**SÖZLÜK PROJESİ:**

dictionary={  
 **"this"**: [(**"zamir"**, **"Bu, Şu"**)],  
 **"is"**: [(**"fiil"**, **"Olmak"**)],  
 **"a"**: [(**"belirteç"**, **"Bir"**)],  
 **"book"**:[ (**"isim"**, **"Kitap"**), (**"fiil"**, **"Rezerve etmek"**)],  
 **"pen"**: [(**"isim"**, **"Kalem"**)]  
}  
  
**while True**:  
 sentence = input (**"Bir cümle giriniz: (Çıkmak için ENTER'a basınız)"**)  
 **if** sentence == **""**:  
 **break  
  
 for** signCharacter **in ".!?,=:-)("**:  
 sentence = sentence.replace(signCharacter, **''**)  
  
 words = set(sentence.lower().split())  
  
 wordsMaxLength = 1  
 **for** word **in** words:  
 **if** wordsMaxLength < len(word):  
 wordsMaxLength = len(word)  
  
 **for** word **in** words:  
 wordDefinitions = dictionary.get(word, **"[ANLAMI BULUNAMADI]"**)  
 **if** len(wordDefinitions) == 1:  
 print(**f'{**word.upper()**:<{**wordsMaxLength**}} : ({**wordDefinitions[0][0]**:^8}) {**wordDefinitions[0][1]**}'**)  
 **elif** len(wordDefinitions) > 1:  
 print(**f'{**word.upper()**:<{**wordsMaxLength**}} : '**, end=**''**)  
 **for** wordDefinition **in** wordDefinitions:  
 print(**f'({**wordDefinition[0]**}) {**wordDefinition[1]**} '**,end=**''**)  
 print()  
  
 print()  
print(**"Kullanıcı çıkışı."**)

ARAŞTIRMA KONULARI

1.HAFTA: Web programlama nedir?

Hangi dillerde ve editörlerde yazılır?

Webin bileşenleri nedir?

Katmanlı mimari nedir?

HTML ve CSS kavramlarında yeni teknolojiler nelerdir?

2.HAFTA: MVC ve .Net Core nedir?

3.HAFTA: Mobil programlama nedir?

Nasıl yapılır?

Hangi dilleri bilmek gerekir?

React Native hakkında bilgiler

4.HAFTA: Veri madenciliği nedir?

Hangi alanda ve ne için kullanılır?

Veri madenciliği yöntemleri hakkında genel bilgiler

5.HAFTA: Derin öğrenme nedir?

Derin öğrenme için neler yapılır?

6.HAFTA: Javascript nedir? Jquery nasıl kullanılır?

Node.js farkı nedir? Kullanım örnekleri

**1.HAFTA ARAŞTIRMA KONULARI**

1. WEB PROGRAMLAMA NEDİR?
2. HANGİ DİLLERDE VE EDİTÖRLERDE YAPILIR?
3. WEBİN BİLEŞENLERİ NELERDİR?
4. KATMANLI MİMARİ NEDİR?
5. HTML VE CSS KAVRAMLARINDA YENİ TEKNOLOJİLER NELERDİR?

***WEB PROGRAMLAMA NEDİR?***

Web Programlama, temel ve basit anlamıyla bir web sitesi oluşturmak için yazılan kaynak koddur.

Günümüzde kullanılan facebook, twitter (her ne kadar mobil appleri olsa da 🙂 ), alışveriş yaptığımız e-ticaret siteleri hep web programlamanın eseridir.

Web Programlama temelde 2 ye ayrılır. İstemci ( Front-End) ve Sunucu (Back-End) taraflı programlamadır.

Sunucu Taraflı

Adından da anlaşılacağı gibi sunucu ile web sitesi arasındaki iletişimi sağlar. Ağırlıklı olarak veritabanı işlemleri; üye kayıdı, üye girişi, veri giriş-çıkışı gibi işlemlerdir.

Kullanılabilecek programlama dilleri: PHP, ASP.Net

İstemci Taraflı

Kullanıcıya gösterilen ve kullanıcıyla etkileşime girilmesi sağlanır. Etkileşim dediğimiz olaya değinirsek kullanıcının yazılara yorum yapması, resmin üzerine geldiğinizde büyümesi, kullanıcının girdiği verilerle işlemlerin yapılması gibi olaylardır.

Kullanılabilecek tek bir programlama dili vardır “javascript”.

JQuery,Ajax,Angular JS var desekte aslında bunların hepsi bir javascript kütüphanesidir.

HTML ve CSS diye bir şey var bunlar nedir?

Birçok kişi tarafından programlama dili olarak kabul edilmese de, bir kısım hala programlama dili olarak görmekte bu ikisini.

HTML ( Hypertext Markup Language )

Web sayfalarını oluşturmak için kullanılan metin işaretleme dilidir.

HTML sayesinde web sayfamızın iskeletini oluştururuz.

CSS (Cascading Style Sheets)

Kısa tanımıyla stil şablonudur. HTML ile oluşturulan yazılara, görsellere, tablolara, vs. stil vermek için kullanılır. Siteye bir görsellik kazandırmış oluruz. Yani sitemizin tasarımını yaptığımız kısımdır.

Kullanılan birçok CSS kütüphanesi olmasına rağmen günümüzde en çok kullanılanı Twitter geliştiricileri tarafından oluşturulmuş olan Bootstrap’tir.

Hangi sırayla ne öğrenmeniz gerektiğini kısaca söyleyeyim:

1.HTML

2. CSS

3.Javascript

4.Herhangi bir JS kütüphanesi

5. Back End’e geçecekseniz PHP,ASP.Net, Java, Python veya Ruby dillerini öğrenmeniz gerekecek.

***HANGİ DİLLERDE VE EDİTÖRLERDE YAPILIR?***

Web sitenizin tasarım çalışması bittikten sonra bu çalışmanın arka planının yazılması işlemi başlar. *Web yazılım* çalışması yapılırken de tıpkı web tasarımda olduğu gibi bazı *editörler*den faydalanılarak bu işlem gerçekleştirilir. Her editörün kendine göre avantajları veya dezavantajları bulunabilir. Önemli olan sizin kullanımınıza en uygun olanını seçmektir.

**Web Yazılım Editörleri**

Aşağıda sizlerle yazılım editörlerinin çok kullanılanlarından bazılarını paylaşıyor olacağız. Bu editörler haricinde kullanılan editörler de olduğunu eklemek gerekiyor. Bazı editörlerin donanım özelliği düşük olan bilgisayarlarda ağır açıldığını ve çalıştığını da belirtelim. Hangi editör sizin için kullanışlı geliyorsa o editörle çalışmanızı öneriyoruz. Büyük projeler için kapsamlı özelliklere sahip, gerekirse de ücretli editör kullanmanızda fayda var.

**Notpad ++**

En yaygın bilinen ve genelde yazılıma yeni başlayanlar tarafından kullanılan editörlerdendir. Bildiğimiz not defterinin gelişmiş versiyonu olan bu editör kısayolları ile de yazım hızınızı artırmanızda yardımcı olabiliyor. Ücretsiz yayınlanan ve sürekli geliştirilen editörü kendi sitesinden indirebilirsiniz.

**Dreamwiever**

**Web tasarım** çalışmalarında kullanılan bu editör yazılım çalışmalarında da kullanılabiliyor. Teknik özellikleri gereği iki çalışmayı da aynı anda yürütebildiği için popüler editörler arasında yer alıyor.

**PHP Designer**

Ücretli olan bu editör PHP dilinde yazılım için kolaylıklar sağlasa da diğer yazılım dillerinde de kod yazmanıza müsade ediyor. Bir çok editörden farklı olarak hatanın olduğu satırı size gösterebiliyor. Zaman olarak tasarruf etmenizi sağlayabilir.

**Visual Studio Code**

Yazılım editörleri arasında çok kullanılan editörlerden. Unuttuğunuz fonksiyonları tamamlama, hata satırlarını gösterme gibi özellikleri bulunuyor. Diğer editörlere göre başlangıç aşamasında biraz yavaş açılsa da sonrasında sizi tatmin edecek özelliklere sahip bir editör.

**BlueFish Editör**

Web yazılım için özel bir editör olan Bluefish Editor, webe özel birçok seçeneği içinde barındırıyor. Bunlar arasında sözdizimi vurgulama (ASP.NET, CSS, HTML, JavaScript, PHP ve fazlasını destekliyor), kod katlama, güçlü arama ve değiştirme araçları, otomatik tamamlama ve biraz daha fazlası.

Bu editörlerin haricinde internette onlarca editör bulabilirsiniz. Editör kullanımında eğer başlangıç aşamsındaysanız tavsiyemiz Notpad++’dır. Bir çok editör kod tamamlama özelliğine sahip olduğundan işin mantığını kavramanız biraz geç olabiliyor. Notpad++’da kodları hep manuel olarak girerek yazım mantığını kısa zamanda çözebilirsiniz. Daha sonrasında isterseniz Notpad++’in bazı özelliklerini açarak kolaylıklar sağlayabilir, isterseniz de farklı bir editör üzerinden devam edebilirsiniz.

**Editörlerin Faydaları Nelerdir?**

* Daha hızlı bir şekilde kod yazmanızı sağlar.
* Kod tamamlayıcı özellikleri ile unuttuğunuz fonksiyonları size hatırlatır.
* Hata satırlarını bulma konusunda işinizi kolaylaştırır.
* Kullandıkları dosya gösterim mantığı ile projenizdeki tüm dosyaları tek ekrandan hızlıca yönetebilirsiniz.

Sonuç olarak editörler sunduğu özelliklerle web yazılımcının işini kolaylaştırmak ve proje süresini kısaltmak için varlar. Bu nedenle size en kullanılabilir gelen editörle çalışmanız önemli. Fakat bir editörü kullanmaya kadar vermeden önce bir kaç deneme mutlaka yapın.

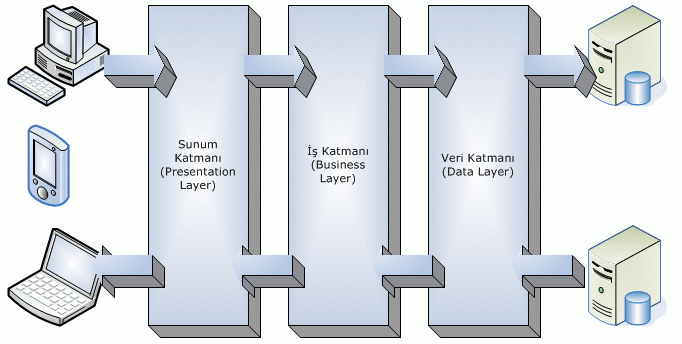
***WEB TASARIM BİLEŞENLERİ NELERDİR?***

***Web tasarım*** bileşenleri, **web sitenizi**“*iyi bir çözüm*” haline getirmemizi sağlayacak özelliklerdir. Siteye giren herhangi bir kullanıcının sitenin vermeye çalıştığı hizmeti, kısa bir süre içerisinde algılayabilmesi gerekir. Bir başka deyişle, oluşturduğumuz *web sitesi,*kullanıcıda bir “*farkındalık*” yaratmalıdır. Kırık bağlantılar, kötü renk seçimleri, açılmayan resim dosyaları, karışık yazılar, yaptığımız tasarımı sabote edecek, hedefe ulaşmamızı engelleyecektir.

1. **Web**Sitesinin İşlevselliği: **Web siteleri**, **web sayfalarının** işlevsel olarak birbirlerine bağlanmasıyla oluşurlar. Bu yüzden bir **web sitesinin** uğrayabileceği en talihsiz durum, bu bağlantılardaki kırılmalardır.
2. İçerik Metinleri: Pek çok kişinin metinler konusunda düştüğü hata, onları yalnızca seo açısından değerlendirmektir. Oysa **web tasarım** açısından bakıldığında, kullanıcıyı çeken şey, kolay okunması ve doyurucu olmasıdır. Kendini tekrar eden cümleler ve uzun metinler, siteyi üst sıraya yükseltse bile, kullanıcı tarafından tercih edilmeyecektir.
3. Renk Denetimi: Kötü renk seçimleri tasarımı kullanışsız yapar. Tasarımcı son derece zevkli biri olsa da renkleri kendi isteği doğrultusunda kullanıyorsa, yanlış yoldadır. Yeşil renkten hoşlanan bir tasarımcı, kadınlara yönelik bir sitede renk seçimini yeşilden yana yapmak yerine, “kullanıcı gözlüğü” takıp pembe ve morla donatmalıdır.
4. Görsel Seçimler: **Web sitesi**, kullanıcıda “farkındalık” yaratacak bir görsel bütünlük içerisinde olmalıdır. Grafikler, fotoğraflar, videolar, yazı tipleri, ilgi çekici olmalı, birbirini desteklemelidir.
5. **Web Tasarım** Eklentileri: **Web sitesinin kuruluşu** sırasında, amacımıza yönelik seçtiğimiz eklentiler, hedefe daha kolay ulaşmamızı sağlarlar. Onları kullanarak hızımızı ve performansımızı artırabilir, **web sitesini görünür kılabiliriz**.
6. Sunucu Hizmeti: Sunucu hizmeti, **web sitesinin performansını**etkilediği için, tasarım içerisinde büyük rolü vardır.

*KATMANLI MİMARİ*

Katmanlı mimari temelde 3 katmandan oluşmaktadır. Bu 3 katman genelde her projenizde olması gereken katmanlardır. Siz bu 3 katmandan daha fazla bir katmanlı yapıda oluşturabilirsiniz bunada çok katmanlı mimari denilmektedir. Hadi gelin temeldeki 3 katmanı inceleyelim.



**Katmanlı Mimari**

**Data Access Layer :** Bu katmanda sadece veritabanı işlemleri yapılmaktadır. Bu katmanın görevi veriyi ekleme, silme, güncelleme ve veritabanından çekme işlemidir. Bu katmanda bu işlemlerden başka herhangi bir işlem yapılmamaktadır.

**Business Layer :** Bu katmanda iş yüklerimizi yazıyoruz. Öncelikle şunu söylemeliyim bu katman Data Access tarafından projeye çekilmiş olan verileri alarak işleyecek olan katmandır. Biz uygulamalarımızda Data Access katmanını direk olarak kullanmayız. Araya Business katmanını koyarak bizim yerimize Business’ın yapmasını sağlarız. Kullanıcıdan gelen veriler öncelikle Business katmanına gider oradan işlenerek Data Access katmanına aktarılır. Bu katmanda ayrıca bu verilere kimlerin erişeceğini belirtiyoruz. Örneğin IT ve Muhasebe bölümü var. IT bölümünün veri tabanına ekleme işlemleri yapmasını istiyoruz ama Muhasebe bölümünün sadece verileri çekmesini istiyorsak bunu Business Katmanında gerçekleştiriyoruz.

**Presentation Layer:**Bu katman kullanıcı ile etkileşimin yapıldığı katmandır. Burası Windows form da olabilir, Web’te olabilir veya Bir Consol

uygulamasida olabilir. Burada temel amac kullaniciya verileri göstermek ve kullanıcıdan gelen verileri Business Katmanı ile Data Access’e iletmektir.

**Entities :** Entities Katmanında ise genelde domain olarak adlandırılan classlarımızı tanımlıyoruz. Bu Entities katmanının ismini domain olarakta değiştirebilirsiniz veya Common katmanıda yapabilirsiniz. Ben Entities ismini tercih ediyorum. Bu katmanda proje boyunca kullanacağımız ana classlarımızı belirliyoruz yani gerçek nesnelerimizi belirlediğimiz yer burası. Daha anlaşılabilir bir şekilde anlatmak için birkaç örnek vereyim. Örneğin bir Stok veritabını sistemi yapmak istiyorsunuz. Bu sistemde Ürün bilgileriniz, Kategori bilgileriniz ve Satış bilgilerinizin olduğunu varsayalım. İşte bu bilgilerinizi burada tanımlıyorusunuz. Ürün classınız içerisinde Property olarak ÜrünAdı, ÜrünID si, ÜrünStokMiktari gibi, ÜrünFiyatı gibi propertyler olabilir. Bu katmanı hem Data Access hem Business hemde Presentation katmanı kullanmaktadır.

*CSS3 Nedir?*

Şu an için en gelişmiş “**C**ascading **S**tyle **S**heets” yani Türkçesi Basamaklı Stil Şablonudur. Önceki versiyonlarının güçlü taraflarının yanı sıra @media-queries, box-shadow, translate, transition gibi yeni özellikleri getirmiş bir teknolojidir. Makalenin devamında bunları inceleyeceğiz.

CSS3 hangi yenilikleri getirdi?

Adı üstünde stil şablonu olan bu teknoloji ile öncelikle [mobil](https://bilisimhareketi.com/etiket/mobil) tasarımda ve görsellikte çığır açmayı ve özgürleşmeyi sağlayan seçici tanımları geldi. CSS seçiciler olmadan pek bir işe yaramayan bir teknolojidir. HTML sayfası içerisinde class ve id kullanarak ya da [HTML5](https://bilisimhareketi.com/etiket/html5) ile gelen header,section,footer,aside gibi seçiciler ile birlikte kullanılmaktadır. Tasarımsal geçişlerin sağlandığı kodlar ile birlikte flash gibi artık açıklarından dolayı tehlikeli içeriklere mahal veren teknolojiler kullanılmaya gerek kalmamıştır.

transition

Bu da CSS3 içerisine yeni katılmış olan bir teknoloji. Adının tam Türkçe’ye çevrimi “geçiş” yani efekt olarak kullanılabilecek kabiliyetlerin ekstra programlamaya ya da flash teknolojisine gerek kalmadan buton gibi elemanlara uygulanması anlamına geliyor. Buton üstüne gelince sağ tarafı kıvrılıyormuş gibi bir efekt oluşuyor.

keyframes

İşte benim en sevdiğim tanım. Elemanların efekt olarak ileri geri, köşeden köşeye ya da istediğiniz şekilde tekrar eden ya da bir kere çalışan şekilde hareketlenmesini sağlayan animasyon altyapısı.

Flexbox Modelleme

Grid sisteminden sonra en büyük CSS3 yeniliği olarak görülebilir. Flexbox altyapısı sitenizin tasarımsal özelliklerinin -ki grid yerleşimi buna dail- web tarayıcısı tarafından derlenmesi ve esnek altyapı kullanarak elemanları sağa sola yaslamak için uğraşmanıza gerek kalmayan flex kapsayıcısının kullanılmasını sağlamaktadır.

***KAYNAK:***

1. : <https://www.damlakayali.net/web-programlama-nedir-nerden-baslamaliyim/>
2. : <https://www.atolyeajans.com.tr/web-tasarim-ve-yazilim.html>
3. <https://medium.com/kodcular/katmanl%C4%B1-mimari-9fb34ef8c376>
4. : <https://bilisimhareketi.com/web-teknolojileri-giris-css3>

2.HAFTA:

.NET Core nedir?

.NET Core, Microsoft ve .NET topluluğu tarafından geliştirilmiş ve geliştirilmeye devam eden, Github üzerinde bulunan açık kaynaklı, çapraz platform geliştirme yapmaya olanak sağlayan bir platformdur. .Net Core çapraz geliştirmeye verdiği olanak sayesinde Windows, macOS ve Linux’u destekler. Ayrıca bulut ve IoT uygulamaları oluşturulmasında kullanılabilir.

.NET Core Neler İçerir?

**Çapraz-platform:** Windows, macOS ve Linux işletim sistemlerinde çalışabilen.

**Farklı mimariler arasında tutarlı:** Yazdığınız kodları aynı davranışlarla x64, x86 ve ARM mimarilerinde çalıştırır.

**Komut satırı araçları:** Yerel geliştirme ve sürekli entegrasyonun yer aldığı senaryolarda kullanılabilecek, kullanımı kolay, modern komut satırı araçlarını içerir.

**Esnek geliştirme:** Uygulamanıza eklenebilir sistem çapında veya kullanıcı çapında kurulumlar sunmakla birlikte Docker kapsayıcıları ile de kullanım imkanı sunar.

**Uyumluluk:** .NET Core, .NET Framework, Xamarin ve Mono ile .NET Standard kütüphanesi sayesinde uyum sağlar.

**Açık kaynak:** .NET Core açık kaynak kodludur ve MIT, Apache 2 lisansları kapsamındadır. .NET Core bir .NET Kuruluşu projesidir.

**Destek:**.NET Core, Microsoft tarafından desteklenir.

.NET Core ile Hangi Yazılım Dilleri Kullanılabilir?

**C#, Visual Basic** ve **F#** dilleri ile uygulamalar veya kütüphaneler geliştirirken .NET Core’dan yararlanılabilir. **Visual Studio**, **Visual Studio Code**, **Sublime Text** ve **Vim** gibi editörler ile geliştirme yapılabilir.

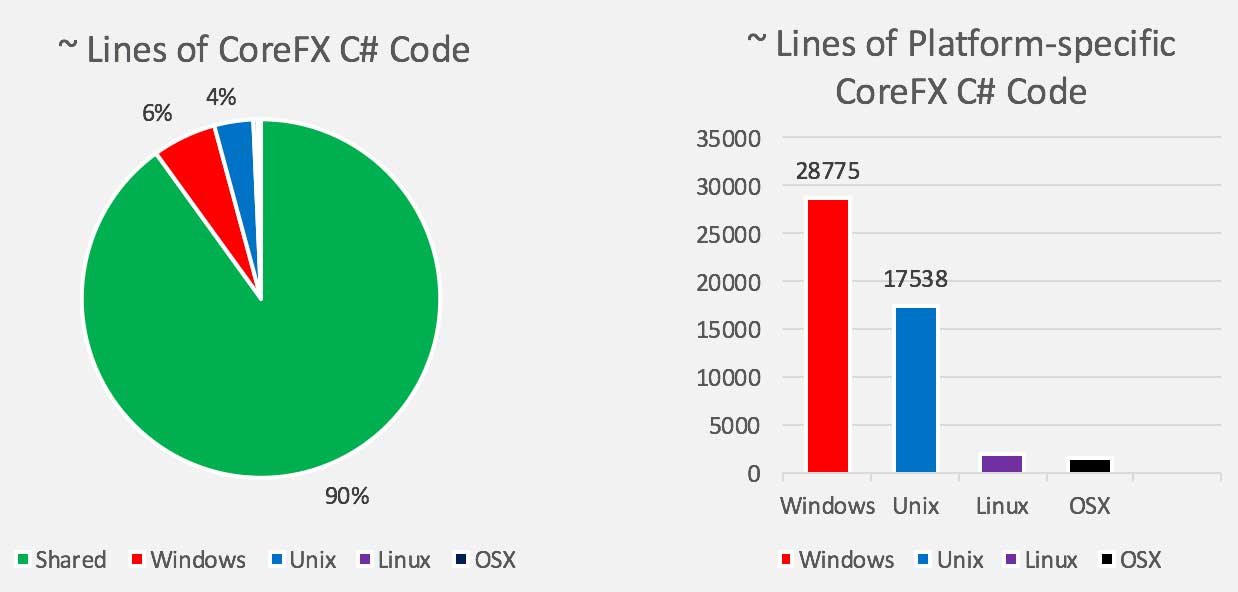
.NET Core ile Geliştirilen Frameworkler Nelerdir?

ASP.NET Core, Windows 10 Universal Windows Platform (UWP) ve Tizen frameworkleri .NET Core ile geliştirilmişlerdir.

.NET Core’un Uyarlanabilirliği Nasıl?

.NET Core, diğer .NET ürünlerine göre çok benzer olmakla birlikte benzersiz bir ürün olarak geliştirilmiştir. Yeni platformlarına ve iş yüklerine geniş kapsamlı adaptasyon sağlayabilir.

.NET Core, çeşitli parçaların farklı zamanlarda yeni plarformlara ve iş yüklerine adapte edilmesini sağlamak amacıyla birkaç parçaya bölünür. Çalışma zamanı ve platformlara özgü temel kütüphanelere bir birim olarak taşınmaktadır. Platform-agnostik kütüphaneleri, tüm platformlarda olduğu gibi çalışmalıdır. Geliştiricinin verimliliğini arttırmak adına platforma özgü uygulamaları azaltma yönünde .NET Core’un uyarlanabilirliği tasarlanmıştır.



.NET Core ve .NET Framework Karşılaştırması

**Uygulama-modelleri:** .NET Core, tüm .NET Framework uygulama modellerini desteklemez. Özellikle ASP.NET Web Form ve ASP.NET MVC’yi desteklemez. Ancak ASP.NET Core MVC’yi destekler. Microsoft .NET Core 3 ile birlikte WPF ve Wİndows Forms’u destekleyeceğini duyurdu.

**API:** .NET Core, farklı bir fakroinge sahip. Büyük bir .NET Framework temel sınıf kütüphanesini alt kümesi olarak içerir. Derleme adları farklıdır, türlere maruz kalan üyeler önemli durumlarda farklılık gösterir. Bu farklılıklar, bazı durumlarda port kaynağında .NET Core’da değişiklik yapılmasını gerektirebilir. .NET Core, .NET Standard API spesifikasyonunu uygular.

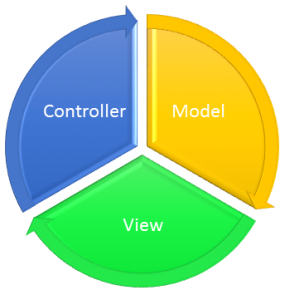
**Platform ve Açık Kaynak:** .NET Framework açık kaynak kodlu olmamakla birlikte sadece **Windows** ve **Windows Server**‘ı destekler. .NET Core ise açık kaynak kodlu olup ayrıca **macOS** ve **Linux** işletim sistemlerini de destekler.

ASP.NET MVC Nedir, Ne İşe Yarar?

*ASP.NET MVC*, MVC pattern’ini ASP.NET’e eklemek için Microsoft’un geliştirdiği framework’tür. ASP.NET MVC’nin ne olduğunu anlamak için öncelikle MVC’nin ne olduğunu incelemekte fayda var.

*MVC*, uygulama geliştirmede (özellikle web uygulaması geliştirmede) önemli yere sahip mimari desenlerden biridir. Günümüzde MVC denince akla Microsoft’un geliştirdiği ASP.NET MVC Framework gelmektedir, oysa 1979 yılından beri (Microsoft 1975 yılında kurulmuştur) yazılım dünyasında yer almaktadır.

*MVC*, Model , View , Controller kelimelerinin baş harflerinden oluşur ve her kelime MVC’nin farklı bir katmanını ifade eder.

[](http://www.borakasmer.com/wp-content/uploads/2014/07/MVC_01.png)

*Model View Controller*

**Model**

MVC dünyasında model uygulama verisinin veya durumunun saklandığı yerdir, genellikle veritabanı veya xml/json dosyası formatındadır.  
Model, veri katmanını (database, xml, json dosyası, vb.) uygulamadan izole eder, böylece diğer katmanlarda veri katmanının neresi olduğunun bilinmesine gerek kalmaz.  
Model katmanı sıklıkla Entity Framework, Nhibernate, LLBLGen, vb. gibi araçlar kullanılarak oluşturulur.

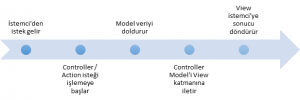
**View**

View, istemcinin gördüğü arayüzü içeren katmandır, genellikle Model katmanındaki verinin kullanılması ile oluşturulur. View katmanının Model ve Controller katmanlarından ayrılması ile arayüz değişikliklerinin uygulamanın diğer katmanlarını değiştirmeye gerek kalmadan yapılabilmesi sağlanmıştır.

View katmanında HTML5 ve CSS3 gibi son versiyon teknolojiler kullanmak mümkündür. HTML5 ve CSS3 ile masaüstü ve mobil tarayıcılarda çalışabilen uygulamalar geliştirmek çok kolaylaşmıştır. *Hatta Windows Store uygulamaları geliştirmek için HTML5 ve CSS3 teknolojilerinden yararlanılabilir.*

**Controller**

Controller, istemciden gelen isteği işlemek, Model ve View katmanları arasında köprü olmak gibi görevleri yerine getirir. Controller içerisinde bir veya daha fazla Action olabilir, genellikle her Action bir web sayfası üretmek için kullanılır.

[](http://www.borakasmer.com/wp-content/uploads/2014/07/MVC_02.png)

*MVC Life Cycle*

MVC’nin diğer bir önemli yapıtaşı Routing mekanizmasıdır.

**Routing**

Routing, istemci’nin uygulamaya yaptığı isteği uygun Controller ve Action’a yönlendiren yapıdır. İstemci, isteği uygulamanın belli bir adresine gönderir, routing mekanizması sayesinde ilgili adres için en uygun Controller ve içerisindeki Action tespit edilir ve çalıştırılır.

**Neden ASP.NET MVC tercih etmeliyim?**

MVC ile istemci’nin isteğine karşılık üretilen çıktı üzerinde çok büyük kontrol imkanı vardır. Bu sayede her alanına müdahele edebildiğimiz ve isteğe en uygun çıktının üretilebilmesi sağlanmıştır.

MVC ile tekrar kullanılabilir (reusable) kod üretmek mümkündür. MVC’nin katmanları birbirinden ayrıldığı için her bir katmanın başka projelerde kullanılabilmesi sağlanmıştır.

MVC ile istemci’nin istek göndereceği adresler üzerinde çok büyük kontrol imkanı vardır. Bu sayede adres’in içerik ile tam bir ilişki içerisinde olması sağlanmıştır. Arama motorları için adres-içerik ilişkisi önemli olduğu için uygulamanın bulunabilirliğine katkısı yüksektir.

MVC ile test edilebilir uygulamalar geliştirme çok kolaylaşmıştır. Katmanların birbirinden ayrı olması ve Test Driven Development yapmayı kolaylaştırıcı mimarisi sayesinde test edilebilir uygulama geliştirilebilmesi sağlanmıştır.

**Sonuç**

MVC uzun yıllardır bir çok framework’te ve programlama dilinde kullanılmış (Java, PHP, vb.) ve olgunlaşmış bir desendir. ASP.NET MVC sayesinde .Net framework dilleri ile MVC pattern kullanılarak hızlı çalışan, test edilebilir, tekrar kullanılabilir parçaları olabilen web uygulamaları geliştirilebilmesi sağlanmıştır.

**KAYNAK**

1. <https://tukanajans.com/blog/web-yazilim/net-core-nedir/>

<http://www.borakasmer.com/asp-net-mvc-nedir-ne-ise-yarar/>

***3.HAFTA:***

***MOBİL UYGULAMA GELİŞTİRME NASIL YAPILIR, AŞAMALARI NELERDİR?***

Kullanımı günden güne artan ve hayatımızı kolaylaştıran mobil uygulamaları geliştirmek için ilk olarak hedef kitlenizi belirleyin.20 Aralık 2018

Günümüzün olmazsa olmazlarından biri olan Mobil Uygulamalar, hayatımızın her alanında, yaşamımızı kolaylaştırıyor.**Mobil Uygulama**kullanımı günden günde hızlı bir şekilde artıyor. Farklı firmalar kullanıcılarının, müşterilerinin veya çalışanlarının hayatlarını kolaylaştırabilmek için mobil uygulamalaröne sürüyor.

E-girişimciler artık sadece e-ticaret ile sınırlı kalmıyorlar. Bunun yerine mobil uygulamalara ağırlık vererek çok daha fazla hedef kitleye ulaşma çabası içerisindeler. Mobil cihaz kullanım sürelerinin artması, kullanıcı yaş aralığının genişlemesi de mobil uygulama geliştirmeye olan ilgiyi arttırıyor.

Mobil uygulamalar, yazılımcılar tarafından geliştirilerek sanal marketlerde müşterilerin hizmetine sunulur. Bu uygulamaların kimisi ücretlidir, kimisi ise ücretsiz olarak kullanıma açıktır ancak her halükarda yazılım geliştiricileri bu uygulamalardan para kazanmaktadır.

**Mobil uygulama geliştireceklere tavsiyeler;**

**Hedef Kitlenizi Belirleyin**

Mobil uygulama geliştirirken sorulması gereken soruların başında mobil uygulamanın kimlere hitap edeceği ve ağırlıklı olarak kimler tarafından kullanılacağıdır. Mobil uygulama geliştirirken hedef kitlenizin ilgisini çekecek detayları atlamayın.

**Basitlik ve Kolaylık**

Bilgiye kolayca ulaştığımız günlerdeyiz. Bunu göz önüne alarak karmaşık ve sıkıcı süreçleri olan uygulamalardansa basit ve kolay olana odaklanmak ve kullanıcı dostu bir mobil uygulama geliştirmek, kullanıcıların uygulama ile etkileşimini arttırmada büyük önem taşır.

**Farklı İşletim Sistemlerine Göre Tasarlayın**

Farklı işletim sistemlerikullanan kullanıcılar hedef kitleniz belirlendiğinde her platforma uyum sağlayabilen bir uygulama geliştirmek de zorunluluk haline gelmekte. Android ve iOS gibi işletim sistemlerine uygun ve sorunsuz çalışan bir uygulama çok daha fazla kişiye hitap etmektedir.

**Uyumluluk**

Mobil uygulamanızın işletim sistemi çeşitliliğinin yanında masaüstü, tablet ve akıllı telefon gibi platformlarda da problemsiz bir şekilde çalışıp çalışmadığını kontrol edin. Mobil uygulamanızın birden fazla cihazdan ulaşılabilirliğinive fraklı cihazlardan girildiğinde uygulama içindeki görsel, yazı gibi etkenlerin ekran genişliğine göre yeniden şekillenmesini sağlayın.

**İyi Tasarım**

İlgi çekici ve canlı tasarım ögeleri ile mobil uygulamanızı çok daha çekici hale getirebilirsiniz. Farklı tasarım ögelerini bir araya getirerek ve renk uyumları ile uygulamanızı ön plana çıkarmayı deneyin.

**Bir hedefe odaklanın**

Mobil uygulama tasarlarken ekranda aynı anda pek çok seçeneğin olması karışıklığa neden olur. Bu sebeple uygulamanızı her sayfada tek hedefe odaklayarak minimalist davranmanız kullanıcılarınızın kafa karışıklığını önler.

**Grid kullanımı önemli**

Mobil uygulamanızı grid yapı ile tasarlamak birbirleri ile orantılı bir deneyim sağladığı için zihindeki görsel bağlantıyı arttırarak rahat bir kullanım sunar.

**Eclipse ve AndroidStudio**

Android bir yazılım için mobil uygulama geliştirebilecek entegre geliştirme ortamı (IDE) olan Eclipse, XML ve Java dosyalarınızı düzenlemeye alabilir ve uygulamanıza ait olan kanalları oluşturmaya başlayabilirsiniz. Eclipse’i Google’dan edinebilir, uygulamanızı geliştirebilir ve hatta uygulamanızı bir üst sürüme taşıyabilirsiniz.

AndroidStudio, Eclipse’e göre daha hızlı hizmet veren bir alt yapıya sahiptir ancak Eclipse’e göre daha fazla hata çıkarması söz konusudur. Bu yönden genel olarak, daha hızlı bir şekilde geleceğe ilerlemek isteyenlerin tercih ettikleri platformun AndroidStudio olduğunu söyleyebilmekteyiz.

**Android SDK Aracılığı ile ADB**

ADB’nin amacı uygulama geliştiricilerini asiste etmektir. ADB aracılığı ile Android cihazların yazılımları üzerinde değişiklik gerçekleştirebilir ve uygulama yüklemesi yapabilirsiniz.

**Android Developer Guidelines**

Bir rehber görevi gören Android Developer Guidelines resmi bir yazılım olarak uygulama geliştiricilere yardımcı olmakta ve ipuçları vermektedir. Temel olarak IDE ve ADB bilgisine sahip olmak yetecektir.

**Android Design Guidelines**

Android yazılım işi için mucizevi nitelikte olan bir mobil uygulama geliştirme platformları arasında estetik ve görsel bilginizi konuşturmaya başlayacağınız platform Android Design Guideliness’tir.Temel animasyonlar, sayfalar ve daha birçok içeriği oluştururken görsel güçten faydalanmak için bu rehberi kullanmak oldukça faydalı olacaktır.

**GitHub ve BitBucket**

Android uygulamaları çok sayıda dosya ve klasörden meydana gelmektedir ve her birinin düzeni, varlığı, ismi, numarası gibi ince detayları yazılımın doğru işlemesi için çok büyük önem taşımaktadır.

**IOS’dan bahsedecek olursak:**

JS ve HTML5 kullanarak ios’ta mobil uygulama geliştirmek mümkün. Ancak HTML5 ve JS’yi kullanarak mobil uygulama geliştirmek mantıklı görünüyor olabilir fakat bunu yapmak için framework kullanmamız gerekmekte. Frameworkten kısaca bahsedecek olursak, kullanılmaya hazır kod parçalarının birleşmesine denir. Bu frameworkler, tıpkı native uygulamalardaki gibi cihazın donanımına erişebilme, tasarım bütünlüğü oluşturabilme vb. gibi mobil uygulama ihtiyaçlarını karşılayabilmeyi sağlıyor. Bunlar olmaksızın bir uygulama geliştirmek oldukça zor ve kullanışsız olacaktır.

JS, CSS3 ve HTML5 yazılım dillerini kullanarak olağanüstü efektler ve cihazın neredeyse tümüne erişim imkanısağlanması sayesinde birçok işlem içeren uygulamalar yapılabilmektedir

***MOBİL PROGRAMLAMA DİLLERİ NELERDİR?***

Bugün dünya üzerindeki toplam uygulama sayısı milyonlarla ifade ediliyor. Bu gerçekten müthiş bir iletişim, oyun, haber, teknoloji ve eğlence havuzu demek. Üstelik her geçen gün bu uygulamaların sayısı biraz daha artıyor. Mobil cihazların aktif bir şekilde kullanıldığı çağımızda uygulamalar gerçekten de büyük ilgi uyandırıyor.

Teknoloji şirketleri web uygulamalarının ardından doğal olarak mobil uygulamalara da çok hızlı bir şekilde adapte olmak zorunda kaldı. Mobil uygulamalarla şirketler kadar bireysel geliştiriciler de yakından ilgileniyor. Gerek yaratıcı fikir düzeyinde gerekse de teknik düzeyde bireysel üretimler oldukça değerli. Kısa süre içinde sadece bir mobil uygulama sayesinde yüksek kazançlar elde etmek mümkün. Tam manasıyla sektör haline gelen bu alanda rekabet de üst seviyeye çıkmış durumda.

**1. C Dili Programlama**

C dili 1970’li yılların hemen başında ortaya çıkarılmıştır. Aynı dönemde sadece bu dilin detaylarını ve inceliklerini anlatan 800 sayfa civarında bir kitap bile basılmıştır. İlk yıllarda Unix işletim sistemine yönelik olarak yoğun şekilde tercih edilmiştir. Uygulama süreçlerinde en sık tercih edilen dildir. Bu dilin kullanım süresi 30 yıldan fazladır. Gerek işletim sistemleri gerekse de mobil uygulamaları büyük ölçüde bu dili benimsiyor. Bu dil diğer dilleri öğrenme kolaylığı da sağlamaktadır. Pek çok açıdan en temel programa dili olarak kabul görür. Genelde yeni başlayanlar için zorlayıcı olabilmektedir; ancak benimseyince deyim yerindeyse çocuk oyuncağına dönüşmektedir diyebiliriz.

**2. C++ Dili Programlama**

Programa dilleri içinde en kolaylarından bir tanesi olarak kabul görür. C’nin nesnel tabanlı bir türüdür. Aynı şekilde hem mobil uygulamalarda hem de işletim sistemlerinde rahatlıkla kullanılabilir. Özellikle mobil uygulama alanında tercih dildiğinde seri programların oluşmasına olanak sağlar. Düşük seviye programlama dilleri arasında ilk sıradadır.

**3. Objective-C Programlama**

Genellikle IPhone ya da İPad benzeri cihazlara yönelik hazırlanan uygulamalarda kullanılır. Bir başka deyişle İOS işletim sisteminde rahatlıkla tercih edilmektedir. Genel anlamda C’yi andıran bu dil, öğrenme aşamasında sizi pek de zorlamayacaktır.

**4. Java Programlama**

Genelde tek bir defa yazıldıktan sonra işlevsel biçimde pek çok yerde kullanılabilen JAVA, son derece verimli ve kullanışlıdır.  Müthiş bir popülerliğe sahip olan JAVA, işletim sistemlerinin hemen hemen tamamına uyumludur. Öte yandan pek çok cihazda sorunsuz şekilde çalışabilmektedir. Akıllı telefonların varlığı Java’nın tahtını bir nebze olsun sallamış olsa da hala itibarını koruyor diyebiliriz. Akıllı telefonların varlığına ve yeni işletim sistemlerine rağmen Java hala en çok tercih edilen dil unvanına sahip. Taşınabilir bir programa dili olan Java ile yazılan kodlar, pek değişikliğe ihtiyaç duymaz. Java’nın geçmişi 1991 senesine kadar uzanıyor. Aslında o dönem akıllı televizyonlar için ön görülmüştü. Dil kadar bir “ortam” niteliğine de sahip olan Java, elektrikli ev aletleri ve oyun aygıtlarında da karşımıza sıklıkla çıkabiliyor.

**5. HTML 5 Programlama**

HTML genelde programla dili olarak görülmez. O daha çok bir işletim sistemi olarak tanımlanır. Sadece mobil uygulama değil web uygulamaları geliştirmek de mümkündür. Öğrenme sürecinin biraz daha kolay olacağı HTML 5, işaretleme dili olarak da bilinir. Metin odaklı bir dil olan HTML, birbirinden farklı türlerde olan materyalleri bağlama işlevine sahip. Yani bir başka deyişle yazı, resim ve video gibi birbirinden ayrı içerikler HTML 5 sayesinde birbirine bağlanmış olur. HTML aynı zamanda uzun vadede geliştirilmeye ve reklam çalışmalarına çok müsaittir.

**6. C# Dilinde Mobil Programlama**

C Sharp olarak bilinen bu dil, Miccrosoft tarafından geliştirilmiştir. Temel amacı C veya C++ üzerinde yer alan bazı hasarları ve çelişkileri yok etmektir. Nesnel tabanlı bir dil olan C Sharp, büyük ölçüde Java’yı anımsatır. Derleyici yönü baskındır ve bu özelliği, hataları minimum seviyeye çeker. Uygulama aktif bir şekilde çalışmaya başladıktan sonra derleyici kısa sürede hataları ayıklar ve farklı bir ekranda özel olarak sunulur. Geliştirici aynı anda bu hataları ortadan kaldırarak büyük bir kolaylık sağlamaktadır.

**7. Swift Dilinde Mobil Uygulama**

Apple firması tarafından ortaya çıkarılan bu dil, İOS işletim sistemine yönelik olarak işlev görmektedir. Açık kayaklı olduğu için çoğu zaman kullanıcıların dostu olarak görülen bu Swift, kolay kullanımına rağmen son derece güçlü bir yapıya sahiptir.

***REACT NATİVE NEDİR?***

React Native için kısaca Facebook tarafından üretilen, cross-platform mobil uygulama geliştirme olanağı sağlayan bir framework diyebiliriz.

“**Cross-platform mobil uygulama geliştirme framework’ü**” ne demek derseniz; “**Ben tek bir dil ile mobil uygulama geliştireyim; ama hem Android, hem iOS, hem Windows … vs ortamlarında çalışsın. Kalkıp da Android için Java, iOS için Swift öğrenmekle uğraşmayayım!**” dediğiniz de sizin imdadınıza koşan diller, framework’lerdir.

React Native’de bunlardan biri. Kendine özgü bir mobil uygulama uzantısı yok, uygulamayı geliştiriyorsunuz ve Android için **.apk**, iOS için **.ipa** uzantılarını alıyorsunuz. Android ortamına uygulama geliştirirken karşılaşacağınız hatalar Java hataları, iOS ortamına uygulama geliştirirken karşılaşacağınız hatalar Objective-C hataları. Ama kodlarken kullanacağınız dil React.

Kısa, öz ve açık bir giriş oldu ise asıl konumuza geçebiliriz.

Kendine has bir dil formatı (JSX) olan React Native bizlere tek bir dil üzerinden kodlama yapabilme ve geliştirilen uygulamanın bir çok platformda çalışma olanağını sunuyor dedik. Geliştiricilere bu desteği sağlayan React Native, cihaz ile arayüz arasında bir köprü görevi görerek geliştirilen mobil uygulamaların sorunsuzca çalışmasını sağlıyor. Siz geliştirme ortamında bir bileşen tanımlıyorsunuz (örneğin; **<Text>**) ve React Native ilgili ortamda o bileşenin karşılığı ne ise (Android’de **TextView**, iOS’da **UIView**) sizin yerinize oluşturuyor. Bunun için de yazılan React (yani Javascript) kodlarını native dile (Java, Objective-C .. vs) çevirirken “babel” adı verilen bir yapı devreye giriyor.

***Ne işe yarar? Nerede kullanılır?***

React Native bir front-end (*ön-yüz)*framework’üdür*.*Bilmeyenler için front-end; bir uygulamanın ara yüzünde ki tasarımsal ve operasyonel işlemlerin bütününü ifade eder. Haliyle de bir front-end framework’ü bu gereksinimleri karşılayan, sağlayan bir yapıdır diyebiliriz. Back-end (*arka-yüz)* ise bir uygulamanın arka planında ki gereksinimleri sağlayan (*makine öğrenmesi işlemleri, veri desteği, restful API servisi vs..*) işlemler topluluğudur. Back-end tarafında dillere odaklanılmaz, sunulan servis hizmetleri göz önünde bulundurulur. Python, nodeJS, .NET gibi diller ile back-end servisleri oluşturabilirsiniz, fakat bu diller back-end için yaratılmamıştır. Oysa React Native front-end’e yönelik bir framework’dür.

Peki React Native tek başına mobil bir uygulama için yeterli mi? Uygulamaya bağlı. Ama açıkçası pek sayılmaz. Güncel bir mobil uygulama sağlam bir back-end, en azından iyi bir servis desteği istemekte ve haliyle bu noktada geliştireceğiniz uygulamanın front ve back end dillerinin ve platformlarının uyumlu bir şekilde çalışmasını sağlamak önemli bir husus. Ne mutlu ki React Native arkasında ki topluluk desteği ile bir çok back-end servislerini destekleyebiliyor. MongoDB, SQLite, firebase, MySQL, GraphQL, realm, Flask, NodeJS ve bunlar gibi bir çok seçenek ile mobil uygulamalarımıza iyi bir back-end desteği verebiliyoruz.

***KAYNAK:***

1. <https://ata.com.tr/arama?key=-%20mobil%20uygulama%20geli%C5%9Ftirme>
2. <https://www.mediaclick.com.tr/blog/mobil-programlama-dilleri-nelerdir>
3. <https://medium.com/kodcular/react-native-nedir-7b333d319597>

**4. HAFTA:**

**Veri Madenciliği Nedir?**

Basit bir tanım olarak veri madenciliği büyük ölçekli veriler arasında bilgiye ulaşma veya bilgiyi madenleme işidir. Büyük veri yığınları içerisinden gelecekle ilgili tahminde bulunabilmemizi sağlayabilecek bağıntıların bilgisayar programı kullanarak aranmasıdır. Bunun dışında bilgi keşfi, bilgi madenciliği, bilgi çıkarımı, veri/model analizi, veri arkeolojisi… Veri Madenciliği için kullanılan alternatif terimlerdir. Kısacası büyük veri yığınları arasında görülmeyen bilgiyi ortaya çıkarma işlemidir.  
**→**Gelecek 5 sene şirketimiz hangi alanda büyüyebilir, bu yıl en iyi 10 müşterimiz kim olacak, hastanın hastalığı ne olabilir… gibi kısacası bilinmeyen bilgiyi ortaya çıkarır.

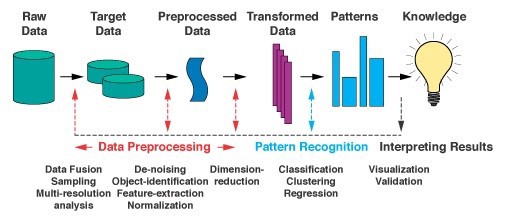
**Veri Madenciliğinin Kısa Tarihçesi**

* 1950’lerde ilk bilgisayarlar matematiksel sayımlarda kullanılıyordu
* 1960’larda Veri Koleksiyonları, Veri tabanı kullanımı başladı
* 1970’lerde İlişkisel veri modeli ve İlişlisel RDMS uygulamaları geliştirildi
* 1980’lerde İlişkisel RDMS kullanımı yaygınlaşmaya başlıyor.
* 1990’larda Günlük işlerde derlenen verinin nasıl değerlendirilebileceği sorgulanmaya başlıyor.
* 1991’de Knowledge Discovery in Real Databases tanımı ve kavramları ortaya konuluyor
* 1992’de Veri Madenciliği konusunda ilk yazılımın geliştirilmesi
* 2000’lerde Veri Ambarları ve Veri Madenciliğinin yaygınlaşması oluyor.

**Uygulama Alanları**

* **CRM Yönetimi**  
  - Pazarlama kampanyasında getirinin maksimizasyonu  
  - Müşteri Sadakatının artırılması
* **Pazarlama**- Müşteri Satın alma alışkanlıkları belirlenmesi  
  - Pazar Sepeti Analizi (Market Basket Analysis)  
  - Satışların Tahmini (Sales Forecasting)
* ***Bankacılık & Finans Sektörü***- Kredi kartı harcalamarına göre müşteri gruplarının belirlenmesi  
  - Kredi taleplerinin değerlendirilmesi
* **Elektronik Ticarette**- Saldırıların Tespiti (Intrusion Detection)  
  - Web Sayfalarına yapılan ziyaretlerin çözümlenmesi  
  - Kullanıcı davranışlarına göre web sitesinin yenilenmesi
* **Sigortacılık**
* **Sağlık & İlaç Sektöründe,**
* **Spor Bilimlerinde…vb**

**Veri Madenciliği Bilgi Keşif Süreci**

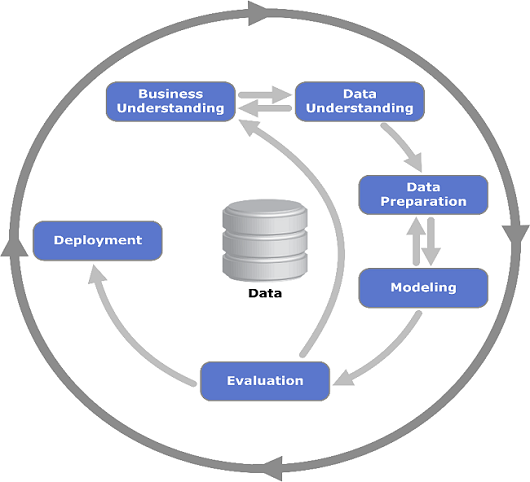


**1.) Veri Temizliği (Data Cleaning)**Bu adımda eksik, gürültülü, tutarsız verilerin temizlenme sürecidir.  
**2.) Veri Bütünleştirme (Data Integration)**Birçok veri kaynağından alınan verilerin birleştirme sürecidir.  
**3.) Veri Seçme (Data Selection)**Veritabanından alınan analiz ile ilgili verilerden, probleme ilişkin olan verileri seçme sürecidir.  
**4.) Veri Dönüştürme (Data Transformation)**Bu aşamada verinin uygun formlara dönüştürülüp veri madenciliğinde kullanılabilecek hale getirme sürecidir.  
**5.) Veri Madenciliği Uygulaması (Data Mining)**Hazırlanan veriler üzerinden amacına göre Veri Madenciliği Algoritmalarının uygulanma sürecidir.  
**6.) Desenler (Pattern Evaluation)**Bazı ölçümlere göre elde edilmiş bilgiyi temsil eden örüntüler tanımlama sürecidir.  
**7.) Bilgi Sunumu (Knowledge Presentation)**Veri madenciliği elde edilmiş bilginin kullanıcıya sunulmasıdır.

**Veri Madenciliği Metodolojileri**

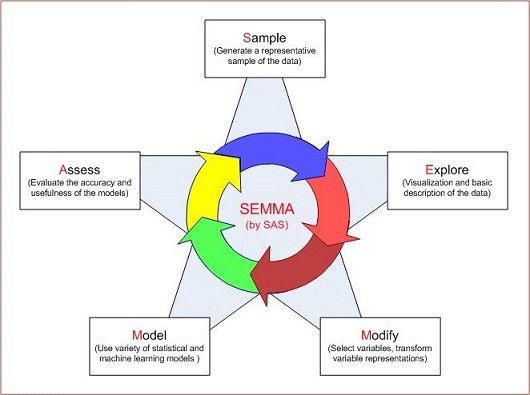
Metodolojiler, önemli veri madenciliği sorunlarını daha iyi anlamak ve süreçlerinin nasıl yapılması gerektiğini ifade eden yöntemlerdir. Bu metodolojilerden en çok kullanılan CRISP-DM ve SEMMA metodolojileri hakkında genel olarak bilgi vermeye çalışacağım. Bu metodolojiler dışında ise şirketlerin kullandığı kendine özel metodolojileri de bulunmaktadır.

* **CRIPS-DM Metodolojisi (Cross-Industry Standard Process for Data Mining)**



CRIPS-DM analitik, veri madenciliği ve veri biliminde en popüler metodolojidir. Veri madenciliği projelerini planlama ve yürütmede kullanılan bir süreç modelidir. Bu model 6 aşamadan oluşmaktadır.  
**1. İşi Tanımlama (Business Understanding):**Başlangıç olarak proje hedeflerini ve ihtiyaçlarını anlama ve bunu veri madenciliği tanımına dönüştürme aşamasıdır.  
**2. Veriyi Anlama (Data Understanding):**Bu aşamada veri toplama işlemiyle başlar, veri kalitesi problemlerini belirleme, veriden ilk görüleri çıkartma.. diye verinin probleme ne kadar çözüm getirdiğiyle devam eder.  
**3. Veriyi Hazırlama (Data Preparation):**Topladığımız veriden veri seçme, veri temizleme, veri dönüştürme… gibi model uygun son veri setini elde etmek için yapılan işlemlerdir. **4. Modelleme (Modeling):**Bu aşamada çeşitli modelleme tekniklerinin ve algoritmalarının seçilmesi, parametrelerin seçilmesi ve uygulama işlemleri gerçekleştirilir. **5. Değerlendirme:** Bu aşamada oluşturulan modelin deneme ve gözden geçirilmesi yapılır, gerekiyorsa iyileştirmeler yapılır.  
**6. Uygulama:** Son aşamada ise modelin analistlere ve son kullanıcılara sunulup iş süreçlerinde kullanılacak hale getirilir.

* **SEMMA Metodolojisi (*Sample*,*Explore*, *Modify*, *Model* and *Assess*)**



*Sample*, *Explore*, *Modify*, *Model*, ve *Assess*kelimelerinin baş harflerinden oluşan bir metodolojidir. İstatistik ve İş Zekası yazılımı geliştiren SAS Enstitüsü tarafından geliştirilen ardışık adımlar listesidir.  
CRISP-DM’den farkı ise CRISP-DM olduğu gibi bütün projenin metodolojisi iken SEMMA ise veri madenciliği yapılan kısmın metodolojisidir.

1. **Sample:**Bu aşamada veri örnekleme ile başlar yani modelleme için veri seti seçilir.
2. **Explore:**Beklenen ve beklenmeyen değişkenler arasında ilişkileri ve anormallikleri keşfedilerek verilerin anlaşılır hale getirilir
3. **Modify:**Modelleme süreci için verilerin temizlenmesi ve dönüştürülmesi yapılır
4. **Model:**Eğilim ve tahminleri keşfetmek için modelin verilere uygulanmasıdır
5. **Assess:**Bu aşamada uyguladığımız modelin sonucumuza uygunluğunun değerlendirilmesi yapılır

**Veri Madenciliği Modelleri (Data Mining Models)**

Veri madenciliğinde kullanılan modeler Tahmin Edici ve Tanımlayıcı olmak üzere ikiye ayrılmaktadır. **Tahmin Edici Modeller (Predictive):**Sonuçları bilinen verilerden hareket ederek bir model oluşturup, sonuçları bilinmeyen veri kümeleri için sonuç değerlerinin tahmin edilmesidir.  
— Sınıflama  
— Regresyon  
— Zaman Serisi Analizi **Tanımlayıcı Modeller (Descriptive):**Karar vermeye rehberlik etmede kullanılabilecek verilerdeki örüntülerin tanımlanmasını sağlamaktadır.  
— Kümeleme Yöntemi  
— Birliktelik Kuralı

**Veri Madenciliği Yöntemleri (Data Mining Techniques)**

**1. Birliktelik Kuralları (Association Rules)**

Veri madenciliğinde en iyi bilinen yöntemlerden biridir. Büyük veritabanlarında birbiriyle ilişkili değişkenleri ve aralarında bağlantının büyüklüğünü tespit etmek için kullanılan bir yöntemdir. Apriori, Carma, Eclat, Sequence, GRI.. birliktelik yönteminde kullanılan algoritmalardır.  
***Örneğin***alışveriş alışkanlıklarında markette ekmek alanlar yüzde kaç oranında süt veya ekmek,süt alanlar yüzde kaç oranında peynir alıyor gibi ilişkileri tespit edebiliriz.

Ekmek → | %70 (satılan ürünlerde %70 oranında ekmekte alınıyor.)  
Ekmek → Süt | %50 (ekmek alanlar %50 oranında sütte alıyor.)  
Ekmek,Süt → Peynir | %40 (ekmek ve süt alanların %40 oranında peynir alıyor)

**2.Sınıflandırma ve Tahmin (Classification and Predicton)**

Gelecekteki veri eğilimlerini açıklamak için bir nesnenin niteliklerini inceleme ve bu nesneyi önceden tanımlanmış bir sınıfa atamaktır. Decision Tree, Random Forest, Navie Bayes, KNN.. sınıflandırma yönteminde kullanılan algoritmalardır.  
***Örneğin***kredi başvurusu yapacak bir müşteriye kredi verilebilirliği, Geçmiş bilgilerden hastalık teşhisi, Ses tanıma, kullanıcı davranışlarını belirleme.. birer sınıflandırma örnekleridir.

**Tahminleme:**Veri seti içinde bilinmeyen veya eksik olan sayısal verilerin tahmin edilmesidir.

**3.Kümeleme Analizi (Cluster Analysis)**

Kümelemede amaç dağınık halde duran verileri özelliklerine göre birleştirip işlenebilir hale getirtmektir. Sınıflandırmaya benzer ama aradaki fark kümelerin önceden belirlenmemiş olmasıdır. Bu analiz sonucu müşteri profili oluşturmak içinde kullanılır. K-Means, K-Metoids.. algoritmalar kümeleme yönteminde kullanılan algoritmalardır.  
***Örneğin***marketlerde farklı müşteri gruplarının keşfedilmesi ve bu grupların alışveriş örüntülerinin ortaya konması, biyolojide bitki ve hayvan sınıflandırmları ve işlevlerine göre genlerin sınıflandırılması, şehir planlamasında evlerin tiplerine,değerlerine ve coğrafi konumuna göre gruplara ayrılması.. kümeleme örnekleridir.

**4. Aykırılık Analizi (Outlier Analysis)**

Verilerin algoritmalar ile kontrol edilerek verilerde aşırı sapma veya aykırı değerlerin bulunma sürecidir. Sıradışı veriler okuma, kayıt etme, ölçüm gibi hatalardan oluşmaktadır.Veri madenciliği algoritmaları ise bu sıradışı verileri en aza indirme veya ortadan kaldırmayı amaçlamaktadır.  
***Örneğin***: kredi kartlarının olağandışı kullanımının tespiti, telekomunikasyon servislerinde olağandışı dolandırıcılık tesipiti, tıbbı tedavilerde olağandışı sonuçları bulmak.. için kullanılmaktadır.

**KAYNAK**

1. <https://medium.com/@furkanalaybeg/veri-madencili%C4%9Fi-ve-y%C3%B6ntemleri-d0e2fd238e44>

***5.HAFTA:***

***DERİN ÖĞRENME***

Yapay zeka alanı, makinelerin deneyimle öğrenebileceği ve insan katılımı olmadan beceriler kazanabileceği makine öğrenimini kapsıyor. Derin öğrenme ise, yapay sinir ağlarının ve insan beyninden ilham alan algoritmaların veriden öğrendiği bir makine öğreniminin alt kümesidir. İnsanların tecrübelerinden öğrendiklerine benzer olarak, derin öğrenme algoritması, sonucu iyileştirmek adına her defasında biraz değişiklik yaparak daha iyi bir iş çıkarmaktadır. Derin öğrenme, düşünce gerektiren herhangi bir problem hakkındaki çözümü gerçekleştirebilir.

Her geçen gün üretilen ve 2,6 milyon bayt olarak tahmin edilen veri miktarı oldukça şaşırtıcı. Derin öğrenme algoritmaları, öğrenmek için çok fazla veri gerektirdiğinden, veri oluşturmadaki bu artış, derin öğrenme yeteneklerinin son yıllarda artmasına neden olmuştur. Buna ek olarak, derin öğrenme algoritmaları, yapay zeka (AI) teknolojisinin geliştirilmesi ile birlikte, bugün mevcut olan daha güçlü bilgi işlem gücünden yararlanır.

AI, bir hizmet olarak küçük kuruluşlara yapay zeka teknolojisine erişim ve büyük bir ilk yatırım olmadan derin öğrenme için gereken algoritmaları vermiştir. Derin öğrenme, makinelerin çok çeşitli, yapılandırılmamış ve birbirine bağlı bir veri kümesi kullanırken bile karmaşık sorunları çözmesini sağlamaktadır. Derin öğrenme algoritmaları ne kadar çok öğrenirse, o kadar iyi performans göstermektedir.

**Derin Öğrenme Örnekleri**

Artık makineler, karmaşık problemleri insan müdahalesi olmadan çözmeyi öğrenebildiğine göre, tam olarak ne ile mücadele ediyorlar? Derin öğrenmenin bugün desteklediği görevlerden sadece birkaçını aşağıda listeledik. Algoritmalar veri akışı yoluyla öğrenmeye devam ettikçe, listenin büyümeye devam etmesi bekleniyor.

**1. Sanal Yardımcılar**



İster Alexa, ister Siri veya Cortana olsun, çevrimiçi servis sağlayıcıların sanal asistanları, konuşmaları ve insanların onlarla etkileşime girdiklerinde kullandıkları dili anlamalarına yardımcı olmak için derin öğrenme kullanıyor.

**2. Çeviriler**

Benzer şekilde, derin öğrenme algoritmaları otomatik olarak diller arasında çeviri yapabiliyor. Bu özellik, özellikle seyahat edenler, iş adamları ve hükümet çalışanları için oldukça önemlidir.

**3. Sürücüsüz Araçlar**

Sürücüsüz bir aracın yolun gerçekliğini anlama şekli, yoldaki dur işaretine veya başka bir araç olup olmadığına nasıl cevap vereceği, derin öğrenme algoritmaları yoluyla gerçekleşir. Algoritmalar ne kadar çok veri alırsa, o kadar çok insan davranışlarını gerçekleştirebilir.

**4. Sohbet Robotları ve Servis Botları**



Birçok şirket için müşteri hizmeti sağlayan sohbet robotları ve servis botları, derin öğrenme sayesinde artan miktarda işitsel ve yazılı sorulara akıllıca ve yardımcı bir şekilde cevap verebiliyor.

**5. Görüntü Renklendirme**

Siyah beyaz görüntüleri renkli hale getirmek, önceleri insan eliyle yapılıyordu. Bugün, derin öğrenme algoritmaları, resimlerdeki kaynağı ve nesneleri, siyah beyaz görüntüyü renkli olarak yeniden oluşturmak için kullanabilmekte. Sonuçlar ise oldukça doğru ve etkileyici.

**6. Yüz tanıma**



Derin öğrenme yüz tanıma özelliği, yalnızca güvenlik amaçlı değil, aynı zamanda Facebook gönderilerindeki insanları etiketlemek için de kullanılıyor. Yakın gelecekte yüzlerimizi kullanarak bile bir mağazadaki ürünler için ödeme yapabiliriz. Yüz tanıma konusundaki derin öğrenme algoritmalarının karşılaştığı zorluk, kişilerin saç stilleri değiştiğinde, sakal bıraktıklarında ya da tıraş olduklarında veya alınan görüntünün kötü aydınlatma gibi nedenlerle zayıf olduğu durumlarda bile aynı kişinin yüz tanımlamasını yapabilmeleridir.

**7. İlaç ve Tıbbı Ürünler**

Hastalık ve tümör tanılarından, bir bireyin taşıdığı genler için özel olarak oluşturulan kişiselleştirilmiş ilaçlara, tıp alanındaki derin öğrenme, büyük ilaç ve tıbbi şirketlerin çoğunun dikkatini çekmektedir.

**8. Kişiselleştirilmiş Alışveriş ve Eğlence**

Netflix daha sonra izlemeniz gerekenler için önerileri nasıl bulabiliyor? Ya da Amazon, bir sonraki alışverişinizde satın almanız gerekenler hakkında nasıl fikir edinebiliyor ve bu öneriler nasıl tam da sizin ihtiyacınız olan şeyler olabiliyor? Bu soruların cevabı; derin öğrenme algoritmaları. Derin öğrenme algoritmaları ne kadar deneyim sahibi olursa o kadar iyi performans sergiliyor. Teknoloji gelişmeye devam ederken, derin öğrenme algoritmalarıyla ilgili ilginç gelişmelerle karşılaşacağız gibi duruyor.

**KAYNAK**

<http://www.yapayzekatr.com/2019/12/16/derin-ogrenme-nedir/>

6.HAFTA:

JavaScript Nedir?

JavaScript, web sayfalarına canlılık ve dinamiklik getirmek için oluşturulmuş bir betik dildir.   
HTML koduna gömülü ya da dahili olarak yazılıp sayfa yüklendiğinde doğrudan çalıştırılabilir. Dolayısıyla kodlar herhangi bir derleyiciye ihtiyaç duymadan çalışabilir.   
Bunları ele aldığımızda JavaScript diğer dillere kıyasen oldukça farklıdır.

JavaScript'in Tarihçesi

İlk olarak 1995 yılında NetScape Şirketinde çalışan Brendan Eich tarafından yazılmıştır.   
JavaScript ilk başta "LiveScript" adıyla yazılmıştır. Fakat Java dili o zamanlarda çok fazla gündemde olmasından dolayı yeni bir dil ve "küçük kardeş" olarak görüldüğünden dolayı adını JavaScript olarak almıştır.  
Fakat bu olay kötü sonuçlar doğurmuştur;  
Java o zamanlarda çok popüler ve güçlü bir dil olduğundan dolayı kullanıcılar JavaScript'i Java dilinin daha basit ve yararsız bir alt dili olarak algılamıştır. Bu algının etkileri günümüzde bile halen etkisini göstermektedir.  
JavaScript'in geliştirilmesi günümüzde NetScape veya günümüz adıyla Mozilla tarafından yapılmaktadır.

*JavaScript'i Eşsiz Kılan Nedir?*

JavaScript'i eşsiz kılan 3 temel neden vardır;

    •    HTML/CSS ile beraber çalışması,

    •    En çok bilinen tüm tarayıcılarda ön tanımlı olarak gelip sorunsuz şekilde çalışması,

    •    Kolay işleri kolay bir şekilde yapabilmeyi sağlamasıdır.

Bu özelliklerin hepsini içinde barındıran başka hiçbir tarayıcı teknolojisi yoktur.  
Bu yüzden JavaScript en çok kullanılan tarayıcı teknolojisidir ve onu eşsiz kılan da budur, günümüzde hemen hemen bütün tarayıcıların altyapısında da JavaScript çalışır.

Özet Olarak

İlk başta sadece ağ tarayıcılarında çalışmak üzere tasarlanmış bir dil olsa da günümüzde birçok yerde(client-side ve native mobil uygulamalar gibi) çalışabilir.  
HTML ve CSS ile kusursuz uyumluluğundan dolayı dengi yoktur.  
Her ne kadar ilk başta Java dilinin yeteneksiz bir alt dili olarak görülse de günümüzde çok gelişmiş bir programlama dilidir ve Java ile alakası yoktur.

***JQUERY***

Jquery kullanımıyla birlikle, internet Explorer ve benzer özelliklere sahip olan 6 farklı tarayıcıda meydana gelen uyumsuzlukların çözümleri kolayla hazırlanmaktadır. Birçok sitede meydana gelen teknik problemlerin kolaylıkla giderilmesinde ve sitelerin hazırlık aşamasında gerek duyulan uzun kodların daha kısa olarak kullanılmasında büyük etkisi bulunan, dijital bir kütüphanedir. Jquery, Google robotlarının yapmış olduğu Ajax Siteleri taramaları esnasında kullanılan yaygın bir tekniktir.

Jquery Özellikleri Nelerdir?

Jquery kütüphanesi, bugünün en gelişmiş yazılım kodlarının yer aldığı uygulamadır ve site kurucuları tarafından oldukça fazla tercih edilmektedir. Jquery hakkında bilgi sahibi olmak isteyenlerin, aşağıdaki özellikleri incelemesi yeterli olacaktır:

* HTML ve DOM nesnelerinin kolaylıkla işlenmesini sağlamaktadır.
* CSS işlemleri hakkındaki tüm formları oluşturmaktadır.
* HTML olay metotlarını işlemektedir.
* Siteler üzerinde çeşitli efekt ve animasyonların oluşturulmasını sağlamaktadır.
* AJAX sitelerinin tüm açıkları için çözüm üretmektedir.
* Hizmet programlarının geliştirilmesini sağlamaktadır.

Jquery kütüphanesi, belirtilen işlemlerin tamamı için farklı eklentileri içinde barındırmaktadır. Bugün kullanılan JavaScript için belirlenen pek çok geliştirici adına framework oluşturulmuştur ancak, Jquery uygulaması bu geliştiriciler arasında en donanımlı olanıdır.

Jquery Kullanımının Sağladığı Avantajlar Nelerdir?

Jquery kütüphanesinin kullanılması sonucunda elde edilen getiriler şunlardır:

* Jquery ile hazırlanan sitelerdeki nesne seçimleri ve etkilenmeleri, daha hızlı şekilde ilerleyecektir.
* HTML içerisinde geliştirilen animasyonların daha kolay şekilde ilerlemesini sağlamaktadır.
* Css programlarıyla birlikte kullanıldığı zaman, kusursuz şekilde işler ve herhangi bir problemle karşılaşılmasına izin vermez.
* Sürekli geliştiği için yeni sürümleriyle daha hızlı ve kapsamlı bir kütüphane haline gelmektedir.
* Jquery kütüphanesi, ücretsiz şekilde işleyen kapsamlı bir açık kaynaktır.
* JavaScript kullanımı için gerekli olan uzun kodların yazılımı yerine, daha kısa ve pratik kodlar ve uygulamalar geliştirmiştir.
* Jquery kütüphanesinin kullanımı öğrenmek, oldukça kolaydır. HTML, CSS ve JavaScript hakkında ufak bilgilere sahip olan kişiler, Jquery kullanımı hemen kavrayacaktır. JavaScript hakkında herhangi bir bilgiye sahip olmayan kişiler de kolayca Jquery uygulamasını kavrayabilir.
* Jquery içerisinde bulunan çeşitli metotlar arasından, istenildiği gibi tercih yapılabilir ve bu elemanlar istenilen alanlara uygulanabilir.

Dünya üzerindeki en büyük sitelerden olan, Google, Facebook ve Microsoft IBM gibi uygulamaların tamamı, yazılım geliştirmeleri esnasında Jquery kütüphanesini tercih etmektedir.

Node.js: Nedir ne değildir?

Node.js, JavaScript’dir.  
Web tarayıcılarına muhtaç kalmadan heryerde çalışabilir.  
Java ile yakından uzaktan ilgisi yoktur.

**Ne işe yarar?**

Temelde iki şeye:

1. Bir yazılımın hem arkayüz (backend) hem önyüz (frontend) kodlarını aynı dilde yazmanın tek yoludur diyecektik ama itiraz edenler çıkacaktır. O yüzden en kolay yoludur deyip geçelim.
2. Günümüzde çoğu uygulama, zamanının çoğunu veritabanlarına veya Internet’deki çeşitli servislere istek yapıp gelen sonuçları beklemekle geçirir. Node.js, yapı gereği asenkrondur. Beklemeyi hiç sevmez. İstekleri teker teker yapmak yerine paralel olarak yapar. İstekler bittikleri zaman Node’u “geri ararlar” (callback). Bu sayede anlık olarak çok sayıda istek hızlı bir şekilde işlenebilir.

**Asenkron Programlamayı Öğrenmesi Zor mudur?**

Hemen her programlama dilinde işler sırayla ilerler. Örneğin “Git veritabanından bir kayıt oku” komutu bitmeden sıradaki komut başlamaz. Node.js’de ise başlar. Bu da alışık olmayanlar için kafa karıştırıcı olsa da sonuçta yaşanacak performans artışına ve ruhsal aydınlanmaya değerdir ya da en azından biz (ve milyonlarca Node.js geliştiricisi) öyle düşünüyor.

**Node.js Ekosisteminin Diğerlerinden Farkı Nedir?**

Node.js, JavaScript’in fonksiyonel programlamaya uygun yapısı sayesinde modüler kod yazmayı teşvik eder. Her bir modül tıpkı Unix felsefesinde olduğu gibi sadece tek bir işi yapmaya çalışır. Ne eksik ne fazla.

Node.js’de kod yazmak lego parçalarını birleştirmeye benzer. Herşeyi yapmaya çalışan hantal bir framework kullanmak yerine yüzbinlerce açık kaynak paketten oluşan NPM (Node Package Manager) ekosisteminden küçük büyük istediğiniz modülleri karıştırıp kullanabilirsiniz.

**Kimler Kullanıyor?**

Startup’lardan büyük şirketlere hemen herkes. Birkaç örnek: Apple, Google, IBM, Microsoft, NASA, Netflix, PayPal, Pinterest.

KULLANIM ÖRNEKLERİ

Hello World – Konsol

Öncelikle **nodejs-examples** isminde bir klasör oluşturup, içerisine **HelloConsole.js**isminde bir Javascript dosyası oluşturalım.

**HelloConsole.js**

function sayHelloFunction(name){

console.log("Hello, "  + name + "!");

}

sayHelloFunction("World");

Node.js bize getirdiği modüllerle ek özellikler sağlasada, alışık olduğumuz Javascript özelliklerini de aynen korumaktadır. Şimdi bu dosyayı aşağıdaki komut ile node.js üzerinde çalıştıralım.

* node HelloConsole.js

Eğer kurulumumuz başarıyla tamamlandıysa bize “Hello, World!” çıktısını üretecektir.

Hello World – Http

Şimdi ise başka bir örnek daha yapalım. Bu sefer bir Http sunucusu yaratıp, internet tarayıcısı üzerinden Hello, World yazdıralım.

Bunun için öncelikle **HelloHttp.js** dosyamızı ve modülümüzü oluşturuyoruz.

**HelloHttp.js**

module.exports = {

startHttpServer : function () {

var http = require("http");

var server = http.createServer(function(request, response){

response.writeHead(200);

response.write("<html><head><title>Hello, World!</title></head><body>Hello, World!");

response.end("</body></html>");

});

server.listen(8080);

console.log("8080 Portundan erişebilirsiniz.");

}

};

Burada gördüğünüz üzere modül içerisinde başka modüller çağrılabiliyor. Http sunucu için Node.js ile gelen Http modülünü kullanıyoruz. Fakat az önce olduğu gibi bu sadece modül, bu dosyayı çalıştırdığımızda ekrana birşey yazmayacaktır. Bunun için tekrar **Main.js** dosyamıza dönüyoruz ve modülümüzü o dosya içerisinde çağrıyoruz.

<http://www.bahadirakin.com/node-js-hello-world/>

**KAYNAK**

* [**https://www.biltektasarim.com/blog/javascript-nedir**](https://www.biltektasarim.com/blog/javascript-nedir)
* **https://firmajans.com.tr/internet/jquery-nedir-ve-nasil-kullanilir/**
* [**https://medium.com/reengen/node-js-nedir-ne-de%C4%9Fildir-9d5781f1fe50**](https://medium.com/reengen/node-js-nedir-ne-de%C4%9Fildir-9d5781f1fe50)
* [**http://www.bahadirakin.com/node-js-hello-world/**](http://www.bahadirakin.com/node-js-hello-world/)