Caraca Mobil Uygulaması Proje Raporu

GitHub: <https://github.com/bartukaraca/Caraca>

Drive: <https://drive.google.com/drive/folders/17oA9bC_iyQwV4uzqIm2AcOG5kFK7AS4p?usp=sharing>

Bartu KARACA  
*211307042*

*Bilişim Sistemleri Mühendisliği   
Teknoloji Fakültesi  
Kocaeli Üniversitesi*Kocaeli, Türkiye  
bartukaraca5328@gmail.com

*Özet*— Bu makale, bulut bilişim teknolojilerinden yararlanarak bir araba tanıtım uygulamasının tasarımını ve uygulanmasını sunmaktadır. Sorunsuz bağlantı ve otomotiv bilgilerine gerçek zamanlı erişim taleplerine yönelik olarak, uygulamamız veriyi depolamak ve işlemek için bulut bilişim teknolojilerini kullanmaktadır. Bulut hizmetlerinin entegrasyonu, kullanıcıların çeşitli cihazlardan uygulamaya erişimini sağlar, araba özellikleri ve teknik özelliklerini sergilemek için bir platform oluşturur.

Bulut tabanlı araba tanıtım uygulamamızın temel bileşenleri, araç bilgilerinin güvenli depolanmasını sağlamak için bir bulut sunucusunda barındırılan bir veritabanını içerir. Bu kolay güncellemeleri ve bakımı mümkün kılar, kullanıcılara en son araba modelleri ve özellikleri hakkında güncel bilgiler sunar. Proje tamamlandıktan sonra gerekli testler yapıldı. Kullanıcı deneyimi test edildi. Testler sonucu gerekli düzeltme ve iyileştirmeler yapıldı.

Uygulama Kotlin dili ile yazılmıştır. Uygulama yazılırken editor olarak Android Studio kullanılmıştır. Uygulama AWS üzerinden kiralanmış Bitnami serverı üzerine kurulmuş ve parse frameworkü kullanılmıştır. Uygulama bir sanal makina üzerinde çalışmaktadır.

Sonuç olarak, bulut tabanlı araba tanıtım uygulamamız, bulut bilişimin potansiyelini dinamik, ölçeklenebilir ve kişiselleştirilmiş bir deneyim sunmak için kullanma konusunda önemli bir ilerlemedir. Bulut hizmetlerinin entegrasyonu sadece erişimi artırmakla kalmaz, aynı zamanda otomotiv sektöründe gelecekteki yeniliklerin temelini oluşturur, teknolojinin araçlarla etkileşimimizi ve keşfetme şeklimiz üzerindeki dönüştürücü etkisini gösterir.

Anahtar Kelimeler— bulut bilişim, AWS, araba, Kotlin, cloud computing

Abstract— This article presents the design and implementation of a car promotion application by leveraging cloud computing technologies. Addressing the demands for seamless connectivity and real-time access to automotive information, our application uses cloud computing technologies to store and process data. The integration of cloud services allows users to access the application from various devices, creating a platform to showcase car features and specifications..

Key components of our cloud-based car promotion application include a database hosted on a cloud server to ensure secure storage of vehicle information. This enables easy updates and maintenance, providing users with up-to-date information on the latest car models and features. After the project was completed, the necessary tests were carried out. User experience tested. As a result of the tests, necessary corrections and improvements were made..

The application is written in Kotlin language. Android Studio was used as editor while writing the application. The application was built on the Bitnami server rented through AWS and the parse framework was used. The application runs on a virtual machine.

All in all, our cloud-based car promotion application is a significant advance in using the potential of cloud computing to deliver a dynamic, scalable and personalized experience. The integration of cloud services not only increases access, but also underpins future innovations in the automotive industry, demonstrating the transformative impact of technology on the way we interact with and explore vehicles.

Keywords— cloud computing, AWS, car, Kotlin, cloud computing

# GİRİŞ

Günümüzde, otomotiv endüstrisi hızla dijitalleşmekte ve kullanıcıların araçlara yönelik talepleri giderek karmaşık hale gelmektedir. Bu bağlamda, bulut bilişim teknolojilerinin benimsenmesi, araç tanıtım uygulamalarının geliştirilmesinde önemli bir dönemeç oluşturmaktadır. Bu makalede, bulut bilişim altyapısı üzerine inşa edilmiş bir araç tanıtım uygulamasının tasarımı ve uygulanması ele alınmaktadır.

Geleneksel araç tanıtım uygulamaları, genellikle yerel cihazlarda çalışan sınırlı özelliklere sahipken, bulut bilişimle entegre bir yaklaşım, kullanıcılara daha geniş bir erişim ve etkileşim imkanı sunar. Bu uygulama, araç özellikleri, performans verileri ve güncellemeleri gibi çeşitli bilgileri depolamak, işlemek ve kullanıcılara sunmak için bulut tabanlı bir altyapı kullanmaktadır.

Bulut bilişimin getirdiği avantajlar arasında, güvenilirlik, ölçeklenebilirlik ve erişilebilirlik bulunmaktadır. Merkezi bir bulut sunucusunda barındırılan veritabanı, kullanıcılara güvenli bir depolama ortamı sunar ve aynı zamanda güncel bilgilere kolay erişim sağlar. Bu, kullanıcıların her an güncel araç bilgilerine ulaşabilmesini sağlayarak, karar verme süreçlerini iyileştirir.

Bu makale, bulut bilişim teknolojileri kullanılarak geliştirilen araç tanıtım uygulamasının tasarımını, altyapısını ve kullanıcı deneyimini detaylı bir şekilde ele alacak

# senaryo

## Uygulamanın Senaryosu

Caraca kullanıcının authentication işlemlerini yapabildiği, kullanıcıların araba ekleyebildiği, bu arabaların fotoğraf , marka, model ve özelliklerini ekleyebildiği ve bunu diğer kullanıclarında görmesi için paylaşabildiği bir araba tanıtım uygulamasıdır.

## Bulut Sistemi Senaryosu

Bulut sistemi olarak Amazon Web Service altyapısı kullanıldı. AWS marketplaceden Bitnami Parse server metin, elektronik donanım, ekran görüntüsü, yazılım içeren bir resim

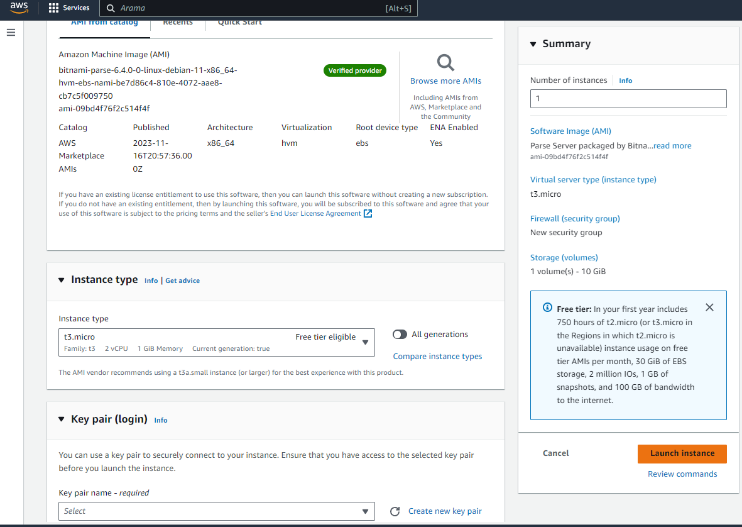
Açıklama otomatik olarak oluşturuldukiralandı. Veritabanı burada bulunmaktadır. Parse frameworkü ile uygulamaya veri giriş çıkışları sağlanmaktadır.

## Sanallaştırma Senaryosu

Sanallaştırma için Oracle VM VirtualBox kullanıldı. VirtualBox üzerinden bir Android işletim sistemi sanallaştırılarak uygulama bunun üzerinden çalıştırıldı.

# PROJE TANIMI

## Bulut Ortamının Hazırlanması

Öncelikle bir AWS hesabı açıldı. Hesap açıldıktan sonra projeye uygun server olarak Bitnami Parse Server seçildi. Server açılırken isim, key pair oluşturma gibi işlemler yapıldı. Key pair sunucuya erişimde kullanılan bir .ppk dosyasıdır. Launch instance butonuna tıklanılır ve server oluşturulur.

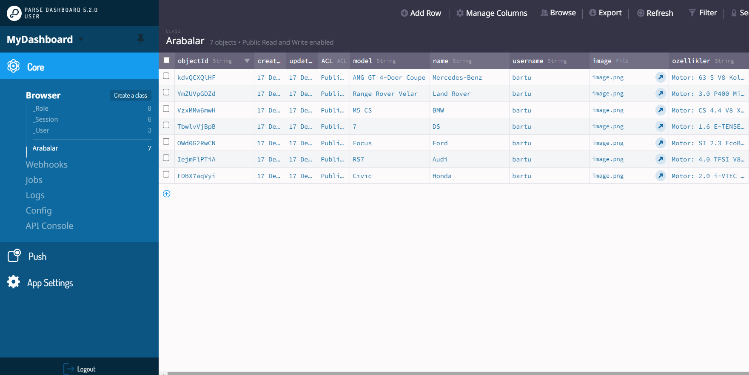
Şekil.1 AWS Server Oluşturma Ekranı

metin, elektronik donanım, ekran görüntüsü, ekran, görüntüleme içeren bir resim

Açıklama otomatik olarak oluşturuldu Sunucu oluşturulunca sunucuların yönetildiği ekrana gidilir. Bu ekranda sunucuya dair işlemleri yapabilir; sunucu ile ilgili bilgileri görebilirsiniz. Sunucuya bağlanmak için “Putty” programı kullanıldı. Bu programda aws’den alınan kullancı bilgisi ve key pair kullanılarak sunucunun komut istemcisine ulaşabilmekteyiz. Bu komut işlemcisinde sırasıyla “cs stack” “cd parse” “nano config.json” komutları girilerek veritabanını uygulamamıza entegre etmek için gerekli olan bilgilere erişilir.

Şekil.2 Putty Ekranı

Şekil.3 config.json içindeki veriler

 AWS’deki sunucu yönetim sayfasında verilen public IPv4 DNS linkine tıklayınca Dashboard ekranına gidilmektedir. Burada kullanıcı adı olarak user, şifre olarakta Puttyden aldığımız masterKey kullanılarak giriş yapılır. Burası Parse dashboardıdır. Veritabanında oluşturduğumuz tabloları vesaire görebilir ve yönetibiliriz.

Şekil.4 Parse Dashboard Ekranı

## Ugulamanın Yapımı

Uygulama Android Studio’da Kotlin dili kullanılarak yazılmıştır. Build Gradle içine gerekli tanımlamalar yapıldı. Gerekli SDK’ler yüklendi.

Parse.initialize Parse SDK'nın kullanımını başlatır. Bu, Parse Configuration Builder kullanılarak yapılandırılmış bir ParseConfiguration nesnesi ile yapılır.

* applicationId: Puttyden aldığımız appID parametresinin değeri girilir.
* clientKey: Puttyden aldığımız masterKey parametresinin değeri girilir.
* server: Puttyden aldığımız serverURL parametresinin değeri girilir.

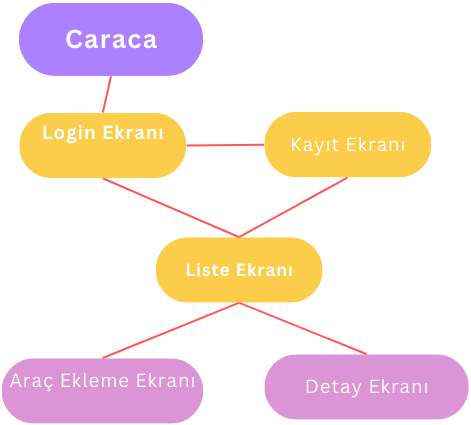
SignUpActivity deki signUp2 fonksiyonun sözde kodu aşağıdaki gibidir.

1. Yeni bir ParseUser örneği oluştur
2. Kullanıcı adını ve şifreyi giriş metin kutularından al
3. Kullanıcıyı arka planda kaydet, hata kontrolü yap
4. Hata varsa hata mesajını göster
5. Hata yoksa başarılı mesajını göster
6. LocationsActivity'ye geçiş için bir Intent oluştur ve başlat

PlaceNameAcrivity araba oluşturmak için kullandığımız activitymizdir. Buradaki ilgili alanlara girilen bilgiler parse aracılığıyla buluttaki veritabanına gönderilir.

MainActivity giriş yapmak için kullandığımız activitymizdir.

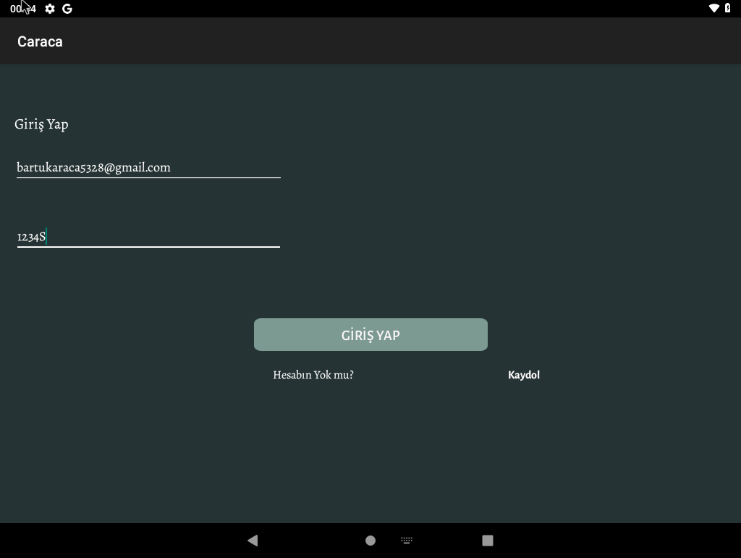
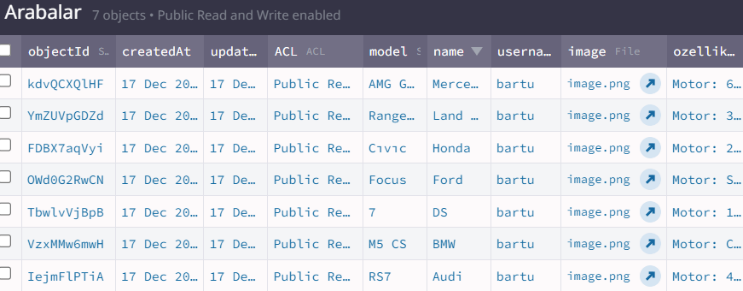
LocationActivity arabaların listelendiği activitymizdir. Burada bir listview kullanılmıştır. Listenelenen arabaların markaları gözükmetedir. Görüntülenmek istenen arabanın üstüne tıklanılarak araba detay sayfasına gidilir.

 DetailActivity arabanın resim, model, marka ve özelliklerinin gösterildiği ekrandır. Parse aracılığı ile veritabanındaki veriler çekilir. İlgili alanlara yazdırılır.

Şekil.5 Uygulamanın Akış Diyagramı

## Veritabanı

Veritabanında Arabalar tablosu bulunmaktadır. Bu tabloda bulunan kolonlar aşağıdaki gibidir.

* objectId: Parse otomatik olarak nesne için bir ıd tanımlar
* createdAt: Oluşturulma tarihini Date biçiminde verir.
* updatedAt: Güncelleme tarihini Date biçiminde verir.
* model: Arabanın modelini string biçiminde verir.
* name: Arabanın markasını string biçiminde verir.
* ozellikler: Arabanın özelliklerini string biçiminde verir.
* image: Arabanın fotoğrafını tutar.

Şekil.6 Veritabanı

## Sanal Makina

Sanal makina olarak Oracle VirtualBox kullanıldı. VirtualBox indirildikten son android ISO dosyası indilirdi. Daha sonrasında aşağıdaki adımlar uygulanarak sanal makina oluşturuldu.

* VirtualBox'u açın.
* "Yeni" düğmesine tıklayın ve sanal makina için temel ayarları yapın.
* İsim ve İşletim Sistemi: İsimi girin, Tür'ü "Linux" olarak seçin, Versiyon'u "Other Linux (64-bit)" olarak seçin.
* Bellek (RAM): Sanal makinenize ayrılacak RAM miktarını seçin (min. 2GB önerilir).
* Sabit Disk: Sanal makine için bir sabit disk oluşturun. "Yeni Oluştur" seçeneğini seçin ve bir VDI (VirtualBox Disk Image) seçin.

Sabit Disk Ayarları:

* Oluşturduğunuz sabit diski seçin ve "Değiştir" düğmesine tıklayın.
* "Disk Dosyası Türü" olarak "Dinamik Büyüyen" seçeneğini belirleyin ve sabit diskin boyutunu ayarlayın.

Android x86 ISO Dosyasını Ekleyin:

* Sanal makinenizi seçin ve "Ayarlar" düğmesine tıklayın.
* metin, yazılım, multimedya yazılımı, bilgisayar simgesi içeren bir resim

  Açıklama otomatik olarak oluşturuldu"Depolama" sekmesine gidin ve "Boş" altındaki disk simgesine tıklayarak Android x86 ISO dosyasını ekleyin.

Şekil.7 VirtualBox Ekranı

Sanal makina üzerinde android işletim sisteminin kurulumları yapıldıktan sonra uygulamanın apk dosyası makina içine atıldı ve uygulama kuruldu. Uygulamanın testleri yapıldı ve sorunsuz çalıştığı belirlendi.

Şekil.8 Uygulamanın Ekran Görüntüsü

# BİLİŞİM ve SANALLAŞTIRMA

## Bulut Bilişim

Bulut bilişim, internet üzerinden bilgisayar sistemlerine erişim ve bu sistemlerin paylaşılan kaynaklarının kullanımını ifade eder. Hizmet modelleri aşağıdaki gibidir:

* Altyapı Sunumu (IaaS): Temel altyapı kaynaklarını (sunucular, depolama) sağlar.
* Platform Sunumu (PaaS): Uygulama geliştirmek için gerekli platformu sağlar.
* Yazılım Sunumu (SaaS): Kullanıcılara hazır yazılımları sağlar.

Dağıtım modelleri aşağıdaki gibidir:

* Public Bulutu: Genel olarak herkese açık ve üçüncü taraf sağlayıcılar tarafından yönetilen bulut hizmetleri.
* Private Bulut: Belirli bir kuruluşa özel, genellikle kendi veri merkezlerinde yönetilen bulut hizmetleri.
* Hybrid Bulut: Public ve Private bulutun bir kombinasyonunu içerir.

Esneklik ve ölçeklenebilirlik, maliyet Tasarrufu, yedekleme ve güvenlik bulut bilişimin en büyük avantajlarındandır. Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), IBM Cloud en popüler bulut bilişim sağlayıcılarıdır.

## Sanallaştırma

Sanal Makine (VM): Bir fiziksel bilgisayar üzerinde çalışan sanal bir bilgisayar sistemidir. Her VM, kendi işletim sistemine ve uygulamalarına sahiptir.

Hypervisor (Sanallaştırma Yazılımı): Sanal makineleri yöneten ve fiziksel kaynakları sanal makineler arasında paylaştıran yazılımdır. Tip 1 (bare-metal) ve Tip 2 (hosted) hypervisorlar vardır.

Sanallaştırma, fiziksel donanımın üzerine inşa edilen sanal ortamlar oluşturarak, bir bilgisayar veya sunucunun birden çok sanal makine (VM) olarak çalışmasını sağlayan bir teknolojidir.

Avantajları:

* Kaynak Kullanımı: Daha etkin kaynak kullanımı sağlar.
* İzolasyon: Farklı sanal makineler birbirinden izole edilir.
* Yedekleme ve Kurtarma: Sanal makinelerin anlık görüntüleri alınabilir ve geri yüklenebilir.

# SONUÇ

Bu raporda bulut bilişim ve sanallaştırma teknolojileri kullanılarak oluşturulmuş olan Caraca uygulamasının tasarım ve mimari yapım aşamaları açıklanmıştır.

Projede karşılaşılan en büyük sorun sunucuyu oluşturken ve sunucuya bağlanırken yaşanılan zorluklar olmuştur. Projede AWS, Bitnami, Parse, VirtualBox, Kotlin, Android Studio araçları kullanılmıştır.

# Kaynaklar

[1] https://www.udemy.com/course/android-o-mobil-uygulama-dersi-kotlin-java/

[2] <https://docs.bitnami.com/>

[3] <https://docs.bitnami.com/aws/>

[4] <https://docs.aws.amazon.com/>

[5] https://wiki.uhem.itu.edu.tr/w/index.php?title=Putty\_kullan%C4%B1m\_k%C4%B1lavuzu