

Gerçek Zamanlı Konum Takibi

*Çok Katmanlı Bir Web Uygulaması

1st Muhammet Cüneyd Kurtbaş
Kocaeli Üniversitesi
Bilgisayar Mühendisliği
200201132
cuneydkurtbas@gmail.com

2nd Rana Dudu Kabak
Kocaeli Üniversitesi
Bilgisayar Mühendisliği
190201084
ranadudukabak@gmail.com

Abstract—Bu çalışmada; birden fazla kaynaktan akan anlık GPS verilerinin bir message broker ile NoSQL veri tabanında tutulmasının sağlanması, Model, View, Controller kalıbının kullanılması ve SQL veri tabanında yönetimi yapılan kullanıcıların dağıtık bir sistemde subscribe oldukları konulardan yayınlanan verileri bir web uygulamasıyla görerek, çok katmanlı gerçek zamanlı konum takibi yapılabilmesi amaçlanmıştır.

Index Terms—mqtt, mosca, redis, noSql, MVC

I. GİRİŞ

Araç takip hizmeti vermek isteyen bir A firması, müşterilerine araçlarına takabilecekları GPS destekli bir takip donanımı ve ilgili hizmetleri ücret karşılığı sunmak istemektedir. Anılan takip donanım üzerinde GSM modülü bulunmakta olup, araçların mevkileri GSM üzerinden, A firmasının sunucularına aktarılacaktır. A firması, müşterilerine web arayüzü üzerinden araçların mevki bilgilerini sunacaktır. A firması ilgili donanımı hazır ticari ürün olarak temin edecektir.

Proje kapsamında ilgili donanıma ait veriler için İsveç taksi verisi[1] kullanılacaktır. Söz konusu taksi verisinde yaklaşık 100 civarında taksinin bir aylık verileri bulunmaktadır. Genel olarak veriler 1 dakika aralıklarla ile kayıt edilmiştir. Ancak gerçek veriler olması nedeniyle kayıt hatası, gönderim hatası gibi nedenler ile hatalı veriler güzergah içinde çıkabilmektedir. Bazen veride 1 dakikada yüzlerce millik bir atlama yapıp geri dönmektedir. Bu tür atlamaları veya bariz hatalı kısımları veriyi inceleyip, önceden temizleyebilirsiniz. Gerçek verilerde bu tür hatalar, mesleki yaşamınızda karşılaştığınız ve çözüm bulmanız gereken meselelerdir.

II. YÖNTEM

A. Node.js ve Npm

Asenkron, olay tabanlı JavaScript çalışma ortamı olan Node.js, ölçeklenebilir ağ uygulamaları oluşturmak için tasarlanmıştır. İşletim sistemi iş parçacıklarının(thread) kullanıldığı günümüzün yaygın eşzamanlılık modelinin aksinedir. İş parçacığı tabanlı ağ iletişimi nispeten yetersiz ve kullanmak için zordur. Node.js kullanıcıları işlemlerin kilitlenmesinden endişe duymaz, çünkü kilitler yoktur. Node.js'deki neredeyse hiçbir işlev doğrudan G/Ç gerçekleştirmez, bu yüzden işlem hiç bloklanmaz. Hiçbir şey engellemediğinden, ölçeklenebilir sistemlerin Node.js'de geliştirilmesi çok makuldür.[2]

Npm; Node Package Manager ya da Node Packaged Modules olarak da denmektedir. Aslında npm projemizdeki paketlerin yönetimini otomatikleştiriyor diyebiliriz. Npm ile temel olarak yapabileceğimiz şeyler ise; Otomatik ya da manuel olarak paketleri yükleme, sistemdeki paketleri silme, listeleme, güncelleme.Npm komut satırı üzerinden çalışan bir uygulamadır.[3]

Proje kapsamında kullanılan Node versionu v10.16.3 ve aynı paket içinde gelen Npm versionu v6.9.0'dır.

B. Message Queuing Telemetry Transport (MQTT)

MQTT mesajın karşı tarafa gönderilmesi için kullanılan bir haberleşme protokoldür. Bu haberleşme trafiğini kontrol eden yöneticiye Broker, mesaj yayınına Publish ve bu mesaj yayınına abone olanlara Subscribe denmektedir. MQTT de asenkron bir haberleşme kullanılmaktadır. Mesajı yayınlayan ve mesaja abone olanlar arasında veriler asenkron (eş-zamansız) olarak taşınmaktadır. İnternet üzerindeki çeşitli Broker'lara belli konularda abone olabilirsiniz. MQTT diğer haberleşme protokollerine göre daha basit bir yapıya sahiptir ve kolayca projelerinize entegre edilebilirsiniz. Minimum kaynak tüketimi sayesinde özellikle M2M (Machine-to-machine) haberleşmesinde kullanılmaktadır. Bu da MQTT yi IoT projeleri için vazgeçilmez bir mesajlaşma protokolü haline getirmektedir.[4]

Mosca ise; bağımsız Hizmet olarak kullanılabilen veya başka bir Node.js uygulamasında yerleşik olarak kullanılabilen bir Node.js MQTT aracıdır. "Mesaj Kuyruğu" araçları olarak sınıflandırılabilir.

Proje kapsamında kullanılan ve package.json altında tanımlanan mqtt sürümü v4.3.6 ve mosca sürümü v2.8.3'tür.



Fig. 1. Mqtt

C. Remote Dictionary Server (Redis)

Redis, geliştiriciler tarafından en çok kullanılan ve bilinen NoSQL veritabanlarından birisidir. Redis, açık kaynaktır ve kaynak kodlarına GitHub üzerinden erişilebilmektedir. C dili ile yazıldığı için yüksek performanslı sonuçlar vermektedir. Redis günümüz sistemlerinde en çok kullanılan anahtar-değer veritabanıdır ve genellikle caching, session yönetimi, pub/sub, message broker amacıyla kullanılmaktadır.[5]

Redis bir milisaniyenin altında yanıt süreleriyle oyun, reklam teknolojisi, finansal hizmetler, sağlık hizmetleri ve IoT gibi sektörlerde gerçek zamanlı uygulamalar için saniyede milyonlarca isteğin işlenmesini sağlıyor. Hızlı performansı sayesinde Redis; önbelleğe alma, oturum yönetimi, oyun, lider tabloları, gerçek zamanlı analiz, jeo-uzamsal, yolcu taşıma, sohbet/mesajlaşma, medya akışı ve yayınlama/abonelik uygulamaları için popüler bir seçenektir.[6]

Proje kapsamında kullanılan Redis sürümü v4.0.4'tür.

D. MySQL, PHP ve Apache Web Server

MySQL bir ilişkisel veritabanı yönetim sistemidir ve çift lisanslı bir yazılımdır. Yani hem Genel Kamu Lisansı'na (GPL) sahip özgür bir yazılım, hem de GPL'in kısıtladığı alanlarda kullanmak isteyenler için ayrı bir lisansa sahiptir. Ayrıca MySQL işlemlerini SQL adı verilen, veritabanlarına erişmek için kullanılan en yaygın ve standart dil ile yapıyor. MySQL UNIX, OS/2 ve Windows platformlarında kullanılabilir. Fakat Linux altında daha yüksek performans sergilemektedir. MySQL içerisinde ODBC sürücüler de bulunduğu için birçok geliştirme platformunda rahatlıkla kullanılabilir. Çok esnek ve güçlü bir kullanıcı erişim kısıtlama/yetkilendirme sistemine sahiptir. Güçlü bir veritabanı yönetim sistemi olan MySQL özellikle web sunucularında en çok kullanılan veritabanıdır, ASP, PHP gibi birçok web programlama dili ile kullanılabilir.

PHP (Hypertext Preprocessor) geniş bir kitle tarafından kullanılan, özellikle sanal yöreler üzerinde geliştirme için tasarlanmış HTML içine gömülebilen bir betik dilidir. PHP'yi Javascript gibi kullanıcı tarafında çalışan dillerden ayıran, sunucu tarafında çalıştırılıyor olmasıdır. PHP temel olarak sunucu-tarafı programlamaya odaklanmıştır, bu nedenle CGI uygulamalarının yaptığı her şeyi, örneğin formdan veri toplama, dinamik sayfa içeriği oluşturma, ya da çerez alıp gönderme gibi işlemleri yapabilirsiniz. PHP'nin gelişimi sunucu-tarafı programlamaya odaklanmışsa da, çok daha fazlasını yapmanıza olanak tanıyan araçlara da sahiptir.[7]

Apache, açık kaynaklı bir web sunucusu yazılımıdır. Apache'yi bir web sunucusu olarak adlandırmamıza rağmen, bu fiziksel bir sunucu değil, sunucu üzerinde çalışan bir yazılımdır. Görevi, bir sunucu ile web sitesi ziyaretçilerinin tarayıcılar arasında bir bağlantı kurmak ve aralarında dosyalar (istemci-sunucu yapısı) iletmektir. Apache, platformlar arası bir yazılımdır.

Proje kapsamında MySQL ve Apache Server kurulumları, PHP geliştirme ortamı sağlayan XAMPP v8.0.2 paketinin kurulumu ile sağlanmıştır.[8]

III. SİSTEM MİMARISI

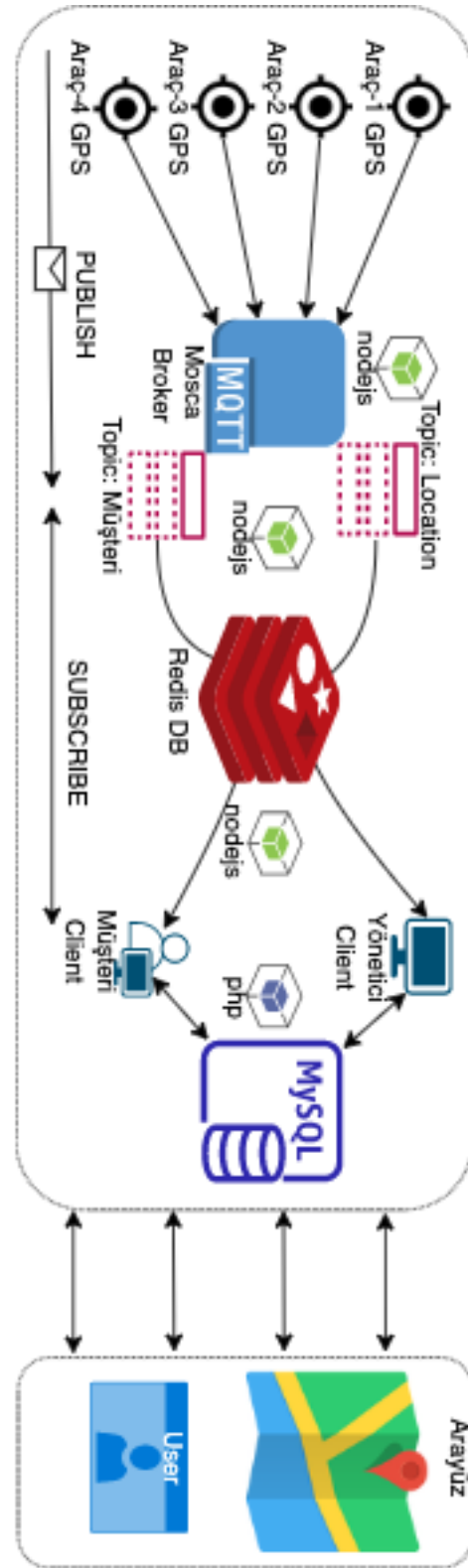


Fig. 2. Mimari

SÖZDE KOD

A. Mosca Brokeri Ayaga Kaldır

```
var redis = require('redis');
var mosca = require('mosca')
var ascoltatore = {
  type: 'redis',
  redis: require('redis'),
  db: 12,
  port: 6379,
  return_buffers: true,
  host: "localhost"
};

var moscaSettings = {
  port: 1883,
  backend: ascoltatore,
  persistence: {
    factory: mosca.persistence.Redis
  },
  http: {
    port: 3000,
  },
};

// mosca server ayaga kaldırma
var server =
new mosca.Server(moscaSettings);
server.on('ready', setup);

// bir client bağlandığında
server.on('clientConnected',
  function (client) {
    console.log('client connected',
      client.id);
  });

// bir mesaj alındığında
server.on('published',
  function (packet, client) {
    console.log('Published',
      packet.topic, packet.payload);
  });

// Mosca hazır olduğunda
function setup() {
  console.log('MOSCA SERVER AYAGA KALKTI')
}
```

B. MQTT Connection, Publish, Subscribe

```
var mqtt = require('mqtt');
var client =
  mqtt.connect('ws://127.0.0.1:3000');
client.on('connect', function () {
  console.log('MQTT Broker Calisiyor!');
});
```

```
client.subscribe('LOCATION');
client.on('connect', function() {
  console.log('Client Konuma Subscribe Oldu');

  client.subscribe('LOCATION', function () {
    coords.forEach(function (element) {
      var dataStr = element.Lat, element.Lng;
      client.publish('LOCATION', dataStr, {
        retain: true,
      });
    });
  });
});
```

C. Google MAP API

```
var map, marker;
function initMap() {
  myLatLng =
    { lat: 40.821955, lng: 29.923659 };
  map = new google.maps.Map
    (document.getElementById('map'), {
      center: myLatLng,
      zoom: 9,
    });
  marker = new google.maps.Marker({
    position: myLatLng,
    map,
    title: "Benim konumum!",
  });
}
```

D. Kullanıcı İşlemleri

```
function registerUser(){
  let name = $('#nameRegister').val()
  let email = $('#emailRegister').val()
  let password = $('#passwordRegister').val()
  $.ajax({ //ajax çağrısı
    url: "api/addUser.php",
    type: 'POST',
    data: {
      name: name,
      email: email,
      password: password,
    },
    success: function (response) {
      swal({
        title: "",
        text: "Kayıt tamamlandı",
        type: "success"
      },function () {
        window.location.href
        ='index.php';
      });
    }
  });
}
```

IV. ÇIKTILAR

A. Mosca Terminal Ekranı

```
cuneyd@ck-macbook% nodemon server.js

[nodemon]2.0.15
[nodemon]to restart at any time,enter`rs`
[nodemon]watching path(s): *.*
[nodemon]watching extensions: js,mjs,json
[nodemon]starting `node server.js`
MOSCA SERVER AYAGA KALKTI

client connected mqttjs_8bf2847c
Published $SYS/8lfQ6tI/new/clients
      mqttjs_8bf2847c
Published $SYS/8lfQ6tI/new/subscribes
{"clientId":"mqttjs_8bf2847c",
"topic":"LOCATION"}
```

B. Redis-Cli Monitor

```
cuneyd@ck-macbook % redis-cli monitor
OK

1647296910.417634 [0 127.0.0.1:49572]
"get" "client:sub:mqttjs_8bf2847c"

1647296910.420214 [0 127.0.0.1:49572]
"del" "client:sub:mqttjs_8bf2847c"

1647296910.420376 [0 127.0.0.1:49572]
"del" "packets:mqttjs_8bf2847c"

1647296910.449176 [12 127.0.0.1:49571]
"publish" "$SYS/8lfQ6tI/new/clients/"
"\x82\xa7message\xafmqttjs_8bf2847c\
\xa7options\x82\xa3qos\xc0\
\xa9messageId\xa7ws4mBT3"

1647296910.467872 [12 127.0.0.1:49570]
"subscribe" "LOCATION/"

1647296910.470353 [0 127.0.0.1:49572]
"hget" "retained" "LOCATION"

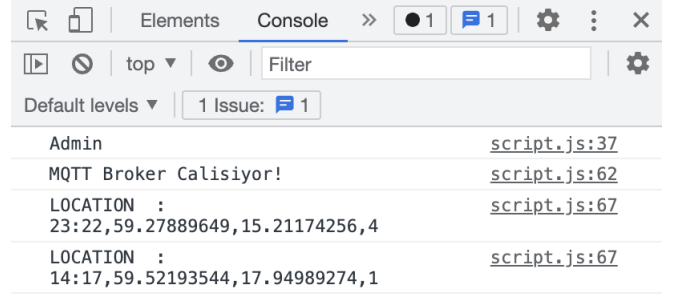
1647296910.474365 [12 127.0.0.1:49571]
"publish" "$SYS/8lfQ6tI/new/subscribes/"
"\x82\xa7message\xd91{\"clientId\":\
\"mqttjs_8bf2847c\", \"topic\":\
\"LOCATION\"}\xa7options\x82\xa3qos\
xc0\xa9messageId\xa7qAd25nb"
```

C. Subscribe Terminal Ekranı

```
cuneyd@ck-macbook% node client.js
Client Konuma Subscribe Oldu
LOCATION:23:22,59.27889649,15.21174256,4
```

D. Arayüz

(*) PHP ile hazırlanan stil katmanlı web sayfası ile kullanıcı kayıt, güncelleme, silme, oturum yönetimi ve MySQL veri tabanına ait ekran görüntülerine yer verilmemiştir.



Admin	script.js:37
MQTT Broker Calisiyor!	script.js:62
LOCATION : 23:22,59.27889649,15.21174256,4	script.js:67
LOCATION : 14:17,59.52193544,17.94989274,1	script.js:67

Fig. 3. Tarayıcı Consol Log

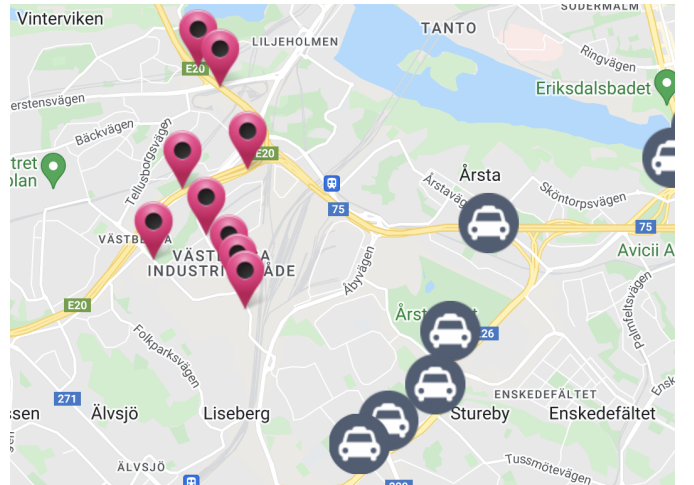
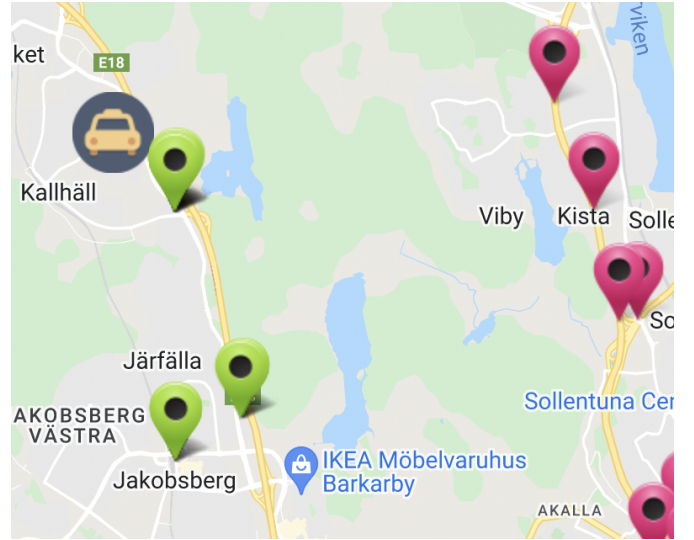


Fig. 4. Konum Takip Web Arayüzü

V. SONUÇ

Bu projenin gereklenmesi sonucunda; Model, Man- zara/Arayüz ve Yönetim (Model,View,Controller) kalıbının kullanılması (Arayüz, İşlem Mantığı ve Verinin ayrıştırılması), mesaj broker kullanarak yazılım fonksiyonel bölümleri arasındaki bağımlılıkların azaltılması, dağıtık bir sistem mi- mari kullanarak çözüm oluşturma, güncel web kütüphaneleri, NoSQL ve SQL veri tabanlarını kullanma ve veri ta- banı tasarımı konularında çok katmanlı bir web uygulaması başarıyla geliştirilmiştir.

Proje; macOS Monterey 12.0.1 (M1) işletim sistemi üzerinde, Visual Studio Code IDE’si kullanılarak NodeJS ve PHP dilleri ile geliştirilmiştir. Uygulama kapsamında gerekli olan uyumlu paketlerin kurulumunda ve son haliyle çalıştırıldığında herhangi bir Runtime Hatası alınmamış olup tamamen açık kaynaklı araçlar kullanılmıştır. Projeye ait tüm kodlar GitHub repomuzdan çekilebilir.

REFERENCES

- [1] taxiMovementConcatenated — Kaggle url: <https://www.kaggle.com/henrikengdahl/taximovementconcatenated>.
- [2] Node.js — url: <https://nodejs.org/tr>
- [3] Npm — url: <https://www.npmjs.com>
- [4] MQTT Nedir? IoT ile Bağlantısı Nedir? — url: <https://www.ebi.com.tr/blog/mqtt-nedir-iot-ile-baglantisi-nedir/>
- [5] Redis Nedir? — url: <https://devnot.com/2020/redis-nedir-temel-kullanim-alanlari-nelerdir>
- [6] Redis — url: <https://aws.amazon.com/tr/redis>
- [7] PHP: What Can Do — <https://www.php.net/manual/tr/intro-whatcando.php>
- [8] XAMPP — url: <https://www.apachefriends.org/tr/>