'dan kayıtlarımızın başarılı olduğunu gördük.





Şekil 3.12 : Robomongo

### 3.3 WebRTC

Projemizin temel gereklerinden biri kullanıcıların gerçek zamanlı olarak karşılıklı görüşmesi idi. Bu hedefimizi gerçekleştirmek için WebRTC denen bir teknolojiyi kullanmayı seçtik. WebRTC kullanma sebebimiz, WebRTC'nin kullanılabilmesi için sadece internet tarayıcınızdan ilgili sitenin adresine girmenizin yeterli olmasıdır. Tarayıcınıza hiçbir üçüncü parti uygulama veya plugin kurmanıza gerek yok. Ayrıca WebRTC'nin kullandığı API'lerin hepsi kullanılan popüler internet tarayıcıların içinde geliyor. Bu görevi yerine getirirken bazı kütüphaneler ve fonksiyonlar kullanılır. Bunlardan biri socket.io kütüphanesidir. Bu sunucu tarafında olan bir dosyadır. İkincisi bir fonksiyondur. Adı getusermedia()'dır. İnternet tarayıcısı ile yüklü gelir. Bunun ile internet tarayıcısından video ve ses bilgileri alınır. Bunlar 2 müşteri arasında daha sonra bahsedeceğim yöntem ile birbirine aktarılır.

```

1  // use strict;
2
3  var os = require('os');
4  var nodeStatic = require('node-static');
5  var http = require('http');
6  var socketIO = require('socket.io');
7
8  var fileServer = new(nodeStatic.Server)();
9  var app = http.createServer(function(req, res) {
10     fileServer.serve(req, res);
11  }).listen(8080);
12
13  var io = socketIO.listen(app);
14  io.sockets.on('connection', function(socket) {
15
16     // convenience function to log server messages on the client
17     function log() {
18         var array = ['Message from server:'];
19         array.push.apply(array, arguments);
20         socket.emit('log', array);
21     }
22
23     socket.on('message', function(message) {
24         log('Client said: ', message);
25         // for a real app, would be room-only (not broadcast)
26         socket.broadcast.emit('message', message);
27     });
28
29     socket.on('create or join', function(room) {
30         log('Received request to create or join room ' + room);
31
32         var numClients = io.sockets.sockets.length;
33         log('Room ' + room + ' now has ' + numClients + ' client(s)');
34
35         if (numClients === 1) {
36             socket.join(room);
37             log('Client ID ' + socket.id + ' created room ' + room);
38             socket.emit('created', room, socket.id);
39         } else if (numClients === 2) {
40             log('Client ID ' + socket.id + ' joined room ' + room);
41             // io.sockets.in(room).emit('join', room);
42             socket.join(room);
43             socket.emit('joined', room, socket.id);
44             io.sockets.in(room).emit('ready', room);
45             socket.broadcast.emit('ready', room);
46         } else { // max two clients
47             socket.emit('full', room);
48         }
49     });
50
51     // ...
52
53     // ...
54
55     // ...
56
57     // ...
58
59     // ...
60
61     // ...
62
63     // ...
64
65     // ...
66
67     // ...
68
69     // ...
70
71     // ...
72
73     // ...
74
75     // ...
76
77     // ...
78
79     // ...
80
81     // ...
82
83     // ...
84
85     // ...
86
87     // ...
88
89     // ...
90
91     // ...
92
93     // ...
94
95     // ...
96
97     // ...
98
99     // ...
100
101     // ...
102
103     // ...
104
105     // ...
106
107     // ...
108
109     // ...
110
111     // ...
112
113     // ...
114
115     // ...
116
117     // ...
118
119     // ...
120
121     // ...
122
123     // ...
124
125     // ...
126
127     // ...
128
129     // ...
130
131     // ...
132
133     // ...
134
135     // ...
136
137     // ...
138
139     // ...
140
141     // ...
142
143     // ...
144
145     // ...
146
147     // ...
148
149     // ...
150
151     // ...
152
153     // ...
154
155     // ...
156
157     // ...
158
159     // ...
160
161     // ...
162
163     // ...
164
165     // ...
166
167     // ...
168
169     // ...
170
171     // ...
172
173     // ...
174
175     // ...
176
177     // ...
178
179     // ...
180
181     // ...
182
183     // ...
184
185     // ...
186
187     // ...
188
189     // ...
190
191     // ...
192
193     // ...
194
195     // ...
196
197     // ...
198
199     // ...
200
201     // ...
202
203     // ...
204
205     // ...
206
207     // ...
208
209     // ...
210
211     // ...
212
213     // ...
214
215     // ...
216
217     // ...
218
219     // ...
220
221     // ...
222
223     // ...
224
225     // ...
226
227     // ...
228
229     // ...
230
231     // ...
232
233     // ...
234
235     // ...
236
237     // ...
238
239     // ...
240
241     // ...
242
243     // ...
244
245     // ...
246
247     // ...
248
249     // ...
250
251     // ...
252
253     // ...
254
255     // ...
256
257     // ...
258
259     // ...
260
261     // ...
262
263     // ...
264
265     // ...
266
267     // ...
268
269     // ...
270
271     // ...
272
273     // ...
274
275     // ...
276
277     // ...
278
279     // ...
280
281     // ...
282
283     // ...
284
285     // ...
286
287     // ...
288
289     // ...
290
291     // ...
292
293     // ...
294
295     // ...
296
297     // ...
298
299     // ...
300
301     // ...
302
303     // ...
304
305     // ...
306
307     // ...
308
309     // ...
310
311     // ...
312
313     // ...
314
315     // ...
316
317     // ...
318
319     // ...
320
321     // ...
322
323     // ...
324
325     // ...
326
327     // ...
328
329     // ...
330
331     // ...
332
333     // ...
334
335     // ...
336
337     // ...
338
339     // ...
340
341     // ...
342
343     // ...
344
345     // ...
346
347     // ...
348
349     // ...
350
351     // ...
352
353     // ...
354
355     // ...
356
357     // ...
358
359     // ...
360
361     // ...
362
363     // ...
364
365     // ...
366
367     // ...
368
369     // ...
370
371     // ...
372
373     // ...
374
375     // ...
376
377     // ...
378
379     // ...
380
381     // ...
382
383     // ...
384
385     // ...
386
387     // ...
388
389     // ...
390
391     // ...
392
393     // ...
394
395     // ...
396
397     // ...
398
399     // ...
400
401     // ...
402
403     // ...
404
405     // ...
406
407     // ...
408
409     // ...
410
411     // ...
412
413     // ...
414
415     // ...
416
417     // ...
418
419     // ...
420
421     // ...
422
423     // ...
424
425     // ...
426
427     // ...
428
429     // ...
430
431     // ...
432
433     // ...
434
435     // ...
436
437     // ...
438
439     // ...
440
441     // ...
442
443     // ...
444
445     // ...
446
447     // ...
448
449     // ...
450
451     // ...
452
453     // ...
454
455     // ...
456
457     // ...
458
459     // ...
460
461     // ...
462
463     // ...
464
465     // ...
466
467     // ...
468
469     // ...
470
471     // ...
472
473     // ...
474
475     // ...
476
477     // ...
478
479     // ...
480
481     // ...
482
483     // ...
484
485     // ...
486
487     // ...
488
489     // ...
490
491     // ...
492
493     // ...
494
495     // ...
496
497     // ...
498
499     // ...
500
501     // ...
502
503     // ...
504
505     // ...
506
507     // ...
508
509     // ...
510
511     // ...
512
513     // ...
514
515     // ...
516
517     // ...
518
519     // ...
520
521     // ...
522
523     // ...
524
525     // ...
526
527     // ...
528
529     // ...
530
531     // ...
532
533     // ...
534
535     // ...
536
537     // ...
538
539     // ...
540
541     // ...
542
543     // ...
544
545     // ...
546
547     // ...
548
549     // ...
550
551     // ...
552
553     // ...
554
555     // ...
556
557     // ...
558
559     // ...
560
561     // ...
562
563     // ...
564
565     // ...
566
567     // ...
568
569     // ...
570
571     // ...
572
573     // ...
574
575     // ...
576
577     // ...
578
579     // ...
580
581     // ...
582
583     // ...
584
585     // ...
586
587     // ...
588
589     // ...
590
591     // ...
592
593     // ...
594
595     // ...
596
597     // ...
598
599     // ...
600
601     // ...
602
603     // ...
604
605     // ...
606
607     // ...
608
609     // ...
610
611     // ...
612
613     // ...
614
615     // ...
616
617     // ...
618
619     // ...
620
621     // ...
622
623     // ...
624
625     // ...
626
627     // ...
628
629     // ...
630
631     // ...
632
633     // ...
634
635     // ...
636
637     // ...
638
639     // ...
640
641     // ...
642
643     // ...
644
645     // ...
646
647     // ...
648
649     // ...
650
651     // ...
652
653     // ...
654
655     // ...
656
657     // ...
658
659     // ...
660
661     // ...
662
663     // ...
664
665     // ...
666
667     // ...
668
669     // ...
670
671     // ...
672
673     // ...
674
675     // ...
676
677     // ...
678
679     // ...
680
681     // ...
682
683     // ...
684
685     // ...
686
687     // ...
688
689     // ...
690
691     // ...
692
693     // ...
694
695     // ...
696
697     // ...
698
699     // ...
700
701     // ...
702
703     // ...
704
705     // ...
706
707     // ...
708
709     // ...
710
711     // ...
712
713     // ...
714
715     // ...
716
717     // ...
718
719     // ...
720
721     // ...
722
723     // ...
724
725     // ...
726
727     // ...
728
729     // ...
730
731     // ...
732
733     // ...
734
735     // ...
736
737     // ...
738
739     // ...
740
741     // ...
742
743     // ...
744
745     // ...
746
747     // ...
748
749     // ...
750
751     // ...
752
753     // ...
754
755     // ...
756
757     // ...
758
759     // ...
760
761     // ...
762
763     // ...
764
765     // ...
766
767     // ...
768
769     // ...
770
771     // ...
772
773     // ...
774
775     // ...
776
777     // ...
778
779     // ...
780
781     // ...
782
783     // ...
784
785     // ...
786
787     // ...
788
789     // ...
790
791     // ...
792
793     // ...
794
795     // ...
796
797     // ...
798
799     // ...
800
801     // ...
802
803     // ...
804
805     // ...
806
807     // ...
808
809     // ...
810
811     // ...
812
813     // ...
814
815     // ...
816
817     // ...
818
819     // ...
820
821     // ...
822
823     // ...
824
825     // ...
826
827     // ...
828
829     // ...
830
831     // ...
832
833     // ...
834
835     // ...
836
837     // ...
838
839     // ...
840
841     // ...
842
843     // ...
844
845     // ...
846
847     // ...
848
849     // ...
850
851     // ...
852
853     // ...
854
855     // ...
856
857     // ...
858
859     // ...
860
861     // ...
862
863     // ...
864
865     // ...
866
867     // ...
868
869     // ...
870
871     // ...
872
873     // ...
874
875     // ...
876
877     // ...
878
879     // ...
880
881     // ...
882
883     // ...
884
885     // ...
886
887     // ...
888
889     // ...
890
891     // ...
892
893     // ...
894
895     // ...
896
897     // ...
898
899     // ...
900
901     // ...
902
903     // ...
904
905     // ...
906
907     // ...
908
909     // ...
910
911     // ...
912
913     // ...
914
915     // ...
916
917     // ...
918
919     // ...
920
921     // ...
922
923     // ...
924
925     // ...
926
927     // ...
928
929     // ...
930
931     // ...
932
933     // ...
934
935     // ...
936
937     // ...
938
939     // ...
940
941     // ...
942
943     // ...
944
945     // ...
946
947     // ...
948
949     // ...
950
951     // ...
952
953     // ...
954
955     // ...
956
957     // ...
958
959     // ...
960
961     // ...
962
963     // ...
964
965     // ...
966
967     // ...
968
969     // ...
970
971     // ...
972
973     // ...
974
975     // ...
976
977     // ...
978
979     // ...
980
981     // ...
982
983     // ...
984
985     // ...
986
987     // ...
988
989     // ...
990
991     // ...
992
993     // ...
994
995     // ...
996
997     // ...
998
999     // ...
1000
1001     // ...
1002
1003     // ...
1004
1005     // ...
1006
1007     // ...
1008
1009     // ...
1010
1011     // ...
1012
1013     // ...
1014
1015     // ...
1016
1017     // ...
1018
1019     // ...
1020
1021     // ...
1022
1023     // ...
1024
1025     // ...
1026
1027     // ...
1028
1029     // ...
1030
1031     // ...
1032
1033     // ...
1034
1035     // ...
1036
1037     // ...
1038
1039     // ...
1040
1041     // ...
1042
1043     // ...
1044
1045     // ...
1046
1047     // ...
1048
1049     // ...
1050
1051     // ...
1052
1053     // ...
1054
1055     // ...
1056
1057     // ...
1058
1059     // ...
1060
1061     // ...
1062
1063     // ...
1064
1065     // ...
1066
1067     // ...
1068
1069     // ...
1070
1071     // ...
1072
1073     // ...
1074
1075     // ...
1076
1077     // ...
1078
1079     // ...
1080
1081     // ...
1082
1083     // ...
1084
1085     // ...
1086
1087     // ...
1088
1089     // ...
1090
1091     // ...
1092
1093     // ...
1094
1095     // ...
1096
1097     // ...
1098
1099     // ...
1100
1101     // ...
1102
1103     // ...
1104
1105     // ...
1106
1107     // ...
1108
1109     // ...
1110
1111     // ...
1112
1113     // ...
1114
1115     // ...
1116
1117     // ...
1118
1119     // ...
1120
1121     // ...
1122
1123     // ...
1124
1125     // ...
1126
1127     // ...
1128
1129     // ...
1130
1131     // ...
1132
1133     // ...
1134
1135     // ...
1136
1137     // ...
1138
1139     // ...
1140
1141     // ...
1142
1143     // ...
1144
1145     // ...
1146
1147     // ...
1148
1149     // ...
1150
1151     // ...
1152
1153     // ...
1154
1155     // ...
1156
1157     // ...
1158
1159     // ...
1160
1161     // ...
1162
1163     // ...
1164
1165     // ...
1166
1167     // ...
1168
1169     // ...
1170
1171     // ...
1172
1173     // ...
1174
1175     // ...
1176
1177     // ...
1178
1179     // ...
1180
1181     // ...
1182
1183     // ...
1184
1185     // ...
1186
1187     // ...
1188
1189     // ...
1190
1191     // ...
1192
1193     // ...
1194
1195     // ...
1196
1197     // ...
1198
1199     // ...
1200
1201     // ...
1202
1203     // ...
1204
1205     // ...
1206
1207     // ...
1208
1209     // ...
1210
1211     // ...
1212
1213     // ...
1214
1215     // ...
1216
1217     // ...
1218
1219     // ...
1220
1221     // ...
1222
1223     // ...
1224
1225     // ...
1226
1227     // ...
1228
1229     // ...
1230
1231     // ...
1232
1233     // ...
1234
1235     // ...
1236
1237     // ...
1238
1239     // ...
1240
1241     // ...
1242
1243     // ...
1244
1245     // ...
1246
1247     // ...
1248
1249     // ...
1250
1251     // ...
1252
1253     // ...
1254
1255     // ...
1256
1257     // ...
1258
1259     // ...
1260
1261     // ...
1262
1263     // ...
1264
1265     // ...
1266
1267     // ...
1268
1269     // ...
1270
1271     // ...
1272
1273     // ...
1274
1275     // ...
1276
1277     // ...
1278
1279     // ...
1280
1281     // ...
1282
1283     // ...
1284
1285     // ...
1286
1287     // ...
1288
1289     // ...
1290
1291     // ...
1292
1293     // ...
1294
1295     // ...
1296
1297     // ...
1298
1299     // ...
1300
1301     // ...
1302
1303     // ...
1304
1305     // ...
1306
1307     // ...
1308
1309     // ...
1310
1311     // ...
1312
1313     // ...
1314
1315     // ...
1316
1317     // ...
1318
1319     // ...
1320
1321     // ...
1322
1323     // ...
1324
1325     // ...
1326
1327     // ...
1328
1329     // ...
1330
1331     // ...
1332
1333     // ...
1334
1335     // ...
1336
1337     // ...
1338
1339     // ...
1340
1341     // ...
1342
1343     // ...
1344
1345     // ...
1346
1347     // ...
1348
1349     // ...
1350
1351     // ...
1352
1353     // ...
1354
1355     // ...
1356
1357     // ...
1358
1359     // ...
1360
1361     // ...
1362
1363     // ...
1364
1365     // ...
1366
1367     // ...
1368
1369     // ...
1370
1371     // ...
1372
1373     // ...
1374
1375     // ...
1376
1377     // ...
1378
1379     // ...
1380
1381     // ...
1382
1383     // ...
1384
1385     // ...
1386
1387     // ...
1388
1389     // ...
1390
1391     // ...
1392
1393     // ...
1394
1395     // ...
1396
1397     // ...
1398
1399     // ...
1400
1401     // ...
1402
1403     // ...
1404
1405     // ...
1406
1407     // ...
1408
1409     // ...
1410
1411     // ...
1412
1413     // ...
1414
1415     // ...
1416
1417     // ...
1418
1419     // ...
1420
1421     // ...
1422
1423     // ...
1424
1425     // ...
1426
1427     // ...
1428
1429     // ...
1430
1431     // ...
1432
1433     // ...
1434
1435     // ...
1436
1437     // ...
1438
1439     // ...
1440
1441     // ...
1442
1443     // ...
1444
1445     // ...
1446
1447     // ...
1448
1449     // ...
1450
1451     // ...
1452
1453     // ...
1454
1455     // ...
1456
1457     // ...
1458
1459     // ...
1460
1461     // ...
1462
1463     // ...
1464
1465     // ...
1466
1467     // ...
1468
1469     // ...
1470
1471     // ...
1472
1473     // ...
1474
1475     // ...
1476
1477     // ...
1478
1479     // ...
1480
1481     // ...
1482
1483     // ...
1484
1485     // ...
1486
1487     // ...
1488
1489     // ...
1490
1491     // ...
1492
1493     // ...
1494
1495     // ...
1496
1497     // ...
1498
1499     // ...
1500
1501     // ...
1502
1503     // ...
1504
1505     // ...
1506
1507     // ...
1508
1509     // ...
1510
1511     // ...
1512
1513     // ...
1514
1515     // ...
1516
1517     // ...
1518
1519     // ...
1520
1521     // ...
1522
1523     // ...
1524
1525     // ...
1526
1527     // ...
1528
1529     // ...
1530
1531     // ...
1532
1533     // ...
1534
1535     // ...
1536
1537     // ...
1538
1539     // ...
1540
1541     // ...
1542
1543     // ...
1544
1545     // ...
1546
1547     // ...
1548
1549     // ...
1550
1551     // ...
1552
1553     // ...
1554
1555     // ...
1556
1557     // ...
1558
1559     // ...
1560
1561     // ...
1562
1563     // ...
1564
1565     // ...
1566
1567     // ...
1568
1569     // ...
1570
1571     // ...
1572
1573     // ...
1574
1575     // ...
1576
1577     // ...
1578
1579     // ...
1580
1581     // ...
1582
1583     // ...
1584
1585     // ...
1586
1587     // ...
1588
1589     // ...
1590
1591     // ...
1592
1593     // ...
1594
1595     // ...
1596
1597     // ...
1598
1599     // ...
1600
1601     // ...
1602
1603     // ...
1604
1605     // ...
1606
1607     // ...
1608
1609     // ...
1610
1611     // ...
1612
1613     // ...
1614
1615     // ...
1616
1617     // ...
1618
1619     // ...
1620
1621     // ...
1622
1623     // ...
1624
1625     // ...
1626
1627     // ...
1628
1629     // ...
1630
1631     // ...
1632
1633     // ...
1634
1635     // ...
1636
1637     // ...
1638
1639     // ...
1640
1641     // ...
1642
1643     // ...
1644
1645     // ...
1646
1647     // ...
1648
1649     // ...
1650
1651     // ...
1652
1653     // ...
1654
1655     // ...
1656
1657     // ...
1658
1659     // ...
1660
1661     // ...
1662
1663     // ...
1664
1665     // ...
1666
1667     // ...
1668
1669     // ...
1670
1671     // ...
1672
1673     // ...
1674
1675     // ...
1676
1677     // ...
1678
1679     // ...
1680
1681     // ...
1682
1683     // ...
1684
1685     // ...
1686
1687     // ...
1688
1689     // ...
1690
1691     // ...
1692
1693     // ...
1694
1695     // ...
1696
1697     // ...
1698
1699     // ...
1700
1701     // ...
1702
1703     // ...
1704
1705     // ...
1706
1707     // ...
1708
1709     // ...
1710
1711     // ...
1712
1713     // ...
1714
1715     // ...
1716
1717     // ...
1718
1719     // ...
1720
1721     // ...
1722
1723     // ...
1724
1725     // ...
1726
1727     // ...
1728
1729     // ...
1730
1731     // ...
1732
1733     // ...
1734
1735     // ...
1736
1737     // ...
1738
1739     // ...
1740
1741     // ...
1742
1743     // ...
1744
1745     // ...
1746
1747     // ...
1748
1749     // ...
1750
1751     // ...
1752
1753     // ...
1754
1755     // ...
1756
1757     // ...
1758
1759     // ...
1760
1761     // ...
1762
1763     // ...
1764
1765     // ...
1766
1767     // ...
1768
1769     // ...
1770
1771     // ...
1772
1773     // ...
1774
1775     // ...
1776
1777     // ...
1778
1779     // ...
1780
1781     // ...
1782
1783     // ...
1784
1785     // ...
1786
1787     // ...
1788
1789     // ...
1790
1791     // ...
1792
1793     // ...
1794
1795     // ...
1796
1797     // ...
1798
1799     // ...
1800
1801     // ...
1802
1803     // ...
1804
1805     // ...
1806
1807     // ...
1808
1809     // ...
1810
1811     // ...
1812
1813     // ...
1814
1815     // ...
1816
1817     // ...
1818
1819     // ...
1820
1821     // ...
1822
1823     // ...
1824
1825     // ...
1826
1827     // ...
1828
1829     // ...
1830
1831     // ...
1832
1833     // ...
1834
1835     // ...
1836
1837     // ...
1838
1839     // ...
1840
1841     // ...
1842
1843     // ...
1844
1845     // ...
1846
1847     // ...
1848
1849     // ...
1850
1851     // ...
1852
1853     // ...
1854
1855     // ...
1856
1857     // ...
1858
1859     // ...
1860
1861     // ...
1862
1863     // ...
1864
1865     // ...
1866
1867     // ...
1868
1869     // ...
1870
1871     // ...
1872
1873     // ...
1874
1875     // ...
1876
1877     // ...
1878
1879     // ...
1880
1881     // ...
1882
1883     // ...
1884
1885     // ...
1886
1887     // ...
1888
1889     // ...
1890
1891     // ...
1892
1893     // ...
1894
1895     // ...
1896
1897     // ...
1898
1899     // ...
1900
1901     // ...
1902
1903     // ...
1904
1905     // ...
1906
1907     // ...
1908
1909     // ...
1910
1911     // ...
1912
1913     // ...
1914
1915     // ...
1916
1917     // ...
1918
1919     // ...
1920
1921     // ...
1922
1923     // ...
1924
1925     // ...
1926
1927     // ...
1928
1929     // ...
1930
1931     // ...
1932
1933     // ...
1934
1935     // ...
1936
1937     // ...
1938
1939     // ...
1940
1941     // ...
1942
1943     // ...
1944
1945     // ...
1946
1947     // ...
1948
1949     // ...
1950
1951     // ...
1952
1953     // ...
1954
1955     // ...
1956
1957     // ...
1958
1959     // ...
1960
1961     // ...
1962
1963     // ...
1964
1965     // ...
1966
1967     // ...
1968
1969     // ...
1970
1971     // ...
1972
1973     // ...
1974
1975     // ...
1976
1977     // ...
1978
1979     // ...
1980
1981     // ...
1982
1983     // ...
1984
1985     // ...
1986
1987     // ...
1988
1989     // ...
1990
1991     // ...
1992
1993     // ...
1994
1995     // ...
1996
1997     // ...
1998
1999     // ...
2000
2001     // ...
2002
2003     // ...
2004
2005     // ...
2006
2007     // ...
2008
2009     // ...
2010
2011     // ...
2012
2013     // ...
2014
2015     // ...
2016
201
```

Şekil 2.12’ de görünen kod normal kodumuzun sadece başlangıcıdır. Bu kodun görevi daha önce bahsettiğimiz 2 client arasında görüşme başlamadan önce aradaki haberleşme için gerekli ayarların karşılıklı olarak iletilmesidir. Ardından gelen süreç video ve ses aktarımında kullanılan çözücülerin transfer edilmesidir. Bu çözücüler olmadan karşı taraftan gelen video ve sesin hangi formatta geldiği anlaşılamaz ve veriler aktarılsa bile karşı taraf bu veriyi anlayamayacağı için görüşme yapılamaz. Bundan sonra karşılıklı olarak veriler gönderilmeye başlanır ve görüşme başlamış olur.

## BÖLÜM 4. GELİŞTİRİLEN SİSTEMİN ANLATIMI

Uygulamanın giriş sayfasında kullanıcıdan kullanıcı adı ve parola istenerek giriş yapması istenecektir. Eğer kullanıcı kayıtlı değilse Kayıt Ol butonuna tıklayarak kayıt olma sayfasına yönlendirilecektir. Bu sayfada kullanıcıdan bir kullanıcı adı ve parola belirlenmesi istenecektir. Başarılı bir şekilde kayıt gerçekleştirildikten sonra tekrar giriş sayfasına yönlendirilecek ve ardından belirlenen kullanıcı adı ve parolayla sisteme giriş yapılacaktır.

Oturum açıldıktan sonra ana sayfamız olan mesajlaşma sayfası ekrana gelecektir. Bu ekranda solda arkadaş listemiz, orta panelde ise seçtiğimiz kişiyle mesajlaşmalarımız ve en altta mesaj kutusu yer alacaktır.

Görüntülü görüşme kısmı ise şu şekilde olacaktır. Kullanıcı adı girildikten sonra karşı tarafın id numarası alınacak ve sisteme girilecektir. Karşı taraf bizim id'mizi otomatik olarak görecektir. Böylece kullanıcılar eşleşecek ve görüntülü görüşme başlayacaktır.

Uygulamanın giriş sayfasında login.ejs açılacaktır. İki adet metin kutusu belirecektir. Burada kullanıcı adı ve parola girilecektir. Eğer Kayıt Ol butonuna tıklanırsa sign-up.ejs açılacaktır. Burada üç adet metin kutusu belirecektir. Kullanıcı adı, şifre, şifre yeniden yazıldıktan sonra kayıt oluşturulur. Eğer aynı kullanıcı adı daha önce alınmışsa uyarı verecek ve kullanıcı kaydı yapmayacaktır.

Görüntülü görüşme için ise index.html açılacaktır. Öncelikle kamera ve mikrofonu kullanmak için izin alınır. Ardından yine login işlemi yapılacak, name ve peer id kısımları girilecektir.

## BÖLÜM 5. SONUÇLAR VE TARTIŞMA

Yapacağımız proje geniş bir proje olduğu ve araştırılması gereken fazla konu olduğu için konularımızı üçe ayırdık. Utku WebRTC, Yasin Robomongo ile MongoDB veri tabanı geliştirme, Ali Rıza Node.js ve Socket.io uygulamaları ile ilgili araştırmalarda bulundu. Ali Rıza sayfa tasarımlarını, sayfa geçişlerini ve socket.io'nun projeye dahil edilmesini sağladı. Yasin veri tabanını tasarladı ve projeye ekledi. Utku WebRTC modüllerini yazdı.

Projemizde tüm bunları yaptık ve bundan sonra nasıl ilerleyeceğimiz hakkında yol haritası belirledik. Sonuçta kullanıcı sayfaya ilk olarak geldiğinde kullanıcı adı ve parolasıyla giriş yapacak. Eğer daha önce kaydolmadıysa onu kayıt sayfasına yönlendireceğiz. Kaydolduktan sonra tekrar login ekranına dönen kullanıcı, kullanıcı adı ve parolasını girdikten sonra ana sayfamız olan chat sayfasına yönlendirilecektir. Chat sayfasında ise kişinin arkadaşları, kimle mesajlaşacağı, atacağı mesajların olduğu mesaj kutusu gibi ekranlar olacaktır. Sisteme kaydolun kullanıcı veri tabanına kaydedilecek, bu sayede bilgileri silinmeyecektir. Projeye eklediğimiz WebRTC modülleri ile gerçek zamanlı görüntülü görüşme yapılmaya başlanacaktır.

## KAYNAKLAR

- [1] [www.jskoleji.com](http://www.jskoleji.com)
- [2] <https://socket.io>
- [3] <https://nodejs.org>
- [4] <https://github.com/dwyl/learn-WebRTC>
- [5] <https://codelabs.developers.google.com/codelabs/webrtc-web/#0>
- [6] <http://io13webrtc.appspot.com/#1>
- [7] <https://www.html5rocks.com/en/tutorials/webrtc/basics/>
- [8] <https://www.mongodb.com/>
- [9] <https://www.sitepoint.com/webrtc-video-chat-application-peerjs/>

**Projenin Github adresi:**  
**<https://github.com/alirizaoyan1/chatexample>**

## ÖZGEÇMİŞ

Yasin YALÇIN, 05.11.1995'te Nevşehir'de doğdu. İlk, orta ve lise eğitimini İstanbul'da tamamladı. 2013 yılında Handan Hayrettin Yelkikanat Anadolu Teknik ve Endüstri Meslek Lisesi'nden mezun oldu. 2013 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği bölümünü kazandı. 2016 yılında Eflatun Yazılım'da yazılım stajını Unity oyun programlama konusunda yapmıştır. SAÜ Bilgisayar Mühendisliği bölümünde eğitimine hala devam etmektedir.

Utku Akgüngör, 11.03.1996'da Bursa'da doğdu. İlk, orta ve lise eğitimini Bursa'da tamamladı. 2014 yılında Bursa Hürriyet Anadolu Lisesi'nden mezun oldu. 2014 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği bölümünü kazandı. 2017 yılında Federal-Mogul'da donanım stajını yapmıştır. SAÜ Bilgisayar Mühendisliği bölümünde eğitimine hala devam etmektedir.

Ali Rıza OYAN, 29.06.1994'te İzmir'de doğdu. İlk, orta ve lise eğitimini Bornova'da tamamladı. 2012 yılında Bornova Kız Teknik ve Anadolu Lisesi'nden mezun oldu. 2014 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği bölümünü kazandı. SAÜ Bilgisayar Mühendisliği bölümünde eğitimine hala devam etmektedir.

## BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI

KONU :

ÖĞRENCİLER (Öğrenci No/Ad/Soyad):

Değerlendirme Konusu	İstenenler	Not Aralığı	Not
<b>Yazılı Çalışma</b>			
<b>Çalışma klavuza uygun olarak hazırlanmış mı?</b>	x	0-5	
<b>Teknik Yönden</b>			
<b>Problemin tanımı yapılmış mı?</b>	x	0-5	
Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı?	x	0-5	
Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı?			
Yazılımın gereksinim listesi oluşturulmuş mu?			
Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı?	x	0-10	
Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi?			
UML ile modelleme yapılmış mı?	x	0-10	
Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.)	x	0-10	
Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı?			
Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı?			
Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı?			
Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)?	x		
Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı?	x		
Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi?			
Yazılımın sızma testi yapılmış mı?			
Performans testi yapılmış mı?			
Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı?			
<b>Yapılan işlerin zorluk derecesi?</b>	x	0-15	
<b>Sözlü Sınav</b>			
<b>Yapılan sunum başarılı mı?</b>			
<b>Soruları yanıtlama yetkinliği?</b>			
<b>Devam Durumu</b>			
<b>Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?</b>	x	0-15	
<b>Diğer Maddeler</b>			
<b>Toplam</b>			

DANIŞMAN:

DANIŞMAN İMZASI: