|  |  |
| --- | --- |
|  | **T.C.**  **GEBZE TEKNİK ÜNİVERSİTESİ**  **MÜHENDİSLİK FAKÜLTESİ** |

**STAJ DEFTERİ**

**Öğrencinin;**

|  |  |
| --- | --- |
| **Adı, Soyadı** | **Onur KÜÇÜK** |
| **Numarası** | **151044059** |
| **Bölümü** | **Bilgisayar Mühendisliği** |
| **Staj Yaptığı Yer** | **NETAŞ** |
| **Staj Tarihleri** | **20-06-2018 / 17-07-2018** |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **-** | | **Özet** |
| Staj NETAŞ şirketinde uluslararası ar-ge departmanında yapılmıştır. Bu departmanda appstacle adındaki eclipse’in ve birçok şirketin ortağı olduğu bir IOT Application platformu olarak tasarlanan proje üzerinde çalışıldı.  Stajın ilk günlerinde projede kullanılan toollardan eclipse hono ve eclipse ditto üzerinde çalışmalar ve bazı testler yapılmıştır. Hono için küçük bir consumer yazılmıştır. Sonraki günlerde ise önce jsp ile java ve javascript kullanılarak, daha sonra ise springboot ile sadece java kullanılarak web servisleri yazılmıştır. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **20/06/2018** | | **Oryantasyon** |
| Şirketin stajyerler için hazırladığı oryantasyona katılındı. Oryantasyonda şirket, şirketin yapmakta olduğu projeler, bu projeler için kullanılan teknolojiler ile ilgili bilgiler edinildi. Stajyerin çalışacağı departman açıklandı. Departmanda çalışılacak çalışanlarla tanışıldı ve proje hakkında bilgiler alındı.  Stajyer Uluslararası AR-GE departmanında, APPSTACLE adlı projede çalışacaktır. APPSTACLE cloud ve IOT teknolojilerini içeren, arabalar ve toplu taşıma araçları için geliştirilen projedir. Bir uygulama platformu da bulunacak olan projede farklı kullanım amacı olan araçlar için farklı uygulamalar geliştirilecektir.  Projenin bulunduğu durumu gösteren bir demo yapıldı. Demo’da sisteme bağlı olan araçlar bir harita üzerinde gösterilmektedir ve araçların hangi doğrultu üzerinde hareket ettiği, aracın hızı, iç ve dış sıcaklığı, motor harareti vb. gibi bir çok önemli bilgi verilmektedir. Proje henüz başlangıç aşamasında olduğundan henüz gerçek araçlarla test etmeye başlanılmamış olup, sanal araçlar kullanılmaktadır. Sanal araç bir simülasyon üzerinden kontrol edilmektedir. İlerleyen zamanlarda gerçek araçlarla çalışılmaya başlanacaktır.  Demo bittikten sonra projede kullanılan ve kullanılacak olan teknolojiler, kullanım amaçları ile birlikte anlatıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **21/06/2018** | | **Bilgisayar teslimi ve Eclipse Hono** |
| Stajyerin bilgisayarı verildi. Projede kullanılan bir tool olan Eclipse Hono’yu araştırması ve kurması istendi.  Eclipse Hono birçok IOT cihazın bağlanabildiği, bu cihazlara uzaktan etkileşimde bulunabileceğiniz bir servis sağlar. Cihazlarla HTTP, MQTT gibi mesajlaşma protokolleri aracılığıyla telemetry mesajları kullanarak iletişimde bulunur. Henüz versiyon 1 yayınlanmamış olan Eclipse Hono halen geliştirilmektedir.  Önce Eclipse Hono’yu kurmak için gerekli ortam hazırlandı. Linux tabanlı bir işletim sistemi gerektiğinden Oracle VM VirtualBox kuruldu. Sanal işletim sistemi olarakta Ubuntu kuruldu.  İşletim sistemi yeni kurulduğundan önce gerekli araçları kurmak gerekti. Eclipse Hono java tabanlı bir maven projesi şeklinde yayınlandığından ve ayağa kaldırmak için docker gerektiğinden java, maven ve docker kuruldu. Eclipse Hono kendi github sayfasından indirildi. Compile işlemi sırasında bir hata oluştu. Çalışanlarla neden kaynaklanabileceği hakkında tartışıldı ve tekrar denendi fakat yine başarılı olmadı. Çalışma saati bittiğinden dolayı bir sonraki güne bırakıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **22/06/2018** | | **Eclipse Hono kurulumu ve araştırma** |
| Ubuntu yerine Linux Mint kuruldu ve Eclipse Hono’yu kurmak için gerekli adımlar tekrarlandı. Eclipse Hono kuruldu ve ayağa kalktı.  Hono’nun web sayfasındaki ‘Getting Started’ bölümü okunuldu ve burada bulunan rehber uygulandı. Rehberde Hono’nun kurulumuyla birlikte gelen ‘DEFAULT\_TENANT’ adlı bir tenantla çalışılıyor.  Tenant belirli bir tür cihazların kayıt olduğu alan gibi düşünülebilir. A tipi cihazlar a tenantına, B tipi cihazlar b tenantına kayıt yapılırsa, farklı tiplerdeki cihazlar daha kolay bir şekilde kontrol edilebilir. İsteğinize göre yeni bir tenant oluşturulabilir. Her tenantın consumerları vardır. Bu consumerlar o tenanttaki cihazlardan gelen telemetry mesajlarını alır. Bu consumerları özelleştirilerek alınan mesaja göre belirli bir iş yapılması sağlanabilir.  Rehberin devamında DEFAULT\_TENANT’a sanal bir cihaz oluşturuluyor ve bu cihazdan telemetry mesajları yollanıyor. Henüz özelleştirilmiş bir consumer yazılmadığından consumer panelinde sadece cihazdan mesaj alındığı gözüküyor, ekstra bir iş yapılmıyor.  Rehber tamamlandıktan sonra mesai bitene kadar Hono ile ilgili detaylı şeyler araştırıldı ve test edildi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **25/06/2018** | | **Agile Eğitimi ve Eclipse Hono araştırma** |
| İki gün sürecek olan agile eğitiminin ilk gününe katılındı. Bu rehberde agile’ın temelleri anlatıldı ve agile’ı daha iyi anlamak için geliştirilmiş bazı uygulamalarda bulunuldu.  Uygulamalarda iki farklı grup seçildi ve bir grup yapması gerekeni normal bir şekilde yaparken diğer grup yapması gerekeni agile yöntemine uygun bir şekilde yaptı ve bu iki grup karşılaştırılarak hangi grubun daha verimli olduğu tartışıldı. Burada her iki grubunda avantajlar ve dezavantajları olduğu fakat daha avantajlı olan grubun agile grubu olduğu farkedildi.  Agile kısaca müşteri ile iletişim halinde bulunulan, müşterinin isteklerini daha iyi anlanılan, projede yapılan değişikliklere daha hızlı adapte olunan bir proje geliştirme yöntemidir. Agile’ın kendi içinde bazı uygulama yöntemleri vardır. Bunlardan kısaca bahsedildikten sonra eğitimin ilk günü tamamlandı.  Eğitimin ilk günü tamamlandıktan sonra Eclipse Hono ile ilgili araştırmalara devam edildi. Yeni tenantlar, cihazlar ve consumerlar oluşturulup yeni öğrenilen bilgiler test edildi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **26/06/2018** | | **Agile eğitimi ve Eclipse Hono testler** |
| Agile eğitiminin ikinci gününe katılındı. Eğitimin ikinci gününde agile’ın içinde bir yöntem olan scrum anlatıldı ve yine bir önceki gün gibi uygulamalar yapıldı.  Scrum’da proje sprintlere ayrılır. Genelde maksimum 1 ay süren sprintler için önceden belirlenen bir noktaya gelinmeye çalışılır. Her sprint sonunda ne kadar başarılı olunduğuyla ilgili, her sprint başında o sprintin maddeleri belirlenmek üzere toplantı yapılır. Bunun haricinde günlük toplantılar olur ve bu günlük toplantılarda proje çalışanları dün yaptıklarını, bugün yapacaklarını anlatır ve önünde bir engel olup olmadığını belirtir. Eğer bir engel varsa bu engele bir çözüm bulunur.  Eğitim tamamlandıktan sonra stajyerden Eclipse Hono’da bazı şeyleri test ettmesi istendi. Eclipse Hono henüz stabil bir tool olmadığından kullanılacak özelliklerde hatalar olup olmadığı test edilmesi gerekiyordu.  Aynı tenant için birden fazla consumer konsolu açılıp test edilebilecek tüm yöntemler uygulandı ve mesaj kaybı veya herhangi bir hata var mı diye kontrol edildi. Herhangi bir sorun bulunamadı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **27/06/2018** | | **Eclipse Hono testler ve proje toplantısı** |
| Farklı tenantlar için consumer konsolları açıp bir önceki gün yapılan testler uygulandı. Daha sonra hem farklı hem aynı tenantlar için konsollar bir arada açılıp yine aynı testler uygulandı. Herhangi bir sorun bulunulmadı.  Tüm bu testler bittikten sonra diğer çalışanlara rapor edildi ve testler üzerine tartışıldı ve sonuçlar değerlendirildi.  Günün devamında projenin gidişatı ile ilgili bir toplantı yapıldı. Proje üzerinde genel bir konuşma gerçekleştirildi ve izlenmesi gereken yolun kararlaştırılması gerektiği konuşuldu. Projede uygulanacak yapının belli olması gerektiği söylendi ve bir sonraki gün için büyük bir toplantı gerçekleştirileceği, bu toplantı için herkesin proje için aklına gelen yapıları avantaj ve dezavantajlarını belirterek bir çizmeleri istenildi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **28/06/2018** | | **Proje yapısı toplantısı ve Eclipse Ditto** |
| Daha önce konuşulan toplantı gerçekleşti. Toplantıda çalışanlardan toplamda 5 tane yapı önerisi sunuldu. Yapılar sırayla anlatıldı. Yapıların avantajları ve dezavantajları tartışıldı. Seçim için düşünülmesi gerektiği ve gün içerisinde tekrar toplanılması kararlaştırıldı. İncelenmesi için herkese yapıları içeren bir mail atıldı.    Toplantıda konuşulan yapılara bir örnek.  Yapılar detaylıca incelendikten sonra yeniden toplanıldı. Bazı yapılarda öneriler sonucunda değişiklikler yapıldı. Sonuç olarak bir yapı seçildi ve toplantı sonlandı.  Günün geri kalanında kullanılacak olan toollardan Eclipse Ditto kuruldu ve çalıştırıldı. Eclipse ditto bir IOT cihazın dijital ikizini oluşturan ve bunu bir cloudda tutan bir tooldur. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **29/06/2018** | | **Server kurulumu ve Hono-Ditto çalışmaları** |
| Projeye ait bir server bilgisayarı kurluması gerekiyordu. Projeye ait toolları stabil çekilde çalışacak şekilde kuran son kişi stajyer olduğundan bilgisayarı kurma görevi ona verildi. Önce şirketin server odasına gidildi. Bu oda şirkete ait tüm serverların bulunduğu büyük bir oda. Burada projeye ait olan bilgisayara Linux Mint kuruldu. Bilgisayara IP adresi verildi ve dışarıdan erişebilmek için gerekli işlemler yapıldı.  Günün devamında bu bilgisayara ssh ile bağlanılıp eclipse hono, eclipse ditto ve bunları çalıştırmak için gereken programlar yüklenildi. Böylece çalışanların ve şirket dışı proje ortaklarının kullanabileceği bir bilgisayar kurulmuş oldu.  Stajyerden nasıl consumer yazılacağını araştırması ve devamında bir consumer yazması istendi. Günün geri kalanında consumerın nasıl yazılacağı araştırıldı.  Hono’nun default consumer’ı üzerinden yeni bir consumer yazılabileceği farkedildi ve Hono’nun source kodları incelenildi ve anlaşılmaya çalışıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **02/07/2018** | | **Hono Consumer** |
| Bu günden itibaren her günün başlangıcında scrum yapmaya karar verildi.  Bugünün scrumında herkes ne durumda olduğu ve ne yapacağını konuştu.  Stajyerden basit bir hono consumer’ı yapması istendi ve bu consumerda neler istendiği belirtildi.  Hono consumer’ı yazılmaya başlandı. Hono mesajlaşmalarını json formatında yaptığı için bir json kütüphanesi kullanmak işi kolaylaştıracaktı. Bunun için json kütüphaneleri araştırıldı ve en işe yarayacak olan seçildi.  Seçilen kütüphanenin maven dependency’si:  <dependency>  <groupId>org.json</groupId>  <artifactId>json</artifactId>  <version>20180130</version>  </dependency>  Gelen mesajlarda bazı kısımlarda null değerler oluştuğu farkedildi. Bu nedenle json formatına tam olarak uygun olmadığından hatalar alındı. Mesajı uygun formata dönüştürecek bir fonksiyon yazıldı.  Daha sonra gelen mesaj bu kütüphane yardımıyla parse edildi. İlk etapta mesajlar düzgün bir şekilde iletiliyor mu, parse işlemi düzgün yapılıyor mu diye kontrol edildi ve hatalar giderildi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **03/07/2018** | | **Hono Consumer** |
| Günlük scrum gerçekleştirildi. Bu scrumda projenin gidişatıyla ilgili verilmesi gereken kararlar konuşuldu. Projenin ortaklarıyla görüşmek üzere bazı kararlar üzerinde düşünüldü ve fikirler alındı.  Consumer yazılmaya devam edinildi. Gelen mesajlardan tenant ve cihaz bilgileri alındı. Alınan bilgiler kullanılarak her tenant ve cihaz için gelen mesajlar ayrı ayrı sistemde dosyalanacak şekilde bir fonksiyon yazıldı. Örneklemek gerekirse:  Hono mesajları:  -Tenant1:  -Cihaz1mesajları  -Cihaz2mesajları  -Tenant2:  -Cihaz1mesajları  gibi bir dosyalama oluşturuldu. Böylece hangi tenantta hangi araca ne mesaj geldiği tek tek sistemde tutulmuş olacaktı. Mesajların ne zaman geldiğini belirtmek için de, mesajların alındığı zamanı epochtime cinsinden “Instant.now().toEpochMilli();” fonksiyonu aracılığıyla hesaplayıp bu dosyalara yazıldı. Böylece gelen mesajların zamanı da belirtildi.  Şimdilik sistemde dosyalanarak tutulan bu veriler daha sonra oluşturulacak olan cloud sisteminde database şeklinde tutulacaktır. Consumer içerisinde de gelen mesajların içeriğine göre işlemler yapılacaktır. Örneğin: araba iç sıcaklığı belli bir değerin üstündeyse klimayı çalıştır, arabanın hızı bulunduğu yolun hız sınırını aşıyorsa arabayı yavaşlat vb. gibi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **04/07/2018** | | **JSP Web Servisi** |
| Günlük scrum yapıldı. Bu scrumda proje için yapılacak uygulamalardan birisi hakkında konuşuldu. Bu uygulamada kayıtlı araçlar ve sürücüler bulunmakta. Bu sürücüler sadece kendine atanan aracı, kendine atandığı süre içerisinde, kendine atandığı yetkiler ile kullanabilecek. Her araçta bir cihaz bulunacak ve sürücüler kimlik doğrulamasını bu cihaz ve mobil uygulamadaki QR kod ile sağlayacak. Adminin sürücüleri ve araçları kontrol edebileceği bir arayüz olacak. Stajyerden bu uygulama için bir admin web arayüzü yapılması istendi.  Stajyerin daha önceden web ile bir tecrübesi olmadığından önce bir araştırma yapıldı. Araştırmanın sonucunda jsp ve java kullanarak güzel bir servis yapabileceği kararına vardı.  İlk olarak bir kimlik doğrulama servisi ve arayüzü yazıldı. Bir sonraki sayfaya kimlik doğrulamadan direk geçilememesi için jsp nin session özelliği kullanıldı. Giriş yapıldığında ‘session.setAttribute("username", username);  session.setAttribute("password", password);’ şeklinde username ve password tutuluyor ve her sayfada ‘session.getAttribute("username").toString();’ şeklinde bir kontrol yapılıyor. Burada eğer kimlik doğrulama yapılmamışsa session içinde “username” adlı değişken bulunmadığından bir exception fırlatıyor. Bu exception kullanıcıyı otomatik olarak kimlik doğrulama sayfasına yönlendirecek şekilde handle edildi.  Böylece kullanıcı kimlik doğrulama yapmadan diğer sayfalara giremeyecek. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **05/07/2018** | | **Web servisi anasayfa** |
| Günlük scrum yapıldı. Bu scrumda herkes ne durumda olduğunu, neler yaptığını ve neler yapacağını tartıştı.  Web arayüzü yapılmaya devam edildi. Kimlik doğrulama yapıldığında girilecek ana sayfa tasarlandı. Bu sayfa dashboard tarzında tasarlandı. Sayfada ana sayfaya yönlendiren bir tuş, giriş yaptığın kullanıcıyı belirten bir kısım ve çıkış kısmı içeren bir üst bar, yapacağın işlemleri içeren bir yan bar ve sayfanın içeriğinin olduğu kısım bulunmakta. Bu sayfa servisin genel sayfası olacak ve yapılacak işlemler seçildiğinde jsp’nin %@include file="file\_name.jsp"% özelliğini kullanarak diğer sayfaların bu sayfada gözükmesi sağlanacak.  Sayfa üzerinde geliştirmeler yapıldı. Yetki ekleme sayfası açıldı ve tasarlanmaya başlandı.    Ana sayfanın şuan ki hali.(Rule = yetki ekleme sayfası) | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **06/07/2018** | | **Web servisi ditto araçlar** |
| Günlük scrum gerçekleştirildi. Projenin ortaklarıyla yapılan, projenin yapısıyla ilgili olan toplantıda ki kararlar konuşuldu. Herkesin durumu, ne yaptığı konuşuldu ve scrum sonlandırıldı.  Yetki ekleme sayfasında yetki eklenecek araç ve yetkilendirilecek sürücülerin çekilmesi gerekiyordu. Araçlar eclipse ditto üzerinde kayıtlı tutulduğundan ordan çekilmesi doğru bulundu. Ditto üzerinden araçları curl ile kolayca çekilebildiğinden, java için curl işlemini yapan bir kütüphane araştırıldı. Sonuç olarak ‘<dependency>  <groupId>org.toile-libre.libe</groupId>  <artifactId>curl</artifactId>  <version>LATEST</version>  </dependency>' kütüphanesi kullanılmaya karar verildi. Bu kütüphanede ki curl(String parameter) fonksiyonu sayesinde araçlar json formatında elde edildi.  Araçları daha kolay kullanabilmek için bir class oluşturuldu. Bu class araçlarla ilgili verileri tutacak, araçla yapılacak işlemleri kolaylaştıracak fonksiyonları içerecek şekilde tasarlandı. Daha sonra curl ile aldığımız jsondan araçların verilerini çekip parse edilerek bu classın tipindeki bir arrayliste atıldı. Böylece tüm araçlar kolayca işlenebilecek şekilde tek bir yerde tutulmuş olundu. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **09/07/2018** | | **Web Servisi yetki kaydı** |
| Web servisi için yetki verme işlemlerine devam edildi. Araç listesini alınıp araç için oluşturduğumuz class’ı içeren bir arraylistte tuttulmuştu. Sürücü listesini alma işlemi yapılması gerekiyordu. Serverdaki sürücü listesi proje grubundan proje için appstore hazırlayan kişi tarafından sağlanacaktı. Henüz hazır olmadığı için şimdilik geçici olmak üzere üç sürücü içeren bir array tanımlandı. Sürücü listesi servisi hazır olana kadar bu array ile çalışılacak.  Yetki kaydı için bir pop-up hazırlandı. Bu pop-up ta ditto’dan çektiğimiz araç listesini ve geçici sürücü listesini dropdown menüsü şeklinde yerleştirildi. Verilen yetkiye bir isim vermek için bir textbox yerleştirildi. Yetkinin verileceği süreyi belirtmek için bir takvim ve saat seçeneği yerleştirildi. Her aracın alabileceği yetkiler farklıdır ve bu yetkiler araç kayıt olurken atılan json mesajında belirlenir. Ditto’dan çektiğimiz araç listesinde araçlara verilecek yetkilerde bulunmaktadır. Dropdown menüsünden bir aracı seçtiğinde o araca vereceğin yetkiyi seçebilmek için jsp nin içinde iç içe javascript ve java kodu yazıldı. Yetki listesini içeren bir dropdown menüsü oluşturuldu.  <%devices = (ArrayList<Device>)session.getAttribute("devices");%>  var div = document.getElementById("commands");  var device = document.getElementById("device");  var index = parseInt(device.options[device.selectedIndex].value);  var devicenames = [];  var commands = [];  <%for(int i=0; i<devices.size(); ++i){%>  devicenames.push("<%=devices.get(i).getName()%>");  commands.push("<%=devices.get(i).getCommands()%>");  <%}%>  var array = commands[index].split(',');  Kodun bir kısmı bu şekilde. <% - %> arasında yazılan kısım jsp’nin özelliği olan html dosyasının içinde java kodu yazma özelliğidir. Bununla birlikte yetki verme sayfası tamamlandı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |
| --- | --- |
| Yetki ekleme sayfasının son hali. | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** |
| **Onur KÜÇÜK** | **Caner AKSOY** |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **10/07/2018** | | **Springboota giriş** |
| Günlük scrum yapıldı. Bu günlük scrumda herkesin o an meşgul olduğu şeylerden, varsa problemlerden bahsedildi. Proje yöneticisi stajyerin yapmakta olduğu web servisine detaylıca baktı. Diğer çalışanın yaptığı web servisi olan appstore springboot ile yazıldığı için, bu web servisininde springboota geçirilmesi kararı alındı.  Springboot hiç bilinmediğinden hakkında araştırmalar yapıldı. Birkaç örnek proje incelendi ve tutorial videoları izlendi. Giriş için önerilen birkaç servis denendi. Anlaşılmayan yerler spring kullanan çalışanlara soruldu. Gün boyu spring’i öğrenmeye yönelik buna benzer çalışmalar yapıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **11/07/2018** | | **Springboota giriş** |
| Bugün de gün boyu spring çalışıldı. Bir çok siteden örnek spring projeleri incelendi ve sıfırdan bir kaç basit projeler yazıldı. Spring’de kullanılması gereken yapılar incelendi.  Proje için yazılan springboot web servisi olan appstore incelendi. Kullanılan yapı, kullanılan frameworkler araştırıldı. Günün sonuna kadar bu şekilde spring’i öğrenmeye yönelik ve web servisini yazabilmeye yönelik çalışmalar yapıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **12/07/2018** | | **Spring web service** |
| Appstore’u yazan çalışanla birlikte springboot çalışıldı. Appstore’u yazılırken kullanılan vaadin adındaki java web framework çalışıldı. Web servisini yazmaya başlamak için yeterli spring öğrenildikten sonra yazılmaya başlandı.  Öncelikle kullanıcının özelliklerini tutmak üzere bir User class’ı yazıldı. Spring’in çalışması için gerekli olan UserRepository classı ve UserService classı yazıldı. Daha sonra ui olarak LoginView classı yazıldı. Bu class’ta kullanıcı giriş ekranı bulunmakta. Spring ve vaadin özellikleri kullanılarak tamamen java kodundan oluşan bir ui.  @SpringView(name = LoginView.***VIEW\_NAME***)  **public** **class** LoginView **extends** CustomComponent **implements** View {  **public** **static** **final** String ***VIEW\_NAME*** = "login";  **public** **static** **final** String ***TITLE\_NAME*** = "Login";  // private Navigator navigator;  **private** TextField username = **new** TextField();  **private** PasswordField password = **new** PasswordField();  **private** Button login = **new** Button("Login");  @Autowired  UserService userManagerService;  @Autowired  **public** LoginView() {    com.vaadin.server.Page.*getCurrent*().setTitle(***TITLE\_NAME***);  CustomLayout sample = **new** CustomLayout("login-layout");  username.setWidth(100.0f, Unit.***PERCENTAGE***);  sample.addComponent(username, "username");  password.setWidth(100.0f, Unit.***PERCENTAGE***);  sample.addComponent(password, "password");  login.setClickShortcut(ShortcutAction.KeyCode.***ENTER***);  sample.addComponent(login, "okbutton");  login.addClickListener(**new** ClickListener() {  @Override  **public** **void** buttonClick(ClickEvent event) {  User loggedUser = userManagerService.findByUserNameAndPassword(username.getValue(), password.getValue());    **if** (loggedUser != **null**) {    VaadinSession.*getCurrent*().setAttribute("user", loggedUser.getUserName());  VaadinSession.getCurrent().setAttribute("isCurrentUserAdmin", loggedUser.getAdminuser());  Page.getCurrent().setUriFragment("!" + RuleView.VIEW\_NAME);  } **else** {  **new** Notification("Login Failed", "Invalid username or password!", Notification.Type.ERROR\_MESSAGE).show(Page.getCurrent());  }  }  });  setCompositionRoot(sample);  }  LoginView ui’ının kodu.  Sonrasında jsp için yazılan web servise’te de yazılan, Device classı’ı ve araçları dittodan çeken class spring’e uygun şekilde yazıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **13/07/2018** | | **Spring web service** |
| Günlük scrum yapıldı. Bu scrumda stajyerin durumu kontrol edildi. Yapılacak olan web servisinin tasarımı çizildi.  Kullanıcı login olduktan sonra ekrana gelecek olan ui çizildi. Bu çizime göre o sayfa yazılacaktı. Öncesinde güvenlik önlemi olması için bir kod parçası yazıldı. Bu kod parçasında anasayfanın linki girildiğinde giriş yapmış herhangi bir kullanıcı var mı yok mu o kontrol edilmekte. Eğer giriş yapmış bir kullanıcı varsa sayfaya devam ediyor fakat giriş yapmış bir kullanıcı yoksa kullanıcıyı giriş ekranına atıyor.  **private** **void** router(String route) {  **if** (getSession().getAttribute("user") != **null**) {  User loggedUser = userRepository.findByUserName(getSession().getAttribute("user").toString());  getNavigator().addView(RuleView.***VIEW\_NAME***, RuleView.**class**);  getNavigator().addView(UserEditView.***VIEW\_NAME***, UserEditView.**class**);  getNavigator().addView(SignUpView.***VIEW\_NAME***, SignUpView.**class**);  **if** (route.equals("!main")) {  getNavigator().navigateTo(RuleView.***VIEW\_NAME***);  } **else** {  String gotopage;  gotopage = com.vaadin.server.Page.*getCurrent*().getUriFragment();  **if** (gotopage == **null**) {  gotopage = "!main";  }  com.vaadin.server.Page.*getCurrent*().setUriFragment(gotopage);  }  } **else** {  **if** (route.equals("!signup")) {  om.vaadin.server.Page.*getCurrent*().setUriFragment("!" + SignUpView.***VIEW\_NAME***);  } **else** {  com.vaadin.server.Page.*getCurrent*().setUriFragment("!" + LoginView.***VIEW\_NAME***);  }  }  }  Kod parçasının bir kısmı. Bu kısım yazılırken bir kullanıcı kayıt olma sayfası ve kullanıcı ayar değiştirme sayfası da yazıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **16/07/2018** | | **Spring web service** |
| Günlük scrum yapıldı. Bu scrumda herkesin durumu, o an ne yaptığı ve ne yapacağı vb. konuşuldu.  Yetki ekleme sayfası yazılmaya başlandı. JSP’de yazılan yetki ekleme sayfasına göre çok daha farklı bir sayfa tasarlandı. Bu sayfada üstte yetki eklemek için bir buton bulunacak ve tıklanıldığında bir pop-up açılacak. Bu pop-up JSP’de yazılan yetki sayfasıyla aynı olacak. Butonun alt tarafında ise o an etkin olan yetkilerin bir listesi bulunacak ve bu yetkilerin tüm detayları burada gözükecek. Yetkilerin yan tarafında ise yetkileri düzenleme butonu olacak bu şekilde yetkiler değiştirilebilecek.  İlk iş olarak yetkinin verilerinin tutulduğun Rule classı, RuleRepository ve RuleService yazıldı. Daha sonra sayfa düzeni yazılmaya başlandı. Sayfaya rule ekleme butonu eklendi. Şimdilik bu buton herhangi bir iş yapmamakta. Daha sonra eklenen rule’ların gözükeceği tablo eklendi. Bu tabloda “id – rulename – devicename – drivername – datetime – command” gibi bilgiler bulunmakta. Tablonun yanına da burda bulunan yetkiyi değiştirmek – silmek için bir buton koyuldu. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ:** | | **KAPSAMI:** |
| **17/07/2018** | | **Spring web service** |
| RuleEditor sayfası yazılmaya başlandı. Bu sayfa bir pop-up şeklinde belirecek şekilde tasarlandı. Yeni bir yetki ekleme ya da bir yetkiyi değiştirmek için butonlara basıldığında bu sayfa açılacak. Her iki buton için farklı şekilde çalışacak.  JSP’de yapılan web servisine benzer bir pop-up tasarımı yapılmaya çalışıldı.  @SpringComponent  @UIScope  **public** **class** RuleEditor **extends** VerticalLayout **implements** View{  @Autowired  RuleService ruleService;  **public** Rule rule;  **public** TextField rulename = **new** TextField("Rule Name");  **public** Button save = **new** Button("Save", ~~FontAwesome~~.~~SAVE~~);  **public** Button cancel = **new** Button("Delete",~~FontAwesome~~.~~CLOSE~~);    /\*Dittodan çekilen araçların listesi\*/  ArrayList<Device> devices;  /\*Rule eklerken veya düzenlerken seçtiğimiz aracın komutları\*/  String commands;    HorizontalLayout hlayout = **new** HorizontalLayout();  HorizontalLayout commandlayout = **new** HorizontalLayout();  VerticalLayout vlayout = **new** VerticalLayout();  Binder<Rule> binder = **new** Binder<>(Rule.**class**);  NativeSelect<String> devicelist = **new** NativeSelect<>();/\*Araçların bulunduğu dropbox\*/  NativeSelect<String> driverlist = **new** NativeSelect<>();/\*Sürücülerin bulunduğu dropbox\*/  NativeSelect<String> commandlist = **new** NativeSelect<>();/\*Komut isimlerinin bulunduğu dropbox\*/  NativeSelect<String> commandselect = **new** NativeSelect<>();/\*Komut seçeneklerinin bulunduğu dropbox\*/  InlineDateField date = **new** InlineDateField();/\*Takvim\*/    @Autowired  **public** RuleEditor() {  devices = **null**;  /\*Dittodan araç listesini çekiyor. Çekemezse ditto ayakta değildir. Konsola bunu belirtiyor\*/  **try** {  devices = DeviceList.*getDevices*();  }**catch**(Exception e) {  System.***out***.println("================");  System.***out***.println("Ditto is not up!");  System.***out***.println("================");  }    rulename.setWidth("200px");  vlayout.addComponents(rulename);    Collection<String> devicenames = **new** ArrayList<>();  **for**(**int** i = 0; i < devices.size(); ++i) {  devicenames.add(devices.get(i).getName());  }  devicelist.setItems(devicenames);    devicelist.addValueChangeListener(event -> {  rule.setDevicename(event.getValue());  **for**(**int** i = 0; i < devices.size(); ++i) {  **if**(event.getValue().equals(devices.get(i).getName())) {  commands = devices.get(i).getCommands();  }  }  commandlist.setItems(commands.split(","));  });    /\*Burda appstore'dan sürücülerin listesini alıp eklemek gerekiyor  Şuanlık ben elimle 3 tane ekledim\*/  Collection<String> drivernames = **new** ArrayList<>();  drivernames.add("Abbas");  drivernames.add("Ahmed");  drivernames.add("Ali");  driverlist.setItems(drivernames);    driverlist.addValueChangeListener(event -> {  **if**(event.getValue() != **null**)  rule.setDrivername(event.getValue());  });    commandlist.addValueChangeListener(event -> {  **if**(event.getValue() != **null**) {  rule.setSelectedcommand(event.getValue());  commandselect.setItems(commandselector(event.getValue()).split(","));  }  });    commandselect.addValueChangeListener(event -> {  **if**(event.getValue() != **null**)  rule.setCommand(rule.getSelectedcommand() + ":" + event.getValue());  });    vlayout.addComponent(devicelist);  vlayout.addComponent(driverlist);  commandlayout.addComponent(commandlist);  commandlayout.addComponent(commandselect);  vlayout.addComponent(commandlayout);    date.setDateFormat("yyyy-MM-dd'T'HH:mm:ss.SSSZ");  date.addValueChangeListener(event -> {  rule.setDatetime(event.getValue().toString());  });  date.setRangeStart(**new** java.util.Date().toInstant().atZone(ZoneId.*systemDefault*()).toLocalDate());    vlayout.addComponent(date);  vlayout.addComponents(save,cancel);  hlayout.addComponent(vlayout);    addComponents(hlayout);  binder.bindInstanceFields(**this**);  setSpacing(**true**);  save.setStyleName(ValoTheme.***BUTTON\_PRIMARY***);  save.setClickShortcut(ShortcutAction.KeyCode.***ENTER***);  setVisible(**false**);  }    /\*Her yeni command için buraya ekleme yapılması gerekiyor\*/  **private** String commandselector(String command) {  /\*Seçenekleri on ve off olan komutlar\*/  **if**(command.equals("motor") || command.equals("light")) {  **return** "on, off";  }/\*Seçenekleri 1 ve 9 arasında rakamlar olan komutlar\*/  **else** **if**(command.equals("run")) {  **return** "1,2,3,4,5,6,7,8,9";  }  **return** **null**;  }  RuleEditor classının bir kısmı bu şekilde. Bu classta bittiğinde bu servisin şuanlık istenen kısmı bitmiş oldu.    Servisin genel yapısı bu şekilde.  Servisin istenilen kısımları bittiğinden küçük bir demo yapıldı. Demoda yapılan şeyler detaylı bir şekilde anlatıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
| **Onur KÜÇÜK** | **Caner AKSOY** | |