**ERZURUM TEKNİK ÜNİVERSİTESİ**

**Mühendislik ve Mimarlık Fakültesi**

**Bilgisayar Mühendisliği**



**Staj Defteri**

Beyza PARLAK

180303036

Ekim, 2023

|  |  |  |
| --- | --- | --- |
|  | |  |
| **Öğrencinin** | | |
| Adı Soyadı | Beyza PARLAK | |
| Fakülte Numarası | 180303036 | |
| Firmanın (Kurumun) Adı | Yazıcı Demir-Çelik | |
| Staja Başlama ve Bitiş Tarihi | 09.10.2023-17.11.2023 | |
| Staj Süresi (İş günü olarak) | 30 | |
| Staj Değerlendirmesi (Üniversite tarafından doldurulacaktır) | | |
| Kabul edilen gün sayısı |  | |
| Onaylayan | İmza / Tarih | |

Bu staj defterinde stajıma destek veren firmanın ticari değeri olan özel bilgileri içermediği doğrularım. 17/11/2023

İmza

Beyza PARLAK

Bu staj defteri aşağıda bilgileri yer alan öğrenciniz tarafından firmamızda yapmış olduğu staj süresince doldurulmuş olup, yapılan çalışmaları ve öğrencinin stajına devam ettiğini onaylarım. 17/11/2023

İmza

Ahmet ÇAĞLAR

|  |  |  |
| --- | --- | --- |
| 1. Hafta | 1.Gün | Yapılan Çalışmalar |
| 09.10.2023 | Pazartesi |  |
| Yazıcı Demir Çelik San. ve Turizm Tic. A.S, 1991 yılında ülkenin artan inşaat demiri talebini karşılamak amacıyla Haddehane Tesisi olarak kurulmuştur [4].  1994 yılında modern bir Çelikhane tesisi devreye alınmıştır [4].  Yazıcı Demir Çelik San. ve Turizm Tic. A.S, yıllık kapasitesi 1.050.000 ton olan Çelikhane, 1.024.000 ton olan Haddehane ile 7.500.000 ton olan iki liman, 220 ton/gün kapasiteli Gaz oksijen, 6.8 ton/gün kapasiteli sıvı oksijen, 300 ton/gün kapasiteli Gaz azot, 6 ton/gün kapasiteli sıvı azot ve 8.5 ton/gün kapasiteli sıvı argon tesisleri, Su Tesisleri ve Toz tutma gibi birçok yardımcı ünitelerden meydana gelmektedir [4]. Çelikhane, haddehane, laboratuvar gibi üretim bölümleri bulunmaktadır.  Yazıcı Demir Çelik, Diler Holding aracılığıyla müşteri isteklerine göre ürünleri standart paketlerde ve firkete olarak ABD, İtalya, İngiltere, Almanya, Hollanda, Portekiz, İspanya, Tunus, Cezayir, Mısır, Ürdün, Lübnan, Suriye, Irak, Suudi Arabistan, Kuveyt, Birleşik Arap Emirlikleri, Yemen, Israil, Malezya, Singapur ve Hong Kong başta olmak üzere yaklaşık 50 ülkeye ihracat yapmaktadır ve iç piyasaya arz etmektedir [4].  Stajımı gerçekleştireceğim bölüm Bilgi İşlem Bölümüdür. Bilgi İşlem Bölümünde, şirketin donanım ve yazılım çalışmaları yapılmaktadır. Ben yazılım tarafında yer alacağım. Bilgi işlem bölümünde toplamda sekiz çalışan ve müdür bulunmaktadır. 5 mühendis ve 3 memur çalışmaktadır.  İlk staj günümde; benim gibi fabrikadaki stajyerler ile birlikte bazı derslere girdik. 14:00 saatine kadar İş Güvenliği, Çalışanların Yasal Hak ve Sorumlulukları, Çevre Kirliliği, Sağlık eğitimlerini almış bulunmaktayım.  Ardından da Bilgi İşlem bölümünde çalışma arkadaşlarımla tanıştım. Şirket için yapılan ve kullanılan programların bazıları anlatıldı, ne işe yaradıkları konuşuldu. SAP nedir ve ne işe yarar araştırmam söylendi.  SAP açılımı Systems, Applications, and Products in Data Processing (Veri İşlemede Sistemler, Uygulamalar ve Ürünler). SAP, şirketlerin birden fazla işin sürecini yönetmelerine ve optimize etmelerine yardımcı olmak için tasarlanan bir yazılım paketi olarak bilinmektedir.  SAP’in görevleri:   * İş Süreci Yönetimi: Şirketin tüm işlerini bir sistem üzerinden yönetilmesini sağlar. * Tedarik Zinciri Yönetimi: Malzemelerin yönetimi, envanterlerin tedarikleri ile ilgili en iyi planlamayı yaparak, düzenli ve sistematik bir şekilde yönetilmesini sağlar. * Analitik ve Veri Raporlama: SAP, hayal ettiğimizden daha büyük verileri analiz edebilir. Analiz ettiği verileri de en iyi şekilde görselleştirebilir. Analiz ettiği veriler çerçevesinde kararlar alabilir. * Müşteri İlişkileri Yönetimi: SAP, müşteri ilişkilerini yönetmede ve müşterilerin şirkete ya da ortaya koyulan hizmete karşılık memnuniyetlerinin arttırılması konuları için araçlar sunar. * Bulut Tabanlı ve Mobil Uygulamalar: SAP, bulut tabanlı hizmetler ve mobil uygulama geliştirme imkanı sunar. Bulut tabanlı hizmetlerin sonucu olarak, kullanıcılar istedikleri zamanda ve istedikleri cihaz ile erişim ve yönetim sağlarlar. * İş Sürekliliği ve Güvenlik: SAP, şirket işlerinin sistematik ve doğru bir şekilde ilerlemesi ve güvenlikle ilgili konularda güçlü önlemler sunmaktadır. Şirketler, verilerini güvenli bir şekilde işleyebilirler ve işlemlerini kesintisiz bir şekilde sürdürebilirler.   SAP, çok büyük ve karmaşık işletmelerin iş süreçlerini daha verimli hale getirmelerine, verileri daha etkili bir şekilde yönetmelerine ve kararlarını daha iyi bir temele dayandırmalarına yardımcı olan kapsamlı bir iş yazılımı platformudur. SAP sistemi yüksek ücretli bir yazılımdır.  Birden fazla alanda çalışan büyük şirketlerin kullandığı SAP sistemi, yine birden fazla alanla ilgilenen şirketin tek bir program üzerinden verileri görmelerini ve iş süreçlerini daha sistematik bir şekilde yönetmelerini sağlamaktadır. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 1. Hafta | 2.Gün | Yapılan Çalışmalar |
| 10.10.2023 | Salı |  |
| AGILE, SCRUM, KANBAN, SAP MII ve SAPUI5 kavramlarını araştırmam söylendi. Staj sürem boyunca bu konular üzerinden öğrenim kazanmam gerektiği ve bu konularda uzman olma yolunda neler yapabileceğim ile ilgili Murat Bey ( Bilgi İşlem Müdürü) ile sohbet ettik. Bu alanda ilerlemek istersem bir yol haritasının olduğunu ve bu yol haritasının içeriğini anlattı.  **AGILE:** Agile kelimesi dilimizde çevik anlamına gelmektedir. Yinelemeli geliştirmeye dayalı bir metoddur. Ekipler her sprintten sonra, bir sonraki sprint için geliştirilecek bir şey var mı diye düşünür ve geriye bakarlar. Agile da süreç hızlı ilerlemektedir fakat ekip üyelerini tüketecek kadar hızlı ilerleme beklenememelidir. Kendi kendini organize eden ekiplerle çalışılmaktadır.  Müşteri ile ürün, ne kadar çabuk temas ederse o kadar çabuk geri dönüş alınabilir. Alınan geri dönüşe göre de proje o kadar aktif hale gelmektedir.  Yapılacak işler, time boxing (zaman kutusu) denilen olaylara bölünmektedir ve bu periyotta ürün gelişimi ilerlemektedir. Proje süresince, her time boxing sonunda çalışan bir ürün ortaya çıkmalıdır.  En yaygın olanlar; Scrum, Kanban, XP(Extrem Programming).  image 1[1]  Agile’nin 4 temel özelliği şunlardır:   1. Süreçler ve araçlara odaklanmak yerine bireylere odaklanır. İşbirliği önemlidir. 2. Kapsamlı dokümantasyon yerine yazılıma odaklanılır. 3. Müşterilerle işbirliği çok önemlidir. 4. Bir plan çerçevesinde ilerlemek yerine gelişime ve değişime odaklanılır.   Agile’ın avantajları ise şöyledir; ekip çalışmasına teşvik eder, değişen gereksinimler için esneklik sağlar, yazılım oluşurken geri bildirime izin verir.  Agile’ın dezavantajları ise şöyledir; proje başında maliyetini tahmin edemeyiz çünkü ortada gelişen bir proje vardır, müşteri odaklı sorun yaşanabilir(geri bildirim, iletişim vb.). Sözlü iletişim sağlandığı için güven çok önemlidir.  Sonuç olarak Agile metodunda değişen gereksinimler projenin son aşamasında bile kabul edilebilir oldu, daha kısa süreli bildirimler ile hata payı azaltıldı ve çalışan yazılım müşteriye sunuldu, iş sahibi ve yazılımcılar süreç boyunca birlikte çalıştı, yüz yüze iletişime dayandığı için bilgi alışverişi sağlandı. Projenin en başında her şey planlanmalı ve gereksinim değişikliği göz önüne alınarak değişimi kucaklayan bir yapı inşa edilmeli.  **SCRUM:** Scrum, projeleri sprint adı verilen küçük yapılara böler. Bu sprintler aracılığıyla yazılım geliştirmede işbirliğini destekleyen Agile’ın bir çerçevesidir(framework). Scrum, her geliştirme sprintinden sonra yazılımı teslim eder ve işi en kısa zamanda sunmayı amaçlamaktadır. Scrum Agile’a göre daha yapılandırılmış ve daha az esnektir çünkü sonuçları hızlı bir şekilde teslim etmeye odaklıdır.  Scrum’ um 3 temel prensibi vardır. Bu prensipler; şeffaflık, denetleme ve uyarlamadır. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 1. Hafta | 3.Gün | Yapılan Çalışmalar |
| 11.10.2023 | Çarşamba |  |
| Takımlar kendilerini organize ederler ve takımlar genellikle 6-10 kişiden oluşmaktadır.  Scrum ile Agile arasındaki farklar:   * Scrum daha katı bir özelliğe sahipken Agile daha esnektir. * Scrum daha kısa ve ayrı projeler sunarken Agile sürecin sonunda her şeyi teslim eder. * Agile’ın basit tutulması amaçlanırken Scrum yenilikçi ve deneysel olabilir.   Agile liderler hayati bir rol oynarken Scrum kendi kendine işleyen çapraz işlevli bir ekibi teşvik eder.  Scrum avantajları şöyledir:   * Kısa iterasyonlarla (yenileme) tamamen geliştirilmiş ve test edilmiş özellikler * Sürecin uygulama basitliği * Açıkça tanımlı kurallar * Kendi kendini düzenleme * Her takım üyesinin sorumluluk bilincinde olması   Scrum dezavantajları ise şöyledir:   * Yazılı doküman azlığı * Sorumluluk ihlali olasılığı (iyi niyet üzerine kurulu)   **KANBAN:**  Kanban, görsel bir proje yönetimi aracıdır ve iş akışını tahtada sürükleyerek gösterilir.  Görevleri kartlar halinde temsil eder ve tahta üzerindeki sütunlar, işin durumunu yansıtır, örneğin, "Bekleme," "Üzerinde Çalışılıyor," "Tamamlandı" olarak gösterim yapılmaktadır.  Kanban, işin akışını izlemek, iş önceliklerini belirlemek ve sürekli iyileştirme sağlamak için kullanılan bir proje yönetimi aracıdır. Kanban'ın amacı, sürecinizdeki olası darboğazları belirlemek ve bunları düzeltmektir, böylece iş, optimum hızda veya verimde gerçekleşebilir [2].  Kanban İlkeleri:   * Ne yapıyorsan oradan başla - Kanban, belirli bir kurulum gerektirmez ve direkt iş süreciniz ile birebir uyum sağlayabilir. * Değişimi kabul et - Kanban, sürekli, artan ve evrimsel değişiklikleri teşvik ederek en az şekilde mevcut projenize zorluk çıkaracak şekilde tasarlanmıştır. * Her düzeyde liderliği destekler - En az bilgili ekip üyesinin parlak bir fikri varsa, bu kabul edilmeli ve uygulanmalıdır. Çalışanların en fazla performansa çıkabilmesi için herkesin sürekli iyileştirmesi (Kaizen) zihniyetini teşvik etmesi gerekir.   image 2 [2] | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 1. Hafta | 4.Gün | Yapılan Çalışmalar |
| 12.10.2023 | Perşembe |  |
| Bilgi işlem müdürümüz Murat Bey, staj sürecim boyunca Javascript ve SAP Fiori içeriğinden SAP UI5 öğrenmem ve bu teknolojilerle ilgili çalışmalar yapmamı söyledi. Bende bu doğrultuda Javascript öğrenmeye ilk adımı attım. Öğrendiklerim:  Öncelikle VS Code programını kurdum ve Javascript ile ilgili birkaç extension ekledim. Ardından Node JS kurdum ve versiyon kontrol ettim.  Bir dosya oluşturdum ve aşağıdaki görselde göründüğü gibi ilk kodumu derledim. Aslında console iki türlü çalıştırılabilir. İlk olarak vs code üzerindeki console, ikinci olarak da chrome üzerinde sağ tıklayıp sayfayı incele ekranını açtığımızda oradaki console üzerinden. Bana daha açık geldiği için chrome üzerinden console ekranına devam edeceğim.    script.js kodumuzun tarayıcıda görünmesi için html sayfasına aşağıda koddaki gibi adres vermeliyiz.  <script src="scripts.js"></script>  **Değişkenler:** Javascriptte değişkenler iki türlü tanımlanır. İlki var, ikincisi let. Değişkenlerle basit bir zam programı yazacağım.  var maasAli = 10000;  var maasCan = 8000;  var zam = 0.3;  console.log("Ali'nin maaşı " + (maasAli + (maasAli \* zam)));  console.log("Can'ın maaşı " +(maasCan + (maasAli \* zam)));  Yukarıda yazdığım kodun çıktısının görseli aşağıya eklenmiştir.  Aşağıdaki kodda string toplama işlemi yapılmıştır.  let sayi12 = "10";  let sayi22 = "50";  console.log(sayi12.toString() + sayi22.toString());  Aşağıdaki kodda stringi integera çevirip toplama işlemi yapılmıştır.  let sayi3 = "56";  let sayi4 = "77";  console.log(Number(sayi3) + Number(sayi4));  Yukarıdaki iki toplama işleminin sonucu görselde verilmiştir. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 1. Hafta | 5.Gün | Yapılan Çalışmalar |
| 13.10.2023 | Cuma |  |
| let not\_1 = 70; let not\_2 = 70; let not\_3 = 80;  let ortalama = (not\_1 + not\_2 + not\_3) / 3;  console.log(parseInt(ortalama ));  console.log(ortalama >= 50);  Yukarıdaki kodda bir öğrencinin üç notunun ortalaması hesaplanmıştır. ParseInt ile integer değer döndürür. Ortalama 50’ den büyük veya eşitse ortalamayı yazdırır ve true döndürür. Kodun çıktısı aşağıdadır.  let suankiYil = new Date().getFullYear();  let ogr\_1Yas = suankiYil - parseInt(ogr\_1DogumTarih);  let ogr\_2Yas = suankiYil - parseInt(ogr\_2DogumTarih);  Yukarıdaki kodun ilk satırında getFullYear() metodu ile bilgisayarın şuanki saatini alır. Önceden aldığım ogr\_1DogumTarih ve ogr\_2DogumTarih değerlerinin şimdiki yaşlarını hesaplıyorum. ParseInt() metodu ile integer değer döndürüyorum.  Operatörler: Bazı operatörler; +, -, \*, /, %, =, ++, -- gibidir. Operatörler kendi içlerinde ayrılır.   * Aritmetik Operatörler (+, -, \*, /, %, \*\*, ++, --) * Atama Operatörleri (=, +=, -=, \*=, /=, %=, \*\*=) * Karşılaştırma Operatörleri (==, >, <, ?, !=, >=, <= gibi) * Mantıksal Operatörler * Tip Operatörleri   Operatörlerle ilgili örnek çalışmalarım aşağıda bulunmaktadır.  let isLoggedIn = true;  if(isLoggedIn){  console.log("Uygulamaya giriş yapıldı..");  }else{  console.log("Uygulamaya giriş yapıldı..");  }  Yukarıdaki kodda boolean türünde bir değişken tanımlanmıştır. Bu değişken true ise if içerisini, false ise else içerisini çalıştıracaktır. Sonuç true dönmüştür.  let userName = "BeyzaParlak";  let password = "1234";  if(userName == "BeyzaParlak"){  if(password == "12345"){  console.log("Kullanıcı girişi başarılı.");  }else{  console.log("Kullanıcı girişi başarısız.");  }  }  Yukarıdaki kodda bir kullanıcı girişi simulasyonu yapılmıştır. userName ve password doğru ise Kullanıcı girişi başarılı, yanlış ise Kullanıcı girişi başarısız yazdıracaktır. Console ekran görüntüsü aşağıda verilmiştir. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 2. Hafta | 6.Gün | Yapılan Çalışmalar |
| 16.10.2023 | Pazartesi |  |
| **String Metodları:** Öncelikle aşağıdaki gibi iki değişken içerisinde string tanımlıyorum ve string metodlarını bu iki değişken üzerinden gerçekleştireceğim.  let url = "https:/www.beyzaparlak.com.tr/";  let tag = "Blog Sitesi";  Url değişkeninin kaç karakterden oluştuğunu bulalım. Bir değişkenin uzunluğunu bulmak için lenght metodu kullanılır.  console.log(url.length);  Yukarıdaki kodun çıktısı **30** olarak verilmiştir. Tag stringi kaç kelimeden oluştuğunu bulalım. Split() metodu ile kelimeler arasındaki boşlukları tespit edip, boşluk sayısını sayacağız.  console.log(tag.split(" ").length);  Yukarıdaki kodun çıktısı **2** olarak verilmiştir. url değişkeni “https” ile mi başlıyor bulalım.startsWith() metodu ile stringin başlangıcını kontrol edebiliriz.  console.log(url.startsWith("https"));  Yukarıdaki kodun çıktısı **true** dönmektedir. tag değişkeninin tüm karakterlerini büyük harfe çevirelim. Bu işlemi toUpperCase() metodu ile yapabiliriz.  console.log(tag.toUpperCase());  Yukarıdaki kodun çıktısı “BLOG SİTESİ” olmuştur. Tag stringi içerisinde “x” harfi var mı kontrol edelim. Bu işlemi includes() metodu ile yapabiliriz.  if(tag.includes("x")){  console.log("Var");  }else{  console.log("Yok");  }  Yukarıdaki kodun çıktısı “Yok” olarak verilmiştir. Url stringini “beyza.parlak.com.tr” ile değiştirelim. Bu işlemi replace() metodu ile yapabiliriz. İki stringi değiştirme işlemini yapar.  console.log(url.replace("https:/www.beyzaparlak.com.tr/"," beyza.parlak.com.tr "));  **Date:** Date konu başlığı altında zaman ve tarih ile ilgili işlemler yapılmaktadır. Yapılan işlemler metodlar vasıtasıyla gerçekleşmektedir. Kullanılan bazı metodlar; getFullYear(), getDate(), getTime(), setDate(), setFullYear() gibi birçok metod bulunmaktadır. Aşağıdaki kodun çıktısı aşağıda verilmiştir. Simdi değişkenine yeni zaman nesnesi atanmıştır.  Date ile ilgili örnekler:  let simdi = new Date();  console.log(simdi);  console.log(simdi.getDay());  Yukarıdaki kodun çıktısı 2 olarak verilmiştir. getDay() metodu haftanın hangi gününde olduğumuzu sayısal olarak vermektedir. 2 çıktısı alındığında Salı gününde olunduğu anlaşılır. Aşağıdaki kod, set eder yani günceller. Yıl bilgisini 2025 olarak günceller.  console.log(simdi.setFullYear(2025));  **Diziler ve Dizi Metodları:** Dizi birden fazla değer tutabilen bir yapıdadır. Diziler, aynı veri türünden veya farklı veri türlerinden öğeleri içerebilirler. Dizi tanımı aşağıda verilmiştir.  let urunler = ["Asus Zenbook", "Redmi Note 12 Pro", "Iphone 13","Lenovo Klavye"];  Yukarıda urunler adında, dört elemanlı bir dizinin tanımlandığını görmekteyiz.  console.log(urunler[3]);  console.log(urunler[0]); | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 2. Hafta | 7.Gün | Yapılan Çalışmalar |
| 17.10.2023 | Salı |  |
| Önceki sayfadaki kodu çalıştırdığımızda çıktı şöyle olacaktır; “Lenovo Klavye”, “Asus Zenbook”. Yani ilk satırdaki kod ile urunler dizisindeki 3. ve 0. indeksteki elemanlara ulaşıyoruz. console.log(urunler.push("Logitech Mouse"));  Yukarıdaki kod ile urunler dizisinin sonuna “Logitech Mouse” elemanını ekleriz. Eleman eklemek için push() metodu kullanıldı.  console.log(urunler.lenght);  Yukarıdaki kod ile urunler dizisinin uzunluğunu öğrenebiliriz. Lenght metodu dizinin uzunluğunu bulmaktadır.  for (var i = 0; i < urunler.length; i++) {  console.log( urunler[i]);  }  Yukarıdaki kod urunler[i] ile urunler dizisinin elemanlarına tek tek ulaşıp yazdırır.  **Objects:**  Objects, key - value ilişkisini temsil eden yapılardır. Aşağıda verilen kodda object tanımlama işlemi yapılmıştır. Öncelikle let ile object tanımlanmıştır ve ardından key – value şeklinde tanımlamalar yapılmıştır. Adress ve hobies’ de iç içe bir yapıdan oluşmaktadır. Bir objectin içinde başka bir object veya dizi olabilir.  let users = {  "name" : "Beyza",  "lastname" : "Parlak",  "age" : 23,  "adress" : {"city" : "Hatay",  "county" : "Dortyol",  "license plate" : "31"  },  "hobies" : ["cinema","video game","books"],  }  Objectlere erişim kodları aşağıda verilmiştir. Aşağıdaki kodun çıktısı tanımda da görüldüğü üzere 23 olacaktır.  let yas = users.age;  Yine object erişimine örnek vermek gerekirse aşağıdaki gibi bir erişim sağlanır ve console’ a yazdırma işlemi yapılır.  console.log(users.name +" "+ users.lastname +" - "+ users.adress.city);  Object ile ilgili bir uygulama; order\_1 adında bir object tanımlıyorum ve içerisine order\_ID, order\_date, payment\_method, shipping\_address, products tanımlayıp içlerini dolduruyorum ve order\_1 deki siparişte her ürünün KDV dahil toplam sipariş fiyatını hesaplayan kod verilmiştir. (KDV: %18)  let order\_1 = {  "order\_ID": 10001,  "order\_date": "01.10.2023",  "payment\_method": "bank card",  "shipping\_address":{  "province":"Erzurum",  "County":"Palandöken"  },  "products":[  {  "product\_ID": 5,  "product\_title": "Lenovo Notebook", | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 2. Hafta | 8.Gün | Yapılan Çalışmalar |
| 18.10.2023 | Çarşamba |  |
| "product\_url": "https://beyzaparlak.com/lenovo-notebook-blue?123",  "product\_color": "white",  "product\_price": 750  },{  "product\_ID": 9,  "product\_title": "Razer Keyboard",  "product\_url": "https://beyzaparlak.com/razer-keyboard?109",  "product\_color": "blue",  "product\_price": 50  }]  };  let order1\_sum = (order\_1.products[0].product\_price + order\_1.products[1].product\_price) \* 1.18;  console.log("Order 1 Amount: $ " + order1\_sum);  Yukarıdaki kodun çıktısı aşağıda verilmiştir.  C:\Users\User\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Ekran görüntüsü 2023-11-01 084533.png  **DOM (Document Object Model):** JavaScript DOM, web tarayıcılarının içeriğini ve yapısını programlar tarafından daha iyi yönetilebilir hale getiren bir uygulama programlama arabirimidir (API). DOM, bir ağaç yapısı olarak tasarlanmıştır ve bu ağaç yapısı, her HTML veya XML belgesinin temsil edildiği bir nesne koleksiyonunu içerir. Bu ağaç yapısı, temel bir düğüm olan document düğümü ile başlar ve altında diğer elementler ve özellikler bulunur. Örnek vermek gerekirse, belirli bir HTML elementini seçebilir, bu elementin özelliklerini veya içeriğini değiştirebilir, yeni elementler ekleyebiliriz. Web geliştiricilere web sayfalarını dinamik içerikli şekilde oluşturma seçeneği sunar.  **Single Element Selection (Tek Eleman Seçimi):**  <script>  let sonuc;  sonuc = document.getElementById("title");  sonuc = document.getElementById("title").id;  sonuc = document.getElementById("title").className;  document.getElementById("title").style.fontSize = "40px";  document.getElementById("title").style.color = "blue";  console.log(sonuc);  </script>  Yukarıdaki kod parçasından önce basit düzeyde bir HTML yapısı oluşturulmuştur. Ardından <script> tagı içerisinde bir değişken tanımlanmıştır ve bu değişkene değiştirilen değerler ya da istenilen sonuçlar atanmıştır. Sonuc değişkeni adım adım console ile çağırılmıştır fakat yukarıda kod toplu bir şekilde verilmiştir. getElementById() metodu, sayfanın belirli bir öğesine erişmek ve üzerinde işlemler yapmak için sıklıkla kullanılır. “title” ile title tagı içeren satırlara erişim sağlarız. İd ve className ile de <title> tagı içeren satırların id ve className değerlerine erişim sağlarız. Style.fontSize ile <title> scope altında olan ve HTML sayfasında görünen kodlara 40px değerini veririz. Style.color ile de bu satırların text renklerini belirlemiş oluruz. Son olarak console.log ile bu sonuçları tek tek yazdırıp çıktılarını gördüm.  HTML kodu ve çıktı ekran görüntüsü diğer sayfada verilmiştir. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 2. Hafta | 9.Gün | Yapılan Çalışmalar |
| 19.10.2023 | Perşembe |  |
| sonuc = document.querySelector("#title");  sonuc = document.querySelector(".card-header");  sonuc = document.querySelector("div");  sonuc = document.querySelector("li:first-child");  console.log(sonuç);  Yukarıda gördüğünüz kod parçasındaki querySelector() metodunun görevi, belirli bir öğeyi seçmek ve üzerinde işlem yapmak için kullanılır. İlk satırda title a sahip ilk etiketi yazdırır. İkinci satırda ise classı card-header olan etikete ulaşırız. Üçüncü satırda ise sadece <div> etiket ismine göre seçim yaparız. Son satırda ise <li> etiketindeki first-child yani ilk satır üzerinden seçim yaparız.  **Multiple Elements**: Çoklu eleman seçimleri üzerinden değişiklik yapılmasını sağlar.  taskList = document.getElementsByClassName("task");  for(let i=0; i < taskList.length; i++){  console.log(taskList[i]);  }  Yukarıdaki kod parçasında bir görev listesi oluşturulmuştur ve class name = “task” olan scopeları seçip yazdırır. Aşağıda çıktının ekran görüntüsü verilmiştir.  for(let task of taskList){  task.style.color="red"  task.style.fontSize = "35px";  task.innerText = "Item";  }  Yukarıdaki kod parçasında taskList içindeki elemanlar gezilir ve döngü içerisindeki değişiklikler bu elemanlara uygulanır. Bütün elemanların rengini, font büyüklüğünü ve textini değiştirir. Kod çıktısının ekran görüntüsü aşağıda verilmiştir.  **Creating Elements:** Creating Elements, aynı yapıda olan elemanları, liste yardımıyla ekrana yazdırma işlemidir. Aşağıda kodlar ve ekran görüntüsü verilmiştir.  let taskList = [  {"id": 1, "taskName": "Task 1"},  {"id": 2, "taskName": "Task 2"}, | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 2. Hafta | 10.Gün | Yapılan Çalışmalar |
| 20.10.2023 | Cuma |  |
| {"id": 3, "taskName": "Task 3"},  {"id": 4, "taskName": "Task 4"},  ];  let ul = document.getElementById("task-list");  for(let task of taskList) {  let li = `  <li class="task list-group-item">  <div class="form-check">  <input type="checkbox" id="${task.id}" class="form-check-input">  <label for="${task.id}" class="form-check-label">${task.taskName}</label>  </div>  </li>`;  ul.insertAdjacentHTML("beforeend", li);  }  Öncelikle bir değişkene taskList adı verilmiştir ve liste elemanları hakkında bilgiler eklenmiştir. Ardından for döngüsü ile taskList üzerinde gezinme ve işlem yapmak için kodlar yazılmıştır. Her bir eleman aynı yapuda olacağı için HTML üzerinden yazdığım kodları li değişkeni içerisine attım ve ${task.id}" ile taskin id sine göre numaralandırma işlemi yapmasını ve sayısını belirlemesini, ${task.taskName} ile de taskin adının yazdırılmasını sağladım. Kodun çıktısı aşağıdaki görselde verilmiştir. ul.insertAdjacentHTML("beforeend", li); bu satırda kullandığım beforeend, baştan sona sıralar.  **Event Listener:** Event listener, HTML sayfasındaki bir olay gerçekleştiğinde tetiklenir ve yazılan işlemi gerçekleştirir. Kısacası event bir olayla ilişkilendirilmelidir. Benim yapacağım işlem, add butonuna her bastığımızda click event sayısı artacak ve sayfa yenilenmeyecek. İlk satırdaki kod sayesinde add butonununa basınca console a yazdıracaktır. İkinci satırdaki kod sayesinde clear butonuna basınca console a yazdıracaktır.  document.querySelector("#btnAddNewTask").addEventListener("click", newTask);  document.querySelector("#btnClear").addEventListener("click", newTask);  function newTask(event){  console.log("click event: clear");  event.preventDefault();  }  Kodun çıktısı aşağıdaki görselde verilmiştir. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 3. Hafta | 11.Gün | Yapılan Çalışmalar |
| 23.10.2023 | Pazartesi |  |
| **Adding New Task:** Bu aşamadaki görevim yeni taskler eklemektir. Öncelikle bir fonksiyon yazıyorum. taskName leri textInput değişkenine atıyorum. İf içerisinde text alanı boşken ekleme yapmak istenirse alert ile uyarı vermesini istiyorum. Else içerisinde id ve text içerisindeki numara bilgilerini arttırıyorum ki sayılara ulaşayım.  function newTask(event) {  let taskInput = document.querySelector("#txtTaskName");  if(taskInput.value == ""){  alert("Task the enter!!");  }else{  taskList.push({"id": taskList.length + 1, "taskName": taskInput.value})  taskInput.value="";  displayTasks();  } event.preventDefault();  }  Ardından sadece add butonuna basınca task eklemesin, aynı zamanda enter tuşuna basınca da task eklesin istediğim için onun için de ayrı bir kod yazıyorum.  document.querySelector("#btnAddNewTask").addEventListener("keypress",function(){  if(event.key == "Enter"){  document.getElementById("#btnAddNewTask").click();  }});  **Delete Task:** Bu aşamadaki görevim, task silme işlemidir. TaskList teki id ye göre fonksiyon ile id bulup iki id yi birbirine eşitleyeceğim ve o id ye sahip elemanı sileceğim. Bu aşamada HTML ile ilgili birkaç ekleme yaptım. Splice() metodu, belirtilen indexten itibaren bir index siler. displayTask() metodunu çağırma sebebim ise görev listesi elemanlarını tekrar yüklemek istememdir. Koda ve çıktısına aşağıda yer verdim.  function deleteTask(id){  let deletedId;  for(let index in taskList){  if(taskList[index].id == id){  deletedId = index;}  }  taskList.splice(deletedId, 1);  displayTasks();  }  **Update Task:** Bu aşamadaki görevim, task güncelleme işlemidir. Dropdown içerisindeki Edit e basınca taski yazdığımız alana getirecek ve tekrar add butonuna basınca ya da enter a basınca güncelleme işlemini yerine getirecek işlemi yapacağım. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 3. Hafta | 12.Gün | Yapılan Çalışmalar |
| 24.10.2023 | Salı |  |
| NewTask fonksiyonuna aşağıdaki kodu ekledim. Öncelikle for içerisinde elemanları tek tek geziyorum. Taskin id si editId ye eşit ise bilgileri değiştiriyoruz. isEditTask değişkenini false yapıyorum ve edit işlevini kapatıyorum.  else{  if(!isEditTask){  taskList.push({"id": taskList.length + 1, "taskName": taskInput.value})  }else{  for(let task of taskList){  if(task.id == editId){  task.taskName = taskInput.value;  }  isEditTask = false;  }  }  taskInput.value="";  displayTasks();  }  **Delete All Task:** Bu aşamada, taskListteki tüm taskleri silme işlemini yapacağım. Clear butonuna tıklandığında tüm taskler silinecek. İlk indexten taskList sayısına kadar tüm elemanları siler. Kod ve çıktısı aşağıda verilmiştir.  btnClear.addEventListener("click",function(){  taskList.splice(0, taskList.length);  displayTasks();  })  **Check Task:** Bu aşamada, durum güncelleme işlemi yapacağız yani check edilmiş (completed) mi edilmemiş (pending) mi onu kontrol edip task üzerinde değişiklik yapacağım. nextElementSibling() aynı hizada olan elemanları söyler. İf içerisinde eğer task secildiyse tamamlandı olarak işaretler, secilmediyse tamamlanmadı olarak işaretler.  function updateStatus(selectedTask){  let label = selectedTask.nextElementSibling;  if(selectedTask.checked){  label.classList.add("checked");  state = "completed";  } else{  label.classList.remove("checked");  state = "pending";  } | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 3. Hafta | 13.Gün | Yapılan Çalışmalar |
| 25.10.2023 | Çarşamba |  |
| for(let task of taskList){  if(task.id == selectedTask.id){  task.state = state;  }  console.log(taskList);  }}  **Filters:** Bu aşamada tüm taskler üzerinden işlemler yani filtreleme işlemi yapacağım. Örneğin; tamamlanan tasklar için COMPLETED, tamamlanmayan tasklar için TO DO tag içerisinde yazdıracağım. Bu işlemleri “pending” ve “completed” üzerinden yapacağım. Aşağıda if içerisinde task durumu all ya da diğer durumlar ise li içerinde önceden yaptığımız işlemleri gerçekleştirecek. For içerisinde span etiketiyle tıklandığında o filtrenin active olmasını sağlayacaktır ve bunu ekrana yazdıracaktır.      **Local Storage:** Local storage, web tarayıcılarında kullanılan bir veri depolama yöntemidir ve tarayıcıda kullanılan web uygulamalarının verilerini kullanıcının cihazında saklar.Kullanıcı tarayıcıyı yenilediğinde, önceden yazdığı taskler silinmesin ve tarayıcıyıda saklansın istiyorum. Local storage a tarayıcının application kısmından ulaşılabiliyor. taskListi local storage da saklayabilirim ve bu bilgileri taskListe aktarabilirim.  Aşağıdaki kodda set veri aktarımı - get varolan veriyi alma işleminde kullanılıyor. Veriyi string olarak alıp JSON verisine parse ediyorum.  if(localStorage.getItem("taskList") !== null){  taskList = JSON.parse(localStorage.getItem("taskList"));  }  Aşağıdaki kod, taskList objesini JSON formatından stringe çevir ve setItem() metodu ile local storage a aktarmaya yarar.  localStorage.setItem("taskList", JSON.stringify(taskList));  Bu kodları da yazdıktan sonra TO DO uygulamasının sonuna geldim. Artık eklediğim taskler tamamlandı ya da tamamlanmadı olarak doğru görünüyor ve istediğimde hepsini silebiliyorum ya da seçtiğimi silebiliyorum. Sayfayı her yenilediğimde en son yazdığım taskler kayıtlı bir şekilde geliyor. Uygulamanın son halinin ekran görüntüsüne bu sayfaya sığmadığı için diğer sayfada yer verdim. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 3. Hafta | 14.Gün | Yapılan Çalışmalar |
| 26.10.2023 | Perşembe |  |
| Yapacağım ikinci uygulama Form Validator (Form Doğrulayıcı) olacaktır. Bu uygulama için öncelikle, HTML iskeletini oluşturdum. CSS ve Javascript dosyalarını koda ekledim. Ardından form validation görüntüsü iç,n HTML kodlarımı yazdım. Container içerisine card ekledim ardından cardın içine de formları ekledim. Form içini form-grouplara böldüm ve her birinin içine <label> ve <input> ekledim, son olarak kontrol için button ekledim. Kodların küçük bir kısmına aşağıda yer verdim. Username için yaptığım işlemleri; password, re-password ve email için de yazdım. My-5 ile alltan ve üstten forma boşluk verdim, justify-content-center ile form içindekileri ortaladım.  <div class="container my-5 d-flex justify-content-center">  <div class="col-sm-5">  <div class="card">  <div class="card-header">Create User</div>  <div class="card-body">  <form id="form" novalidate>  <div class="form-group">  <label for="username">Username</label>  <input type="text" class="form-control" id="username" placeholder="Enter your username">  </div>  <button type="submit" class="btn btn-success btn-block">Register</button>  </form>  </div>  </div></div></div>  Script.js dosyası oluşturuyorum. Ardından form, username, email, password, re-password bilgilerini alıyorum. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 3. Hafta | 15.Gün | Yapılan Çalışmalar |
| 27.10.2023 | Cuma |  |
| const form = document.getElementById('form');  const username = document.getElementById('username');  const email = document.getElementById('email');  const password = document.getElementById('password');  const repassword = document.getElementById('repassword');  Bir event oluşturacağım. Submit butonuna tıklayınca bir olay gerçekleşecektir. Username için yaptıklarımı email, password ve re-password için de yazıyorum. preventDefault() metoduyla, formun varsayılan özelliğini kapatıp, submit etmemiş oluyoruz. Eğer username değeri boş ise error fonksiyonunu çağır, değilse success fonksiyonunu çağır diyorum.  form.addEventListener('submit', function(e){  e.preventDefault();  if(username.value === ''){  error(username,'Enter a valid username!');  }else{  success(username);  }});  Error() fonksiyonunda, girdi ve bir mesaj alacaktır. Değer girilmediğinde kırmızı rengi olacaktır(is-invalid).  function error(input,message){  input.className = 'form-control is-invalid';  const div = input.nextElementSibling;  div.innerText = message;  div.className = 'invalid-feedback';  }  Success() fonksiyonunda, girdi başarılı ise tik işareti ile doğru olduğunu gösterecektir.  function success(input){  input.className = 'form-control is-valid';  }  Card a arkasında gölge olması için style.css dosyasını oluşturup içerisine aşağıdaki kodu ekliyorum. Tüm kodlar sonrası çıktısı aşağıya eklenmiştir.  .card {  box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2);} | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 4. Hafta | 16.Gün | Yapılan Çalışmalar |
| 30.10.2023 | Pazartesi |  |
| **Modern Javascript:** İkinci Revizyon ES6 (2015) ile Modern Javascript e adım atılmıştır.  Arrow Function: Daha kısa bir şekilde fonksiyon yazmaya olanak tanır. Aşağıdaki kodların soldakinde temel düzeyde fonksiyon oluşturulmuştur. Sağdaki kodda ise arrow function ile fonksiyon oluşturulmuştur.    Filter() Metodu: Bir koşul tarafından sağlanan verilerden yeni bir dizi oluşturur. Filter metodu orijinal dizide değişiklik yapmaz. Aşağıdaki kodlar; bir dizideki çift sayıları buluyor ve filter() metodu ile yeni bir diziye ekliyor. Kodun çıktısı ekran görüntüsündedir.  const numbers = [12,69,36,63,82,18,21,85,87,97];  const couple\_numbers1 = numbers.filter((number) => number % 2 == 0);  console.log(couple\_numbers1);  Map() Metodu: Bir koşul tarafından sağlanan veriler için yeni bir dizi döndürür.  const couple\_numbers2 = numbers.map((number) => number % 2 == 0);  console.log(couple\_numbers2);  Maps: Verilerin çiftler halinde saklanmasına yardımcı olan bir veri yapısıdır. Çift, benzersiz bir anahtar ve anahtara eşlenen bir değerden oluşur [3]. Const ile provinces adında bir maps tanımladım. Ardından set ile değer ataması yaptım. Sonra bu atamalar üzerinden bazı metodlar kullandım. Get verileri getirir, size uzunluğunu verir, has listede veri var mı bakar, delete belirtilen veriyi siler. Aşağıdaki kodun çıktısının ekran görüntüsü verilmiştir.  const provinces = new Map();  provinces.set(1,"Adana");  provinces.set(31,"Hatay");  provinces.set(28,"Giresun");  let result = provinces.get(31);  result = provinces.size;  result = provinces.has(34);  result = provinces.delete(1);  console.log(result);  console.log(provinces);  Aşağıdaki kod, dizideki tüm elemanlar üzerinde gezinebilir. Console.log ile tüm verileri yazdırır.  for(let province of provinces.entries()){  console.log(province);  } | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 4. Hafta | 17.Gün | Yapılan Çalışmalar |
| 31.10.2023 | Salı |  |
| Dizideki key ve valuelere aşağıdaki kodlarla ulaşabiliriz. Kodların çıktısının ekran görüntüsü verilmiştir.  for(let province of provinces.keys()){  console.log(province);  }  for(let province of provinces.values()){  console.log(province);  }  Sets: Her türdeki benzersiz değerleri saklamanızı sağlar. Aşağıdaki kodda dizi değerlerini yazdırdım. Numbers yazsaydım da aynı sonucu alıyorum. (key – value yok)  const numbers = new Set();  numbers.add(11);  numbers.add(22);  console.log(numbers.has(22));  for(let number of numbers.values()){  console.log(number);  }  Classes: OOP için daha okunaklı kod yazmaya yardımcı olur. Aşağıdaki kodda bir class oluşturdum, ardından constructor(yapıcı) metod oluşturdum ve alacağım değişkenleri tanıttım. ageCalculate ve information adında iki fonksiyon yazdım. ageCalculate şimdiki yaşını getFullYear yardımı ile hesaplıyor. Information ise id ve name döndürüyor.  class Person{  constructor(id, name, job, birthYear){  this.id = id;  this.name = name;  this.job = job;  this.birthYear = birthYear;  console.log("new object...");  };  ageCalculate(birthYear){  let date = new Date().getFullYear();  return date - this.birthYear;  }  information(id,name){  return this.id + " - " + this.name;  }  Diğer sayfadaki kodda, iki tane yeni nesne oluşturdum ve bu nesnelere değerler verdim. Bu değerleri fonksiyon çağırarak yazdırdım. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 4. Hafta | 18.Gün | Yapılan Çalışmalar |
| 01.11.2023 | Çarşamba |  |
| let person1 = new Person("1098","Beyza Parlak","Computer Engineer",2000);  let person2 = new Person("1101","Aslı Gültekin","Psychologist",2001);  console.log(person1.information()+" - "+person1.ageCalculate());  console.log(person2.information()+" - "+person2.ageCalculate());  Getter – Setter: Bir nesnenin içindeki verilere (sınıfın özellikleri veya üyeleri olarak adlandırılır) erişmek ve bu verileri değiştirmek için kullanılan yöntemlerdir. Aşağıdaki kodda örnek verilmiştir. Set değer almaktadır, get ise sadece değer döndürür. Kodun bir bölümü ve çıktısı aşağıda verilmiştir.  get ID(){  return this.id;  }  get Language(){  return this.language;  }  set Name(name){  return this.name = name.toLowerCase();  }  Inheritance: Bir sınıfın kendisine ait özellikleri ve işlevleri bir başka sınıfa aynen aktarması ya da bazı özellik ve işlevlerini diğer sınıfların kullanmasına izin vermesidir. Extends anahtar kelimesi kullanılır.  **Yeni Proje – Music Player**  Javascript, HTML ve CSS ile Music Player Application kodlayacağım. İleri, geri ve duraklat-oynat butonları ile kontrolünü sağlayacağım. Her şarkı için bir mp3 ve images objesi olacaktır.  Öncelikle bir images, mp3 klasörlerini açıp içlerini dolduruyorum. Ardından app.js, music.js, musicplayer.js, style.css ve index.html dosyalarını oluşturuyorum.  Constructor Kodlanması: Constructor, yapıcı metodlardır. Program çalışıtğında ilk constructor metodlar çalışır. Music adında bir class oluşturdum, içerisini constructor ve getName metodları ile doldurdum. getName metodu, şarkıcı ve şarkı adını döndürecektir.  class Music {  constructor(title, singer, images, way){  this.title = title;  this.singer = singer;  this.images = images;  this.way = way;  }  getName(){  return this.singer + " - " + this.title;  }  }  musicList adında bir liste oluşturuyorum ve bu liste title, singer, images, way bilgilerinin değerini alıyor.  const musicList = [  new Music("Çok Yazık","Çağan Şengün","1.jpeg","1.mp3"), | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 4. Hafta | 19.Gün | Yapılan Çalışmalar |
| 02.11.2023 | Perşembe |  |
| new Music("Tövbe","Derya Bedavacı & UZİ","2.jpeg","2.mp3"),  new Music( "Harika","Ebru Gündeş","3.jpeg","3.mp3"),  new Music("Lafı Mı Olur","Kibariye","4.jpeg","4.mp3"),  new Music( "Ateşe Düştüm","Mert Demir","5.jpeg","5.mp3"),  new Music("Basit Numaralar","Zerrin Özer", "6.jpeg","6.mp3")  Ardından musicplayer dosyası için bir MusicPlayer adında class oluşturuyorum ve içerisini constructor, getMusic(), next(), previous() metodları ile dolduruyorum. getMusic() müzik bilgisini indexe göre getirecek, next() müzik sayısı son müzige gelene kadar arttırır. Son müzikten sonra tekrar başa dönecek. previous() ise önceki müziği getirecektir.  class MusicPlayer{  constructor(musicList){  this.musicList = musicList;  this.index = 0;  }  getMusic(){  return this.musicList[this.index];  }  next(){  if(this.index + 1 != this.musicList.length) {  this.index++;  }  else{  this.index = 0;  }  }  previous(){  if(this.index != 0) {  this.index--;  }else {  this.index = this.musicList.length - 1;  }  app.js dosyasının içerisinde ise player adından bir obje oluşturdum ve music değişkenine getMusic metodunu çağırdım. Ardından console üzerinde görmek için yazdırdım. Kodların çıktısı aşağıdadır.  Display: Bu aşamada mp3 sayfasının görüntüsü üzerinde çalışacağım. Butonların görünmesi, image eklenmesi gibi. Öncelikle <container> ile çerçeve belirliyorum. Image, audio dosyalarını alıyorum. Ardından getName metodunun çağrılacağı music details kısmını yazıyorum. Ardından buton ile kontrollerin yapılacağı kısmı control adında bir <div> etiketi açıyorum ve butonları <i> ile ekliyorum. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 4. Hafta | 20.Gün | Yapılan Çalışmalar |
| 03.11.2023 | Cuma |  |
| <div class="container">  <img id="music-image" src="" alt="">  <audio id="audio" src=""></audio>  <div id="music-details">  <p class="title"> </p>  <p class="singer"> </p>  </div><div id="controls">  <i id="prev" class="fa-solid fa-backward-step"></i>  <i id="play" class="fa-solid fa-play"></i>  <i id="next" class="fa-solid fa-forward-step"></i>  </div>  </div>  Ardından const ile ulaşmak istediğim etiketleri seçiyorum. Bu seçimi ise querySelector metodu ile yapıyorum.  const title = document.querySelector("#music-details .title");  const singer = document.querySelector("#music-details .singer");  const prev = document.querySelector("#controls #prev");  const play = document.querySelector("#controls #play");  const next = document.querySelector("#controls #next");  Music dosyasındaki bilgileri DOM üzerinden aktarmak için bir event listener oluşturuyorum. DisplayMusic adında bir function oluşturuyorum ve burada mp3 ve image dosyalarının yolunu gösteriyorum. Text ile yazdırma işlemini ekliyorum. Son olarak click olayı gerçekleştiğinde müziğin oynatılması işlemini addEventListener ile gerçekleşmesini sağlıyorum.  window.addEventListener("load", () =>{  let music = player.getMusic();  displayMusic(music); });  function displayMusic(music){  title.innerText = music.getName();  image.src = "images/" + music.images;  audio.src = "mp3/" + music.way;}  play.addEventListener("click", () => { audio.play(); })  Bu aşamada yazdığım kodların çıktısı diğer sayfaya eklenmiştir.  Ekran görüntüsü 2023-11-11 153025  Controllerin Yapılması: Bu aşamada oynat butonuna tıklandığında müziğin başlaması, durdura bastığımızda(yine aynı buton) durması, önceki ve sonraki şarkıya geçme işlemleri yapılacaktır. pauseMusic() metodu ile playe tekrar tıkladığımızda classa playing özelliği silinir. Play.classListe tıklandığında pause  ikonu aftif olur. playMusic() metodunda playe tıkladığımızda playing özelliği eklenir. Play.classListe tıklandığında play ikonu aktif olur. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 5. Hafta | 21.Gün | Yapılan Çalışmalar |
| 06.11.2023 | Pazartesi |  |
| function pauseMusic(){  container.classList.remove("playing");  play.classList = "fa-solid fa-pause";  audio.pause();}  function playMusic(){  container.classList.add("playing");  play.classList = "fa-solid fa-play";  audio.play();}  Ardından önceki ve sonraki butonlarının işlevleri için kodlar yazdım. prev.addEventListener komutunda önceki butonuna tıklandığında precMusic() fonksiyonunu çağırır. prevMusic() fonksiyonu ise müziği başlatır, müzik bilgilerini getirir. Değiştirdikten sonra playMusic() dediğimiz için müzik direkt başlar.  prev.addEventListener("click", () => {  prevMusic();});  function prevMusic(){  player.prev();  let music = player.getMusic();  displayMusic(music);  playMusic();}  Sonraki butonuna tıklandığında nextMusic() fonksiyonunu çağırır. nextMusic() fonksiyonu, müziği başlatır ve müzik bilgilerini getirir. Değişim gerçekleştikten sonra playMusic() dediğimiz için müzik direkt başlar.  next.addEventListener("click", () => {  nextMusic();});  function nextMusic(){  player.next();  let music = player.getMusic();  displayMusic(music);  playMusic();}  Önceki sayfada önceki ve sonraki müziğe geçme işleminin kodlarını yazmıştım. Kodların çıktısı aşağıda ekran görüntülerinde verilmiştir.  Ekran görüntüsü 2023-11-11 180458 Ekran görüntüsü 2023-11-11 180516  Progress Bar Eklenmesi: Bu aşamada müziğin süresini görmek ve bu süre üzerinde ileri geri saniyeleri görmek ve o saniyeler üzerinde işlem yapmak için kodlar yazdım.  Öncelikle index.html üzerine progress bar ekledim. İçerisine başlangıç ve bitiş zamanını ekledim.  <div id="progress">  <input type="range" id="progress-bar"> | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 5. Hafta | 22.Gün | Yapılan Çalışmalar |
| 07.11.2023 | Salı |  |
| <div class="times">  <span id="current-time">0:00</span>  <span id="duration">0:00</span>  </div>  Ardından duration, currentTime, progressBar ögelerini html sayfasından seçip const ile değişkene atadım. Toplam müzik süresi saniye üzerinden hesaplandığı için bunu dakika cinsinden gösterim yapılması için kod yazdım. Ardından bu hesaplanan süreleri dakika ve saniye cinsinden sonuç adında değişkene atadım. audio.addEventListener üzerinden yüklenen veriyi müziğin ilişkilendirildiğini garanti altına alıyorum. Maximum süreyi ve duraklama zamanını gösteriyorum.  const calculateTime = (sumSeconds) => {  const minutes = Math.floor(sumSeconds / 60);  const seconds = Math.floor(sumSeconds % 60);  const updateSeconds = seconds < 10 ? `0${seconds}`: `${seconds}`;  const sonuc = `${minutes}:${updateSeconds}`;  return sonuc;}  audio.addEventListener("loadedmetadata", () => {  duration.textContent = calculateTime(audio.duration);  progressBar.max = Math.floor(audio.duration);});  audio.addEventListener("timeupdate", () => {  progressBar.value = Math.floor(audio.currentTime);  currentTime.textContent = calculateTime(progressBar.value);});  Süre Kontrolü: Bu aşamada progress bar üzerinde bir noktaya tıklandığında, progressBar a tıklandığında o tıklanan yerin saniyesini hesaplar ve oradan şarkıya devam eden kodu yazacağım.  progressBar.addEventListener("input", () =>{  currentTime.textContent = calculateTime(progressBar.value);  audio.currentTime = progressBar.value;});  Progress Bar Üzerinden Süre ve Ses Kontrolü: Bu aşamada progress bar üzerinden süre ve ses kontrolü yapacağım. volumeBar a tıklandığı yere göre ses düzeyi ayarlansın ve volumeBarda input bilgisi 100 üzerinden alınsın. audio da değer 0 ile 1 arasındadır bu yüzden 100 e bölünür. İlk if içinde ses sliderında 0 a tıkladığımda ikon değişsin istiyorum.  volumeBar.addEventListener("input", (e) =>{  const value = e.target.value;  audio.volume = value / 100;  if(value == 0){  audio.muted = true;  muteState = "unmuted";  volume.classList = "fa-solid fa-volume-xmark";}  else{  audio.muted = false;  muteState = "muted";  volume.classList = "fa-solid fa-volume-high";}});  Ses butonuna tıklandığında ses kapansın, 0 durumuna gelsin ve ikonu değişsin. Tekrar basıldığında ses  butonuna tıklandığında ses açılsın, 100 durumuna gelsin ve ikonu değişsin.  volume.addEventListener("click", () => {  if(muteState === "muted"){ | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 5. Hafta | 23.Gün | Yapılan Çalışmalar |
| 08.11.2023 | Çarşamba |  |
| audio.muted = true;  muteState = "unmuted";  volume.classList = "fa-solid fa-volume-xmark";  volumeBar.value = 0;}else{  audio.muted = false;  muteState = "muted";  volume.classList = "fa-solid fa-volume-high";  volumeBar.value = 100;}});    Müzik Listesi: Bu aşamada müzik listesinde tüm şarkıları görmek istiyorum. for içinde döndürüyorum.  const displayMusicList = (list) => {  for(let i=0; i < list.length; i++) {  let liTag = `<li li-index='${i}' onclick="selectedMusic(this)" class="list-group-item d-flex justify-content-between align-items-center">  <span>${list[i].getName()}</span>  <span id="music-${i}" class="badge bg-primary rounded-pill"></span>  <audio class="music-${i}" src="mp3/${list[i].file}"></audio></li>`;  ul.insertAdjacentHTML("beforeend", liTag);  let liAudioDuration = ul.querySelector(`#music-${i}`);  let liAudioTag = ul.querySelector(`.music-${i}`);  liAudioTag.addEventListener("loadeddata", () => {  liAudioDuration.innerText = calculateTime(liAudioTag.duration);});}}  Müzik listesindeki müziğin üzerine tıklandığında, tıklanan müziği oynatacaktır ve üzerine geldiğinde Mouse imleci cursor pointer olacak kodu yazacağım.  const selectedMusic = (li) => {  player.index = li.getAttribute("li-index");  displayMusic(player.getMusic());  playMusic();  isPlayingNow();}  const isPlayingNow = () => {  for(let li of ul.querySelectorAll("li")) {  if(li.classList.contains("playing")) {  li.classList.remove("playing");}  if(li.getAttribute("li-index") == player.index) {  li.classList.add("playing");  Müzik bittiyse diğer müziğe geçmesi için bir event listener yazacağım ve nextMusic() metodunu çağıracağım.  audio.addEventListener("ended", () => {  nextMusic();}) | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 5. Hafta | 24.Gün | Yapılan Çalışmalar |
| 09.11.2023 | Perşembe |  |
| Muzik player application ekran görüntüsü aşağıdadır. Müzik değiştirme, ses kontrolü, müzik listesi gibi uygulamalar gerçekleştirdim.  C:\Users\User\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Ekran görüntüsü 2023-11-12 201221.png C:\Users\User\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Ekran görüntüsü 2023-11-12 170349.png  Yazıcı Bilgi İşlem bölümü müdürümüz Murat Can ile bu zamana kadarki yaptığım uygulamalar ve bundan sonra hangi uygulamaları yapmam, hangi yönde ilerlemem gerektiği hakkında konuştum. Bilgilerimi güncel tutmam gerektiği, SAP' nin günümüzdeki kullanım alanları, şirketlerin SAP' ye olan taleplerinde yüksek oranda artış olduğu gibi konulara değindik.  Staj sonuna kadar SAP UI5 teknolojisini öğrenmeye başlayacağım. SAP, büyük işletmeler için iş süreçlerini otomatize etmek, verileri yönetmek ve iş zekası sağlamak için kullanılan bir yazılım ve teknoloji şirketidir.    **SAP Fiori-UI5-UX:** SAP Fiori, SAP tarafından geliştirilen modern ve kullanıcı dostu bir kullanıcı arayüzü ve deneyimi sunan bir teknoloji ve tasarım yaklaşımıdır. Bu yaklaşım, iş süreçlerini daha kolay ve etkili bir şekilde yönetmek için mobil cihazlar ve tarayıcılar üzerinden erişilebilen şık ve basit uygulamalar geliştirmeyi amaçlar.  SAP Fiori içerisindeki UI5; kullanıcı arayüzü oluşturulması ve geliştirilmesinde kullanılan açık kaynaklı web teknolojisi ve Javascript tabanlı bir frameworktür. Platform bağımsız sorunsuz çalışabilen duyarlı (responsive) web uygulamaları geliştirmenizi sağlar. UX ise kullanıcıların bir uygulama veya web sitesi ile etkileşimini değerlendiren ve geliştiren tasarım sürecini ifade eder.  **SAPUI5:** SAPUI5, JavaScript, CSS ve HTML5'i temel alan bir client teknolojisidir. Platform bağımsızdır, yani mobil cihaz, tablet, masaüstü bilgisayar ayırt etmeden çalışır. Kullanıcılar bir SAPUI5 uygulamasına eriştiğinde, uygulamanın tarayıcıya yüklenmesi için sunucuya istek gönderilir. View ilgili kitaplıklara erişir. Genellikle model de başlatılır ve business data veritabanından alınır.  Bir UI öğesi, kullanıcı arayüzlerimizin temel yapı taşıdır; özellikleri, olayları, yöntemleri ve ilişkileri olan yeniden kullanılabilir bir varlıktır. En önemli ilişkiler, diğer kullanıcı arayüzü öğeleriyle yapılan toplamalardır ve bu şekilde öğelerin ağaç yapısı oluşmaktadır.  Dosyalar: SAPUI5, farklı kullanım durumları için çeşitli önyükleme dosyaları sağlar. Bu dosyalar ve ne işe yaradıklarını açıklayacağım.  sap-ui-core.js - jquery-ui-positionZaten jQuery'yi ve çekirdek kütüphanenin ( ) yalnızca gerekli minimum parçalarını içerir.  sap-ui-core-nojQuery.js - Modüllerin çoğunu içerir. Hemen değil, yalnızca talep üzerine ayrıştırılır ve yürütülür.  sap-ui-core-lean.js - Bu önyükleme dosyası dosyaya benzer sap-ui-core.jsancak bu kullanım durumunda yalnızca jQuery ve bir SAPUI5 dosyası hemen yüklenir ve diğer dosyalar dinamik olarak yüklenir. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 5. Hafta | 25.Gün | Yapılan Çalışmalar |
| 10.11.2023 | Cuma |  |
| XML Şablonu Oluşturma: Bu xml şablonu view içeriğini oluşturmaktan sorumludur. return ifadesi, View.extend fonksiyonu "some.package.RouteTargetView" adında yeni bir view oluşturur. Bu fonksiyon, temel View sınıfını genişleterek createContent adlı bir metod sağlar. Then ise XMLView başarıyla oluşturulduğunda çalıştırılır. then bloğunda oluşturulan görünüm (oView) geri döndürülür.  createContent: function () {  return XMLView.create({  preprocessors: {  xml: {  // ...  }},  viewName: "some.package.TemplateView"  }).then(function (oView) {  return oView;  });  }  Modüller ve Bağımlılıklar: sap.ui.define modüller, fonksiyona dayalı olarak önceden tanımlanmış bir yapıya sahiptir. sap.ui.define, kaynakların eş zamansız olarak yüklenmesini sağlamaya yardımcı olur.  Her modülün, geri çağırma işlevi yürütülmeden önce çözülmesi gereken bağımlılıkların bir listesini tanımlanır. Bağımlılık bildirimleri çalışma zamanında değerlendirilebilir, ancak aynı zamanda derleme zamanında veya sunucudaki dağıtım zamanında da analiz eder.  Aşağıdaki kodda, sap.ui.define ile bir modül adı ve bağımlılıklar belirlenir. Modül adını, diğer dosyalardan bu modülü çağırmak için kullandım. İkinci parametre olarak verilen dizi, modülün bağımlı olduğu diğer modüllerdir. Bu modüller yüklenmeden önce, sap.ui.define fonksiyonunun içindeki callback fonksiyonunun çalışmasını beklerler. "SomeClass" modülü ve onun bir metodu olan foo() fonksiyonu tanımlanmıştır.  sap.ui.define("SomeClass", ['sap/mylib/Helper', 'sap/m/Bar'], function(Helper, Bar) {  var SomeClass = function () {};  SomeClass.prototype.foo = function () {  var mSettings = Helper.foo();  return new Bar(mSettings);  };  Tema: Uygulamanızın temasını seçmek için çeşitli seçenekler vardır.  Temel olarak renk şemasını değiştiren, ancak anında canlı önizlemeyle çok kolay olmayan bir şekilde değiştiren kullanıcı arayüzü tema tasarımcısını kullanarak mevcut bir temayı uyarlanabilir. Uyarlama parametreleri sınırlıdır, ancak kullanıcı arayüzü tema tasarımcısı aynı zamanda özel CSS eklemenize de olanak tanır. Daha sonra yüklenecek olan CSS'nin her parçasını yazarak sıfırdan yeni bir tema oluşturulabilir.  Label: Vertical layout oluşturuyorum. Bu düzen içinde label ve bir input ekliyorum. (V.view.xml)  <l:VerticalLayout class="sapUiContentPadding" width="100%">  <Label text="Name labelFor="input-a" />  <Input id="input-a" required="true" />  </l:VerticalLayout> | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 6. Hafta | 26.Gün | Yapılan Çalışmalar |
| 13.11.2023 | Pazartesi |  |
| Combo Box: "/CountriesCollection" veri modeli kullanılır ve bu veri modelindeki öğeler "text" özelliğine göre sıralanır. (view/ComboBox.view.xml)  <ComboBox  items="{path: '/CountriesCollection',  sorter: {path: 'text' }}">  <core:Item key="{key}" text="{text}" />  </ComboBox>  Oluşturulan JSON modeli, bu Controller'a bağlı olan görünüme (View) eklenir. Bu sayede, ComboBox'ın içeriği bu modele bağlı olarak görüntülenebilir ve güncellenebilir. ( controller/ComboBox.controller.js)  return Controller.extend("sap.m.sample.ComboBox.controller.ComboBox", {  onInit: function () { var oModel = new JSONModel(sap.ui.require.toUrl("sap/ui/demo/mock/countriesExtendedCollection.json"));  this.getView().setModel(oModel);}});  Text Area: Text area oluşturacağım. Xml ve json dosyalarını barındırır. Rows 8 olarak verilmiştir. Content yani içerik olduğu anlamına gelmektedir. Aşağıdaki kod xml dosyasıdır. Ekran görüntüsünde json dosyasının bir kısmı yer almaktadır. Görseldeki kodda bağımlılıklar ve dosya yolları verilmiştir.  <l:content>  <TextArea value="Bu bir text alanıdır." rows="8" />  </l:content>    Date and Time: Bu aşamada saat ve tarih seçme olaylarını, bu olayların html ile görünümüne değineceğim. DateTimePicker kontrolü, kullanıcının bir tarih ve saat seçmesine olanak tanır. Çıktının görüntüsü aşağıdadır.  <DateTimePicker  id="DTP1"  width="100%"  placeholder="Enter Date &amp; Time"  class="sapUiSmallMarginBottom" /> | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 6. Hafta | 27.Gün | Yapılan Çalışmalar |
| 14.11.2023 | Salı |  |
| Radio Buton: Bu aşamada iki çeşit radio buton koyacağım ve hangi seçeneğe tıklanırsa diğerinin işareti silinecektir. Çıktının görüntüsü aşağıda verilmiştir.  <RadioButtonGroup id="GroupA">  <RadioButton text="A" selected="true" />  <RadioButton text="B" selected="true" />  </RadioButtonGroup>    Panel: Bu aşamada bir panel örneği kodlayacağım. Panel, resim ve metni düzenli bir şekilde bir araya getirerek yanısıtır. Kodun çıktısının görüntüsü aşağıda verilmiştir.  <Panel width="auto" class="sapUiResponsiveMargin" accessibleRole="Region">  <headerToolbar>  <OverflowToolbar>  <Title text="Speaker Image"/>  </OverflowToolbar>  </headerToolbar>  <content>  <l:HorizontalLayout>  <Image src="{img>/products/pic1}" width="10em" />  </l:HorizontalLayout>  <Text text="This is an image of the speaker.This is an image of the speaker. This is an image of the speaker." />  </content></Panel>  controller/Panel.controller.js dosyasında image yolu verilmiştir ve image objesi getirilmiştir. Oluşturulan oImgModel JSON modelini, "img" adında bir isimle viewe bağlar. this.getView().setModel(oImgModel, "img") satırıyla model, viewe bağlanır ve "img" adı altında erişilebilir hale getirilir.  onInit : function (evt) {  var oImgModel = new JSONModel(sap.ui.require.toUrl("sap/ui/demo/mock/img.json"));  this.getView().setModel(oImgModel, "img");}  Message Strip: Her bir MessageStrip kontrolü, kullanıcıya bir bildirim veya mesaj göstermek için kullanılır. İki farklı bildirim tipini temsil eden iki MessageStrip yazılmıştır. Kodun çıktısı verilmiştir.  <MessageStrip  text="Error with default icon and close button:"  type="Error" showIcon="true" showCloseButton="true" class="sapUiMediumMarginBottom">  </MessageStrip>  <MessageStrip  text="Success with default icon and close button:"  type="Success" showIcon="true" showCloseButton="true" class="sapUiMediumMarginBottom">  </MessageStrip> | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 6. Hafta | 28.Gün | Yapılan Çalışmalar |
| 15.11.2023 | Çarşamba |  |
| Hata ya da başarılı bir durumda bu mesaj geldiğinde çarpı tuşuna basarak mesaj kapatılabilir.  Şirket tarafından Cross Feature Projesi yapmam istenildi. Bu doğrultuda projeyi yapıp şirkete teslim edeceğim. Kodlarım beğenilirse SAP Fiori ekranına ilgili mühendis tarafından dahil edilecektir.  Projede kullandığım dataset örnek datasettir.  Öncelikle Settings Panel tasarımını xml üzerinden yapacağım. <content> içerisine <VBox>, <HBox> ile class ve width değeri verdim. /dataset/values yolu üzerinden çektiğim veriyi metin olarak yazdırdım. <RadioButtonGroup> ile Zoom Initialization üzerinden, kullanıcının tek bir seçeneği seçmesini sağladım.  <Label text="{/dataset/name}" design="Bold" class='settingsLabel'></Label>  <RadioButtonGroup id='datasetRadioGroup' buttons="{path: '/dataset/values'}">  <buttons><RadioButton class='settingsRadio' text="{name}" select="onDatasetSelected"/></buttons>  </RadioButtonGroup>  /series/values yolu üzerinden çektiğim veriyi metin olarak yazdırdım. Ve Axis Type altındaki tek bir seçeneği seçmesini sağladım.  <Label text='{/series/name}' design="Bold" class='settingsLabel'></Label>  <RadioButtonGroup id='seriesRadioGroup' buttons="{path: '/series/values'}" ><buttons>  <RadioButton class='settingsRadio' text="{name}" select="onSeriesSelected"/>  </buttons></RadioButtonGroup>  Düzenlenmiş veri seti üzerinden {Store Name} ve {Revenue} değerlerini aldım.  <viz.data:dimensions>  <viz.data:DimensionDefinition name="Store Name" value="{Store Name}" />  </viz.data:dimensions>  <viz.data:measures>  <viz.data:MeasureDefinition name="Revenue" value="{Revenue}" />  </viz.data:measures>  initPageSettings dosyamda; kullanıcı telefonda ise ayarlar paneli otomatik daralacak. ChartContainer ve ChartContainerContent etiketleriyle bir çubuk grafik örneğini içeren bir "ChartContainer" oluşturularak gerçekleştirilir. ChartContainerContent oluşturdum. İcon ve title oluşturdum. ChartContainerContent etiketi ise content e sahip olmalıdır. Bu etket FlexBox ile yer değiştirir.  const vizframe = oView.byId("idVizFrame");  const oChartContainerContent = new ChartContainerContent({  icon : "sap-icon://horizontal-bar-chart",title : "vizFrame Bar Chart Sample",content : [ vizframe ]});  const oChartContainer = new ChartContainer({content : [ oChartContainerContent ]});  ZoomInitialization.controller.js dosyamda kontrollerin kodlarını yazacağım.  Öncelikle Sap UI define ettim ve bunların formatlarını belirttim. Ardından model ayarlarını yapmak için settingsModel adında bir dizi oluşturdum. Datasetin name, defaultSelected, ve values değerilerini tanımladım. Values içinde, name, vizProperties, plotArea, window değerlerini içerir.  Ardından kodun ilk başladığı zaman çalıştırılması için onInıt adında bir fonksiyon yazdım. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 6. Hafta | 29.Gün | Yapılan Çalışmalar |
| 16.11.2023 | Perşembe |  |
| Format.numericFormatter(ChartFormatter.getInstance()); satırı ile sayfa başlatıldığında kullanılacak sayı biçimlendirme işlemlerini yapılandırmak için ChartFormatter sınıfının bir örneğini alır ve sayı biçimlendirme için kullanılır. var oModel = new JSONModel(this.settingsModel) ile settingsModel adlı bir JSON modeli oluşturdum ve bu modeli viewe bağladım. var oVizFrame = this.oVizFrame = this.getView().byId("idVizFrame") ile "idVizFrame" adlı bir VizFrame nesnesini viewden aldım ve oVizFrame değişkenine atadım. oVizFrame.setVizProperties(props) ile oVizFrame'in görselleştirme özelliklerini props değişkenindeki ayarlarla yapılandırır. var dataModel = new JSONModel(this.dataPath + "/revenue\_cost\_consume/large.json") ile veri modelinidataModeli oluşturdum ve "dataPath" ile belirtilen yol altındaki "large.json" adlı bir JSON dosyasını kullanarak bu modele veri ekledim. oVizFrame.setModel(dataModel) ile oluşturulan dataModel'i VizFrame'e bağladım ve VizFrame, bu modele dayalı olarak verileri görselleştirebildi.  onInit:function (evt) {  Format.numericFormatter(ChartFormatter.getInstance());  var formatPattern = ChartFormatter.DefaultPattern;  var oModel = new JSONModel(this.settingsModel);  this.getView().setModel(oModel);  var oVizFrame = this.oVizFrame = this.getView().byId("idVizFrame");  var props = this.settingsModel.series.values[0].vizProperties;  oVizFrame.setVizProperties(props);  var dataModel = new JSONModel(this.dataPath + "/revenue\_cost\_consume/large.json");  oVizFrame.setModel(dataModel);  var oPopOver = this.getView().byId("idPopOver");  oPopOver.connect(oVizFrame.getVizUid());}  Kullanıcının bir veri seti seçimini işleyip, seçilen veri setine özgü görselleştirme özelliklerini ayarlayacağım.  Eğer bir veri seti seçimi yapılmamışsa veya seçim iptal edilmişse, fonksiyonu sonlandırdım. İkinci if bloğunda this.oVizFrame varsa ve radio düğmesi seçiliyse, içerideki kod bloğunu çalıştırır. this.oVizFrame.setVizProperties(vizProperties) ile vizProperties değişkenindeki özellikleri kullanarak VizFrame'in görselleştirme özelliklerini ayarladım. Yani, kullanıcı farklı bir veri seti seçtiğinde, görselleştirmenin özellikleri dinamik olarak değiştirilir.  onDatasetSelected : function(oEvent){  if (!oEvent.getParameters().selected) return;  var datasetRadio = oEvent.getSource();  if (this.oVizFrame && datasetRadio.getSelected()){  var bindValue = datasetRadio.getBindingContext().getObject();  var vizProperties = bindValue.vizProperties[this.seriesRadioGroup.getSelectedIndex()];  this.oVizFrame.setVizProperties(vizProperties);  }  } | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| 6. Hafta | 30.Gün | Yapılan Çalışmalar |
| 17.11.2023 | Cuma |  |
| Proje çalışmam, bilgi işlemdeki SAP Fiori ile çalışan Bilal Bey tarafından başaralı bulundu ve bir kısmı şirket projesine eklendi. Şirkette katkıda bulunduğum birden fazla C# form projesi vardır fakat şirket gizliliği nedeniyle ayrıntılarından bahsedemiyorum. Bazı projelerin, şirket datası girilmemiş halinin ekran görüntülerini aşağıya ekleyeceğim. İlk ekran görüntüsü gelir-gider takibi için yapılan bir uygulamadır. İkinci ekran görüntüsü ise Rehber Uygulamasıdır, şirkette çalışanların telefonlarının ve bazı bilgilerinin takibinin yapılabileceği bir uygulamadır. | | |
| PARAF | |  |

|  |  |  |
| --- | --- | --- |
| **İÇİNDEKİLER** | | |
| Bölüm | Yapılan Çalışmalar | Sayfa No |
| 1 | Kurum Tanıtımı ve İş Güvenliği Eğitimi | 3 |
| 2 | Proje Yönetim Metodları | 4-5 |
| 1 . Hafta | Javascript: Kurulum, Değişkenler, Operatörler | 6-7 |
| -2 . Hafta | String Metodları, Date, Diziler ve Dizi Metodları, Objects, DOM (Single, Multiple Selection Elements, Creating Elements,  Event Listener) | 8-12 |
| 3 . Hafta | DOM (Adding New Task, Delete-Update-Delete All Task, Check Task, Filters, Local Storage) | 13-17 |
| 4 . Hafta | Modern Javascript(Filter, Map, Maps, Sets, Classes, Getter–Setter), Music Player Project | 18-22 |
| 5 . Hafta | Music Player Project, SAP Fiori-UI5-UX, SAPUI5 Temelleri | 23-27 |
| 6 . Hafta | SAPUI5 Temelleri, Cross Feature Projesi, Yapılan Diğer Proje Detayları | 28-32 |
|  | Verimli bir staj süreci geçirdim. Web sitesi,  Javascript, C# Form içeren projelere katkıda bulundum. Şirketten ve stajdan memnun bir şekilde staj sürecim bitti. Şirkette işe başlama hakkına sahip olduğum için mutluluk içerisindeyim. |  |

**Kaynakça:**

1. <https://analyticahouse.com/tr/blog/agile-proje-yonetimi-ve-scrum-metodu-neden-onemlidir>
2. <https://medium.com/@cdbozdemir/kanban-nedir-scrum-vs-kanban-ea0993d6018a>
3. <https://www.geeksforgeeks.org/map-vs-object-in-javascript/>
4. <https://www.dilerhld.com/grup.asp?anagrup_no=3&grup_no=13>