

## ----- SORU 1 -----

### 1) FUNCTIONAL TESTING USING SELENIUM (1. seçim)

#### a) Fonksiyonel test nedir?

Fonksiyonel Test, testini gerçekleştirdiğimiz uygulamanın her bir fonksiyonunun verilen gereksinimlere uygun olarak çalışıp çalışmadığını doğrulayan test türüdür.

#### b) Selenium nedir?

Selenium, farklı tarayıcılarda web uygulamalarını test etmek için kullanılan açık kaynaklı ve ücretsiz test aracıdır. Sürekli kendini tekrar eden testlerin manuel olarak yapılması ekstra iş yükü oluşturduğundan testleri otomatikleştirmek zaman kazanmamızı sağlayacaktır. Kısacası, test komut dosyası dili öğrenmeye gerek kalmadan çoğu modern web tarayıcısında işlevsel testler yazmamızı sağlar.

#### d) Selenium Tarihçesi

2004 yılında Jason Huggins tarafından ilk versiyonu ortaya çıkmıştır. Huggins web uygulamalarını test etmek için çok fazla zaman ve enerji harcıyordu ve bundan çok sıkılmış ve yorulmuştu. Çözüm olarak ise manuel testlerde kendini tekrarlamayan bir JavaScript çerçevesini (kütüphanesi) buldu. İlk olarak JavaScriptTestRunner olarak bilinen bu tasarım, testleri direk olarak bir tarayıcıda çalıştırmaktadır. Bunun sayesinde birden fazla tarayıcı da testlerin otomatik olarak çalışmasına izin vermektedir. Böylece daha önce kullanılan test araçlarının hiçbirinde olmayan bir özellik ortaya çıktı. Özet olarak, kullanıcıları kendi belirledikleri herhangi bir programlama dilini kullanarak bir tarayıcıyı kontrol altına almasını sağlayan ilk araç oldu. Ancak zamanla olumsuz yönleri ortaya çıktı. JavaScript tabanlı olduğu için bazı şeyleri yapmakta yetersiz kaldı. Web uygulamaları belirli bir zaman sonra karmaşık bir hal almaya başladı ve bundan kaynaklanan kısıtlamaların sayısı arttı. Birkaç yıl sonra, Google’ da çalışan ve test aracını kullanan Simon Stewart isimli bir mühendis, bu kısıtlamalardan bıktı. Stewart, tarayıcıyla direk olarak konuşabilecek bir araç seti istiyordu. İsteddiği de oldu WebDriver ortaya çıktı. Selenium ile WebDriver’ın birleşmesi birkaç yılı buldu. Bununla birlikte WebDriver’e ek olarak, Selenium IDE, Grid ve Uzaktan Kumanda (RC) ortaya çıktı. Böylece bu dörtlü Selenium projesi için harika bir takıma dönüştü.

#### e) Selenium Tools

Selenium Entegre Geliştirme Ortamı (IDE): Firefox eklentisidir. Komut dosyalarını kaydetmemizi ve oynatmamızı sağlar. Selenium IDE’yi kullanarak komut dosyaları oluşturmak, daha gelişmiş ve sağlam test senaryoları yazmak istiyorsak Selenium RC veya Selenium WebDriver kullanmamız gerekir. Selenium IDE’nin özelliklerini sayacak olursak;

1. Normal bir kullanıcı gibi davranıp işlemleri ona göre yapıyor ve kaydediyor.
2. Fonksiyonel testler yazmak için kullanılabilir.
3. Firefox plug-in’i olarak çalışmaktadır.
4. Birçok dil desteği mevcut (Java, .NET, Python, Ruby, PHP,Perl)
5. Selenium’un open source olması sayesinde birçok platformda (Windows, Linux, IOS) herhangi bir sorun yaşamadan çalışmaktadır.
6. Birçok dil desteği ve platform desteği sayesinde diğer test araçlarına göre daha çok tercih edilmektedir. (UFT, QTP)

Selenium RC (Uzaktan Kumanda): Birden fazla testin sürekli olarak yapılmasını sağlar. Farklı programlama dillerinde web uygulaması testi yazmak için geliştirilmiştir. Günümüzde resmi olarak kullanımdan kaldırılmıştır.

Selenium WebDriver: Selenium WebDriver, komutları kabul eden ve bunları bir tarayıcıya göndererek tarayıcıyla iletişim kuran ve onu kontrol eden, test case oluşturmak ve yürütmek için tasarlanmış bir tarayıcı otomasyonu framework'üdür. Case'ler WebDriver metodlarındaki element locator'lar aracılığı ile oluşturulur ve yürütülür. Selenium WebDriver bir IDE değil, programlama kütüphanesidir. Selenium WebDriver, Java, C#, PHP, Perl ve JavaScript gibi çeşitli programlama dillerini destekler. WebDriver'ın geliştirilmesindeki en büyük amaç, dinamik web sitelerinin içeriklerinin belli bir kısmının değişimi sırasında tüm sayfanın tekrardan yüklenmeden, yapılan değişiklikleri görerek test caselerinizi devam ettirmemize yardımcı olmaktır.

Selenium Grid: Selenium Grid, Selenium RC ile birlikte kullanılan bir araçtır. Buradaki asıl amaç, farklı işletim sistemi, donanım, cihaz gibi kombinasyonlar üzerinde test sonuçlarını görmek, dağıtık ortamda paralel olarak test süreçlerini yürütmek ve test sonuçlarını hızlıca almaktır. Selenium Hub ve Node yapısını kullanarak testlerini yapmaktadır.

*Hub*: Selenium'da kullanılan Hub tek bir tane olmak koşuluyla çalışmaktadır. Sunucu gibi davranan bu yapı kendi üzerinde birçok işlemi barındırarak farklı istemcilerden gelen istekleri yanıtlayarak aynı kodu farklı platformlarda ve browserlarda test edebilirsiniz.

*Node*: Hub'a bağlı olan bir veya birden fazla istemciden oluşan yapıya node denir. Node'lar ile selenium-grid kullanarak birçok Node'tan tek bir Hub'a istekte bulunarak test yapabilirsiniz.

### **c) Selenium ne zaman ve nasıl kullanılır?**

Selenium'u zamanla tekrarlanan bir web işleminiz varsa, bu işlemi otomatikleştirmek için kullanabilirsiniz. Selenium IDE'yi kurmak için tek yapmamız gereken Chrome Web Mağazası üzerinden Selenium IDE pluginini Chrome tarayıcımıza eklemektir. Selenium test komut dosyaları Java, Python, C# ve daha pek çok farklı programlama dillerinde yazılabilir.

## 2) TEST MANAGEMENT USING JIRA (2. seçim)

### a) Test Management (Test Yönetimi) nedir?

Testlerin etkili ve verimli şekilde yapılması iyi şekilde organize edilmiş farklı gruplara, iyi bir planlamaya, görev ve sorumlulukların net, açık bir şekilde tanımlanmasına bağlıdır. Test Yönetimi, Yazılım test süreci ve bu süreç içerisinde yer alan kişi/kişilerin, ekibin yönetimini kapsamaktadır

### b) Test yönetimi ne için kullanılır?

Test yönetimi, *test yönetimi araçları* ile testin nasıl yapılacağına ilişkin bilgileri depolamak, test etkinliklerini planlamak ve kalite güvencesi etkinliklerinin durumunu raporlamak için kullanılır.

### c) JIRA nedir?

JIRA, görev ve bug izleme, yazılım proje yönetimi gibi süreçleri bir arada yönetebilen bir platformdur. JIRA, farklı ölçekteki işletmeler için de uygun olmakla birlikte takımlar, paydaşlar ve proje yöneticilerinin iş birliğini arttıran ve işletmeye değer katan bir uygulamadır. JIRA, mevcut projelerdeki hataları tespit etmek ve düzeltmek için kullanılır.

JIRA'nın en temel amaçları:

- Sorun ve Hata Kayıtları,
- Sorun ve Hata Takibi,
- Gelişmiş Proje Yönetimi,
- Derinlemesine Çevik (Agile) Raporlama,
- Çevik (Agile) Yol Haritası Çıkarma,
- Koşu (Sprint) Planlaması.

### d) JIRA tarihçesi

Jira, 2002 yılında kurulmuş, hizmet olarak yazılım SaaS hizmetleri sağlayan Avustralya merkezli teknoloji şirketi Atlassian tarafından geliştirilen bir proje yönetim aracıdır.

### e) Ne tür ekipler JIRA kullanıyor?

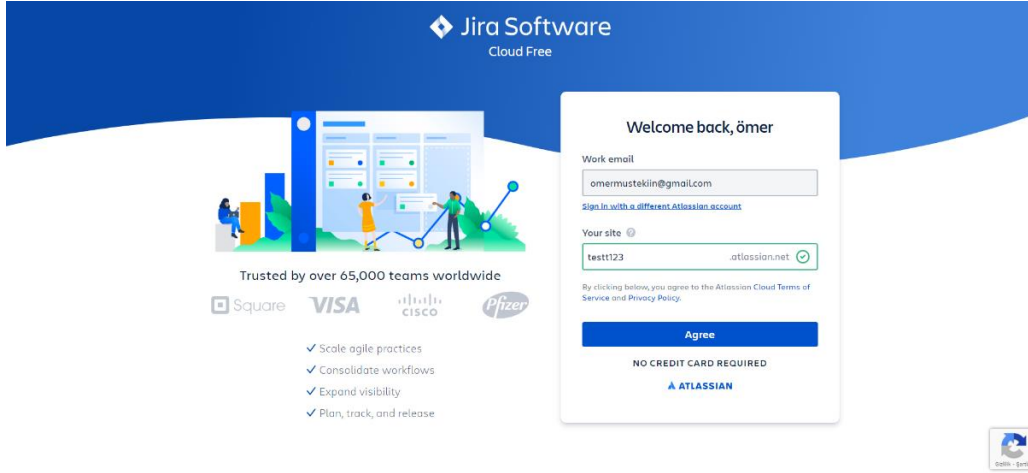
- Gereksinimler ve test senaryosu yönetim ekipleri
- Çevik ekipler
- Proje yönetim ekipleri
- Ürün yönetimi ekipleri
- Yazılım geliştirme ekipleri
- Görev yönetimi ekipleri
- Hata izleme ekipleri

JIRA, kuruluşların %25'i tarafından kullanılıyor ve bu da onu kapsamlı bir sistem arayanlar için bir çözüm haline getiriyor.

## f) JIRA nasıl kullanılır?

1. JIRA yazılımını açın ve JIRA Home simgesine gidin
2. Proje oluşturma seçeneğini seçin
3. Kitaplıktan bir şablon seçin
4. Pano ayarlarından ihtiyacınıza göre sütunları ayarlayın
5. Bir hata oluşturun
6. Ekip üyelerinizi davet edin ve çalışmaya başlayın

[www.atlassian.com](http://www.atlassian.com) sitesine giriş yaparak JIRA'yı kullanmaya başlayabilirsiniz.



### **3) MOBILE TESTING USING APPIUM (3. seçim)**

#### **a) Mobil Uygulama Testi nedir?**

Taşıyabilir mobil cihazlar için geliştirilen uygulama yazılımının işlevselliği, kullanılabilirliği ve tutarlılığı açısından test edildiği bir süreçtir. Mobil uygulama testi, otomatik veya manuel bir test türü olabilir. Mobil Uygulama Testi, mobil uygulamaların birden fazla mobil cihazda işlevsellik, kullanılabilirlik, görsel çekicilik ve tutarlılık açısından analiz edilmesini içerir. Uygulamaya erişmek için kullanılan cihazdan bağımsız olarak optimum bir kullanıcı deneyimi sağlanmasına yardımcı olur.

#### **b) Appium nedir?**

Appium, bir web sürücüsü kullanarak Android veya iOS'ta komut dosyalarını çalıştırmak ve yerel uygulamaları, mobil web uygulamalarını ve hibrit uygulamaları test etmek için açık kaynaklı bir otomasyon aracıdır. Appium birçok C#, Java, PHP, Ruby ve Python gibi dilleri destekler.

#### **c) Appium tarihçesi**

Appium, ilk olarak 2011 yılında Dan Cuellar tarafından "iOSAuto" adı altında C# programlama dili ile yazılarak geliştirilmiştir. Program, Ağustos 2012'de Apache 2 lisansı kullanılarak açık kaynaklı hale getirilmiştir. Ocak 2013'te Sauce Labs, Appium'un gelişimini finanse etmeyi kabul ederek ve kodunun Node.js kullanılarak yeniden yazılması için motive ederek geliştirilip desteklenmesini sağlamıştır.

Appium, en iyi açık kaynaklı masaüstü ve mobil yazılım dalında InfoWorld'ün 2014 Bossie ödülünü kazandı. Appium ayrıca Black Duck Software tarafından Yılın Açık Kaynak Çaylağı (Open Source Rookie of the Year) seçildi.

#### **d) Appium nasıl kullanılır?**

Appium, iki yoldan biriyle kurulabilir: NPM aracılığıyla veya Appium sunucusunu başlatmak için masaüstü tabanlı bir yol olan Appium Desktop'ı indirerek.

<https://appium.io/downloads.html> üzerinden Appium uygulamasını kurup kullanabiliriz.

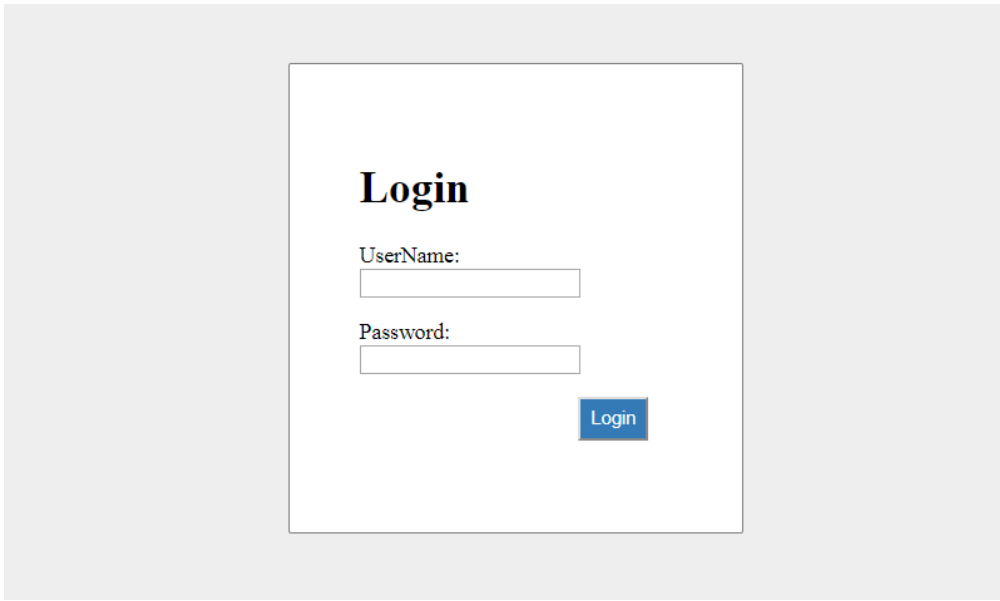
## ----- SORU 2 -----

### 1) FUNCTIONAL TESTING USING SELENIUM ÖRNEĞİ

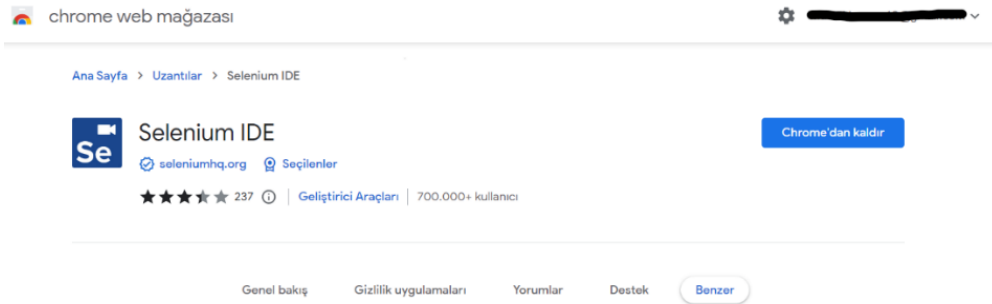
Bu kısımda basit bir oturum açma fonksiyonunun test senaryosunu Selenium IDE aracını kullanarak oluşturacağım. Bu senaryo sürekli tekrarlanan otomatik bir işlem olduğu için Selenium ile bu fonksiyonun testini yapacağım. Oturum açma test senaryom şu adımlardan oluşmaktadır:

1. Kullanıcı adını gir.
2. Parola gir.
3. Giriş yap butonuna bas.

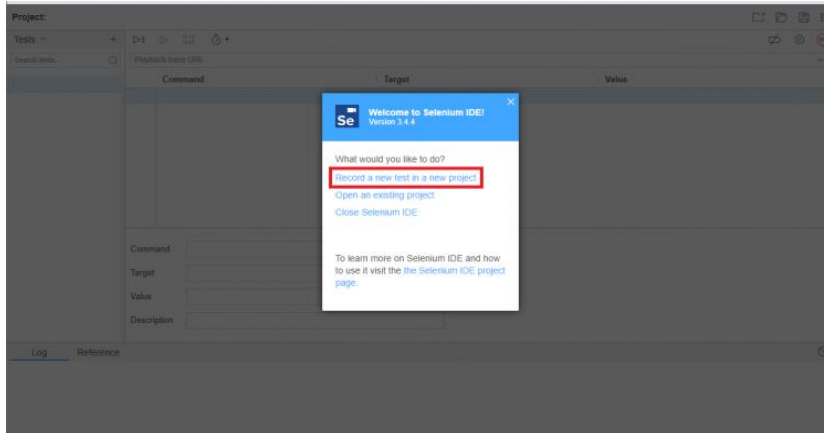
Test edilecek sitemin arayüzü:

A screenshot of a login form titled "Login". It features two input fields: "UserName:" and "Password:". Below the "Password:" field is a blue button labeled "Login". The form is centered on a light gray background.

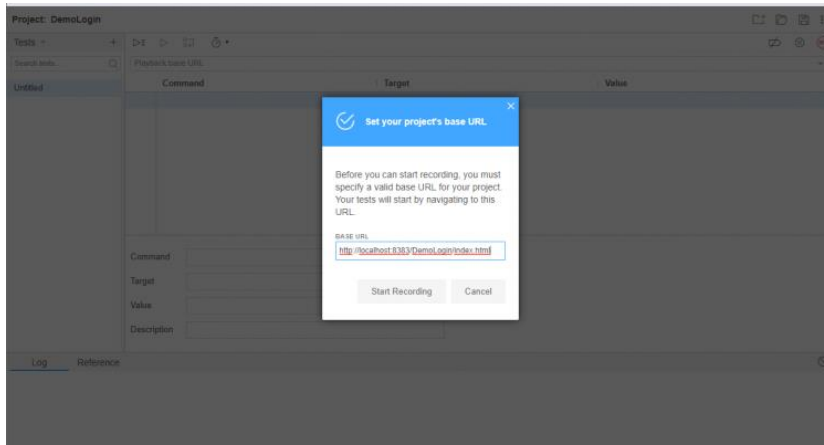
**1.Adım:** Selenium IDE aracını tarayıcıma eklenti olarak yükledim.



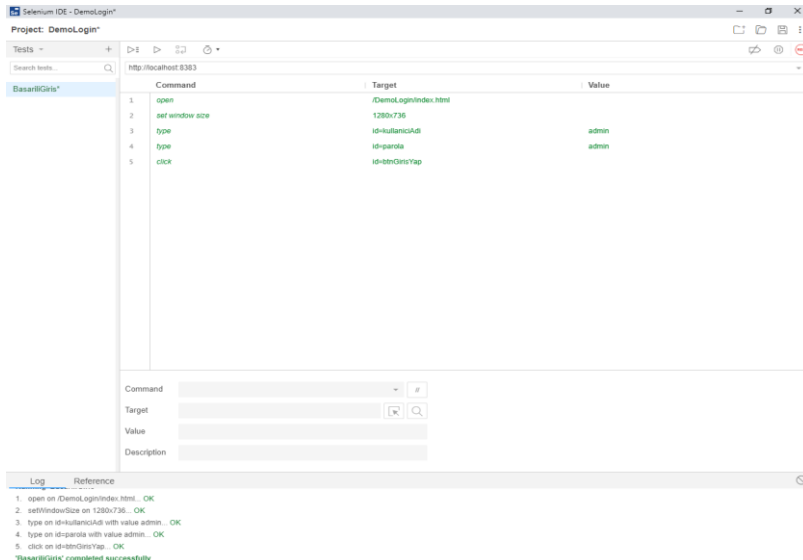
**2.Adım:** Selenium IDE eklentisini çalıştırdığımda aşağıdaki gibi bir ekranla karşılaştım. “Record a new test in a new project” seçeneğine basarak yeni bir proje oluşturdum



**3.Adım:** Sonraki ekranda projeme isim verdim ve test edilecek sayfanın linkini “Base Url” kısmına girdim.



**4.Adım:** Selenium IDE eklentisi otomatik olarak yeni bir tarayıcı penceresi açarak girmiş olduğum linke yönlendi. Kullanıcı adı ve parolamı girme ile kayıt ol butonuna tıklama işlemlerim sonucunda yapmış olduğum bu üç eylem eklenti tarafından otomatik algılanarak test adımları haline getirildi. Her run ettiğimde test yeniden çalışıyor.

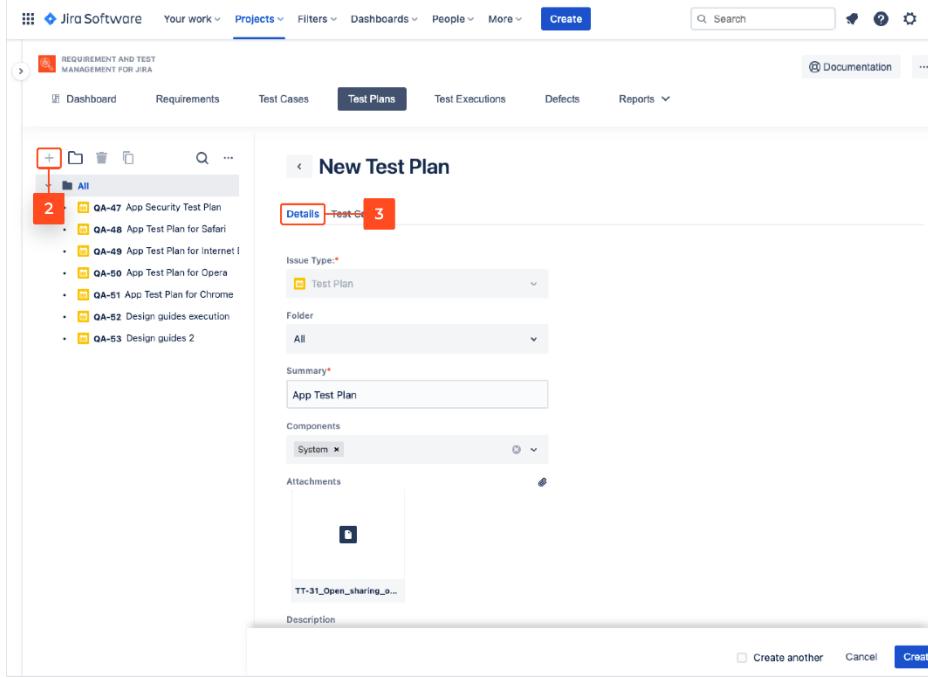


## 2) TEST MANAGEMENT USING JIRA ÖRNEĞİ

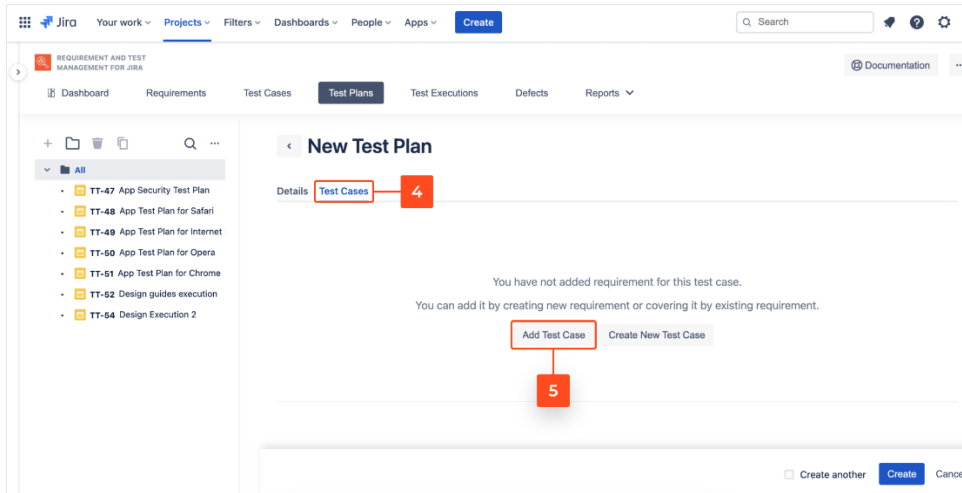
**1.Adım:** JIRA yazılımını kullanmak için [www.atlassian.com](http://www.atlassian.com) sitesine kayıt olduktan sonra giriş yaptım ve Project > Test Management > Test Plans kısmına geldim.

**2.Adım:** ‘+’ ya tıkladım.

**3.Adım:** Ayrıntılar bölümünde ekranda gözüken tüm alanları doldurdum ve yeni bir test planı oluşturdum.

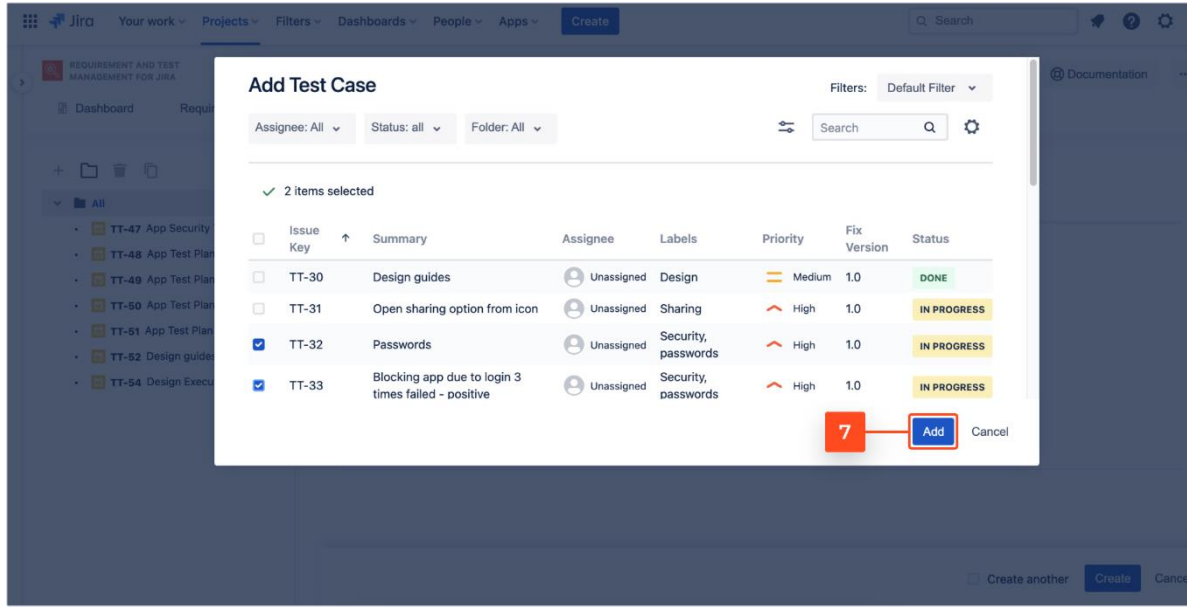


**4.Adım:** Test Cases -> Add Test Case kısmına gittim.

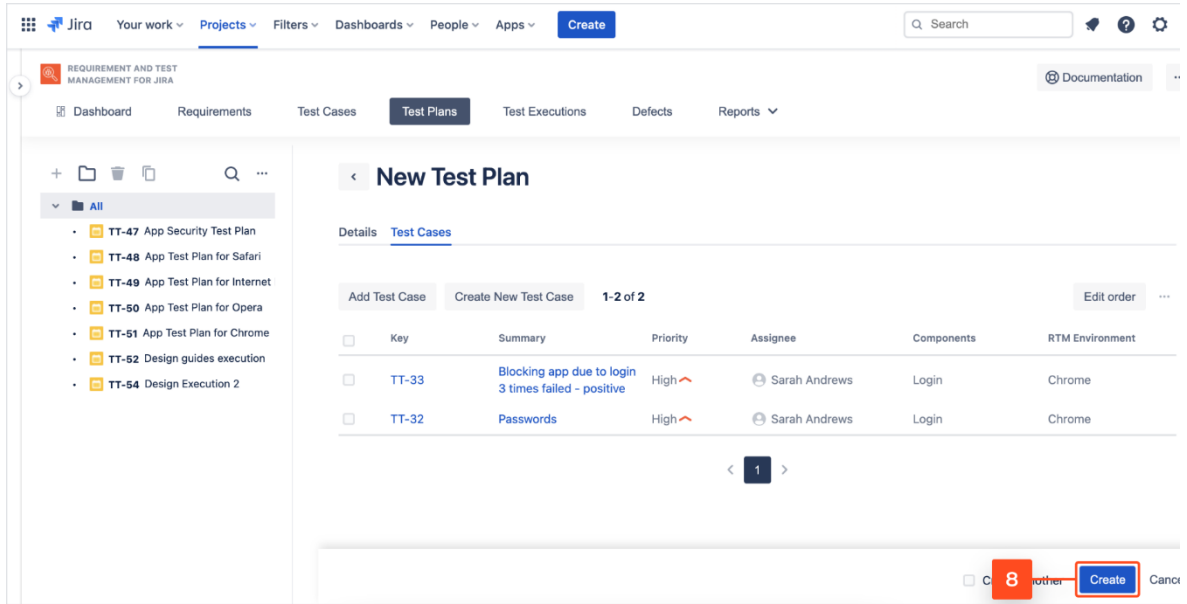




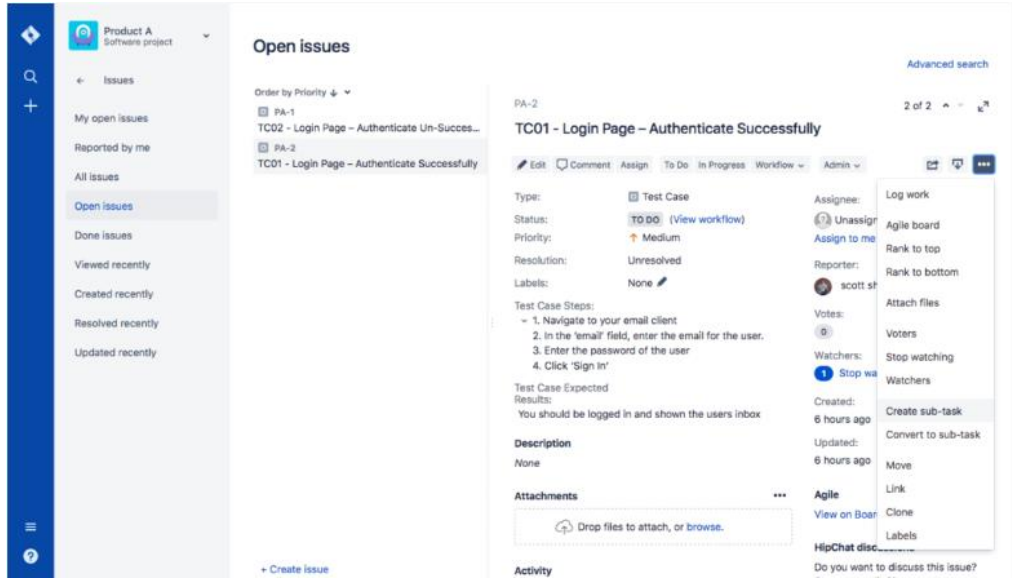
**6.Adım:** Test case'lerimi seçerek "Add" butonuna tıkladım.



**7.Adım:** "Create" butonuna tıkladım ve yeni bir test planı oluşturdum. Test planları test caselerini içeren bir yapı halinde kullanılıyor. Böylece her test case'im için bu adımları izleyerek bir test planı oluşturabiliyorum.



## 8.Adım: Oluşturduğum text caselere gittim



Open issues

Order by Priority

PA-1  
TC02 - Login Page - Authenticate Un-Successfully  
PA-2  
TC01 - Login Page - Authenticate Successfully

TC01 - Login Page - Authenticate Successfully

Type: Test Case  
Status: TO DO (View workflow)  
Priority: Medium  
Resolution: Unresolved  
Labels: None

Test Case Steps:  
1. Navigate to your email client  
2. In the 'email' field, enter the email for the user.  
3. Enter the password of the user  
4. Click 'Sign in'

Test Case Expected Results:  
You should be logged in and shown the users inbox

Description  
None

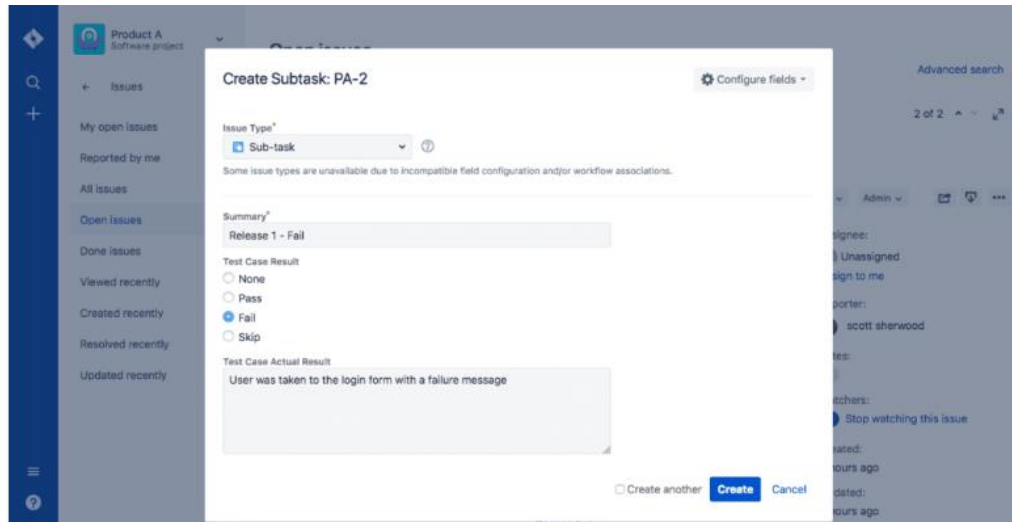
Attachments  
Drop files to attach, or browse.

Activity

Assignee: Unassigned  
Reporter: scott sherwood  
Voters: 0  
Watchers: 1  
Created: 6 hours ago  
Updated: 6 hours ago

Log work  
Agile board  
Rank to top  
Rank to bottom  
Attach files  
Voters  
Stop watching  
Watchers  
Create sub-task  
Convert to sub-task  
Move  
Link  
Clone  
Labels

## 9.Adım: “Create sub-task” diyerek sonuç sayfamı gittim ve test sonuçlarımı gözlemledim.



Create Subtask: PA-2

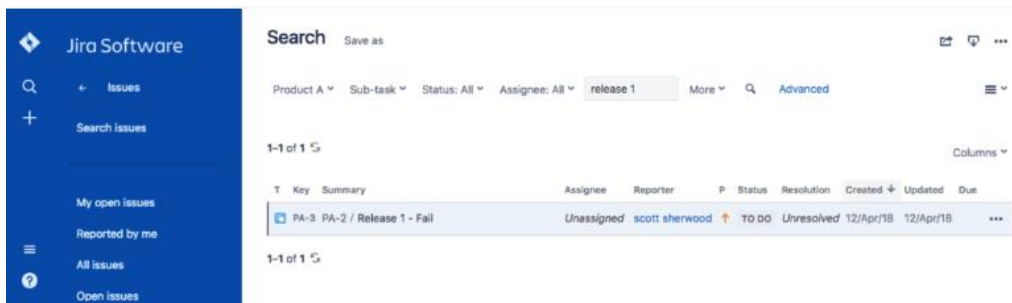
Issue Type: Sub-task

Summary: Release 1 - Fail

Test Case Result: Fail

Test Case Actual Result: User was taken to the login form with a failure message

Create another Create Cancel



Jira Software

Search

Product A Sub-task Status: All Assignee: All release 1

1-1 of 1

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due
	PA-3	PA-2 / Release 1 - Fail	Unassigned	scott sherwood	TO DO	Unresolved		12/Apr/18	12/Apr/18	

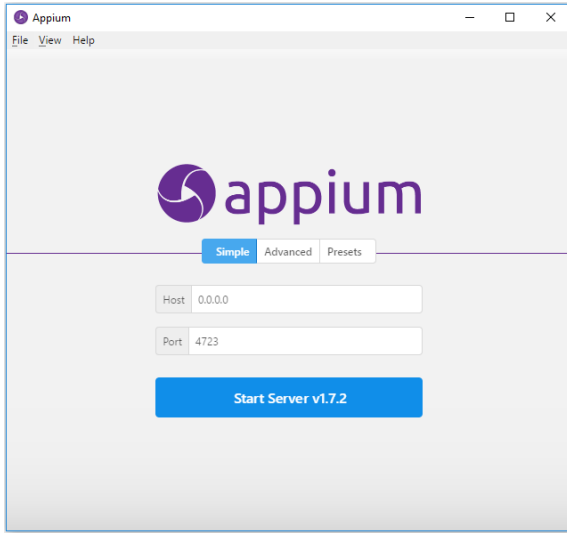
1-1 of 1

### 3) MOBILE TESTING USING APPIUM ÖRNEĞİ

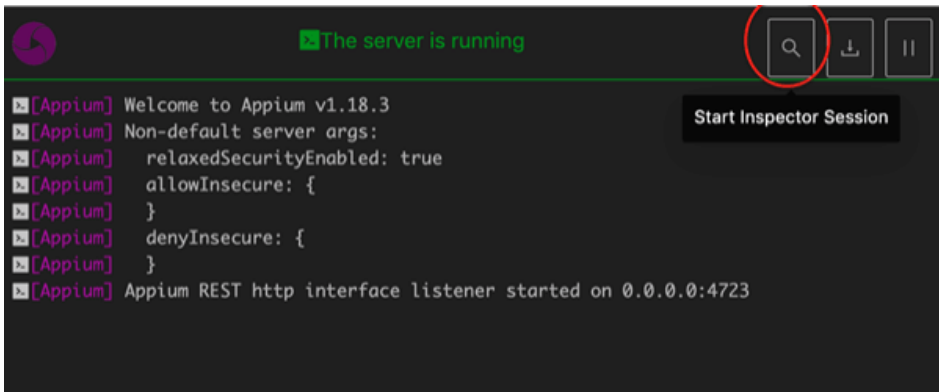
Bu kısımda Android Studio üzerinde bir emulatör çalıştırıp, bu emulatöre “hepsiburada” uygulamasını kurup uygulamanın bilgilerine erişim sağladıktan sonra bu uygulamanın Login senaryosunu test edeceğim. Test edeceğim senaryo şu dört aşamadan meydana gelecektir:

1. “Kapat” butonu tıklanır, animasyonun kapandığı kontrol edilir.
2. Account ikonu tıklanır, NavigationBar açıldığı kontrol edilir.
3. NavigationBar’ da “Giriş Yap veya Üye Ol” tıklanır, Login ekranı geldiği kontrol edilir.
4. Kullanıcı Adı(E-posta) ve Şifre Girilir, Güvenli Giriş butonuna tıklanır, kullanıcının giriş yaptığı kontrol edilir.

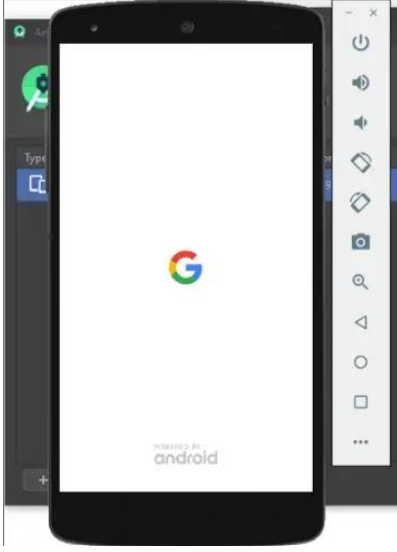
**1.Adım:** Uygulamayı masaüstüne kurduktan sonra başlattım ve Server Adresi default olarak “0.0.0.0” şeklinde karşıma çıktı. Bu aslında gerçek bir cihaz üzerinde çalıştırdığımızda kalması gereken adres. Ancak ben testimi bir emulatör üzerinden çalıştıracığım için bu Server Address alanını “127.0.0.1” olarak değiştirerek giriş yaptım.



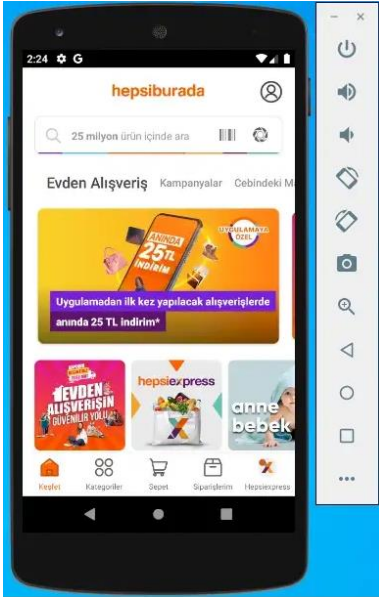
**2.Adım:** Testlerini otomatize edeceğim uygulamanın capabilities bilgilerini girmek için yuvarlak içinde gösterdiğim search ikonuna tıkladım.



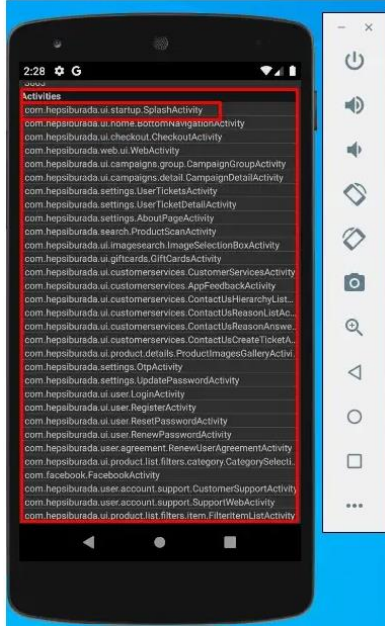
**3.Adım:** Android studio üzerinden gerekli kurulum ve ayarlamaları yaparak sanal bir cihaz çalıştırdım.



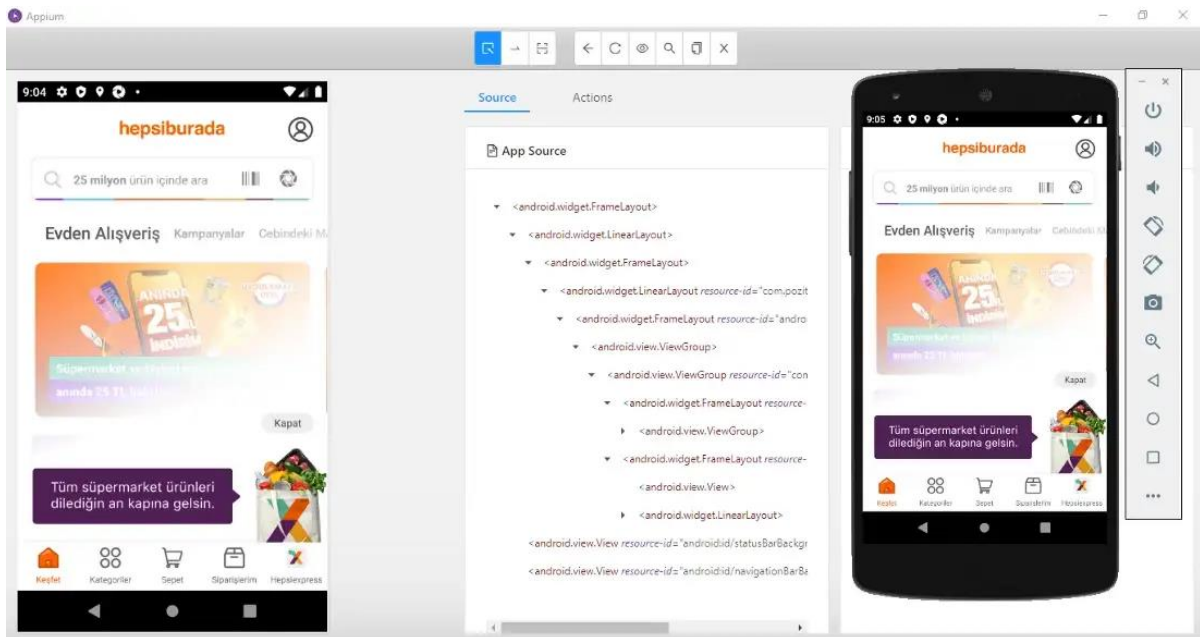
**4.adım:** Test edeceğim uygulama olan Hepsiburada uygulamasının apk kurulumunu sanal cihazımda <https://tr-apk.com/> linki üzerinden gerçekleştirdim.



**5.Adım:** adb shell dumpsys window windows | grep -E 'mCurrentFocus' komutunu terminale yazdım. Aşağıda çıkan komutlar aslında “Hepsiburada” uygulamasını açtıktan sonra çalıştırdığımızda emülatörde çalışan uygulama hakkında bilgiler verir. Ancak çoğu uygulamada SplashScreen’ler bulunduğu için bu activity’i yakalamak gerçekten çok zor. Bu yüzden apkInfo adlı uygulamayı indirerek emülatöre kurdum. Kurduktan sonra apkInfo’yu açtım ve hepsiburada uygulamasını bularak üzerine uzunca basılı tuttum. Gelen seçeneklerden “Detailed Information”ı seçtim ve gelen ekranda artık hepsiburada uygulamasına ait birçok Activity’i görebilir konuma geldim.



**6.Adım:** IntelliJ Idea’da maven projemi başlattım, Appium aracını kurdum ve capabilities bilgilerini girdim. Android cihazım ile appium arasında bir bağlantı kurdum ve bu mobil uygulamanın tüm öğelerinin bilgilerine ulaşılabilir duruma geldim.



**7.Adım:** Bu bilgiler ışığında IntelliJ Idea üzerinde ilk başta belirttiğim Login Senaryolarımın testini Appium aracı ile yazdım ve otomasyonun nasıl test koşacağını (run edeceğini) gördüm.

Aşağıdaki kod ilk başta belirttiğim 4 adet senaryonun testini gerçekleştiriyor:

```
public void loginAccount(){
    WebElement account =
    wait.until(ExpectedConditions.elementToBeClickable(By.
    id("account_icon")));
    account.click();
    WebElement userLogin =
    wait.until(ExpectedConditions.elementToBeClickable(By.
    id("llUserAccountLogin")));
    userLogin.click();
    WebElement loginEmail =
    wait.until(ExpectedConditions.elementToBeClickable(By.
    id("etLoginEmail")));
    loginEmail.sendKeys("***@gmail.com");
    WebElement loginPassword =
    wait.until(ExpectedConditions.elementToBeClickable(By.
    id("etLoginPassword")));
    loginPassword.sendKeys("****");
    WebElement loginButton =
    wait.until(ExpectedConditions.elementToBeClickable(By.
    id("btnLoginLogin")));
    loginButton.click();
}
```

**8.Adım:** Bu fonksiyonları @Test Tagında bulunan loginTest içerisinde çağırdım:

```
@Test
public void loginTest () throws InterruptedException {

    dialogsCloseAndCloseAnimation();
    loginAccount();
}
```

## Final kod:

```
import io.appium.java_client.MobileElement;
import io.appium.java_client.android.AndroidDriver;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

import java.net.MalformedURLException;
import java.net.URL;

public class BaseSetup {

    public AndroidDriver<MobileElement> driver;
    public WebDriverWait wait;

    @BeforeMethod
    public void setup () throws MalformedURLException {
        DesiredCapabilities caps = new DesiredCapabilities();
        caps.setCapability("deviceName", "Nexus 5");
        caps.setCapability("udid", "emulator-5554"); //DeviceId from
        "adb devices" command
        caps.setCapability("platformName", "Android");
        caps.setCapability("platformVersion", "9.0");
        caps.setCapability("appPackage", "com.pozitron.hepsiburada");

        caps.setCapability("appActivity", "com.hepsiburada.ui.startup.SplashAc
        tivity");
        caps.setCapability("noReset", "false");
        driver = new AndroidDriver<MobileElement>(new
        URL("http://127.0.0.1:4723/wd/hub"), caps);
        wait = new WebDriverWait(driver, 10);
    }

    @Test
    public void loginTest () throws InterruptedException {

        dialogsCloseAndCloseAnimation();
        loginAccount();
    }

    public void loginAccount(){
        WebElement account =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("account_ico
        n"))));
        account.click();
        WebElement userLogin =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("llUserAccou
        ntLogin"))));
        userLogin.click();
        WebElement loginEmail =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("etLoginEmai
        l"))));
        loginEmail.sendKeys("***@gmail.com");
        WebElement loginPassword =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("etLoginPass
        word"))));
        loginPassword.sendKeys("*****");
        WebElement loginButton =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("btnLoginLog
        in"))));
        loginButton.click();
    }

    public void dialogsCloseAndCloseAnimation(){
        WebElement check =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("cb_dialog_p
        erm_rationale_location_dont_show"))));
        check.click();
        WebElement noButton =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("button2")))
        ;
        noButton.click();
        WebElement okButton =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("button1")))
        ;
        okButton.click();
        //Click animation close buton
        WebElement element =
        wait.until(ExpectedConditions.elementToBeClickable(By.id("close_butto
        n"))));
        element.click();
    }

    @AfterMethod
    public void teardown(){
        driver.quit();
    }
}
```