Sayfa 78

SALATALIK & GHERKIN

1. Bana Salatalık hakkında daha fazla bilgi verin, Salatalığı kullanmaya nasıl karar verdiniz?

- Geçtiğimiz birkaç yılda, giderek daha fazla BT ekibi, geliştirme süreçlerinde Agile metodolojisini takip ederek
  pazarın hızlı değişimleri. Bu aynı zamanda test ekibi için test senaryolarını ve test komut dosyalarını yönetmede bir zorluktur.
  gereksinimler aylık olarak güncellendiğinde değişti. Başlangıçtan itibaren uygun bir test yöntemi bulmak,
  Agile yazılım projesinin başarısının anahtarları.
- Birçok Agile ekibi, aşağıdakileri kullanarak test sürecinde Davranış Odaklı Geliştirme (veya BDD) yaklaşımını başarıyla uygulamıştır.
  Salatalık aracı. Peki Salatalık nedir? Ve Agile projelerinde neden birlikte kullanılan iyi yaklaşımlardan biridir?
  BDD?
- Salatalık, davranış odaklı geliştirme tarzında yazılmış otomatik kabul testlerini yürütmek için bir araçtır. Biri
  harika ana özellikler, düz metin işlevsel açıklamayı (Gherkin adlı dilde yazılmış) şu şekilde yürütme yeteneğidir.
  otomatik testler. İşte bir örnek:

Özellik : Şifreyi güncelle
Senaryo : Yönetici kullanıcı, kullanıcı şifresini güncelleyebilir
Verilen Bir Yönetici hesabıyla İK sistemindeki duyuyorum
Ne zaman ben başka şifresini güncellemek "kullanıcı"
Ardından şifreyi başarıyla güncellemek için bir mesaj alıyorum
Ve kullanıcının şifresi yeni şifreyle güncellenir

- Bu harika özellik, BDD yaklaşımını aşağıdaki **avantajlarla** desteklemede birincil rol oynamıştır :
  - o Alan modeli etrafında yapılandırılmış ve tüm ekip tarafından kullanılan bir dil olan Ubiquitous dilinde BDD testleri yazmak
    geliştiriciler, test uzmanları, iş yöneticileri vb. dahil üyeler
  - o Bir yazılım ekibinin teknik ve teknik olmayan üyeleri arasında köprüler kurmak
  - o Geliştiricilerin koduyla doğrudan etkileşime izin verir, ancak iş paydaşlarının yapabileceği bir dilde yazılmıştır.
    anlama
  - o Son olarak, Cucumber, davranış odaklı bir şekilde yazılmış testleri çalıştıran bir Otomatik Kabul Test Aracıdır.
    geliştirme (BDD) tarzı.
- **Salatalık Aracı, bir projedeki teknik ve teknik olmayan üyeler arasındaki iletişimi geliştirmeye yardımcı olur.**

2. Bana Salatalığın en önemli şeylerini söyleyin, onu benzersiz kılan nedir?

- Özellikler dosyası, Adım Tanımları, Koşucu Sınıfları, Kanca Sınıfı, Etiketler

3. Raporlarınızı salatalıkta nasıl görebilirsiniz?

- Çerçevem, raporları içeren hedef klasörde salatalık raporları klasörü oluşturur.
- Jenkins üzerinde testleri çalıştırdığımızda, Jenkins her çalışmanın raporunu kaydeder.
- Jenkins işinin ana sayfası her zaman son çalıştırılan raporları gösterir.
- Önceki çalıştırmalar için tüm raporlar yapı numarası altında bulunabilir.
- Hedef klasöre gidin
- Sistem gezgini ile açın
- Hedefe git> salatalık raporu> dizin, yaptığınız testleri gösterir

4. Kornişon nedir?

- Özellik dosyaları tarafından kullanılan dil
- Özellik, Senaryo, Verilen, Sonra, Ne Zaman, Ve, Ama, Arka Plan, Senaryo Özeti

Sayfa 79

5. Cucumber BDD çerçevesinin bileşenleri nelerdir?

1. Özellik dosyaları

- o Belirli bir özelliği veya işlevi test eden senaryolardan oluşur
- o Özelliğin ana hikayesi, senaryolar ise hikayenin (uzun metrajlı filmin) test durumlarıdır.

2. Cukes Koşucusu

- o Testleri kesinlikle çalıştıran, adım tanımı için kodlar üreten bir sınıf
- o @smoketest
- o Cukesrunner → CUCKESRUNNER'DA ÖZELLİĞİMİN NEREDE OLDUĞUNU GÖSTEREN BİR ÖZELLİK KONUMUM VAR
  KONUM

3. Adım tanımı

- o Kornişon dili ile başlayan adımlardan oluşan bir sınıf
- o Adım tanımının cukes Runner veya alt paket (ebeveyn veya kardeş değil) ile aynı pakette olduğundan emin olun

- TEKNİK OLMAYAN PPL'NİN ANLAŞILMASI İÇİN

- BAĞIMSIZ BDD DİNLENMİŞTİR
- POM.XML DOSYASINDA MVN DEPOSU
- CUCUMBER.IO'DAN SALATALIK BDD
- TDD teknolojilerini birleştirin
- Davranış odaklı
- Müşteri davranışının akışını ifade edin → Unsurlara odaklanmayın

6. @CucumberOptions ne işe yarar?

- Salatalık testlerinin çalışmasını özelleştirmek için kullanılan etiket
- @CucumberOptions içine şunları ekleyebilirsiniz:
  - o dryRun
  - o Eklenti
    - **" Güzel "**
      - Konsola daha fazla bilgi ekler → Size etiket, senaryo, yöntem bilgisi verir.
      - "html: hedef / salatalık raporu" → Hedef / salatalık rapor klasöründe bulunan html raporu oluşturur

        "Json: target / cucumber.json
    - **Etiketler**
      - Etiketler özellik yolunda yer almalıdır
      - Birden çok etiket ekleyebilir ... etiketler = "@Dog, @Cat"
    - **Özellik** dosyalarının bulunduğu konumu gösterir
    - Adım tanımlama adımlarının **aranacağı** yeri **yapıştırın** . kanca sınıfı da tutkalın bir parçasıdır.

7. JUnit ile Salatalık nasıl çalıştırılır?

- Salatalık JUnit bağımlılığı ekleyin
- cukesRunner sınıfının üstüne @RunWith (Cucumber.class) ekleme

8. Salatalık TestNG ile nasıl çalıştırılır?

- Salatalık testiNG bağımlılığı ekleyin
- CukesRunner'ı AbstractTestNG Salatalık Testlerine genişletmek

9. Koşucu sınıfınızı etiket olmadan çalıştırırsak ne olur?

- Tüm özellik dosyaları yukarıdan aşağıya doğru çalışır, ancak yalnızca @CucumberOptions içinde bulunan özellik dosyaları

  "Özellikler ="

Sayfa 80

10. Salatalıktaki Kanca nedir?

- Salatalık kancası, kod iş akışını daha iyi yönetmemizi sağlar ve kod fazlalığını azaltmamıza yardımcı olur. Söyleyebiliriz

  senaryolarımızı veya testlerimizi gerçekleştirmemize izin veren görünmeyen bir adım olduğunu.
- Kullanan sınıf
  - o @Before → her salatalık senaryosundan önce çalışır
  - o @After → her senaryodan sonra çalışır (Senaryo başarılı veya başarısız olursa olsun her zaman çalışır)
- Sınıf, stepDefinition ile aynı pakette olmalıdır
- Hook sınıfına ekran görüntüleri ekledim
- dryRun = true ise Kanca Sınıfı çalışmaz
- Senaryoyu önce / sonra yöntemimde parametre olarak kullanıyorum

11. Salatalıkta nasıl ekran görüntüsü alıyorsunuz?

- Aftermethodumda bir kod kullanıyorum:
- TakeScreenShot arayüzünü kullanıyorum
- Ekran görüntüsünü bayt veya dosya olarak saklayabilirsiniz
  - o @ Sonra

```
public void tearDown (Senaryo senaryosu) {
    eğer (scenario.isFailed ()) {
    // ekran görüntüsü almak
son bayt [] ekran görüntüsü = ((AlırScreenshot)
                                    Driver.getDriver ()). GetScreenshotAs (OutputType.BYTES);
// ekran görüntüsünü rapora eklemek
scenario.embed (ekran görüntüsü, "image / png");}
```

12. DDT ile Salatalık nasıl çalıştırılır?

- Salatalık sofraları kullanıyorum:

  | Ana Sayfa | E-postalar | Belgeler | Projeler |

- Yöntemi (DataTable arg1) ile alırsınız
- DataTable parametresinde bunu şu şekilde değiştirebilirsiniz:

Liste < YourType > , Liste <Liste < E >> , Liste <Harita < K , V >> , ve Harita < K , V >

- Liste sırasına göre yazdırır
- Harita için sipariş yok

13. Arka Plan nedir?

- Salatalığın kendine ait bir yöntemi vardır
- Kancalı olan java içindir
- Özellik dosyası içinde bir senaryodan ÖNCE çalışan bir adım
- Tüm senaryolardan önce yalnızca üstüne koyabilir
- Arka planlara boru hatları yerleştirilemez (Yalnızca senaryo taslağında)

14. Senaryo Taslağı nedir? Senaryo mu?

- Salatalıkta senaryo bir kez çalışır.
- Veriye dayalı testler için kullanılır
- Aynı salatalık adımlarını uygulayın, ancak senaryodan sonra verileri anahtar kelime örneklerini kullanarak tablo olarak sağlarız

Sayfa 81

15. Geçebileceğim değişken türlerini nasıl sınırlayabilirim?

- Kornişon parantezinde (İşbirliği | Satış | Markalama, vb.)
- Ör: @When ("^ Fareyle (İşbirliği | Satış | Pazarlama | Etkinlikler | Tümü) menüsünün üzerine geliyorum $")

```
public void i_hover_over_the_Collaboration_menu (Dize menüsü) {
anahtarı (menü) {
durum "Satış":
    BrowserUtils.hover (dashboard.sales); kırmak;
durum "Pazarlama":
    BrowserUtils.hover (dashboard.marketing); kırmak;
durum "İşbirliği":
    BrowserUtils.hover (dashboard.collaboration); kırmak;
durum "Aktiviteler":
    BrowserUtils.hover (dashboard.activities); kırmak;
durum "Tümü":
    BrowserUtils.hover (dashboard.all); mola;};
```

16. İki parametresi (sınırlayıcı parametre, tablo parametresi) olan bir senaryonuz varsa ne olur?

- Misal :
  - o Senaryo: İşbirliğini Doğrula menü seçeneklerini
  - o suiteCRM'de oturum açtığım için
  - o İşbirliği menüsünün üzerine geldiğimde
  - o Daha sonra İşbirliği için aşağıdaki menü seçenekleri görünmelidir:
    | Ana Sayfa | E-postalar | Belgeler | Projeler |
  - o Bu senaryoda bir masam var, işbirliğini sadece işbirliği ve diğer menüler kategorileriyle sınırlamak istiyorum.
- Çözüm:
  - o @ Sonra ("^ aşağıdaki menü seçenekleri görünür olmalıdır
    (İşbirliği | Satış | Pazarlama | Aktiviteler | Tümü): $ ")
  - o public void following_menu_options_should_be_visisble_for_Collaboration (String menüsü, List <String> seçenekleri) {
  - o Dize menüsü 5 menü seçeneğini temsil eder (İşbirliği | Satış | Pazarlama | Aktiviteler | Tümü)
    Liste <string> seçenekler tables; temsil | Ana Sayfa | E-postalar | Belgeler | Projeler |

17. DDT için salatalık senaryosunu nasıl kullanırım?

- Mevcut projemde Örneklerle Senaryo Taslağı kullanıyorum
- Senaryo özellik dosyamda, veriye dayalı olarak bir değişkeni kullandığımda, "<variable>" kullanıyorum
- Sonra Örneklerde:

| **değişken** | sütun adı
| veri1. | satır1
| veriler 2 | 2. sıra
| veri3 | satır3

20. Veriye dayalı

- Test verileri koddan ayrılır ve harici kaynaklarda saklanır: Salatalık Örnekleri tablosu, Excel dosyaları, CSV dosyalar, Veritabanı.
- Veri miktarı o kadar büyük değilse, Örnekler tablosu ile Hıyar Senaryosu taslağını kullanıyorum.
- Diğer zamanlarda test verilerini Excel dosyalarında tutuyorum ve verileri okumak ve yazmak için Apache POI kitaplığını kullanıyorum
- Veriler bir veritabanından geliyorsa veya veritabanı doğrulaması yapmam gerekiyorsa, java'daki JDBC kitaplığı ile birlikte SQL sorguları kullanıyorum.

Sayfa 82

18. Haritalar salatalıkta nasıl kullanılır?

- Senaryo dışı bir Taslak Kullanma
- Senaryo: Bir harita kullanarak kişi oluşturun
  - o suiteCRM'de oturum açtığım için
  - o Yeni bir kişi oluşturduğumda:
    - | ilk_adı | John |
    - | last_name | Smith |
    - | cell_phone | 801 888 8889 |
  - o O zaman "John Smith" için iletişim bilgilerini görmeliyim
  - o Sol taraf anahtardır ve sağ yalnızca değer 2 sütunudur
- Senaryo Taslağı Kullanma
  - o Senaryo Taslağı: Bir harita kullanarak kişi oluşturun
  - o suiteCRM'de oturum açtığım için
  - o Yeni bir kişi oluşturduğumda:
    - | ilk_adı | <ilk_adı> |
    - | last_name | <iname> |
    - | cell_phone | <cell_phone> |
    - | office_phone | <office_phone> |
  - o Sonra "<first_name> <lname>" için iletişim bilgilerini görmeliyim
  - o Örnekler: | **ilk_adı | lname | cell_phone | office_phone |**
    - | Michael | Jackson | 1234567890 | 2345678891 |
    - | Bonnie | Garcia | 4569871234 | 4567890987 |
- Def adımda yazıyorum;

```
@When ("^ Yeni bir kişi oluşturuyorum: $")
 public void i_create_a_new_contact (Map <String, String> contact) {
 // kişi oluşturma iletişim kutusunu aç
```

21. POJO salatalıkta nasıl kullanılır?

- **contactBean** sınıfı oluşturun
  - o Tüm değişkenleri ekleyin
  - o Alıcı / ayarlayıcıları ekleyin
- Fasulye unsur dosyası oluşturun
- contactBean sınıfındaki değişkenleri içeren ilk satırı içeren bir tablo oluşturun
  - o Tablonun altına değerler ekleyin
  - o Parametre ile uygulama yöntemi (List <ContactBean> kişileri)
- Senaryo: Kişi oluştur
  - o suiteCRM'de oturum açtığım için
  - o Yeni bir kişi kaydettiğimde:
    - | **firstName | lastName | officePhone | cep telefonu | e-posta** |
    - | Steve | Kapılar | 3456758888 | 1234329999 | SteveGates123@gmail.com |
  - o O zaman "Steve Gates" için iletişim bilgilerini görmeliyim

22. TestNG kullanarak bir grup test senaryosu nasıl çalıştırılır?

```
@Test (gruplar = {"smokeTest", "FunctionalTest"})
public void loginTest () {
                System.out.println ("Oturum başarıyla açıldı");
}
```


Sayfa 83

23. Veriye Dayalı Test

- ***NE ZAMAN*** *:* Bir uygulamadaki bir işlev veya modül birden fazla veri kümesiyle (Parametrelendirme) test gerektirdiğinde, Birden çok girdi daha sonra veriye dayalı test ve otomasyon gerçekleştirmemiz gerekir.
- Bu senaryolar, otomatikleştirilmesi gereken şeylerden biridir.
- ***NASIL*** *:* Test verileri koddan ayrılır ve harici kaynaklarda saklanır: Salatalık Örnekleri tablosu, Excel dosyaları, CSV dosyalar, Veritabanı.
- ***FAYDALARI*** *:* Daha organize, Veri merkezileştirilmiş, Test verileri üzerinde işbirliği - BA, MT'ler vb.

24. TestNG kullanarak veriye dayalı çerçeveyi nasıl oluşturabiliriz?

- @DataProvider ek açıklamasını kullanarak Veriye Dayalı Çerçeve oluşturabiliriz

```
@DataProvider (name = ”getData”) Genel Nesne [] [] getData () {Nesne [] [] veri = yeni Nesne [2] [2];
Veri [0] [0] = "firstUid"; Veri [0] [1] = "FirstPWD";
Veri [1] [0] = “SecondUid”;
Veri [1] [1] = "SecondPWD"; Verileri döndür;}
```

25. TestNG'de Grup Grubu nasıl oluşturulur?

- Bu gruplara meta gruplar denir.
- Örnek: SmokeTest ve FunctionalTest'i içeren bir grup tanımlamak isteyebilirsiniz.
  Test.xml dosyamızı değiştirelim:

```
<gruplar>
    <tanımla adı = ”tümü”>
                    <include name = ”Duman Testi” />
                    <include name = "fonksiyonelTest" />
    </define>
    <run>
                    <include name = "all" />
    </run>
</groups>
```

26. TestNG kullanarak test senaryoları paralel olarak nasıl çalıştırılır?

- TestNG'de paralel test yürütmeyi gerçekleştirmek için testng.xml'deki "parallel" özelliğini kullanabiliriz
- Paket etiketinin paralel niteliği dört değeri kabul edebilir:
  - Sınıflar → Bir java sınıfındaki tüm test senaryoları paralel çalışacaktır
  - Yöntemler → @Test ek açıklamasına sahip tüm yöntemler paralel olarak yürütülecektir
  - Örnekler → Aynı durumdaki test senaryoları paralel olarak yürütülecek ancak iki farklı örneğin iki yöntemi farklı iş parçacığında çalıştırın. <suite name = ”softwaretestingmaterial” parallel = ”yöntemleri”>

27. TestNG'de bir test senaryosu nasıl göz ardı edilir?

- Test durumunu yok saymak için, enable = false parametresini
- @Test açıklama @Test (etkin = yanlış)

28. Belirli bir test yöntemi, bir test senaryosu yürütmesinin dışında nasıl tutulur?

- Test.xml dosyasına dışlama etiketini ekleyerek

```
<sınıflar>
    <class name = "TestCaseName">
        <methods>
            <exclude name = "TestMethodNameToExclude" />
        </methods>
    </class>
</classes>
```


Sayfa 84

29. Belirli bir test grubu bir test senaryosunun yürütülmesinden nasıl hariç tutulur?

- Test.xml dosyasına dışlama etiketini ekleyerek

```
<gruplar>
    <run>
        <exclude name = "TestGroupNameToExclude" />
    </run>
</groups>
```

30. TestNG sonuçları için rapor oluşturmanın farklı yolu nedir?

- TestNG, bir rapor oluşturmak için iki yol sunar
  - Dinleyiciler, **org.testng** arayüzünü **uygular** . **testListener** ve bir test başladığında gerçek zamanlı olarak bilgilendirilir, geçer, başarısız olur vb.
  - **Raporlayıcılar org.testng.reporter** arayüzünü **uygular** ve tüm süitler tarafından çalıştırıldığında bilgilendirilir. TestNG.
- IReporter örneği, tüm test çalıştırmasını açıklayan nesnelerin bir listesini alır

31. @Listener ek açıklamasının TestNG'de kullanımı nedir?

- raporları ve günlük kaydını yapılandırın.
- yaygın olarak kullanılan dinleyiciler: ITestListener arabirimi.
- OnTestStart, onTestSucess gibi yöntemleri vardır. onTestFailure, TestSkipped'te ...
- bu arayüzü kendi dinleyici sınıfımızı oluşturarak uygulamalıyız,
- Daha sonra, sınıfa dinleyici ek açıklamasını (@Listeners) eklemeliyiz

32. Normal İfade, Normal İfade veya Normal İfade Nedir?

- Normal ifade, bir arama modelini tanımlayan özel bir metin dizesidir.
- Normal ifadeleri steroidlerdeki joker karakterler olarak düşünebilirsiniz.

- Bir dosya yöneticisindeki tüm metin dosyalarını bulmak için muhtemelen * .txt gibi joker karakter gösterimlerini biliyorsunuzdur.
- Normal ifade eşdeğeri. * \. Txt'dir.

33. "Duman" anahtar kelimesini içeren @Test yöntemlerini aramak için test.xml dosyasına düzenli ifade nasıl yazılır?

- "Duman" anahtar kelimesini içeren @Test yöntemini bulmak için normal ifade aşağıda belirtilmiştir

```
<methods>
    <include name = ". * duman. *" />
</methods>
```

34. Test takımlarında ve test senaryolarında belirttiğimiz zaman birimi nedir?

- Zaman birimini test paketlerinde belirtiriz ve test senaryoları milisaniye cinsindendir.

35. @Test'in kullanımı nedir (invocationCount = someInteger)?

```
@Test (invocationCount = 10)
Herkese açık geçersiz örnek olay () {}
```

- // çağrı sayısı özelliği, TestNG'nin bir test yöntemini kaç kez çalıştırması gerektiğini söyler

36. @Test'in kullanımı nedir (threadPoolSize = someInteger)?

- ThreadPoolSize özniteliği, bir iş parçacığı havuzundan test yöntemini birden çok iş parçacığı aracılığıyla çalıştırmasını söyler
- Not: Çağrı sayısı BELİRTİLMEMİŞSE bu öznitelik yok sayılır

Sayfa 85

37. Test zaman aşımı testte ne anlama geliyor?

- Bir test senaryosunun alması gereken maksimum milisaniye sayısı

```
@ Test1 (threadPoolSize = 3, invocationCount = 10, timeOut = 10000)
genel boşluk testi () {}
```

- : // bu örnekte: test1 işlevi üç farklı evreden on kez çağrılacaktır, Ek olarak,
  on saniyelik zaman aşımı, iş parçacıklarından hiçbirinin bu iş parçacığını sonsuza kadar engellemeyeceğini garanti eder.

38. @Factory ve @DataProvider ek açıklaması nedir?

- @Factory → bir test sınıfında bulunan tüm test yöntemlerini sınıfın ayrı bir örneğini kullanarak çalıştırır.
  farklı veri kümesi
- @DataProvider → dataProvider'ı kullanan bir test yöntemi, belirli yöntemleri birden çok sayıda yürütülecektir.
  dataProvider tarafından sağlanan verilere göre zamanlar.

39. ek açıklamalar - öncelik

- Başladığınız sayının önemi yok Örn: @Test (öncelik = 0)
- DependsOnMethods = "test yöntemi adı" Birden fazla test adı ekleyebilirsiniz
- İlki başarısız olursa, 2. test hiç çalışmaz.
- İlk yöntem başarısız olursa, raporunuz 2. testin atlanacağını gösterecektir.

40. testNG'de paralel yürütme

- Xml dosyasında yazılır.
  ○ paralel = "testler" thread-count = "4"
- İş parçacığı sayısı, aynı anda kaç tarayıcı açmak istediğinizdir
- Xml dosyasında her şeyi çalıştırmak için. * Ekleyebilirsiniz
  ○ Ör: <paket adı = ". *"> </package>
- TestNG'nin kendi raporları vardır - xml'yi çalıştırdığınızda, raporu size test-output klasöründe verir.
- Test raporunu html olarak içerir

41. Çerçeve Araçları: Hıyar BDD çerçevesi

- Junit, Salatalık Java, Maven
- Selenium, Log4j ekran görüntüleriyle HTML raporlama,
- JDBC, Huzurlu, Apache POI, Git, Jenkins

42. Çerçeve Araçları: TestNG + Selenium

- Java, Maven, TestNG,
- Selenium, Log4J ekran görüntüleri ile Raporları Genişletin,
- JDBC, Huzurlu, Apache POI, Git, Jenkins

43. Çerçeveniz nasıl raporlar üretiyor?

- Hıyar BDD çerçevemiz HTML raporları oluşturur.
- Rapor, özellik dosyaları, etiketler, adımlar için başarılı / başarısız kapsamını gösterir
- Rapor, her test için tüm adımları içerir Rapor, hatalar için ekran görüntülerine sahiptir

44. Seçici olarak salatalık testleri nasıl yapılır?

- etiketler anahtar kelime the cukesrunner
- özellik anahtar kelimesi the cukesrunner
- etiketler ve özellikler, komut satırı kullanılarak da geçirilebilir
- mvn test -Dcucumber.options = "- etiket @smoke"

Sayfa 86

45. Günlük kaydı için ne kullanıyorsunuz?

- Günlük kaydı için Log4J kullanıyorum. Test yürütmede her zaman önemli adımları kaydederim. Bu, bir başarısızlık.
- Log4J, HTML raporlarının yerini almaz.

```
<bağımlılık>
    <groupId> org.apache.logging.log4j </groupId>
    <artifactId> log4j- core </artifactId>
    <version> 2.11.0 </version>
</dependency>
```

46. ÖZELLİK DOSYASI NASIL ÇALIŞIR?

- **Özellik** → neyin test edildiğinin açıklaması @tags. Örnek özellik dosyası;
  - Özellik: oturum açma işlevi → Arka plan:
  - Giriş sayfasında olduğum için → Senaryo: 1, Senaryo: 2
  - Arka plan her iki senaryodan önce çalışır
- **Senaryo** → test edilen senaryonun açıklaması
  - Giriş sayfasında olduğum için
  - Ve kullanıcı adı ve şifre giriyorum
  - Gönder düğmesine tıkladığımda
  - O zaman profil resmini görebilirim
  - Ancak gönder düğmesi görüntülenmemelidir
- **Verilen** → bir ön koşul
- **Ne zaman** → beklenen sonucu tetikleyen koşul O zaman -> beklenen koşul

47. Test tabanı Sınıfı nedir? ve Çerçevenize nasıl uyguluyorsunuz?

- Test Base sınıfı, testlerimde en çok yöntem kullandığım sınıftır.
- Benim test sınıfları uzatmak testi Ana sınıfı ve dolayısıyla bu yöntemlere erişim hakkına sahiptir. Bu bize yardımcı oluyor yapmak benim yeniden kullanılabilir kod
- Test yöntemlerinden önce / sonra bekleme / senkronizasyon yardımcı programı yöntemleri.
  - SwitchToWindow (başlık)
  - WebDriver sürücüsü;

48. Başarısız olan testler TestNG'de nasıl yeniden çalıştırılır?

- Benim TestNG çerçevemde, başarısız testler hedef klasördeki testng_failed_.xml dosyasında rapor edilir .
- Bu dosyayı pom dosyasına ekleyebiliriz, böylece maven her seferinde başarısız testleri çalıştırmayı deneyecektir.
- If, yalnızca testte başarısızlıklar olduğunda çalışır .

49. Başarısız testler Hıyar'da nasıl yeniden yapılır?

- CukesRunner'da yeniden çalıştır seçeneğini kullanıyoruz.
- Tekrar çalıştırmayı cukes koşucusuna ekleyin.
- Bu seçenek, başarısız testlerin listesini içeren bir dosya oluşturacaktır.
- Başarısız testlerin bir listesiyle dosyaya işaret eden ikinci bir koşucu sınıfı oluşturun
- İkinci koşucuyu pom dosyasına ekleyin

50. Jenkins'te başarısız testler nasıl yeniden çalıştırılır?

- Jenkins'te başarısız olan Birim durumlarını yeniden çalıştıran eklentiler vardır.
- Böylece şu seçeneği kullanarak Jenkins üzerinde Maven derleme yürütmenizi yapılandırabilirsiniz:
- Dsurefire.rerunFailingTestsCount = 2

Sayfa 87

51. SALATALIK TESTLERİNİ PARALEL OLARAK KOŞUYOR MU?

Cucumber + JUnit çerçevesinin paralel olarak nasıl çalıştırılacağına dair birkaç seçenek vardır.

1. bir eklenti bulunmaktadır **salatalık-JVM-paralel-eklentisi**

https://github.com/temyers/cucumber-jvm-parallel-plugin

• Bu eklenti otomatik olarak birden çok cukes runner dosyası oluşturur.

• Konfigürasyona bağlı olarak, bu eklenti özellik dosyası başına bir cukes runner oluşturur.

• Her koşucu bir özellik dosyasını gösterecektir. ve bu cukes koşucuları paralel koşacak.

• Normalde salatalık özellik dosyalarını birbiri ardına çalıştırır. Bu eklentiyi kullandığımızda, onları aynı anda çalıştırıyor.

aynı anda kaç testin çalıştığını belirleyebiliriz

2. Salatalık 4.x paralel seçeneği

Salatalık 4.0'dan başlayarak, salatalık doğal olarak paralelleşmeyi destekler.

https://cucumber.io/blog/2018/09/24/announcing-cucumber-jvm-4-0-0

Resmi belgelere göre, testleri paralel olarak çalıştırmak için, maven surefire eklentisine paralel seçenek eklemeliyiz.

pom dosyası.

```
<build>
    <plugins>
        <plugin>
            <artifactId> maven-surefire-eklentisi </artifactId>
            <yapılandırma>
                <parallel> her ikisi </parallel>
                <threadCount> 4 </threadCount>
            </configuration>
        </plugin>
    </plugins>
</build>
```

Ancak benim özel projemde, bazılarına rağmen testlerin yürütülmeye devam ettiğinden emin olmak için maven arıza korumalı eklentisi ekledik. başarısız. Bu eklenti, testlerin çalışmaya devam etmesini sağlar

```
<plugins>
    <plugin>
        <groupId> org.apache.maven.plugins </groupId>
        <artifactId> maven-failsafe-eklentisi </artifactId>
        <version> 2.18 </version>
        <yapılandırma>
            <testFailureIgnore> doğru </testFailureIgnore>
            <skipTests> false </skipTests>
            <içerir>
                <include> ** / runners / * TestRunner.java </include>
            </includes>
        </configuration>
    </plugin>
<plugins>
```


3. İkinci eklenti **maven-surefire-eklentisidir**

bu eklenti testleri paralel olarak yürütür. bu eklenti yapılandırmasında hangi çalıştırıcı dosyalarını çalıştırmak istediğimizi belirtiyoruz. yapabiliriz eşzamanlı testlerin nasıl yapılacağını da belirtmek isteriz.

<include> ** / runners / * TestRunner * .java </include>. → eklenti bu dosyaları çalıştıracak
<threadCount> 10 </threadCount> → bu, aynı anda kaç tarayıcıya sahip olmak istediğimizi gösterir.
<parallel> sınıflar </parallel> → bu satır cukes runner sınıflarının paralel olarak çalışması gerektiğini söyler

Kaç tane test çalıştırmak istediğimize ve testleri nasıl bozmak istediğimize göre cukes runner dosyaları oluşturduk.
Her cukes koşucusu, belirli kurulum senaryolarına / özellik dosyalarına işaret edecektir.

Nasıl koşulur?

o testleri yalnızca bir maven komutu olarak çalıştırarak çerçevemizde paralel olarak yürütebiliriz

o **mvn doğrula** → bu, testleri çalıştıracak ve raporlar oluşturacaktır

o **mvn temiz doğrula** → önce hedef klasörü silecek, sonra testleri çalıştıracak ve ardından raporlar oluşturacaktır.

Paralelleştirmenin faydaları:

o yürütme süresini kısaltır. UI testleri, özellikle regresyon testinde genellikle uzun zaman alır.

Paralelleştirmenin zorlukları?

o uygulaması zor -> yapması kolay değil.

o load -> aynı makinede çok fazla örnek açarsak, makineyi aşırı yükleyebilir. çalışan testlerle sonuçlanacak yavaşlar ve başarısızlık oranını artırır.

o bu, GRID kullanan farklı makinelerde testler yürüterek ele alınabilir.

o projemde bazı test durumları paralel olarak çalışmadı.

Sayfa 89

API

1. API nedir?

- Bağlantı anlamına gelir. Demek istediğim, API istekleri alan ve bir sisteme ne yapmak istediğinizi söyleyen ve sonra yanıtı size geri döndürür.
- API, **Uygulama Programlama Arayüzünün** (yazılım aracısı olan) kısaltmasıdır. birbirleriyle konuşmak için uygulamalar.

2. API ve Web Hizmetleri mi?

- API = *tarayıcı* : Selenium WebDriver, *veritabanı* : JDBC, *MsOffice* : Apache POI
- Web hizmetleri = bir API iletişim için İnternet kullanıyorsa, bu bir web hizmetidir. * Tüm web hizmetleri API'dir.
- UI yok (kullanıcı arayüzü) → UI ile web uygulaması ve Selenium Webdriver kullanıyoruz
- Kullanıyoruz:
  - SABUN → XML
  - REST → JSON, XML, METİN
  - Postacı, Huzurlu Kütüphane

3. SoapUI nedir? ve bunu mevcut projenizde nasıl kullandınız?

- SOAP UI, önde gelen açık kaynaklı platformlar arası API Test aracıdır
- SOAPUI, test uzmanlarının farklı Web API'sinde otomatikleştirilmiş işlevsellik, regresyon, uyumluluk ve yük testleri yürütmesine olanak tanır.
- SOAPUI, her tür API'yi test etmek için tüm standart protokolleri ve teknolojileri destekler.
- SOAPUI arayüzü, hem teknik hem de teknik olmayan kullanıcıların sorunsuz bir şekilde kullanmasını sağlayan basittir.

4. REST tabanlı mimaride yaygın olarak kullanılan bazı HTTP yöntemlerinin adı?

- Oluştur → POST (sunucuya veri gönder)
- Oku → GET (belirli bir URI kullanarak belirli bir sunucudan veri alır)
- Güncelle → PUT (Hedef kaynağın tüm mevcut temsillerini yüklenen içerikle değiştirir)
- Sil → SİL (Bir URI tarafından verilen hedef kaynağın tüm mevcut temsillerini kaldırır.)

5. HTML Durum Kodları?

- 1xx → Bilgilendirici
- 2xx → Başarılı (istek başarıyla kabul edildi) (200 → Tamam, 201 → Oluşturuldu, 202 → Kabul Edildi, 204 → İçerik Yok)
- 3xx → Yönlendirme
- 4xx → İstemci Hatası (400-Hatalı İstek, 401-Yetkisiz, 403-Yasak, 404-Bulunamadı, 405-Yönteme İzin Verilmiyor)
- 5xx → Sunucu Hatası (500-Dahili sunucu Hatası, 502-Hatalı Ağ Geçidi, 501-Uygulanmadı, 503-Hizmet Kullanılamıyor)

6. Yanıt aldığınızda ilk kontrol ettiğiniz şey nedir?