- 1 -

**129**

**AA tarafından güncellendi**

**Reallife B10 (2019) Mülakat Soruları**

**116**

•

**Java Teknik Soru örnekleri**

**110**

•

**MarufOcak Özeti**

**107**

•

**Çerçevenizi tanımlayın**

**106**

•

**Oluşturduktan sonra Salatalık Raporu**

**104**

•

**Çerçevenizi sıfırdan nasıl oluşturabilirsiniz?**

**103**

•

**AWS**

**98**

•

**SELENIUM GRID & SauceLabs**

**96**

•

**JENKINS**

**94**

•

**JIRA**

**91**

•

**GIT & GITHUB**

**85**

•

**SQL**

**84**

•

**HTML ve CSS**

**75**

•

**API / Web Servisleri**

**64**

•

**salatalık ve kornişon**

**62**

•

**TestNG ve JUnit**

**60**

•

**UZMAN**

**47**

•

**SELENYUM**

**35**

•

**JAVA**

**10**

•

**SDLC ve ÇEVİK**

**02**

•

**DAVRANIŞSAL SORULAR**

**İNDEKS**

•

2

**DAVRANIŞSAL SORULAR**

**1. Bana kendinden bahset?**

Her şeyden önce, bana bu fırsatı verdiğiniz için teşekkür etmek istiyorum ve zaman ayırdığınız için gerçekten minnettarım. benim adım alper

Aslan ama daha çok Albert olarak bilinir.

• 6 yılı aşkın süredir BT sektöründeyim ve şu anda ekibimde **bir SDET'im** .

• Farklı sektörlerde çalıştıktan sonra;

o İletişim, Eğitim ve Finans sektörlerinde **alan bilgisi** edindim .

o Dolayısıyla, güçlü bir **SDLC anlayışına sahibim ve hem Şelale hem de Çevik** ortama aşinayım

o Aşağıdakiler de dahil olmak üzere çeşitli test türlerinde oldukça yetkinim; fonksiyonel, regresyon ve duman testi.

• **Manuel test cihazı** olarak başladım ve sonunda **otomasyon testinde uzmanlaştım**

o Otomasyon çerçevesini sıfırdan geliştiriyorum. Birçok "çok" test komut dosyasını başarıyla tasarladım ve geliştirdim

**Veriye dayalı** , **Davranışa dayalı** ve **Karma Çerçeveler** kullanılarak .

o Core **java** , **Selenium webdriver** , **Maven** , **Cucumber** , **JUnit** , **TestNG** , **Jenkins** ve daha birçok araçta iyiyim.

test otomasyonu. Çoğunlukla açık kaynaklı araçları tercih ederim.

o Kullandım

o Sürüm kontrolü için **GITHUB** , (şu anki şirkette sürüm kontrolü için **SVN kullanıyoruz** )

o Sürekli entegrasyon/dağıtım için **JENKINS ,**

o Proje yönetimi ve hata takibi için **JIRA**

o ve arka uç testi için **SQL**

o Projemde **API** testi üzerinde çalıştım ve **Restful** , **postman** ve **Rest Assured kitaplığını kullandım**

o Otomasyon çerçevesini birkaç kez sıfırdan geliştirdim.

o **Veriye dayalı** , **Davranışa dayalı (BDD)** kullanarak birçok "çok" test komut dosyasını başarıyla tasarladım ve geliştirdim.

ve **Hibrit Çerçeveler** .

o **BDD'yi** destekleyen POM'a dayalı “test çerçevemi” geliştirdim .

o Ayrıca, çerçevem JDBC kullanarak Veritabanı testini destekliyor. Bu günlerde JOOQ öğreniyor ve deniyorum. yazmak gibidir

gerçek SQL sorguları. Biliyor musun bilmiyorum ama harika.

• Scrum master ve Oracle Java Programmer sertifikalarına sahibim.

• Yumuşak beceri söz konusu olduğunda kendimi;

o Olumlu bir **insan** , çapraz fonksiyonel ekip üyesi

o **Hızlı öğrenen** , değişen koşullara uyum sağlayan ve detay odaklı.

o Steve Jobs'un dediği gibi, “ *ne kadar zeki olursanız olun, harika insanlardan oluşan bir ekibe ihtiyacınız vardır* ” ve bay/hanım bana inanın

büyük takım oyuncusu. Ben halk insanıyım.

o Bir **takımda bireysel olarak da iyi çalışabilirim**

o Her zaman **son teslim tarihine uyduğumdan emin** olurum (bunlardan herhangi birini detaylandırmamı isterseniz, daha ileri gitmekten memnuniyet duyarım)

ayrıntılı olarak.?)

• Bu hemen hemen benim hakkımda. Teşekkürler.

3

**2. Rolünüzü tanımlayın?**

• POM'a dayalı “test otomasyonu çerçevemizi” geliştirmekten ve sürdürmekten ve yeni testler eklemekten sorumluyum.

otomatik regresyon paketimize vakalar.

• Ağırlıklı olarak Not Verme, Devam, Davranış, Uygulama, Veli gibi Akademik özelliklerin otomasyonundan sorumluyum.

ve Öğrenci Modülleri. Buna ek olarak Business, “Ödeme Talebi Yönetimi, Ücret Takibi, İK” gibi özelliklere de sahiptir.

Dosyalama Modülleri.

• Çeşitli testler yaptım, örneğin; fonksiyonel test, duman testi, regresyon testi ve arka uç testi

• Şu anki projemde, geliştiriciler projeye yeni işlevler eklediğinde Regresyon testini yürütmekle sorumluyum.

uygulama veya sprint sonu.

• Şirkete katıldığımda, Otomatikleştirmemiz gereken büyük bir regresyon takımımız vardı. Otomasyon için regresyon kapsamı

%10 civarında çok düşüktü. Çok az otomatik test vakamız vardı. Bu nedenle, Koşmaya çok zaman harcıyorduk

gerileme. Çünkü regresyon paketinden tüm test senaryolarını çalıştırmak için yeterli zamanları yoktu, bu yüzden birçok yeni

kusurlar, bu nedenle proje için otomasyona gerçekten ihtiyaç vardı. Katıldığımdan beri uygulama işlevselliğini analiz ediyorlardı.

Son iki yıldır **Regresyon takımını analiz** ediyorum, aşağıdaki gibi öncelikli test senaryoları buldum:

işlevsellik en çok kullanılan, hangi işlevsellik en kritik, hangi işlevlerin manuel olarak yapılması uzun zaman alacak

yürütür, hangi işlevler sıklıkla bozulur.

• Otomasyon çerçevemi sıfırdan geliştirdim.

• Test senaryolarını otomatikleştiriyorum. Test senaryolarını otomatikleştirdiğimde, bazılarını günlük bir duman testinin mümkün olduğu kadar tekrarlanması için planlıyorum.

ihyaç olduğu gibi. Duman testi daha sonra raporunu bana ve ekibime gönderecek.

• **Her uygulama sürümünden önce tüm regresyon takımını çalıştırırım** . Test sonucunu analiz ediyorum. Geçti-kaldı raporu veriyorum. ben

Başarısız olduğunda, herhangi bir şeyin yanlış olup olmadığını görmek için yürütmeyi izleyin. Kodum nedeniyle başarısız olursa kodumu düzeltmem gerekir. (Belki

bu süre zarfında uygulama kapalıydı ve betiğimi yanlış zamanda çalıştırdım.) Gerçekten bir kusur varsa,

kusur ve düzeltilene kadar tekrar test edin.

• İşlevler arası bir ekip üyesi olarak, işlevsellik testçilerine yardım ediyor, onlara temel otomasyon çerçevesini, Java'yı ve

Selenyum onları takımın bir parçası yapmak için, Hepsi takımın verimliliğini artırmak için. En azından test senaryolarını yürütebilirler ve

sonuçları analiz edin.

• ÇAPRAZ FONKSİYONEL ekip üyesi olarak, gerektiğinde işlevsel test ekibine de

manuel test durumları. Ve yeniden üretebileceğim herhangi bir kusur varsa, kusuru JIRA'ya kaydederim.

• Ayrıca her büyük sürümde prodüksiyon desteği de yapıyorum. Normalde üretim desteği hafta sonudur. geleceğim ve

dağıtımdan sonra üretim ortamında 'salt okunur' test senaryolarını yürütün. Dağıtımda herhangi bir sorun varsa ben

sorunu en kısa sürede çözmek için derhal Dev ve QA ile iletişime geçmelisiniz.

• Sprint bakım toplantısında, test edilebilir bir şey olduğundan emin olmak için her zaman kullanıcı hikayelerine geri bildirim veririm ve

ölçülebilir. Örneğin: uygulamada şöyle bir değişiklikten sonra söylenen bir kullanıcı hikayesi vardı.

iyileştirmeli. İş adamlarına performans iyileştirme ile ne demek istediğinizi sordum. nasıl ölçersin

iyileştirme? Bundan sonra daha iyi kullanıcı hikayeleri buldular (çevikte gereklilik)

• Bunun yanında size gerçekten zevk aldığım sorumluluklarımdan birinin kullanıcı hikayesi oluşturma oturumları olduğunu söyleyebilirim, çünkü bu çok

kullanıcı açısından ilginç. Çünkü uygulamayı sürekli test eden biziz. den düşünüyorum

son kullanıcı bakış açısı. Kendimi son kullanıcının bakış açısına koyarak iyi şeyler yaptığımı düşünüyorum. Bu nedenle, denediğimizde

kullanıcı hikayesi oturumları, kabul kriterlerimizi çok daha iyi hale getiriyoruz. Böylece, İş Analisti kullanıcı hikayelerini gözden geçirir,

kabul kriterlerini gözden geçiriyoruz, sorular soruyoruz, geri bildirim veriyoruz, kullanıcı hikayelerini iyileştiriyoruz ve böylece ekibimizi daha iyi hale getiriyoruz

üretken. Çünkü daha iyi, net kabul kriterlerimiz var. Bu bizi, gereksinimlerimizi daha iyi, kodumuzu daha iyi yapar,

daha net ve belirsiz olmak yerine, kullanıcı hikayesi oluşturma oturumunun kendisi açısından bazı kusurlardan kaçınıyoruz.

bir kullanıcı hikayesi, belirsiz kodlar yapmak, bir şeyleri yanlış yapmak.

• Ayrıca, Çevik Scrum Takımının bir parçası olarak, gereksinim gözden geçirmeleri için birkaç adım adım toplantıya katılıyorum

ve BA'ya değerli geri bildirim sağlayın.

• Şu anki projemde otomasyon mühendisi olarak rolümün yaklaşık %80'i bu kadar.

4

**3. Günlük aktivitenizi anlatabilir misiniz?**

• İşyerindeki günlük aktivitelerim, Çoğunlukla sabah erkenden işe giderim ve emin olmak için Duman testinin sonuç raporunu kontrol ederim.

bu ortam çalışır durumda ve uygulama gün için kararlı veya değil.

• Bir şeyler ters giderse, herkes gelmeden en kısa sürede halledebilmeleri için ekibime bir e-posta göndereceğim.

maksimum üretkenliğe ulaşmak için çalışmak.

• Sonra e-postamı, önemli bir görev veya bildirim olup olmadığını kontrol ederim, ayrıca herhangi bir toplantı olup olmadığını da programımı kontrol ederim.

gün için ve ayrıca o gün hangi öncelikte yapılması gerektiğini gözden geçirmek için Jira'yı kontrol edin.

• Daha sonra dün yaptıklarım, bugün yapacaklarımız hakkında konuşmak için scrum ekibimle günlük standup toplantısına katılırım.

ve yolumda herhangi bir engel var mı? Bu toplantı yaklaşık 15 dakika sürer.

• Ondan sonra masama dönüyorum ve regresyon takımlarından test senaryolarını otomatikleştirmeye başlıyorum. Ayrıca, test senaryolarını otomatikleştiriyorum

geçilirse manuel olarak yaptıktan sonra sprint biriktirme listesinden. Sprintimizin sonundayız ve otomasyonumuzu bitiriyoruz, test

Sprint Demo'nun yürütülmesi, raporlanması, hazırlanması ve yürütülmesi

• Ayrıca, haftada bir kez, kodu gözden geçirmek için Kod İnceleme toplantıları yapıyoruz. bu gerçekten faydalı

• Benim için en tatmin edici olanı, genç test uzmanları veya geliştiricilerle, otomasyon ve

bazen teknik olmayan beceriler. Bunu kahverengi bir çanta öğle yemeği olarak yapıyoruz ve genel verimliliğimizi artırmamıza gerçekten yardımcı oluyor,

şirket kaynaklarından ve paradan tasarruf etmenin yanı sıra, bize iş arkadaşlarımızı tanımak için değerli bir fırsat verir ve

günün sonunda, benim için en önemli şey, birlikte çalıştığım insanlar.

**4. Projenizden bahseder misiniz?**

• Şu anda ekibimin özellikle arama FONKSİYONELLİĞİ'ne odaklandığı bir uygulama üzerinde çalışıyorum .

**ÇERÇEVE**

• Çerçevemde JavaScript programlama dili ve İletki kullanarak test kodumu geliştiriyordum.

• Pom.xml'e sahip olduğum bağımlılıklarımı yönetmek ve merkezileştirmek için Jasmine kullandım

• Çerçevem, her sayfa öğesinin bir sınıfa gittiği ve bunu kolaylaştırdığı POM'a dayalı olarak yapılandırılmıştır.

bakım ve kodumu düzenli ve temiz tutmak için.

o Ayrıca, uyguladığım adımı sürdürdüğüm ayrı sınıflarım var ...

o Sürücülerim, koşucularım ve faydalı yöntemlerim için ayrı ayrı klasörler oluşturdum.

• Çerçevem, Davranış Odaklı Geliştirmeyi (BDD) ve senaryo taslağını destekler.

• Hata izleme aracı olarak Jira'yı kullanıyorum.

• Ve sürekli entegrasyon sağlıyor ve testlerimi Jenkins kullanarak planlıyorum. (Y ŞEYİ yaparak X ŞEYİ başardım,

ve elde edilen Z ÖLÇÜLEBİLİR SONUÇ)

• Küçük hoş bir dokunuş: Bu proje sırasında bir otomasyon testçisi olarak gelişmeme gerçekten yardımcı olan başarı,

SORUN X'i keşfetti, Y EYLEMİ yaptı ve ardından ölçülebilir Z sonucu gerçekleşti.

**ÇERÇEVE**

Şirketimin uygulaması, ön uçta SmartClient adlı JavaScript tabanlı bir UI Çerçevesi kullanılarak web uygulaması olarak oluşturuldu.

ve arka uçta C# kodu. Her yerde bulunan ondan fazla sunucuda bulunan Microsoft SQL Veritabanımız var.

dünya. Bunlar web uygulamalarıdır, bunları test etmek çeşitli tarayıcılarda, çeşitli sürümlerde ve çeşitli

okul yapılandırmaları. İşi izlemek için ComindWork adlı bir biletleme sistemi kullandık ve tek testçi olarak görev yaptım

ve her şeyin tasarlandığı gibi çalıştığından emin olmak için kalite kontrol görevlisi. Çeşitli tasarım ve işlevsellik yapardım

tavsiyeler.

5

**5. En büyük Başarınız nedir?**

• Başarılarımdan biri, ekip içinde harika ve güvenilir bir ilişki kurmak olduğunu söyleyebilirim.

• Teknik olarak soruyorsanız : Son projeme katıldığımda uygulamada çok daha az “id” vardı, bu yüzden saatler harcamak zorunda kaldım.

POM projemde bir Web sayfası öğesi bulun, böylece geliştiriciler ve diğer ekip üyeleriyle iletişim kurdum ve tüm

birlikte uygulamaya "Id" koyma yetkisine sahip olduğum çözümü kendim buluyoruz.

• Bu benim için harikaydı, benim ve diğerlerinin zamanını kurtardı. Bu yüzden öğeleri bulmak için zaman harcamak yerine zamanımı

daha fazla otomasyon testi komut dosyaları oluşturma ve bunları yürütme.

• Yeni araçlar bulmayı ve kullanmayı gerçekten çok seviyorum. Ve ben ekibe katılmadan önce sadece PNG formatında ekran görüntüsü almışlardı. Ama ben

sorunu geliştiriciler için daha anlaşılır hale getiren notasyonlu GIF animasyonlu görüntüleri kullanmaya başladı.

**6. Neden iş arıyorsunuz? (Neden pazardasın?)**

• Mevcut projem yakında biteceği için iş arıyorum. Yöneticim bana yeni şeyler aramaya başlamam gerektiğini söyledi.

fırsatlar.

• Şirketimde beni tutmak isterler. Orada bir aile gibiyiz. Ama şirket taşınmayı planlıyor ve ben hissetmiyorum.

bununla rahat. Dallas'tan ayrılmayı planlamıyorum.

**7. Bu pozisyona neden başvurdunuz?**

• İş tanımına baktıktan sonra, günlük faaliyetlerime ve deneyimlerime uygun olduğunu düşünüyorum.

• İş tanımına güveniyordum bu yüzden başvurdum.

• Ayrıca, şirket hakkında biraz araştırma yaptım ve şirketin ürün ve hizmetleri hakkında gerçekten heyecanlıyım...

**8. Kendinizi bundan 5 yıl sonra nerede görüyorsunuz?**

• Daha teknik olmak için mümkün olduğunca çok şey öğrenmek istiyorum. Mobil Geliştirmeye geçmek istiyorum. Ben bir olmak istiyorum

profesyonel Mobil Geliştirici.

• Bundan 5 yıl sonra teknik olarak çok rekabetçi biri olmak istiyorum.

**9. Zayıf yönünüz nedir?**

• Sanırım zayıflığım, bana bazı sorumluluklar verildiğinde ve bunun için bir son tarih olduğunda, her gün çalışıyorum ve

gece, bazen haftanın 7 günü. Bu benim aile hayatım için kötü; gerçek şu ki, işim bitmeden uyuyamıyorum

ödevler.

**EN BÜYÜK zayıflık?**

• En büyük zayıflığım, işime kendini adamış ve tutkulu bir insan olmamdı.

• Bu kulağa iyi bir kalite gibi gelse de, ekip üyelerinin aynı tutkuyu paylaşıp paylaşmadığını bilirsiniz.

senin gibi, çok sinirli olabilir. Mesela, insanların gereksinimlerini takip etmediğini gördüm, her zaman insanları bekledim

karşılaştığımız tüm sorunlara karşı ciddiye alın.

• Bu zayıflığı gidermek için pratik, gerçekçi adımlar atmam gerektiğini anladım.

• Ve tüm ekibin güvence altına alınması için diyalog ve bir tür iletişim yöntemlerinin açık olmasının önemli olduğunu düşünüyorum.

scrum ekibimizdeki üyelerin harika bir kimyası var.

• İş dışında kahve toplantısı, kaçış odası, barbekü organizasyonu gibi birçok etkinlik düzenledim.

sevdiğim :) .

• Bu, ekip üyelerimi aileleriyle ve özellikle çevreleriyle yakın ilişki kurmama ve anlamama yardımcı oldu.

daha iyi ve en önemlisi bunlar, ekip üyelerimin işe olan tutkumu anlamalarına yardımcı oldu.

• Ayrıca, bir ekibin dinamiklerini anlamama yardımcı olması için bazı liderlik kurslarına kaydoldum ve ekibime nasıl akıl hocalığı yapabilirim?

takım üyeleri benim zihniyet almak için.

• Sonuç olarak, scrum ekibimizin kimyasını geliştirdim ve projelerimiz en yüksek kalitede yayınlandı.

Müşteri memnuniyeti.

• Zayıflığımla mücadele etme konusunda gerçekten iyi bir deneyime sahibim ve bir duruma yaklaşma konusunda çok proaktifim.

sorun.

6

**10. Güçlü yönleriniz nelerdir?**

• Ben çok detay odaklı bir insanım. İşime son teslim tarihine göre öncelik verebilirim.

• Ayrıca işime çok bağlı biriyim.

• Ben de dürüst bir insanım ve KG sürecinde beceri ve uzmanlığa sahibim.

• En güçlü yanlarımdan biri, bir grafik becerisine sahip olmamdır, yani

Örneğin; *Bir paragrafı, olayı, vakayı, hatta bir cümleyi gördüğümde veya işittiğimde, zihnimde kolaylıkla grafiğini çizebiliyorum ve*

*gerçek hayatta uygulayın.*

• Sunumlarda gerçekten iyiyimdir. Bir konuyu veya herhangi bir konuyu sunduğumda, bunu çok net ve yapılacaklar şeklinde belirtirim. Bana izin ver

bir örnek ver;

*Önceki şirketimde bir inceleme toplantısı yaptık ve 3 ekip üyesi vardı, üst yönetim her ekipten bize sordu.*

*görüşlerini sunmuştur. Sorumluluklarımızı ayırıyoruz ve tek tek anlatmaya hazırız ama ne yazık ki her ekip*

*üye, şartlar net olmadığı için toplantı konseptini yanlış anladı ve üst yönetimin*

*bizi bir sunum bekliyordu ve ekip üyelerinden biri projemizi sundu ve ilk ekip sunumlarını sunmaya başladıktan sonra*

*gözden geçirin, herkes birbirine baktı ve ekibin biraz gergin olduğunu hissediyorum ve*

*- Slack hakkında bu inceleme için bir grubumuz vardı ve onlara yazdım merak etmeyin bir planım var.*

*- Sadece konuya odaklanıyorum ve bir şablon hazırlıyorum ve ekip üyelerini gönderiyorum ve onlar da parçalarını hazırladılar ve birleştirdim.*

*20 dakika içinde hazırız ve sonra hepsini kolayca sunarım.*

*Bu yüzden gücüm ekibi organize etmeye ve sorunumuzu çözmeye yardımcı oluyor ve tüm ekip üyeleri bunun için mutluydu çünkü biz*

*bitirdim ve çok başarılı bir şekilde sunduk... Biz bir ekiptik ve inanıyorum ki bir ekiple tutkulu bir şekilde çalıştıysanız, her şey*

*ulaşılabilir ve beklenen sonuca zamanında ulaşabiliriz, bu da müşteri memnuniyeti anlamına gelir ve bu da müşteri memnuniyeti anlamına gelir.*

*şirket başarısı.*

• Teknik En büyük güç:

o Java'da en güçlüyüm çünkü onu seviyorum. nedenini açıklayın: artılar, faydalar, vb.

• Nedeninin mantığını açıklayın: Kapanışlar, geri aramalar, vaatler benim gücümdür, çünkü bunlar kapsamlı bilgilerime uygundur.

analitik beceriler, beynim zaten bu şekilde çalışacak şekilde kablolanmıştır, bu yüzden JS dili bana çok doğal ve kolay geldi.

**11. Son projeniz sırasında karşılaştığınız bir zorluğu tarif edin?**

• Sanırım şu anki projemde karşılaştığım en büyük zorluklardan biri...

o ... yeni bir geliştiricimiz vardı. O genç ve ama çok zeki bir çocuk. Ne zaman bir hata bulsam, geliştirici

gergindi ve kabul etmedi ve çoğu zaman onu sakinleştirmem gerekiyordu ve bazen BA'ya sormak zorunda kaldık.

netlik için

o Sonra gereksinimin kendisinin yeterince spesifik olmadığını fark ettim, bu yüzden geliştiriciden farklı anladım

o Sprint Retro'da, gereksinimlerin açıklığa kavuşturulması için daha fazla zaman harcamamız gerektiğini söyledim çünkü bunun anahtar olduğunu biliyorsunuz.

proje başarısı için. Bunu yaptık ve bu sorun çözüldü.

• Bu pek çok firmada çok yaygın değil ama bizim şirketimizde platin paket kullanan müşterilerimiz var. sahip oldukları

gündüzden geceye veya tam tersi bir özellik hakkında yeni bir rapor veya düzeltme isteme hakkı. Bazen saat 11'de bir telefon aldım.

test edilmesi gereken bir özellik veya rapor var. Bu yüzden, bu değişiklikleri test etmek veya

özellikleri. (gerekiyorsa örnek veriniz)

• Sonuç üzerinde daha fazla çalışın. Bence en önemli sorun, ortamdaki yanlış anlaşılma ve iletişim eksikliği.

iş hayatı. Grup olarak bir araya gelip tartışırsak çözemeyeceğimiz hiçbir şey yoktur. gerçekten minnettarım ve mübarek

içinde bulunduğum takımda olmak, çünkü sorunu çözmek için işbirliği yapabildik ve bir araya gelebildik.

• Karşılaştığım teknik zorluk, web sayfalarından doğru HTML kodunu alarak dinamik öğeleri bulmaktı.

Kimlikler sorundu; bazen sayfada görünüyor, bazen de kayboluyor. Sonra ya örtük koymak zorunda kaldım

veya onları bulmak için açık bir şekilde bekleyin.

7

**TEKNİK OLMAYAN Zorluklar:**

**A. Çok fazla işiniz varsa ve son teslim tarihine kadar bitiremezseniz ne yapmalısınız?**

• Geliştiriciler kodlarını zamanında dağıtmadıklarında, test ekibimizin tamamlamak için yeterli zamanı olmaz. ve üst

yönetim bizden bitirmemizi istiyor. - Ekip üyelerimden bazıları yalnızca görevin tamamlanmasına odaklanır,

test kapsamı ve işin kalitesi hakkında. - Bu yüzden Sprint Bakım Toplantısında çok çalışmamız gerektiğini önerdim.

geliştiriciyle yakından ilgilenin ve günlük olarak iletişim kurduğumuzdan emin olun. - Ayrıca geliştiriciler öncelik

önemli görevler ve önce onlar üzerinde çalışın. Geriye kalan tüm senaryolar, eskisi gibi olmadığı için bir sonraki sprint'e aktarılacaktır.

diğerleri kadar önemlidir. - Son olarak, işime öncelik vermeye ve test liderimi ve yöneticimi ne görürlerse onu takip etmeye çalışırım.

ondan başlamam daha önemli.

• Son zamanlarda karşılaştığım zorluklardan biri, aynı zamanda QA olan başka bir iş arkadaşımın kişisel nedenlerle şirketten ayrılmak zorunda kalmasıydı.

muhtemelen bir vize sorunu için ülkesine geri dönmek zorunda kaldı ve o zamandan beri onun işini kendi başıma halletmem gerekiyordu.

sadece iki otomasyoncuyduk, ben ve Jason. - Ve üretim tarihi ertelenemez. - Yani, sorumluluk aldım, ben

yeni bir plan yapmaya başladım ve SM'imle iletişim kurdum, bu elimden gelenin en iyisini yapacaktı, ancak sonuca ulaşmak için biraz yardıma ihtiyacım var.

zamanında üretim hedefi. Her zaman olduğu gibi, iletişim, daha iyi ekip çalışması için birbirimizi anlamanın anahtarıdır. - Ve ayrıca,

geliştiriciler de uygulamayı test etmek için katıldı. Sonunda zamanında başardık. - Gurur duymamın sebeplerinden biri de bu.

scrum ekibinin bir üyesi olmak. Herkes aynı amaç için çalışır ve sorumluluğu paylaşır.

• Teknik bir sorunum yok çünkü teknik olan her konuyu işimin bir parçası olarak görüyorum. Orada olacak

Her zaman üstesinden gelinmesi gereken bir zorluk olacak ve DAİMA üstesinden gelip öğreneceğim. bence gerçekten zor olan ve

düzeltmek her zaman o kadar kolay değildir: İNSANLAR. yine aynı insanlarla yüksek stresli, hızlı tempolu 40+ saat çalışıyoruz

ortamlar. - çoğu zaman, farklı kültürlerden, ülkelerden, geçmişlerden vb.

iletişim son derece zor - doğal olarak, çatışmalar yozlaşıyor ve gözlemleme konusunda geniş deneyime sahibim ve

sonunda bu tür çatışmaları önlemeyi ve çözmeyi öğrenir. - Bunu nasıl yaparım? - Dikkatini vererek, empati kurarak ve üstü

tüm, etkili iletişim. bu, birçok insanla çok iş yapmak anlamına gelir, çünkü çatışmalar genellikle iki kişiyi içerir.

veya daha fazla katılımcı ve ekibin geri kalanını olumsuz etkileyerek üretkenliği azaltabilir. - mutlu çalışanlar

üretken çalışanlar anlamına gelir. - Deneme yanılma yoluyla meslektaşlarımı gerçekten anlamayı ve

ofis-yer siyasetinin zorlu sularında gezinin. örneğin, [belirli bir örneğe gidin]

**B. Çatışmayı nasıl ele alıyorsunuz?**

• Hiçbir şey kişisel değildir. Herkes şirketin çıkarlarını düşünüyor, bu yüzden endişemi ve açıklamasını açıklamak istiyorum.

benim için mantıklı. - Tabii ki şirketime en faydalı olan şeyleri yapabilirim. bu yüzden iletişim kurmaya çalışıyorum

ve endişesini anlamaya çalışırdım. Çünkü herkesin amacı aynıdır ve işini bitirmek ister.

başarılı bir şekilde.

**12. Stresle nasıl başa çıkıyorsunuz?**

• Sprintlerimizden biri, geliştiricim kodu çok geç dağıttı ve işimi halletmek için fazla zamanım olmadı. Ama ben

çok çalışmak, fazladan saatlerce ve özellikle geceleri çalıştım ve görevimi zamanında bitirdim.

• İlk yaklaşımım sakinleşmeye çalışmak ve fazladan saatler çalışmak. Stresten çok durumlara tepki vermeye çalışırım. Bu şekilde,

durum ele alınır ve stresli hale gelmez. Çoğunlukla bunlar stresle başa çıkmama yardımcı oldu. Ben de çikolata yiyorum.

• Ayrıca scrum ortamında ekip olarak çalışıyoruz. Benimle her zaman iyi bir iletişim ve ilişki kurarım.

meslektaşlar. Bu yüzden bana güveniyorlar ve benimle çok rahat iletişim kurabiliyorlar. Her zaman yanlış iletişimden kaçınırım ve

takım her zaman bana inan.

• Bazen bazı gereksinimler anlaşılmaz, bu yüzden gereksinimleri anlamaya ve anlamaya çalışırım. İçinde

Uygulamanın başlangıcında işlevselliği anlamak için ekstra çaba harcıyorum. Bazen anlamak zaman alır.

**13. Baskı altında çalışabilir misiniz?**

• Çalıştığım hiçbir projenin üzerinde baskı olmadığını hatırlamıyorum. Basınç bazen iyi bir şeydir. Seni çalışmaya zorlar

daha sert ve daha akıllı.

• Üzerinde çalışılacak çok sayıda ödev veya yaklaşan bir teslim tarihi gibi iyi baskı, motive olmama ve motive olmama yardımcı oluyor.

8

üretken. Elbette, çok fazla baskının strese yol açabileceği zamanlar vardır; Ancak, dengeleme konusunda çok yetenekliyim

sık sık stresli hissetmemi engelleyen birden fazla proje ve son teslim tarihleri. Örneğin, bir zamanlar üç tane vardı

Aynı hafta içinde teslim edilecek büyük projeler, bu çok fazla baskıydı. Ancak, nasıl yapılacağını ayrıntılı olarak anlatan bir program oluşturduğum için

Her projeyi küçük görevlere bölerdim, üç projeyi de vaktinden önce tamamladım ve

gereksiz stres

**14. Ne kadar erken başlayabilirsiniz?**

• Teklif mektubu ve başlangıç tarihinden 2-3 hafta sonra işe başlayabilirim diye düşünüyorum.

**15. Yarın başlayabilir misin?**

• Mevcut şirketim için adil olmayacak ve yarın ayrılırsam ekibim benden memnun olmayacak ve bence de değil

profesyonel ve bunu daha önce hiç yapmadım. Çok üzgünüm yarın başlayamam.

• Ayrılmadan önce otomasyon çerçevesi bilgisini diğer ekip üyelerine aktarmam gerekiyor.

**16. Bizden ne kadar bekliyorsunuz? ya da bizimle ne kadar çalışmaya hazırsınız?**

• Beklentim 90-100K arası. Bir miktar artış olması benim için büyük bir motivasyon olacak. *(unutmayın 48*

*güven seviyenize bağlıdır. İsterseniz arttırabilirsiniz. Size sormalarının nedeni, oranı onaylamak istemeleridir.*

*Belirli bir bütçeleri olacak ve danışmanlık firması biraz para kazanmak için birkaç dolar işaretlemek zorunda. Yani, her zaman*

*pazarlık edilebilir. Çok düşük diyorsanız, kendinize güvenmiyorsunuz demektir. )*

**17. İşe alınırsanız, ne kadar kalmayı planlıyorsunuz?**

• Çalışmak için bir proje olduğu sürece, mümkün olduğunca uzun süre kalmaya hazırım. Şirketlerimle uzun vadeli çalışmayı seviyorum.

Böylece başarının bir parçası olacaktım. Özgeçmişime bakarsanız son 10 yıldır iki şirkette çalıştım.

**18. Seni işe alırsam ne yaparsın?**

• Biliyorsunuz ilk hafta tüm evrak işlerini halledeceğim, makinaları alacağım ve projeye gerekli ulaşımı sağlayacağım,

veritabanları vb.

• O zaman şirket kültürünü öğrenmem gerekecek. Biniş süreci.

• Projelerim ve takım arkadaşlarım hakkında da daha fazla şey öğrenmeliyim.

• Bence, projenin ne yaptığını anlamak, daha üretken olmak istiyorsam çok önemli.

**19. Referans için mevcut işvereninizle iletişime geçebilir miyim?**

• Özgeçmişimi göndermek istiyorsanız lütfen devam edin (işe alım görevlilerine). Ama başkalarından çok sayıda telefon görüşmesi alıyorum.

işe alımcılar. Özgeçmişimi gönderip göndermeyeceğinden emin olmayan hiç kimsenin işverenimi aramasına izin vermek istemiyorum. *(eğer söylerlerse*

*özgeçmişinizi gönderecek ve işten sonra en kısa sürede göndereceğinizi söyleyecektir. )*

**20. Testle ilgili en çok neyi seviyorsunuz?**

• Test yapmak benim için eğlenceli bir iş çünkü siz müşteri ve son kullanıcılar için çok önemli bir insansınız. Test etmeyi seviyorum çünkü son olarak

kullanıcı Bezelye sanat eseri ve hatasız daha iyi bir ürün satın almak istiyorum. Ayrıca, başkalarının ürünlerinden emin olmalarına yardımcı oluyorum.

üst kaliteye sahiptir. Toyota Camry'nin güvenlik özelliklerini test ettiğinizi ve neredeyse 100 hayat kurtardığınızı hayal edin.

işini yapmak ve işini sevmek.

9

**21. Sizi neden işe almalıyız?**

• Her şeyden önce, bu pozisyonla ilgili kapsamlı bir araştırma yaptım ve iş tanımınızı okudum ve

Bu görüşme sırasında bana çok yardımcı olduğunuz bilgiler, GÜVENLE şunu söyleyebilirim:

bu pozisyon. Yani, iş tanımında gerekli ve tercih edilen tüm teknik ve teknik olmayan uzmanlığa sahibim.

sadece bu pozisyonda başarılı olmak değil, aynı zamanda **gelişmek** .

• Steve J. **“** *ne kadar zeki olursan ol, harika insanlardan oluşan bir ekibe ihtiyacın var* **” dedi** ve inanın ben harika bir takım oyuncusuyum.

• Ancak, benim için temel gereksinim bu ve gerçekten en nitelikli kişi olduğuma ve beklentileri aşacağıma inanıyorum.

not ettiklerimin ötesinde, çünkü her zaman ölçülemeyen çok önemli yumuşak becerilere sahibim. Ve ben

İnanın, günün sonunda, iş tanımından veya ne yaptığımızdan bağımsız olarak, çalışanlarınızın kişiliği,

gerçekten önemli. Bu konuda benzersizim ve öne çıkıyorum çünkü yumuşak becerilerin değerini yıllar önce öğrendim ve

Bu becerileri geliştirme konusunda kapsamlı deneyim.

• Her zaman çalıştığım şirkete atlamaya yardımcı olacak yeni teknikler ve araçlar getiriyorum.

• Birçok kişi kısa bir süre içinde herhangi bir teknik beceri konusunda eğitim alabilir, ancak birini iletişim kuracak şekilde eğitmek

inanılmaz zor. Bolca böyle becerilere sahibim: Ben mükemmel bir iletişimciyim, son derece motive VE motive ediciyim,

ve hepsinden önemlisi, ben bir problem çözücünün tanımıyım. İşimi başarmak için yapılması gereken her ne ise ve

dahası, YAPACAĞIM.

• Bence bu pozisyon için en iyi niteliklere sahip adayı işe almalısınız.

• Diğer adayları tanımadığım için sadece kendimi temsil edebilirim.

• Tecrübe ve teknik uzmanlığımın şirkete ve projeye birçok değer ve fayda sağlayacağını düşünüyorum. Bence

bu yüzden beni işe almalısın.

**22. Bizim için herhangi bir sorunuz var mı?**

• Etrafımda iyi/kibar, çalışkan, zeki insanlarla çevrili olmak benim için çok önemli.

çok çalışmak ve kendimi geliştirmeye devam etmek. İş yerinde iş arkadaşlarımızla harcadığımızdan daha fazla zaman harcadığımızı düşünürsek

arkadaşlarımla veya ailemle bile olsa, iş yerimin kültürüne uyum sağlayabileceğimi ve

orada profesyonel. Peki, şirket kültürünüz nasıl? Takım nasıl?

*(araştırma yaptıysanız, şirket hakkında ilginç bir şey ortaya çıkarabilir ve onlardan detaylandırmalarını/nasıl çalıştığını vb. sorabilirsiniz.*

• aynı zamanda sürekli gelişmem ve pozisyonumda mükemmeli yakalamaya çalışmam ve bunu yapmanın en iyi yolu benim için önemlidir.

bu sürekli öğrenmektir. Her zaman yeni şeyler öğrenmeye veya eski şeyleri daha iyi öğrenmeye çalışıyorum. eğitim veriyor musunuz,

çalışanlarınızın eğitimini desteklemek için seminerler veya herhangi bir şey?

- 10 -

**SDLC ve ÇEVİK**

**1. Yazılım Testi Nedir?**

• İşlevsel ve otomasyon araçlarını kullanarak yazılım hatalarını bulmak amacıyla bir program veya uygulama yürütme süreci

• Bir yazılım programını/uygulamayı doğrulama/doğrulama süreci

• Test ediciler, geçmek için değil, kırmak için yaklaşmayı test etmelidir.

**2. Yazılım Gereksinimleri Belirtimi nedir?**

• Bir yazılım gereksinimleri belirtimi, müşteri ile tedarikçi arasında bir sözleşme işlevi gören bir belgedir.

• Bu SRS, son kullanıcının o uygulamaya ilişkin tüm gereksinimlerini içerir. SRS bir iletişim olarak kullanılabilir

müşteri ve tedarikçi arasındaki ortam.

• Geliştirici ve test eden kişi, uygulamayı SRS'de yazılı gereksinimlere göre hazırlar ve inceler.

• Belgelenen SRS, Müşteri için tüm gereksinimler dikkate alınarak İş Analisti tarafından hazırlanır.

**3. Yazılım Geliştirme Yaşam Döngüsü (SDLC) - SDLC Nedir?**

• SDLC, yazılım veya uygulama **geliştirme** aşamalarını tanımlar .

o Proje Planlama

o Gereksinim Toplama (Projeyi planlamak için kullanılan bilgilerin toplanması, Risklerin belirlenmesi)

o Tasarım (Uygulamanın nasıl oluşturulacağı)

o Kodlama (geliştirme) (Gereksinimlere göre geliştiriciler uygulamayı yazacaktır)

o Test

o Üretim (dağıtım)(Serbest bırakma ürünü)

o Bakım (Ürünün stabil olduğundan emin olmak, hatalarla ilgili müşteri raporuna bakmak ve düzeltmek)

**4. Yazılım Testi Yaşam Döngüsü (STLC) - STLC Nedir?**

• STLC, yazılım veya uygulamanın **test edilmesindeki** aşamaları tanımlar . STLC sürecinde farklı faaliyetler gerçekleştirilmektedir.

ürünün kalitesini artırmak.

o İhtiyaç analizi

o Test Planlaması

o Test Tasarımı

o Test Ortamı Kurulumu

o Testin Yürütülmesi

o Test Raporlama

**5. STLC ve SDLC arasındaki fark nedir?**

• STLC, SDLC'nin bir parçasıdır. STLC'nin SDLC kümesinin bir alt kümesi olduğu söylenebilir.

• STLC, yazılım veya ürün kalitesinin garanti edildiği test aşaması ile sınırlıdır. SDLC'nin eksiksiz ve hayati bir rolü vardır.

bir yazılım veya ürünün geliştirilmesi.

• Ancak, STLC, SDLC'nin çok önemli bir aşamasıdır ve nihai ürün veya yazılım onsuz piyasaya sürülemez.

STLC sürecinden geçerek.

• STLC aynı zamanda yayın sonrası/güncelleme döngüsünün bir parçasıdır, bilinen kusurların giderildiği veya SDLC'nin bakım aşamasıdır.

yazılıma yeni işlevler eklendi.

**6. Gereksinim nedir?**

• Gereksinimler, yazılım veya ürün için kullanıcıların beklentilerini iletir.

• Müşteriden gereksinimleri toplama, bunları analiz etme ve belgeleme süreci, gereksinim mühendisliği olarak bilinir.

• Gereksinim mühendisliğinin amacı, karmaşık ve açıklayıcı SRS 'Sistem Gereksinimlerini geliştirmek ve sürdürmektir.

Şartname' Belgesi

- 11 -

**7. İhtiyaç nereden geliyor?**

• Müşteriler, uygulama için gereksinimleri belirtir

• Bu uygulamayı en çok kullanacak olan son kullanıcılarla konuşun

• Ortaklarla konuşun –

• Etki Alanı Uzmanları ile konuşun – bu uygulamayı daha önce benzer şekilde oluşturmuş olan kodlayıcılar ve geliştiriciler veya

bir uzman, üretilmekte olan ürünün türü

• Sektör Analistleri ve Rakipler Hakkında Bilgi

**8. Test ne zaman başlar?**

• Test, gereksinimlerin test edilmesiyle başlar (en olası yanıt gibi görünen kodlama aşamasından sonra değil).

• İlk etapta gerekliliğin doğru olduğundan emin olmalıyız. Yanlış gereksinimle hata oluşturmak imkansız

ücretsiz uygulama.

**9. Gereksinimin iyi mi kötü mü olduğu nasıl anlaşılır?**

• Gereksinim (SMART) olmalıdır

o Özel à Kullanıcı oturum **açabilmelidir** . Geçerli kullanıcı adı ve şifreye sahip yetkili kullanıcı oturum açabilmelidir.

o **Ölçülebilir** à Kullanıcı çok hızlı oturum açabilmelidir (giriş düğmesine tıkladıktan sonra 2 saniye içinde).

o Ulaşılabilir **\_**

o **Gerçekçi** \_

o Test edilebilir à Kullanıcı **makbuzu** çok hızlı indirebilmelidir (2 saniyede

**10. Neden test ediyoruz?**

• Hatasız uygulama oluşturmak için.

• Memnun son kullanıcı ve müşteriye.

• Daha fazla gelir elde etmek için harika ürünler oluşturmak.

• Test etmeyi seviyorum ve test etmek benim tutkum.

**11. Test edenin ana sorumluluğu nedir?**

• Mümkün olduğu kadar erken hata bulmak. Hatanın çoğunun giderildiğinden emin olun.

• Hatasız ve kullanıcı dostu bir uygulama sunarak son kullanıcı ve müşteri satın alımını tatmin etmek.

**12. %100 test mümkün mü?**

• Hayal bile edemediğimiz sınırsız senaryo olduğu için uygulamayı %100 test edemiyoruz.

• Yazılım testi, mümkün olduğu kadar test edebileceğimiz **işlevselliğin önceliğine** dayanan risk tabanlı bir faaliyettir .

• %100 test mümkün olmasa da %100 müşteri memnuniyetinin kesinlikle mümkün olduğuna inanıyorum.

**13. Test hiyerarşisi nedir?**

• Birim testi à Geliştiriciler, geliştirme sırasında her modülü veya kod bloğunu test eder.

• Bileşen Testi à Bileşen, kendi kendine çalışabilen bağımsız bir işlevdir. Eski. Amazon Alıcı İşlevselliği,

Satıcı İşlevselliği, Prime Video İşlevselliği.

• Entegrasyon Testi à Tüm İşlevleri birleştirin. Bunları entegre ettiğimde, yine de tüm işlevleri kullanabilir miyim? Yapmak

emin olun hepsi hala çalışıyor.

• Sistem Testi à Uçtan Uca test. Her şeyi baştan sona test edin.

• Kabul Testi à Bir UAT (Kullanıcı Kabul Testi) Ekibi Kiralayın veya İş Analisti de Kabul Testi yapabilir.

Test tamamlandıktan sonra, QA ekiplerini onaylayabilmeleri için başka bir ekibin kabul testi yapmasını sağlamalısınız.

test başarılı oldu ve ürünü müşteriye hazır hale getirdi.

- 12 -

**14. Pozitif test nedir? Mutlu Yol testi?**

• Uygulamayı geçerli girişlerle test etme. “ **Mutlu Yol** ” Testi olarak da adlandırılır .

Eski. Geçerli bir kullanıcı adı ve şifre ile giriş yaparsanız, bu pozitif testtir.

**15. 508 Uyumluluk testi nedir?**

• Mülakatta birisine 508 testinin ne olduğunu sorun, sadece ne olduğunu söyleyin. 5-10 yıllık tecrübem var demeyin.

508 Uyumluluk testi.

• **Devlet web siteleri için bir gerekliliktir.**

• Devlet tarafından ve devlet için kullanılan tüm web siteleri. **Engellilerin kullanmasını sağlamak zorundalar.**

• Örnek: Healthcare.gov için Uyumluluk yöneticisi var ve 508 Uyumluluk yapan özel bir QA ekibi var

web sitesinin engelli kullanıcılar için 508 uyumlu olduğundan emin olmak için test.

**16. Riske dayalı test nedir?**

• %100 test yapılamadığından risk analizi yapmak zorundayız. Analize dayanarak testimize öncelik vermeliyiz

Önce aktivite ve yüksek riskli alanı test edin. Örneğin:

o En kritik işlevler

o En sık kullanılan işlevler

o En karmaşık işlevler vb...

**17. Bu regresyon takımını oluşturmak ne kadar sürdü?**

• 3 yıl sürdü; 2 test cihazı 1 manuel test cihazı + 1 otomasyon test cihazı

• çalıştırdığımızda:

o serbest bırakılmadan önce

o büyük hata düzeltmesinden sonra

o büyük yeni işlevlerden sonra

• test senaryolarını tuttuğumuz ve ekip olarak bir sprintte birden fazla uygulanacak kararı aldığımız yer

bazı senaryoları test edersiniz.

**18. Regresyon paketini çalıştırırken bize bir meydan okuma söyle?**

• Başarısızlıklar. Çünkü regresyon takımı çok uzun zaman önce geliştirildi ve neyin değiştiğini bilmiyorsunuz. özellikleri

düğmesi değişmiş olabilir.

**19. Kaç tane ortamınız var?**

• Geliştirme Ortamı

o Birim testi

o Test ortamından daha az kararlı

• Test Ortamı

o Manuel test burada gerçekleşir

o Üretim ortamını tam olarak çoğaltır

o Değişiklikler aralıklarla dağıtılır

o Burada otomatik **duman testleri** yapılır

§ Uygulamanın diğer büyük testleri gerçekleştirmek için yeterince kararlı olduğundan emin olmak için test ortamına karşı çalışır

faaliyetler.

§ Değişiklikler Test ortamına her dağıtıldığında çalıştır

§ Geliştirme ortamında çalıştırılabilir

o Otomasyon testleri burada çalıştırılır

o Otomatik Entegrasyon testleri burada çalıştırılır

• Üretim Öncesi Ortam

- 13 -

o UAT ortamı

o Demo burada gerçekleşir

o yük/performans testi burada gerçekleşir

o Değişiklikler büyük aralıklarla dağıtılır

o Burada otomatik majör **regresyon testleri** (yayınlanmadan önce)

§ UAT ortamına karşı çalışır

§ Yeni değişikliklerin herhangi bir kusurla sonuçlanıp sonuçlanmadığını öğrenmek için

§ Büyük hata düzeltmelerinden ve her sürümden sonra çalışır

§ Bu test, test planında kararlaştırılır

o Çok kararlı

• Üretim ortamı

**20. Regresyon testinin hangi kısmı otomatikleştirilmelidir?**

• Kararlı testler

• Sık tekrarlanan

• Basittir ve test cihazı girişi gerektirmez iyidir

otomasyon adayları

**21. Regresyon testlerinizin etkili olmasını nasıl sağlıyorsunuz?**

• Regresyon testleri, kusurları yakalamaya izin verecek kadar geniş ve ayrıntılı olmalıdır. Yinelenen testi de ortadan kaldırabilirsiniz

durumlarda, test senaryolarını ve otomatik testleri mümkün olduğunca birleştirin.

**22. Yazılımda bir dizi kritik hata düzeltildi. Raporlarla ilgili tüm hatalar tek bir modüldedir. test yöneticisi**

**sadece raporun modülünde regresyon testi yapmaya karar verir.**

• Regresyon testi diğer modüllerde de yapılmalıdır çünkü bir modülün sabitlenmesi diğer modülleri etkileyebilir.

**23. Regresyonunuzu nasıl yürütüyorsunuz? Ne sıklıkla, kaç VM, kaç gün, kaç test?**

• Regresyon her sürümden önce planlanır ve yılda iki kez yayınlarız (İlkbahar sürümü ve sonbahar sürümü).

• Büyük bir hata düzeltmesi olduğunda da gerileme olur.

• Yaklaşık 500 özellik dosyası ve 1300 senaryo.

• Regresyon testleri jenkins tarafından başlatılır. Testler, jenkins sunucusunda (VM) yürütülür. Linux sunucum RedHat.

• En son çalıştırma 12 saatten fazla görünüyor. Bir diğer cevap ise;

o Bir dizi regresyon testi oluşturdum. Regresyon etiketli özellik dosyalarıdır. Ve jenkins'te tekmeleyen bir işim var

regresyon testlerindendir. Testi tetiklemek için maven komutunu kullanır. Maven komutu şu etiket adını içerir:

mvn testi -D salatalık.options =”--tags @Regression”.

o Yürütmenin sonunda, jenkins ayrıntılı test adımları ve ekran görüntüleri içeren HTML raporu oluşturur.

**24. Fonksiyonel test nedir?**

• Fonksiyonel test ekibine manuel test cihazı da denebilir, otomasyon ekibi tarafından da yapılabilir (otomasyon fonksiyonel

test yapmak). Kara kutu testine veya manuel test cihazlarına benzer. Sadece uygulamanın belirli işlevselliğini test edin. Eski. Olabilmek

Kullanıcı Girişi? Kullanıcı oturumu kapatabilir mi? Uygulamanın görünüşünü ve hissini test etmemek.

**25. İşlevsel olmayan test nedir?**

• Performans testi, Güvenlik testi, Ör. 2000 kullanıcısı aynı anda uygulamaya giriş yapabilir mi? Kullanıcı

1 saniyede sonraki sayfa?

**26. Birim testi nedir? Hiç birim testi yaptınız mı?**

• Beyaz kutu testinin bir parçasıdır. Kodu Geliştirme ortamından dağıtmadan önce geliştiriciler tarafından yapılır.

QA ortamına.

• Geliştiriciler tarafından yapıldığı için henüz birim testi yapmadım. Ama bunu öğrenebileceğimi ve gerekirse yapabileceğimi düşünüyorum.

- 14 -

**27. Bileşen testi nedir?**

• Uygulamanın her bir bileşenini ayrı ayrı test etme. Uygulamada bir bileşen olabilir. Bir bileşen vardır

bağımsız işlevsellik. Eski. amazon.com'da Satıcı işlevi bir bileşen olabilir. Alıcı başka olabilir

bileşen. Ayrıca, Amazon prime videoları başka bir bileşen olabilir.

**28. Duman** Testi

• **TEST sırası:** Kod à Birim Testi à Entegrasyon Testi à Sağlık Testi à ***Duman Testi*** à Fonksiyonel Test

• Projemizde; oturum açma, kullanıcıyı görüntüleme, kullanıcı ayrıntı sayfası, yeni kullanıcı oluşturma ve görev oluşturma

• Bu beş modülde, geliştirici ilk olarak tüm ana işlevleri yürüterek duman testini gerçekleştirecektir.

gibi modüller; kullanıcı, geçerli oturum açma bilgileriyle oturum açabilir veya giremez, oturum açtıktan sonra yeni kullanıcı oluşturulabilir veya oluşturulamaz,

yaratıldı, görüntülendi veya görüntülenmedi vb.

**29. Kara kutu testi nedir? Farklı kara kutu test teknikleri nelerdir?**

• Kara kutu testi, yazılımın iç yapısını bilmeden test etmek için kullanılan yazılım test yöntemidir.

kod veya program.

• Bu test genellikle bir uygulamanın işlevselliğini kontrol etmek için yapılır. Farklı kara kutu test teknikleri;

o Eşdeğerlik Bölümleme o Sınır değer analizi o Neden sonuç grafiği

**30. Denklik bölümleme testi nedir?**

• Denklik bölümleme testi, uygulama giriş testi verilerini her birine bölen bir yazılım test tekniğidir.

test senaryolarının türetilebileceği eşdeğer verilerin en az bir kez bölümlenmesi. Bu test yöntemi ile süreyi kısaltır.

yazılım testi için gereklidir.

• Örnek: Bir not hesaplama sistemini test ederken, test eden kişi 90'dan 100'e kadar olan tüm puanların bir not vereceğini belirler.

A, ancak 90'ın altındaki puanlar olmaz.

• Girdi ve çıktı kapsamını elde etmek için hangi teknik kullanılabilir? İnsan girişine uygulanabilir, arayüzler aracılığıyla giriş yapılabilir

entegrasyon testinde bir sisteme veya arayüz parametrelerine.

**31. Sınır değer testi nedir?**

• Giriş ve çıkış denklik sınıflarının kenarlarında, altında ve üstünde sınır koşullarını test edin.

• Örneğin, maksimum 1000$ ve minimum 100$ çekebileceğiniz bir banka uygulaması diyelim.

değer testi, ortada vurmak yerine sadece kesin sınırları test ediyoruz. Bu, maksimumun üzerinde test ettiğimiz anlamına gelir

limit ve minimum limitin altında.

• Örneğin, kredi kartımın: Aktivasyon tarihi alt sınırdır. Son kullanma tarihi 10/2019 üst sınırdır. 0 dolar daha düşük

harcama sınırı için sınır. 25.000$, harcama limiti için üst sınırdır.

**32. Sınır değer analizi neden iyi test durumları sağlar?**

• Çünkü değerler aralığının 'kenarlarına' yakın farklı durumların programlanması sırasında hatalar sıklıkla yapılır.

**33. Karar tablolarını neden kullanırız?**

• Eşdeğerlik paylaştırma ve sınır değer analizi teknikleri genellikle belirli durumlara veya girdilere uygulanır.

Ancak, farklı girdi kombinasyonları farklı eylemlerin gerçekleştirilmesine neden oluyorsa, bunu kullanarak göstermek daha zor olabilir.

daha çok kullanıcı arayüzüne odaklanma eğiliminde olan eşdeğerlik paylaştırma ve sınır değer analizi.

• Spesifikasyona dayalı diğer iki teknik, karar tabloları ve durum geçiş testi daha çok iş odaklıdır.

mantık veya iş kuralları. Bir karar tablosu, şeylerin kombinasyonlarıyla (örneğin girdiler) başa çıkmanın iyi bir yoludur.

• Bu teknik bazen **'neden-sonuç'** tablosu olarak da adlandırılır . Bunun nedeni, ilişkili bir mantık olmasıdır.

bazen karar tablosunun türetilmesine yardımcı olmak için kullanılan **'neden-sonuç grafiği'** adı verilen diyagram oluşturma tekniği .

- 15 -

**34. Beyaz kutu testi nedir ve beyaz kutu testi çeşitlerini listeler misiniz?**

• Beyaz kutu test tekniği, iç yapının analizine dayalı olarak test senaryolarının seçimini içerir (Kod kapsamı,

bir bileşenin veya sistemin dal kapsamı, yol kapsamı, koşul kapsamı vb.)

• Kod Tabanlı test veya Yapısal test olarak da bilinir. Beyaz kutu testinin farklı türleri

o Açıklama Kapsamı o Karar Kapsamı

**35. Beyaz kutu testinde neyi doğrularsınız?**

• Koddaki güvenlik açıklarını doğrulayın

• Koddaki eksik veya bozuk yolları doğrulayın

• Belge spesifikasyonuna göre yapının akışını doğrulayın

• Beklenen çıktıları doğrulayın

• Uygulamanın tam işlevselliğini kontrol etmek için koddaki tüm koşullu döngüleri doğrulayın

• Satır kodlaması ile satırı doğrulayın ve %100 testi kapsar

**36. Gri Kutu Testi Nedir?**

• Gri kutu testi, kara kutu ve beyaz kutu testinin melezidir.

• Gri kutu testinde test mühendisi, bileşenin kodlama bölümü bilgisine sahiptir ve test senaryoları veya test tasarlar.

sistem bilgisine dayalı veriler.

• Bu test cihazında kod bilgisi vardır, ancak bu beyaz kutu testi bilgisinden daha azdır. Bu bilgiye dayanarak

test senaryoları tasarlanır ve test altındaki yazılım uygulaması bir kara kutu gibi davranır ve uygulamayı şuradan test eder:

dışarıda.

**37. Statik ve dinamik testler arasındaki fark nedir?**

• **Statik test** : Statik test sırasında kod yürütülmez ve yazılım belgeleri kullanılarak gerçekleştirilir.

• **Dinamik test** : Bu testi gerçekleştirmek için kodun yürütülebilir bir formda olması gerekir.

**38. Entegrasyon Testi Nedir?**

• Entegrasyon testi, kara kutu testidir. Entegrasyon testi, birimlerin

belirli bir görevi tamamlamak için birlikte çalışın.

• Entegrasyon testinin amacı, uygulamanın farklı bileşenlerinin birbiriyle etkileşime girdiğini doğrulamaktır.

Test senaryoları, bileşenler arasındaki arayüzleri uygulamak amacıyla geliştirilmiştir.

• Gerçek sonuçlar ve beklenen sonuçlar aynı olduğunda entegrasyon testi tamamlanmış sayılır. Entegrasyon testi yapılır

birim testinden sonra. Entegrasyon testi yapmak için temel olarak üç yaklaşım vardır:

o **Yukarıdan Aşağıya Yaklaşım** à bileşenleri yukarıdan aşağıya entegre ederek test eder.

o **Aşağıdan yukarıya yaklaşım** à Kontrol akışının alt kısmından üst düzey bileşenlere doğru gerçekleşir

o **Büyük patlama yaklaşımı** à Burada farklı modüller bir araya getirilerek eksiksiz bir sistem oluşturulur ve ardından test

üzerinde gerçekleştirilmiştir.

**39. Ölçeklenebilirlik Testi Nedir?**

• Ölçeklenebilirlik testi, kullanıcıların işlevsel ve performans yeteneklerini geliştirmek ve iyileştirmek için gerçekleştirilen testtir.

uygulama. Böylece uygulama son kullanıcıların gereksinimlerini karşılayabilir.

• Yük ve stres koşullarında uygulama performansının değerlendirilmesi yapılarak ölçeklenebilirlik ölçümleri yapılır.

Şimdi bu değerlendirmeye bağlı olarak uygulamanın yeteneklerini geliştiriyor ve geliştiriyoruz.

- 16 -

**40. Depolama Testi Nedir?**

• Depolama Testinde, verileri DB'ye depolamaktan sorumlu olan uygulamanın bu işlevlerini test ederiz.

• Son kullanıcı tarafından GUI'de veya ön uçta girilen veriler, veritabanında depolanan verilerle aynıdır.

• Depolama testi, uygulamanın ön ucundan alınan verilerin doğru yerde ve doğru yerde saklandığını belirler.

veritabanında doğru şekilde.

**41. Stres Testi Nedir?**

• Stres testi, yazılım üzerindeki stresi artırırsak uygulamanın çökmediğini kontrol etmek amacıyla yazılımı test eder.

Uygulama üzerinde çalışan çok sayıda kullanıcıyı artırarak uygulama.

• Uygulamanın üstesinden gelemeyeceği çok sayıda işlemi ateşleyen uygulama stresini de uygulayabiliriz.

• Uygulama üzerinde stres testi gerçekleştiririz, uygulama yeteneklerini belirtilen sınırlarda veya ötesinde değerlendiririz.

belirlemek için gereksinimler.

• Genellikle bu, çok yüksek düzeyde yük ve stres koşullarında gerçekleştirilen bir performans testi türüdür.

**42. Bakım testi nedir?**

• Mevcut yazılımın değiştirilmesi, taşınması veya kullanımdan kaldırılmasıyla tetiklenir.

**43. Test Koşum Nedir?**

• Test donanımı, uygulamayı farklı testlerde çalıştırarak test etmek için gereken bir yazılım ve test verileri koleksiyonudur.

stres, yük, veriye dayalı ve davranışını ve çıktılarını izleme gibi koşullar. Test Harness iki ana parça içerir:

o Test yürütme motoru

o Test komut dosyası deposu

• Otomasyon testi, testlerin yürütülmesini kontrol etmek ve gerçek sonuçları beklenen sonuçlarla karşılaştırmak için bir aracın kullanılmasıdır.

Sonuçlar. Ayrıca test ön koşullarının oluşturulmasını da içerir.

**44. Test kapsamı nedir?**

• Test kapsamı, kaç tane test senaryosuna sahip olduğumuz ve bu test senaryolarının hangi fonksiyonel alanı kapsadığı anlamına gelir.

**45. V-Modeli Nedir?**

• Test faaliyetlerinin yazılım geliştirme aşamalarıyla nasıl bütünleştiğini gösteren bir yazılım geliştirme modeli.

**46. Test yakalama ve tekrar oynatma olanakları sağlayan test araçlarının kullanımından en çok aşağıdakilerden hangisinin yararlanması muhtemeldir?**

• Regresyon testi

• Entegrasyon testi

• Sistem testi

• Kullanıcı Kabul Testi

**47. Kabul testi nedir?**

• Kabul testi, KG testinden sonra yapılacaktır. Şu anki projemde UAT ekibi tarafından yapılıyor. UAT ekibinden sonra

kabul testini gerçekleştirerek kod üretime geçecektir.

o Geliştirme ortamı (geliştiricilerin kod yazdığı ve birim testi yaptığı)

o KG ortamı (uygulamayı test ettiğimiz yer.)

o UAT ortamı (kod QA ortamında test edildikten sonra UAT ortamına dağıtılacaktır. UAT testi

ekip, iş gereksinimlerine uyduğundan emin olmak için testler yapacaktır. Aynı zamanda evreleme ortamı olarak da adlandırılır.

o Üretim ortamı (son kullanıcının gerçek uygulamayı görebildiği zamandır)

- 17 -

**48. UAT (Kullanıcı Kabul Testi) ile Sistem testi arasındaki fark nedir?**

• **Sistem Testi** : Sistem testi, sistem bir bütün olarak teste tabi tutulduğunda hataların bulunmasıdır, **bitiş olarak da bilinir.**

**testi bitirmek için** . Bu tür testlerde uygulama başından sonuna kadar devam eder.

• **UAT** : Kullanıcı Kabul Testi (UAT), bir ürünün aşağıdakileri belirleyen bir dizi özel testten geçirilmesini içerir.

ürünün kullanıcılarının ihtiyaçlarını karşılayıp karşılayamayacağı.

**49. Sürekli entegrasyon nedir?**

• Geliştiriciler, uygulamada yeni kod değişiklikleri yaparken kodları sisteme teslim edebilir ve teslim alabilir.

• Bir geliştirici sisteme yeni bir kod girdiğinde, **Sürekli entegrasyon (CI) sunucusu adı verilen bir sunucu vardır** .

• **CI sunucusu** sürekli olarak yeni kod arıyor . Yeni kod uygulamaya eklendiğinde, CI sunucusu

kodun teslim edildiğini hemen anlayın *. (bununla entegre edilmiş bir araç var, belki Jenkins ya da başka bir şey).*

• Bu araç, uygulamanın temel işlevlerini kontrol etmek için otomatik duman testini başlatır. Sonra hava diyecek

bu kod uygulamayı olumsuz etkiledi veya etkilemedi.

**50. Kod, üretim ortamına nasıl dağıtılır?**

• Yerelden

o Git'e çek ve gönder kullanarak kodu kontrol edin (şirketimde bu SVN'dir)

o Birim testlerini çalıştırın

o Değişiklikleri sunucuya dağıtın

§ Geliştirici kodu her kontrol ettiğinde jenkins tarafından otomatik olarak yapılır.

§ Dev ortamındaki değişiklikleri dağıttıktan sonra

• Geliştirme ortamından

o Jenkins tarafından yapılan Test ortamındaki değişiklikleri dağıtın

o Programlanabilir veya manuel olarak tetiklenebilir

• Jenkins tarafından yapılan Test à Deploy değişikliklerinden

• Ön üretimden

**51. Çevik Çerçeve?**

• **Rol** : PO, SM, Takım

• **Törenler** : -Sprint Planlama, Günlük Scrum, Sprint İnceleme, Sprint Retro, Bakım Seansı

• **Artifacts** : Ürün biriktirme listesi, - Sprint biriktirme listesi, - Tükenmişlik tablosu

**52. Çevik Nedir?**

• Çevik, şelale metodolojisine alternatif olan **yinelemeli** ürün **geliştirme** metodolojisidir.

• Scrum : Ekip, bir sonraki sprint için iş miktarını planlar

• Kanban : Sprint planlaması yok, hikayeler olduğu gibi alınır, ancak yine de her şeye sahipsiniz

**53. Neden Çevik'e ihtiyacımız var? Şelale ve Çevik?**

• Şelale metodolojileri aşağıdaki dezavantajlara sahip olduğundan;

o Gereksinim değiştirilemez veya belge imzalandıktan sonra değiştirilmesi zor değildir.

o Şelalede bir aşamayı tamamlamadan bir sonraki aşamaya geçemezsiniz. Örneğin, kodlama aşaması öncesi

tamamlanmış test başlatılamaz.

o Müşteri, geliştirme yaşam döngüsünün son aşamasına kadar ne elde edeceğini göremez.

o Üretime gitmek daha uzun zaman alır. Ürün pazara girdiğinde, zaten modası geçmiş olabilir.

• Çevik aşağıdaki avantajlara sahiptir:

o Değişiklik memnuniyetle karşılanır. Örneğin, sprint demosundan sonra müşteri bir şeyi beğenmediyse, onların isteklerini alabiliriz.

geri bildirimde bulunun ve ürünü iyileştirin. Gereksinim değişikliği tamam.

o Yinelemeli geliştirme süreci olduğundan, geliştirme ekibi işlevsellik geliştirebilir, geri bildirim alabilir

- 18 -

ve sonraki yinelemeyi iyileştirin. Böylece ürün sürekli olarak geliştirilecektir.

o Scrum master yardımıyla atıklar çevik olarak elimine edilir. Örneğin, engellenirsem bekleyip boşa harcamam gerekmez.

zamanım. Ekip üyeleri birbirleriyle verimli bir şekilde iletişim kurduğundan, engelleyerek daha üretken olabiliriz.

çoğaltılmış çaba

o Şelale, C#\_.NET gibi araçları ve platformu vurgular, ancak çevik insanları vurgular. En iyi araca sahip olabilirsiniz ancak

son insanlar bu araçları kullanıyor. İlham veren insanların daha az şeye sahip olsalar bile harika ürünler yapabileceğine inandım.

para veya daha az kaynak.

**54. Önceki projelerinizde ne tür Agile metodolojisi kullandınız?**

• Extreme programlama(XP), Kanban ve Scrum'ı duydum. Ama sadece scrum ile çalıştım.

**55. Scrum, Çevik bir çerçevedir, değil mi? Diğer birkaç Çevik çerçeveyi adlandırın.**

• Evet, Scrum bir Çevik çerçevedir. Diğer birkaç Çevik çerçeve –Özellik Odaklı Geliştirme Test Odaklı

Geliştirme, Kanban

**56. Scrum'daki farklı roller nelerdir?**

• Ürün sahibi aslında projenin paydaşıdır.

o Proje gereksinimlerini ekip önünde temsil eder.

o Ne inşa edileceğine dair bir vizyona sahip olmaktan ve detaylı vizyonunu ekibe iletmekten sorumludur.

o Agile scrum yazılım geliştirme projesinin başlangıç noktasıdır.

• Scrum takımı, belirli bir amacın gerçekleştirilmesi için performans sergileyen bireylerin toplu katkılarından oluşur.

proje.

o Ekip, talep edilen ürünün zamanında teslimi için çalışmakla yükümlüdür.

• Scrum master – Scrum master, scrum takımının olup olmadığını kontrol eden scrum takımının lideri ve koçudur.

taahhüt edilen görevleri düzgün bir şekilde yürütmek.

o Sprint hedefine ulaşabilmeleri için ekibin verimliliğini ve üretkenliğini artırmaktan da sorumludur.

etkili bir şekilde.

**57. Scrum takımını nasıl tanımlarsınız?**

• 5 rock yıldızını bir araya getirirseniz, bu onların bir takım oldukları veya harika ürünler üretebilecekleri anlamına gelmez.

• Benim için ekip, aynı amacı paylaşan, aynı yönde ilerleyen, birbirine güvenen bir grup insandır.

ve harika bir ürün oluşturmak için birbirleriyle etkili bir şekilde iletişim kuracak ve işbirliği yapacak. yıldız olmamalı

bireysel ama bir yıldız takımı.

**58. Bir Scrum Master'ın sorumlulukları nelerdir?**

• İzleme ve izleme

• Gereksinimleri doğru anlamak

• Proje hedefine ulaşmak için çalışın

• Proses kontrol master ve kalite master

• Takımı müfrezelerden koruyun

• Takımın performansını iyileştirmek

• Toplantıları yönetin ve sorunları çözün

• Çatışmaların ve engellerin çözümü

• İletişim ve raporlama

**59. Negatif test durumu nedir?**

• Negatif test senaryoları, yıkıcı bir şekilde test etme fikrine dayalı olarak oluşturulur. Örneğin, ne olacağını test etmek

uygulamaya uygun olmayan girdiler girilirse. Yanlış giriş bilgileri

- 19 -

**60. “Scrum of Scrums” teriminden ne anlıyorsunuz?**

• Şu anda yedi ekibin üzerinde çalıştığı aktif bir projeyi varsayalım. Her takım kendi ekibine liderlik etmekten sorumludur.

Scrum toplantısı. Ancak farklı ekiplerle koordinasyon ve iletişim için ayrı bir organizasyon yapılması gerekmektedir.

Scrum toplantısı. Her takımdan büyükelçi olarak bilinen ve kendi temsilcisini temsil etmekle sorumlu bir takım lideri vardır.

Scrumların telaşındaki takım.

• **Scrum ekipleri arasında koordinasyonu sağlamak için düzenlenen scrum toplantısı, scrum of scrum olarak bilinir.**

**61. Sevk edilebilir ürün/artış?**

• Ürünün parçası yapılır ve her sprintten ek işlevsellik kazanmaya devam eder.

• Artış, geliştirme ekibinin *Bitti Tanımına uygun olmalıdır*

o Ürün eki teslim edildiğinde “Bitti Tanımı”nı karşılaması gerekir.

o Kabul kriterleri karşılandı

o Ürün sahibi, kullanıcı hikayelerini kabul eder.

• Artış, PO tarafından kabul edilebilir olmalıdır

**62. BurnDown Grafiği Nedir?**

• İşin tamamlanma hızının ve ne kadar işin kaldığını gösteren grafik gösterimi

**63. Doğrulama ve Doğrulama Nedir?**

• Doğrulama, test edenler ve geliştiriciler tarafından geliştirme sırasında gerçekleşir; geliştirme aşamasında yazılımı değerlendirme sürecidir.

aşaması ve belirli bir uygulamanın ürününün belirtilen gereksinimleri karşılayıp karşılamadığına karar vermek.

• Test uzmanları tarafından doğrulama; geliştirme sürecinin sonunda yazılımın değerlendirilmesi ve olup olmadığının kontrol edilmesi sürecidir.

müşteri gereksinimlerini karşılar.

**64. Ready'nin Tanımı Nedir?**

• Kabul Kriterleri temizlendi/incelendi

• Puan/saat verilir

**65. Kullanıcı Hikayesi Nedir?**

• *(Not: temel olarak, bir kullanıcı hikayesi sadece bir gerekliliktir)* Kullanıcı hikayesi, kısa ve basit bir tanım olup, minimum **sevk edilebilir** üründür.

• Normalde şöyle görünür: **<end-user> olarak <benefit>** yapabilmek için **< action>** yapmak istiyorum .

o Amazon kullanıcısı olarak internetten alışveriş yapabilmem için giriş yapabilmeliyim

**66. “Sevk edilebilir” dediniz, bununla ne demek istiyorsunuz?**

• Kullanıcı adı alanına kullanıcı adımı koymak istiyorum diyemezsiniz.

• Böylece kullanıcı adımı oraya yazabilirim. Tam işlevsellik olmalıdır. Kullanıcı adını koymak, gönderilebilir bir işlev değildir.

Ancak giriş yapabilmek tam bir işlevselliktir. Gönderilebilir derken bunu kastediyorum.

**67. Otopark nedir?**

• Agile'da bu şu anlama gelir: Diğer insanlarla gerçekten ilgili olmayan bir sorununuz olduğunda toplantıda tutmamamız gerekir.

diğer insanların zamanını boşa harcadığımız için toplantıda o maddeyi tartışmak.< Hadi onu **otopark** maddesi yapalım > demek

bu konuyla ilgilenen herkes toplantıdan sonra konuşabilir.

**68. Sprint iş akışı nedir?**

• Bir hikayenin yapılacaklar arasında nasıl ilerlediği ve yaşam döngüleri - bir şey engellendiğinde ne olur, vb.

- 20 -

**69. En son projenizdeki çevik deneyim?**

• Sprintimiz 4 haftadır ve her 3 sprint'te bir sürüm döngüsü olarak yayın yaptık

• Ekibimde 7 kişi var. 3 geliştirici (Shwan, Simon, Sinan), 1 otomasyon (Me) ve 1 fonksiyonel test cihazı (Usman),

ayrıca 1 SM (Yasin) ve 1 PO (Simon B.).

• Sprint Planlama Toplantısı ile bir sprint başlatıyoruz ve

o ekibin öncelikli özellikleri ve ürün biriktirme listesi kalemleri hakkında tartışırız ve

o Uygulamanın geliştireceğimiz kısmını öğreniyoruz.

o *Hız* ve *kapasiteye* göre hikaye seçimi

§ **Hız** : Bir sprintte teslim edilen hikaye noktalarının/demo'nun sayısı. Örneğin: ekip 30 hikaye puanı planladıysa

(İş değeri); bir sprintte kullanıcı hikayelerinin değeri ve planlandığı gibi teslim edilebiliyorsa takımın hızı 30'dur

§ **Kapasite** : Bir sprint için mevcut toplam saat sayısı Takımın kapasitesidir. Tatil ve PTO saatlerini hesaplar

o Bu toplantı her hafta yapılır ve yaklaşık 1 saat sürer. Sprint Bakım toplantısını yaptığımızdan daha genel bir fikir ediniriz

görevler için bazı tahminler vermek için.

§ Ekip, SM ve PO bir araya gelerek iş öğelerinin alakalı ve kullanışlı olmasını sağlar

§ Kullanıcı hikayeleriyle ilgili PO'ya sorular sorun

§ Kabul kriterlerini yeniden tanımlayın

§ Yeni hikayeler yazmak

§ Destanları kullanıcı hikayelerine bölmek

§ Doğru tahminde bulunmak/eksik/fazla tahminden kaçınmak için hikayeyi anlayın

**Nasıl tahmin edersiniz?**

Tecrübelerime ve hikayenin karmaşıklığına dayanarak ve daha önce üzerinde çalıştığım bir şey.

• Sprint başladıktan sonra Günlük Standup Toplantısı yapıyoruz

o her gün sabah ve dün ne yaptığımızı, bugün ne yapacağımızı ve herhangi bir engelleyici olup olmadığını tartışıyoruz.

o Sadece sprint hakkındaki bilgileri senkronize ediyoruz.

• Sprint sonu, genellikle Sprint Demo/İnceleme Toplantısı yaparız.

o Sadece müşteriye sprint oluşturduğumuzu göstermek içindir (PO geri bildirimde bulunabilir)

o Ekibimde bir SDET olarak, bazen sunum yaptım ve programdaki işlevleri gözden geçirdim.

konferans odası.

o Müşteri veya paydaşlar ya da iş adamları bilmediklerini soruyorlar.

• Sprint Demo'nun ardından Sprint Retrospektif Toplantısı yapıyoruz.

o Sprint Retro'da son sprintte neyin iyi olduğunu, ne gibi hatalar yaptığımızı konuşuyoruz.

o Bunların üzerinden geçer ve aynı hataları tekrar yapmamaya özen gösteririz.

o İyi bir şey ve iyileştirmeler yapsaydık, yapmaya devam ederdik.

o Sprint gözden geçirme toplantısında veya sprint sonunda yapılan bu toplantı; 2-3 saat sürer.

**70. Destan Nedir?**

• Epic, tek bir sprintte tamamlayamayacağınız büyük bir kullanıcı hikayesidir.

• Örneğin, bir kullanıcı olarak yerel mağazayı ziyaret etmek zorunda kalmamak için çevrimiçi satın almak istiyorum. Bu hikaye çok büyük ve olamaz

bir sprintte tamamlandı. Yani kullanıcı hikayesi yerine Epic diyebiliriz. Aşağıdakiler gibi birden çok kullanıcı hikayesine bölünmelidir:

o Bir müşteri olarak hesabımı görebilmek için giriş yapabilmek istiyorum.

o Bir müşteri olarak, satın alabilmek için bir ürün arayabilmek istiyorum.

o Bir müşteri olarak, satın alacağım ürünün ödemesini yapabilmek için ödemeye devam edebilmek istiyorum.

o Bir müşteri olarak hesabımı koruyabilmek için çıkış yapabilmek istiyorum.

o Gördüğünüz gibi< Bir müşteri olarak satın alabilmek istiyorum...> birden fazla kullanıcı hikayesine bölünebilir. Takım seçebilir

her sprintte bir veya daha fazla kullanıcı hikayesi.

- 21 -

**71. Kabul kriterleri nedir?**

• Kabul kriterleri, kullanıcı hikayesinin başarılı bir şekilde geliştirilip geliştirilmediğini bilmemizin yoludur.

• Bir hikayenin ne zaman "bittiğini" belirlemek için kullanıcının bakış açısından tanımlanan gereksinimler beyanları ve

beklendiği gibi çalışmak

• 3 parça örneği

o Giriş à geçerli e-posta adresi

o Süreç à işaretleme mesajlaşma

o Sonuç à pazarlama mesajı tasarımı, pazarlama tarafından sağlanan özelliklerle eşleşir

**72. Fare deliği nedir?**

• Çevik ekipte çok fazla iletişim olduğundan, ekip birçok şeyi tartışmak zorundadır. Ama bazen

tartışma bir konu için çok uzun sürecek ve gerçekten verimli değil. **<rat deliği>** diyeceğiz, bu da yapmamamız gerektiği anlamına geliyor.

Bu konuyu çok uzun tutmaya devam edin ve ilerleyin.

**73. Ne tür Test vakaları?**

• Farklı senaryoları ele alıyorum

o pozitif

o Negatif

o Sınır Değer Analizi

**74. Test Vakası?**

• Test durumu, Test Edilen Uygulamaya karşı kontrol etmek için özel bir koşuldur. Test adımları, ön koşullar,

test ortamı ve çıktılar.

• Test durumu, işlevselliği ve test adımlarını açıklar.

o Test Durumu Kimliği

o Adım numarası

o İşlevin açıklaması

o Beklenen sonuç

o Gerçek Sonuç

**75. Genellikle bir haftada kaç Test vakası (regresyon takımınızda) tamamlarsınız?**

• 10 küçük test vakası, 7-8 orta, 2-3 büyük

• VEYA Projeye bağlıdır. Şirketimde 2000 test vakamız var. 4Stay'de yaklaşık 700 test vakamız var.

**76. Regresyon takımınızı çalıştırmanız ne kadar sürer?**

• Projeye bağlıdır. Şu anki projemde, regresyon paketindeki 2000 test vakasından 1500 civarında zaten

otomatik. Paralel yürütme gerçekleştirmek için 10 sanal makine kullanırsak, otomatik testin yürütülmesi 2 ila 3 gün sürer

vakalar. Ayrıca, manuel test cihazları bazı manuel test senaryolarını yürütür, ancak ne kadarını yürüttüklerinden emin değilim. inanıyorum

sadece önceliklendirmeden sonra bazı önemli test senaryolarını yürütüyorlar.

**77. Otomatik komut dosyanızı çalıştırdığınızda ne yaparsınız veya regresyon çalıştırdığınızda ne yaparsınız?**

• İlk önce betiğimi çalıştırmam gerekiyor. Komut dosyası yürütme işlemi tamamlandıktan sonra, olup olmadığını görmek için çalıştırma sonucunu analiz etmem gerekiyor.

herhangi bir başarısız test vakası. Başarısız test durumları varsa, meşru uygulama sorunu nedeniyle başarısız olup olmadığını belirlemeliyim.

bazı komut dosyası sorunu nedeniyle. (script, otomasyon kodu sorunu nedeniyle de başarısız olabilir) uygulama sorunundan kaynaklanıyorsa,

El ile çoğaltmaya çalışacağım ve çoğaltabilirsem bir kusur kaydedeceğim. Senaryomdan kaynaklanıyorsa, düzeltmem gerekiyor. Ama bu değil

çoğu zaman olan durumdur.

- 22 -

**78. Otomatikleştirmek için attığınız adımlar nelerdir?**

• İşlevi öğrenin

o Okuma gereksinimleri

o BA ile bilgi transferi oturumu

o Takım arkadaşlarına sor

• El ile test edin

o Her adımı doğru anladığımdan emin olmak

o Beklenen sonuçları anlayın

• Otomatikleştirin

o POM sayfaları oluşturun

§ Kullanacağım gerekli öğeleri/yöntemleri ekleyin ve PageFactory tasarım desenini ekleyin

§ Singleton desenli bir sürücü sınıfı oluşturun

• TestNG İddialarını kullanarak testleri doğrulayın

**79. Otomasyona karşı manuel konumun yüzde kaçı?**

• %80-85 otomasyon %15-20 manuel

**80. Manuel test yerine otomatik testi ne zaman seçersiniz?**

• Test senaryoları yüksek öncelikli test senaryolarıysa.

• İşlevsellik kritik işlevsellik ise.

• Shakeout veya duman testi test durumları.

• Test senaryoları çok uzunsa ve yürütülmesi çok zorsa. Önceliğe dayalı regresyon testi durumları.

• Mümkün olduğu kadar otomasyona geçmeliyiz.

**81. Sprint'inizde ne zaman otomasyon yapıyorsunuz?**

• Geliştiricilerin kendi rolleri bittiğinde

• QA/test ortamına kod dağıtıldığında

• Test çerçevesi kurulduğunda

• Tüm manuel testler yapıldığında

• Duman testleri geçiyor

**82. Test Planı Nedir?**

• Test planı, test kapsamını açıklayan bir word belgesidir.

o Yüksek seviye test döngüsü

o Arıza yaşam döngüsü

o Giriş Kriterleri (teste başlamak için gerekenleri tanımlar)

o Çıkış Kriterleri (testin ne bittiğini tanımlar)

**83. Test planlarındaki tablolar nelerdir?**

• Test tasarımı, kapsamı, test stratejileri, yaklaşımı Test plan belgesinin içerdiği çeşitli detaylardır.

o Test durumu tanımlayıcısı

o Kapsam

o Test edilecek özellikler

o Test edilmeyecek özellikler

o Test stratejisi ve Test yaklaşımı

o Test çıktıları

o Sorumluluklar

o Personel ve eğitim o Risk ve Beklenmedik Durumlar

- 23 -

**84. Test planı ile KG planı arasındaki fark nedir?**

• Bir test planı, ürünü test etmek için ne yapılması gerektiğini ortaya koyar ve hataları ve hataları belirlemek için kalite kontrolünün nasıl çalışacağını içerir.

kusurlar.

• Öte yandan bir KG planı, hataların ve kusurların test edilmesi ve düzeltilmesinden daha çok önlenmesi ile ilgilidir.

**85. Akran değerlendirmesi nedir?**

• Akran değerlendirmeleri, aynı ekipte çalışan kişiler arasında yapılan incelemelerdir. Örneğin, yazılmış bir test vakası

bir KG mühendisi tarafından bir geliştirici ve/veya başka bir KG mühendisi tarafından gözden geçirilebilir.

**86. Bir sistemi veya modülü yeterince test etmek için yeterli test senaryosunun oluşturulduğunu nasıl anlarsınız?**

• Her gereksinimi karşılayacak en az bir test senaryosu olduğunda yeterli test senaryosunun oluşturulduğunu söyleyebilirsiniz.

Bu, uygulamanın tasarlanan tüm özelliklerinin test edilmesini sağlar.

• A2-İşte bu nedenle gereksinim izlenebilirlik matrisine ihtiyacımız var. Kaç gereksinim olduğunu söyleyebiliriz

test senaryolarının kapsadığı ve RTM'den kaç tane kaldığını.

**87. Test senaryolarını kim onaylar?**

• Test senaryolarının onaylayıcısı bir kuruluştan diğerine değişir. Bazı kuruluşlarda, QA lideri aşağıdakileri onaylayabilir:

bir başkası bunları akran incelemelerinin bir parçası olarak onaylarken test senaryoları.

**88. Test planları ve test senaryolarını kim yazıyor?**

• Test planları genellikle kalite güvence lideri tarafından yazılırken, test uzmanları genellikle test senaryoları yazar.

**89. Test tasarım tekniğinin amacı nedir?**

• Test koşullarının belirlenmesi ve Test senaryolarının belirlenmesi.

**90. Test senaryosu ve Test komut dosyası arasındaki fark?**

• Test senaryosu terminolojisi çoğunlukla Manuel Test için kullanılırken Test Komut Dosyası çoğunlukla Otomasyon Testi için kullanılır

• **Test durumu, girdi değerlerini, beklenen çıktıyı ve yürütme için ön koşulları belirten bir belgedir.**

**test.** Aynı zamanda senaryonun nasıl test edileceğine ilişkin alt düzey ayrıntıların bir düzenidir.

• Yazılım testindeki bir test komut dosyası, test edilen sistemde, yazılımın test edilip edilmediğini test etmek için gerçekleştirilecek bir dizi talimattır.

sistem beklendiği gibi çalışır.

**91. Bir test stratejisine neler dahil edilmelidir?**

• Test stratejisi, uygulamanın nasıl test edileceğine ve tam olarak neyin test edileceğine *(kullanıcı arayüzü, modüller,*

*süreçleri vb.)* Test için sınırlar belirler ve manuel mi yoksa otomatik testin mi kullanılacağını belirtir.

**92. Komut dosyası başarısız olduğunda ne yapacaksınız?**

• Tecrübelerime göre, başarısızlığı belirleyeceğim,

o Bunun nedeni uygulama hatası, senkronizasyon hatası, komut dosyası sorunu veya ortam çalışmıyorsa, öncelikle sonucu şu şekilde analiz ederim:

Jenkins aracılığıyla yeniden üret, yalnızca başarısız olanı çalıştır,

o Senkronizasyon sorunundan kaynaklanıyorsa, örtük, açık ve bazı özel beklenenleri kullanarak fazladan zaman ekleyeceğim

koşullar,

o Komut dosyası sorunuysa, komut dosyamda hata ayıklayacağım (tanımlayacağım) ve düzelteceğim, istisnaları analiz edeceğim,

o gerçek kusur ise, o zaman kusuru kaydedeceğim.

- 24 -

**93. Test Senaryosu?**

• Test edilen uygulamanın uçtan uca işlevselliğinin beklendiği gibi çalıştığından emin olun

• Test cihazının, eylemi nasıl kullandıklarını kontrol etmek ve gerçekleştirmek için ayağını son kullanıcıların ayakkabılarına koyması gerekir.

test altındaki uygulama

• TS, kendisiyle ilişkili birçok test senaryosuna sahip olabilir, TS'yi çalıştırmadan önce senaryo için test senaryolarını düşünmemiz gerekir.

• Test Senaryosu: Oturum açma sayfasını doğrulayın

o Test Durumu 1: Geçerli bir kullanıcı adı ve şifre girin

o Test Durumu 2: Parolanızı sıfırlayın

o Test Durumu 3: Geçersiz kimlik bilgilerini girin

• Her test durumunda, yürütme için ayrıntılı adımlar ve koşul bulunur

**94. Gereksinim İzlenebilirlik Matrisi (RTM)**

• RTM, tüm test senaryolarının gereksinimi karşılayıp karşılamadığından emin olmak için kullanılır. Excel sayfası gibidir.

**95. İşlevsel özellikler veya herhangi bir sistem ve geliştirme yoksa bir sistem için test geliştirmek için neler yapılabilir?**

**belgeler?**

• İşlevsel spesifikasyonlar veya sistem geliştirme belgeleri olmadığında, test uzmanı kendilerini tanıtmalıdır.

ürün ve kod ile. Piyasada benzer ürünleri bulmak için araştırma yapmak da faydalı olabilir.

**96. Fonksiyonel test türleri nelerdir?**

• Birim Testi

• Duman testi

• Akıl sağlığı testi

• Entegrasyon Testi

• Sistem Testi

• Regresyon Testi

• UAT (kullanıcı kabul testi)

**97. Akıl sağlığı testi ile duman testi arasındaki fark nedir?**

• Akıl sağlığı testi yapıldığında, ürün, test grubuyla bir ön test turuna gönderilir.

düğme işlevi gibi temel işlevleri kontrol etmek için. Duman testi ise geliştiriciler tarafından gerçekleştirilir.

müşterinin gereksinimlerine göre.

**98. Akıl sağlığı testi hangi adımları içerir?**

• Akıl sağlığı testi, duman testine çok benzer. yapmak için yapılan bir bileşenin veya uygulamanın ilk testidir.

en temel düzeyde çalıştığından ve daha ayrıntılı testlere devam edecek kadar kararlı olduğundan emin olun.

**99. WinRunner ve Rational Robot arasındaki fark nedir?**

• WinRunner işlevsel bir test aracıdır ancak Rational Robot hem işlevsel hem de performans testi yapabilir. Ayrıca,

WinRunner'ın 4 doğrulama noktası ve Rational Robot'un 13 doğrulama noktası vardır.

**100. KG ve test arasındaki fark nedir?**

• KG'nin hedefleri, test etme hedeflerinden çok farklıdır.

• KG'nin amacı hataları önlemektir, uygulama ise testin amacı hataları bulmaktır.

**101. Rastgele testi açıklayın.**

• Rastgele test, uygulamanın rastgele oluşturulan girdi verilerini nasıl işlediğini kontrol etmeyi içerir. Veri türleri

tipik olarak yoksayılır ve veri alanına rastgele bir harf, sayı ve diğer karakterler dizisi girilir.

- 25 -

**102. Kalite Kontrol ve Kalite Güvencesi arasındaki fark nedir?**

• Kalite kontrol (QC) ve kalite güvencesi (QA) yakından bağlantılıdır ancak çok farklı kavramlardır. QC, bir

KG'nin amacı, geliştirme sürecinin, ürünün

sistem veya uygulama gereksinimleri karşılayacaktır.

**103. Bir proje geliştirmede KG'nin rolü nedir?**

• QA ekibi, geliştirme için yürütülecek sürecin izlenmesinden sorumludur.

• QA ekibinin sorumlulukları, test yürütme sürecini planlamaktır.

• QA Lead, zaman çizelgelerini oluşturur ve ürün için bir Kalite Güvence planı üzerinde anlaşmaya varır.

• KG ekibi, KG sürecini ekip üyelerine iletti. QA ekibi, test senaryolarının gereksinimlere göre izlenebilirliğini sağlar.

•

**104. Bir QA veya Test yöneticisini iyi yapan nedir?**

• Yazılım geliştirme süreci hakkında bilgi

• Verimliliği artırmak için ekip çalışmasını iyileştirin

• Yazılım, test ve kalite güvence mühendisleri arasındaki işbirliğini geliştirin

• Kalite Güvence süreçlerini iyileştirmek.

• İletişim becerileri.

• Toplantıları yönetebilme ve onları odaklanmış halde tutabilme

**105. Regresyon testi ile yeniden test arasındaki fark nedir?**

• Regresyon testi, bir modül veya sistemde yapılan değişikliklerin olumsuz bir etkisi olmadığından emin olmak için testler yapıyor

önceki sürümlerde. Yeniden test etme, yalnızca aynı testi yeniden çalıştırmaktır. Regresyon testi yaygın olarak sorulan bir kılavuzdur

mülakat sorularını test etmek ve dolayısıyla bu konuyu anlamak için daha fazla araştırmaya ihtiyaç vardır.

**106. Hata şiddeti ile hata önceliği arasındaki farkı açıklayın.**

• Hata ciddiyeti, hatanın uygulama veya sistem üzerindeki etkisinin seviyesini, hata önceliği ise seviyeyi ifade eder.

bir düzeltme ihtiyacında aciliyet.

• Ağırlık genellikle finansal kayıp, çevreye verilen zarar, şirketin itibarı ve can kaybı olarak tanımlanır.

Bir kusurun önceliği, bir hatanın ne kadar hızlı düzeltileceği ve canlı sunuculara dağıtılacağı ile ilgilidir.

**107. Sistem testi ile entegrasyon testi arasındaki fark nedir?**

• Sistem testi için tüm sistem bir bütün olarak kontrol edilir, entegrasyon testi için ise sistem arasındaki etkileşim kontrol edilir.

bireysel modüller test edilir.

**108. İşlevsel ve yapısal testler arasındaki farkı açıklayın.**

• İşlevsel test, test edenin sistemin veya

Spesifikasyona göre uygulama fonksiyonları. Yapısal testler ise koda veya algoritmalara dayalıdır.

ve beyaz kutu testi olarak kabul edilir.

**109. 109. Pilot ve Beta testi arasındaki fark nedir?**

• Bu ikisi arasındaki farklar aşağıda listelenmiştir:

o Ürün son kullanıcıya sunulmak üzereyken bir beta testi, daha önceki aşamada pilot testler yapılır

geliştirme döngüsünden.

o Beta testinde, uygulamanın kullanıcı gereksinimlerini karşıladığından ve yaptığından emin olmak için birkaç kullanıcıya verilir.

herhangi bir gösterici içermez, pilot test durumunda ekip üyesi kaliteyi iyileştirmek için geri bildirimde bulunur

uygulamanın

- 26 -

**110. Alfa testi nedir?**

• Geliştiricinin sitesinde son kullanıcı temsilcileri tarafından yayın öncesi testi.

**111. Başarısızlık nedir?**

• Başarısızlık, belirtilen davranıştan ayrılmadır.

**112. Test karşılaştırıcıları nelerdir?**

• Bir yazılıma bazı girdiler koyarsanız, ancak yazılımın aşağıdakileri üretip üretmediğine asla bakmazsanız, bu gerçekten bir test midir?

doğru sonuç?

• Testin özü, yazılımın doğru sonucu üretip üretmediğini kontrol etmektir ve bunun için karşılaştırma yapmalıyız.

yazılımın ne üretmesi gerektiğini üretmesi.

• Bir test karşılaştırıcısı, bu karşılaştırmanın yönlerini otomatikleştirmeye yardımcı olur.

**113. Yazılım testi sırasında Risk analizinin nasıl gerçekleştirileceğini açıklayın?**

• Risk analizi, uygulamadaki risklerin belirlenmesi ve test edilmeleri için önceliklendirilmesi sürecidir. Aşağıdakilerden bazıları

riskler:

1. Yeni Donanım.

2. Yeni Teknoloji.

3. Yeni Otomasyon Aracı.

4. Kod tesliminin sırası.

5. Uygulamanın mevcudiyeti

kaynakları test edin.

• Bunları üç kategoride önceliklendiriyoruz:

o Yüksek büyüklük: Hatanın uygulamanın diğer işlevleri üzerindeki etkisi.

o Orta: Uygulamada tolere edilebilir ancak arzu edilmez.

o Düşük: tolere edilebilir. Bu tür bir riskin şirket işi üzerinde hiçbir etkisi yoktur.

**114. İpek Testi Nedir?**

• Silk Test, uygulamanın regresyon ve işlevsellik testlerini gerçekleştirmek için geliştirilmiş bir araçtır. İpek Testi bir araçtır

Pencere, Java, web veya geleneksel istemci/sunucu tabanlı uygulamaları test ederken kullanılır.

• Silk Test, test planının hazırlanmasına ve bu test planlarının yönetilmesine yardımcı olmak,

veri tabanı ve alanın doğrulanması.

**115. Ana Test Planı ile Test Planı arasındaki fark nedir?**

• Ana Test Planı, uygulamanın tüm test ve risk içeren alanlarını içerirken, Test senaryosu belgesi şunları içerir:

test vakaları.

• Ana Test planı, genel geliştirme sırasında yürütülecek her bir testin tüm ayrıntılarını içerir.

uygulama, test planı ise test gerçekleştirmenin kapsamını, yaklaşımını, kaynaklarını ve zamanlamasını tanımlar.

• Master Test planı, uygulamada yapılacak her testin açıklamasını içerirken test planı

yalnızca birkaç test senaryosunun açıklamasını içerir. Birim testi, Sistem testi, beta testi vb. gibi test döngüsü sırasında

• Tüm büyük projeler için Master Test Planı oluşturulur, ancak küçük proje için oluşturulduğunda buna test planı adını verdik.

**116. Bir test ne zaman başarılı sayılır?**

• Testin amacı, uygulamanın gereksinimlere göre çalışmasını sağlamak ve mümkün olduğu kadar çok şey keşfetmektir.

hatalar ve hatalar mümkün olduğunca. Bu, daha fazla işlevi kapsayan ve daha fazla hata ortaya çıkaran testlerin

en başarılı olmak.

**117. Kusur nedir?**

• Beklenen sonuç gerçek sonuçla eşleşmediğinde, bu kusurdur.

**118. Kusur yoğunluğunu tanımla?**

• Hata yoğunluğu, kod satırı başına toplam hata sayısıdır.

- 27 -

**119. Kusur Yaşam Döngüsü (DLC) Nedir?**

• Yeni à Atandı à Açık à Düzeltildi à Yeniden test edildi à Kapat

**120. Kusur kategorileri nelerdir?**

• **Yanlış** : Uygulamada gereksinimler yanlış uygulanmış.

• **Eksik** : Müşteri tarafından verilen gereksinim ve uygulama bu uygulamayı karşılayamadığında.

• **Ekstra** : Son müşteri tarafından verilmeyen, ürüne dahil edilen bir gereklilik. Bu her zaman bir farklılıktır

ancak ürünün kullanıcısı tarafından istenen bir özellik olabilir.

**121. Bir kusur bulduğunuzda ne yapmalısınız?**

• Bir kusur bulursam, bunu bildirmeden önce, bunun geçerli bir kusur olduğundan emin olmak için gereken hatayı yeniden oluştururum.

• Küçük bir sorunsa, geliştirici masasına gideceğim ve o hemen düzeltebilir.

• Büyük bir sorunsa, JIRA'mı açarım ve kusuru kaydederim.

• Hata olup olmadığından emin değilsem KOBİ (konu uzmanı yani uygulamayı bilen kişi demektir) ile görüşeceğim.

hiç yoktan iyidir).

**122. Geliştirici kusur yok diyorsa ne yapmalı?**

• Her zaman gerçek bir kusur olduğundan emin oluyorum, bu yüzden onu yeniden üretiyorum.

• Ekran görüntülerini alıyorum ve kusuru yeniden oluşturmak için tüm adımları veriyorum.

• Aslında şu anki projemde karşılaştığım en büyük zorluklardan biri de bu.

.

**123. Bir programı test edip hataların %100'ünü bulabilir misiniz?**

• Bir uygulamadaki tüm hataları bulmak çoğu zaman imkansızdır çünkü kaç tane hata olduğunu hesaplamanın bir yolu yoktur. Orası

programın karmaşıklığı, programcının deneyimi,

ve benzeri. Bu Kılavuz test mülakat soruları, test uzmanları tarafından düşünülen en zor sorulardır.

**124. Hata ayıklama ve test etme arasındaki fark nedir?**

• Hata ayıklama ve test etme arasındaki temel fark, hata ayıklamanın genellikle bir geliştirici tarafından gerçekleştirilmesidir.

hata ayıklama aşamasında hatalar. Öte yandan test etme, hataları düzeltmek yerine bulur. Bir test cihazı bulduğunda

hata, genellikle bir geliştiricinin düzeltebilmesi için bildirirler.

**125. Testler nasıl yapılmalıdır?**

• Test, uygulamanın teknik gereksinimlerine göre yapılmalıdır.

**126. İyi bir test olarak kabul edilen nedir?**

• Bir nesnenin veya sistemin işlevselliğinin çoğunu kapsayan test, iyi bir test olarak kabul edilir.

**127. Testler ne zaman durdurulmalıdır?**

• Test edilen sistemin risklerine bağlıdır. Testi durdurabileceğiniz bazı kriterler vardır.

o Son Tarihler (Test, Yayın)

o Test bütçesi tükendi

o Hata oranı belirli bir seviyenin altına düşer

o Belirli bir yüzde ile tamamlanan test vakaları

o Testlerin sona ermesi için alfa veya beta dönemleri

o Kodun, işlevselliğin veya gereksinimlerin kapsamı belirli bir noktaya kadar karşılanıyor

- 28 -

**128. Yukarıdan aşağıya ve aşağıdan yukarıya testler arasındaki fark nedir?**

• Yukarıdan **Aşağıya** test, sistemle başlar ve birim düzeyine kadar iner.

• Ters yönde **aşağıdan yukarıya** test kontrolleri, genel sisteme arayüz için birim seviyesi. İkisinin de değeri var ama

Aşağıdan yukarıya testler, genellikle, hataları düzeltme maliyetinin daha düşük olduğu geliştirme döngüsünün başlarında kusurları keşfetmeye yardımcı olur.

**129. Oluşturduğunuz yürütülebilir dosyaların ortalama boyutu nedir?**

• Bu, yürütülebilir dosyalarla olan deneyimimiz hakkında basit bir röportaj sorusudur. Yarattığınız herhangi birinin boyutunu biliyorsanız,

sadece bu bilgiyi sağlayın.

**130. Ön uç ve arka uç üzerinde testler yaptınız mı?**

• Front-End'i test ettiğimde, aslında uygulamayı açıp UI üzerinde test yaparak UI'yi test ediyorum. eğer yaptıysam

Kullanıcı arayüzündeki herhangi bir şey, değişikliğin veritabanında da yapılıp yapılmadığını görmek için arka uç testi yapmam gerekiyor. İçin

Örneğin, bir veli iletişim bilgilerini güncellediğimde veya yeni bir uygulama oluşturduğumda, veritabanına bağlanıyor ve

verilerde değişiklik yapılıp yapılmadığı veya yeni uygulamanın oluşturulup oluşturulmadığı.

**131. Ön Uç Testi ile Arka Uç testi arasındaki fark nedir?**

• Ön Uç Testi, Grafik Kullanıcı Arayüzü üzerinde yapılırken Arka Uç Testi, veritabanlarının test edilmesini içerir.

• Ön uç, kullanıcının etkileşimde bulunabileceği web sitesi görünümünden oluşurken, arka uç olması durumunda gerekli olan veritabanıdır.

verileri depolamak için.

• Son kullanıcı, ön uç uygulamasının GUI'sine veri girdiğinde, girilen bu veri veritabanında depolanır. Kaydetmek

bu verileri veritabanına SQL sorguları yazıyoruz.

**132. Test sırasında bulduğunuz en zor problem nedir?**

• *(Bu, örnek vermeniz gereken basit bir mülakat sorusudur}* . Bu, en zor manuel testlerden biridir.

Cevabınız işinizi belirleyeceği için mülakat soruları. Problem çözme becerilerinizi geliştirecek şekilde cevap vermelisiniz.

ve senin işin. Problem çözme beceriniz ve yeni şeyler öğrenme hevesiniz,

ve işe olan bağlılığınız cevaplarınızla belirtilecektir.

**133. Scrum'da karşılaştığınız zorluk nedir?**

• Scrum, çapraz fonksiyonel ekibi vurguladığından (bu, geliştiricinin test edebilmesi ve test edenlerin de geliştirme yapabilmesi gerektiği anlamına gelir)

Geleneksel bir QA test cihazı olarak geliştirme ekibinin bir parçası olmak zordur. Çünkü genellikle QA'lar nasıl kod yazılacağını bilmiyorlar.

Bu yüzden kendimi çok rekabetçi tutmak zorundayım. Ne zaman vaktim olsa Java gibi daha çok kodlama öğreniyorum.

• Zaman değişikliği sorunu à Veritabanına girdiğim tarihi bir tarih kaydettiğimde, onlar daha önce.

**134. Otomasyon Testi Nedir?**

• İnsan müdahalesini azaltan otomatik test yapma süreci, bu otomasyon testidir.

• Otomasyon testi, QTP, Selenium, WinRunner vb. otomasyon araçları yardımıyla yapılır.

• Otomasyon testinde, uygulamayı test etmek için test komut dosyasını çalıştıran bir araç kullanıyoruz; bu test komut dosyası oluşturulabilir

manuel veya otomatik olarak. Test tamamlandığında, araçlar otomatik olarak test raporunu oluşturur ve rapor eder

**135. Ne zaman otomatikleştireceksiniz?**

• Çok fazla manuel çaba gerektiriyorsa. En az bir kez manuel çalıştırıyorum ve ondan sonra otomatikleştiriyorum.

• Otomasyon, çoğu tekrarlayan işlevsellik için iyidir

**136. Hangi testler otomatikleştirilebilir?**

• Regresyon testleri

• Duman testleri

• Fonksiyonel testler

• API

• Veritabanı

- 29 -

**137. Ne zaman otomatikleştirmeyeceksiniz?**

• İşlev sürekli değişiyorsa

• İşlevsellik tüm proje boyunca yalnızca bir kez kullanılıyorsa

• **Geçici test** otomatikleştirilemez.

**138. Bir hücum sprintinin süresi nedir? Sprintin ne kadar sürüyor?**

• Mevcut projemde script döngümüz 4 haftadır. Buradaki sprintin ne kadar sürüyor? 2 hafta mı 4 hafta mı? (bazen iyidir

soru sormak için. ATM gibi davranmamanız gerektiğini unutmayın. Genelde sadece soruya cevap veren insanları unuturlar. Orası

bir denge olmalıdır.)

• Ekibimiz 7 kişiliktir. 1 SM, 1 PO, 3 geliştirici, 1 MT, 1 AT

**139. Hız Nedir?**

• Hız, takımın sprint baskıda ilerleme hızıdır.

• İki farklı scrum takımıyla karşılaştırılamayacağını da söyleyebilirim.

**engeller hakkında ne biliyorsunuz ? Engellere birkaç örnek veriniz.**

• Engeller, scrum ekibinin karşılaştığı ve çalışma hızını yavaşlatan engeller veya sorunlardır.

• Eğer bir şey scrum ekibinin işi “Bitti” hale getirmesini engellemeye çalışıyorsa, bu bir engeldir.

• Engeller herhangi bir biçimde gelebilir. Bazı engeller şu şekilde verilmiştir:

o Kaynak eksik veya hasta ekip üyesi

o Teknik, operasyonel, organizasyonel sorunlar

o Yönetim destekleyici sistem eksikliği

o İş sorunları

o Hava durumu, savaş vb. gibi dış sorunlar

o Beceri veya bilgi eksikliği

• Çözüm : Takım çalışması, sıkı çalışma, iyi iletişim kurma, çevrimiçi bağlantı, mentorluk ve eğitim

**141. Agile ve Scrum arasındaki fark ve benzerlik nedir?**

• Çevik geniş bir yelpazedir, proje yönetimi için kullanılan bir metodolojidir, Scrum ise Çevik'in sadece bir şeklidir.

süreci ve adımlarını daha kısa bir şekilde açıklar.

• Agile bir uygulama iken, scrum bu uygulamayı sürdürmek için bir prosedürdür.

• à Çevik'in projeleri adım adım veya aşamalı olarak tamamlamayı içermesi benzerliği . Çevik metodoloji kabul edilir

doğada yinelemeli olmak. Çevik'in bir formu olan Scrum, Çevik'inkiyle aynıdır. Aynı zamanda artımlı ve yinelemelidir.

**142. Artış nedir? Açıklamak.**

• Artış, **bir sprint sırasında tamamlanan tüm ürün biriktirme listesi öğelerinin toplamıdır** .

• Her artış, kümülatif olduğu için önceki tüm sprint artış değerlerini içerir.

• Hedefimize ulaşmak için bir adım olduğu için sonraki sürümde mevcut modda olmalıdır.

**143. "Yapı kırıcı" nedir?**

• Yapı kırıcı, yazılımda bir hata olduğunda ortaya çıkan bir durumdur.

• Bu ani beklenmeyen hata nedeniyle, derleme işlemi durur veya yürütme başarısız olur veya bir uyarı üretilir.

• Test edenin sorumluluğu, bu durumda, hatayı ortadan kaldırarak yazılımı normal çalışma aşamasına geri getirmektir.

- 30 -

**144. Günlük stand-up'tan ne anlıyorsunuz?**

• Günlük stand-up, tüm ekibin neredeyse bir saat boyunca bir araya geldiği günlük bir toplantıdır (en çok tercihen sabahları yapılır).

Aşağıdaki üç sorunun cevabını bulmak için 15 dakika –

o Dün ne yapıldı? Bugün için planın nedir?

o Görevinizi tamamlamanızı engelleyen herhangi bir engel veya engel var mı?

• Günlük stand-up, ekibi motive etmenin ve o gün için bir hedef belirlemelerini sağlamanın etkili bir yoludur.

**145. Scrumban hakkında ne biliyorsun?**

• Scrumban, yazılım geliştirme için Scrum ve Kanban tabanlı bir modeldir.

• Bu model, sürekli bakım gerektiren, çeşitli programlama hataları olan veya

bazı ani değişiklikler var.

• Bu model, bir programlama hatası veya kullanıcı hikayesi için bir projenin minimum sürede tamamlanmasını teşvik eder.

**146. Çevik kalite stratejilerinden bazılarını belirtin?**

• Yineleme

• Yeniden düzenleme

• Dinamik kod analizi

• Kısa geri bildirim döngüleri

• İncelemeler ve inceleme

• Standartlar ve yönergeler

• Kilometre taşı incelemeleri

**147. Çevik Manifesto ve İlkeleri hakkında bilginiz var mı? Kısaca açıklayın.**

• Çevik/scrum rolüne aday olan çoğu kişinin uçlarda olması gereken teori budur.

• Bu sorunun bir parçası olarak mümkün olduğunca dört manifesto değeri ve 12 ilke açıklanmalıdır.

• %100 doğru bir şekilde açıklanmasa bile iyi olmalı, ancak değerler ve ilkelerin niyetleri gelmelidir.

örneğin dışarı

• Manifesto

o Süreçler ve araçlar üzerinden bireyler ve etkileşimler

o Kapsamlı dokümantasyon üzerinden çalışan yazılım

o Sözleşme müzakeresi üzerinden müşteri işbirliği

o Bir planı takip ederek değişime yanıt verme

• Yol Gösterici İlkeler

o Müşteri Memnuniyeti

o Karşılama Değişen Gereksinimler

o Çalışan Yazılımlar Sıklıkla Teslim Edilir (Aylar yerine Haftalar)

o İş İnsanları ve Geliştiriciler Arasında Yakın, Günlük İşbirliği

o Proje, güvenilmesi gereken motive olmuş bireyler etrafında inşa edilmiştir.

o Yüz Yüze Konuşma en iyi iletişim şeklidir

o Çalışan yazılım, ilerlemenin birincil ölçüsüdür

o Sürdürülebilir kalkınma, sabit bir tempoyu koruyabilme

o Teknik mükemmelliğe ve iyi tasarıma sürekli dikkat

o Sadelik - Yapılmayan iş miktarını en üst düzeye çıkarma sanatı - esastır

o Kendi kendini organize eden ekiplerden en iyi mimariler, gereksinimler ve tasarımlar ortaya çıkar

o Ekip düzenli olarak nasıl daha etkili olunacağını düşünür ve buna göre kendini ayarlar

- 31 -

**148. Yakma ve yakma çizelgelerinin kullanımı nedir?**

• Yakma çizelgesi bir projede tamamlanan işin miktarını gösterirken, bitme çizelgesi miktarı gösterir

bir projeyi tamamlamak için iş kaldı.

• Bu nedenle, bir projenin ilerlemesini izlemek için tükenmişlik ve tükenmişlik çizelgeleri kullanılır.

**149. Çevik modelin herhangi bir dezavantajı var mı? Öyleyse, açıkla.**

• Evet, Çevik modelin bazı dezavantajları var, bazıları şöyle;

o Bir görevi tamamlamak için gereken çaba hakkında bir tahminde bulunmak kolay değildir. durumunda daha sorunlu hale gelir.

Gereken toplam çaba hakkında bir fikir edinmek zorlaştıkça büyük projeler.

o Bazen, projenin tasarımına ve dokümantasyonuna gerektiği gibi odaklanmak mümkün değildir.

o Müşterinin gereksinimlerinin tam olarak anlaşılmaması durumunda nihai proje müşteriyi karşılamayacaktır.

Gereksinimler. Böylece müşteri memnuniyetsizliğine yol açacaktır.

o Yalnızca Çevik metodolojilerde önemli deneyime sahip lider önemli kararlar alabilir. bu

Çok az deneyime sahip veya hiç tecrübesi olmayan ekip üyeleri karar verme sürecine dahil değildir, bu nedenle ilerleme şansları yoktur.

onların bilgisi.

**150. Çevikte Sıfır Sprint ve Spike tanımlayın.**

• Sıfır Sprint, Agile'da ilk sprintin hazırlık aşaması olarak tanımlanabilir.

o Projeye fiilen başlamadan önce yapılması gereken bazı faaliyetler vardır.

o Bu aktiviteler Sıfır sprint olarak kabul edilir; bu tür faaliyetlerin örnekleri şunlardır:

geliştirme, biriktirme listelerinin hazırlanması vb.

• Spike, sprintler arasında alınabilecek hikaye türüdür.

o Sivri uçlar genellikle tasarımla ilgili faaliyetler veya araştırma, tasarım,

prototipleme ve keşif.

o İki tür sivri uç vardır – fonksiyonel sivri uçlar ve teknik sivri uçlar.

**151. Scrum Master'ın rolü nedir?**

• Scrum ustası, Scrum takımının hem lideri hem de koçudur.

• SM, takıma hizmet etmek ve performansı etkileyebilecek her türlü bloktan korumaktan sorumludur.

• SM'nin ana rolü, takımını sprint hedefine ulaşmak için motive etmektir.

• Her üyenin aşağıdakilerin uygulanmasına aşina olduğu, kendi kendini organize eden ve motive olmuş bir ekip oluşturmaya odaklanmıştır.

Çevik ve Scrum ilkeleri ve uygulamaları.

• SM, taahhüt edilen görevleri düzgün bir şekilde yürütüyorlarsa, scrum ekibi üzerinde uygun bir kontrol sağlar.

• Sprint hedefine ulaşabilmeleri için ekibin verimliliğini ve üretkenliğini artırmaktan da sorumludur.

etkili bir şekilde.

**152. Scrum'daki bir hikaye noktası hakkında ne biliyorsunuz?**

• Scrum'daki bir hikaye noktası, belirli bir işlemi gerçekleştirmek veya tamamlamak için gereken toplam çabaların tahmini için birimdir.

görev.

**153. Scrum metodolojisinde Sashimi'nin rolü nedir?**

• Sashimi, Scrum metodolojisinde önemli bir rol oynar.

• Sashimi, geliştiriciler tarafından oluşturulan tüm fonksiyonların tamamlandığını kontrol etmek için Scrum tarafından kullanılan bir tekniktir.

• Bu tekniği kullanarak, kullanılan analiz, tasarım, kodlama, test ve dokümantasyon gibi tüm gereksinimler,

bir ürünün yapısı kontrol edilir ve ancak bundan sonra ürün görüntülenir.

- 32 -

**154. Çevik test teriminden ne anlıyorsunuz?**

• Çevik test, tamamen yazılım geliştirmenin çevik ilkelerine dayanan bir yazılım testi uygulamasıdır. O bir

gereksinimlerin ürün sahibi ve ekip arasındaki işbirliğinin sonucu olduğu yinelemeli metodoloji.

Çevik ilke ve uygulamalar, müşteri gereksinimlerini karşılamak için başarıyla tamamlanarak uygulanır.

proje.

**155. Scrum yerine şelale kullanılması önerildi mi? Evet ise, ne zaman olduğunu açıklayın.**

• Evet, bazen Scrum yerine şelale modeli kullanılması önerilir.

• Müşteri gereksinimlerinin basit, iyi tanımlanmış, tam olarak anlaşılmış, öngörülebilir ve tabi olmadığı durumlarda yapılır.

Projenin tamamlanmasına kadar değişiklik.

**156. Scrum, projeler için otomatik test kullanımını neden teşvik ediyor?**

• Scrum, en hızlı hale getirmek için otomatik (otomatik performans veya otomatik regresyon) testinin kullanılmasını teşvik eder.

projenin olası teslimatı . *otomasyon için kullandığınız bazı araçları açıklayabilirsiniz.*

**157. Çevik için bazı yaygın matrisleri açıklayın.**

• **Velocity** à Velocity, son 3-4 sprintteki ortalama puan sayısıdır. Hepsinin toplamı ile ölçülür.

hikayelerin onaylanmış tahminleri. Kapasite, ilerleme vb. hakkında bir fikir verir.

• **Kümülatif Akış Şeması** à Bunun yardımıyla tek tip iş akışı üzerinde bir inceleme yapılır. Bu diyagramda/grafikte,

x ekseni zamanı temsil ederken, y ekseni çaba sayısını temsil eder.

• **İş Kategorisi Tahsisi** à zaman yatırımı hakkında hızlı bilgi veren önemli bir faktördür, yani nerede

zaman harcanıyor ve bir zaman faktörü olarak hangi göreve öncelik verilmesi gerektiği.

• **Zaman Kapsamı** à Test sırasında bir koda verilen zamandır. faktörü olarak yüzde olarak hesaplanır.

test takımı tarafından çağrılan kod satırı sayısı ve ilgili kod satırlarının toplam sayısı.

• **Sağlanan İş Değeri** à Ekibin çalışma verimliliğini ifade eden bir terimdir. İş hedefleri

öncelik, karmaşıklık ve yatırım getirisi düzeyine göre atanan sayısal değerler 1,2,3 vb.

• **Kusur Giderme Farkındalığı** à Ekibin kaliteli bir ürün sunmasına yardımcı olan faktördür. Bir kimlik

aktif kusur sayısı, bunların farkındalığı ve ortadan kaldırılması, yüksek kaliteli bir ürün sunmada önemli bir rol oynar.

• **Hata Çözüm Süresi** à Ekip üyelerinin kusurları (hataları) tespit ettiği ve bir öncelik belirlediği bir prosedürdür.

kusur çözümü için. Hataları/hataları düzeltme veya kusur çözme prosedürü, aşağıdakiler gibi birden fazla süreçten oluşur:

kusur resminin temizlenmesi, kusur tespitinin planlanması, kusur tespitinin tamamlanması, oluşturulması ve çözümün ele alınması

bildiri.

• **Sprint Burndown Matrisi** à Sprint Burndown tablosu, uygulanmayan veya

Scrum döngüsü sırasında uygulanan sprintler. Bu matris, sprint ile tamamlanan işi takip etmeye yardımcı olur.

**158. Çevik modeli kullandığınız bazı metodolojileri ve geliştirmeleri adlandırın.**

• Çevik modelin kullanılabileceği bazı metodolojiler ve geliştirmeler şunlardır:

o Kristal metodolojileri

o Yalın yazılım geliştirme

o Dinamik geliştirme ve Özellik odaklı geliştirme

**159. Scrum M/Ürün O/Agile ekip üyesi olarak deneyiminizi paylaşın ve birincil sorumluluklarınız nelerdi?**

• Bu sorudaki püf nokta, anlatırken kendi kendini organize eden ve motive eden bir ekip gösterip göstermediğinizdir.

**160. Projenizdeki sprintlerin/yinelemelerin uzunluğu ne kadardı?**

• Buradaki fikir, ne tür bir ortamda çalıştığınızı yargılamaktır. gibi bir takip sorusu kesinlikle olacak

bu uzunluk başlangıçta sabit miydi ve hiç değişmedi mi? Bu uzunluktan daha uzun veya bundan daha az denediniz mi?

- 33 -

**161. “Planning Poker” tekniği hakkında ne biliyorsunuz?**

• Scrum Poker olarak da bilinen planlama pokeri, planlama ve tahmin için kullanılan kart tabanlı çevik bir tekniktir. İle

Poker tekniği planlama bir oturum başlatın, çevik kullanıcı hikayesi ürün sahibi tarafından okunur.

• Poker planlama tekniğinde gerçekleştirilen adımlar şunlardır:

o Her tahmincinin, hikaye noktalarını belirtmek için 0, 1, 2, 3, 5 vb. değerlere sahip bir deste poker kartı vardır.

gün veya ekibin tahmin için kullandığı başka bir şey.

o Her tahminci, ürün sahibiyle bir görüşme yapar ve daha sonra kendi özelliklerine göre özel olarak bir kart seçer.

bağımsız tahmin.

o Tüm tahminciler tarafından aynı değere sahip kartlar seçilirse, tahmin olarak kabul edilir. Değilse, tahmin edici

tahminlerinin yüksek ve düşük değerini tartışır.

o Sonra yine her tahminci özel olarak bir kart seçer ve ortaya çıkarır. Bu poker planlama süreci, bir sonuca ulaşmak için tekrarlanır.

Genel Anlaşma.

**162. Projelerinizde kullanıcı hikaye haritalama ve hikaye tahminini nasıl yaptınız?**

• Poker planlama, tişört, beden ölçülendirme gibi herhangi bir tahmin tekniği kullandınız mı? Hangi tekniği kullanırsan kullan

proje sadece çok net bir şekilde bahseder.

**163. Çevik test metodolojisinin diğer test metodolojilerinden farkı nedir?**

• Çevik test metodolojisi, tüm test sürecinin birden çok küçük kod segmentine bölünmesini içerir. İçinde

her adımda, bu kod segmentleri teste tabi tutulur.

• Ekip iletişimi gibi çevik test metodolojilerinde yer alan bir dizi ek süreç vardır.

optimal sonuçlar için stratejik değişiklikler ve diğerleri.

**164. Scrum ekip üyelerini idare ederken projenizde karşılaştığınız en büyük zorluk nedir?**

• Scrumın ilk aşamalarında genellikle karşılaşılan zorluklar, hızı dengelemek, ekip üyeleri çatışıyor,

zaman boksu vb.

o Uygulama test edilecek kadar kararlı olmalıdır. o Her zaman zaman kısıtlaması altında test etme

o Gereksinimleri anlamak.

o Alan bilgisi ve iş kullanıcı perspektifi anlayışı.

**165. İlk önce hangi testler yapılmalı?**

• Komple Uygulamanın Test Edilmesi. o Regresyon testi.

• Nitelikli test uzmanlarının olmaması. o Değişen gereksinimler.

• Kaynak, araç ve eğitim eksikliği

**166. Scrum Master sertifikanız var mı?**

• Sertifikalı bir scrum ustasıysanız, sertifika sınavı, alınan puan ve sertifika gibi sertifikanızın ayrıntılarını paylaşmanız yeterlidir.

sertifika sınavını geçme yılı. Bir sertifikanız yoksa, deneyiminizden bahsedin ve vurgulayın.

özel alan. Ayrıca, yakın gelecekte sertifikaya yatırım yapmayı planlıyorsanız görüşmeciye bildirin.

**167. Çevik sertifikanız var mı? Neden bu sertifikayı seçtiniz?**

• Bir projeyi en kısa sürede tamamlamak için Çevik ve Scrum metodolojileri kullanılır.

• Çevik ilkelerin uygulanması müşteri memnuniyeti ile sonuçlanırken, scrum esnek özelliği ile bilinir.

Gereksinimler.

- 34 -

**168. Daha önce offshore ekibiyle çalıştınız mı?**

• Hayır, çalışmıyorum...

• Offshore, temel olarak ekibin farklı bir ülkede bulunmasına rağmen hala şirketiniz tarafından istihdam edildiği anlamına gelir.

**169. Ortak UI test otomasyon araçları nelerdir?**

• Selenyum

o salatalık

o TestNG

• Appium

• İletki

• Winium

• UFT/QTP

• **Katalon Stüdyosu**

**170. Test Yazılımı Nedir? Test eşyası mı?**

• Uygulamanın test edilmesinde yardımcı olan yazılımın alt kümesidir.

• Test yazılımları, testleri planlamak, tasarlamak ve yürütmek için gereklidir. Belgeler, komut dosyaları, girdiler, beklenen sonuçlar, kurulum içerir

ve testte kullanılan ek yazılım veya yardımcı programlar.

• Test yazılımı, bir yazılım paketini test etmek için gerekli olan tüm yardımcı programlar ve uygulama yazılımlarının birleşimine verilen terimdir.

Özeldir çünkü;

o Farklı amaç

o Kalite için farklı metrikler ve

o Farklı kullanıcılar

**171. İstemci veya sunucu ortamı testi nasıl etkiler?**

• Veri aktarımı veri aktarım hızı, donanım ve donanım gibi testi etkileyen birçok çevresel faktör vardır.

sunucu vb. istemci veya sunucu teknolojileri ile çalışırken kapsamlı testler yapılacaktır.

• Zaman sınırımız olduğunda entegrasyon testi yapıyoruz. Çoğu durumda yük, stres ve performansı tercih ederiz.

İstemci veya sunucu ortamı için uygulamanın yeteneklerini incelemek için testler.

- 35 -

**JAVA**

**1. Java Sanal Makinesi**

• JVM, derlenmiş java sınıfı dosyaları için bir çalışma zamanı ortamı olan Java Virtual Machine'in kısaltmasıdır.

**2. JavaScript ve Java aynı mı?**

• Java bir OOP programlama dilidir, Java Script ise bir OOP betik dilidir.

• Java, bir sanal makinede veya tarayıcıda çalışan uygulamalar oluştururken, JavaScript kodu yalnızca bir tarayıcıda çalıştırılır.

• JavaScript kodunun tamamı metin içindeyken Java kodunun derlenmesi gerekir.

• Farklı eklentiler gerektirirler.

**3. Java Çalışma Zamanı Ortamı**

• JRE, bir Java programını çalıştırmak için ihtiyacımız olan şeydir ve JVM'nin çalışma zamanında kullandığı kitaplıklar ve diğer dosyaları içerir.

• JRE = JVM + Kütüphane Sınıfları

**4. Java Geliştirme Kiti**

• JDK, Java kaynak kodunu derlemek için ihtiyacımız olan şeydir ve JRE, geliştirme araçlarını içerir.

• JDK = JRE + Geliştirme araçları

**5. Nesneye Yönelik Programlama (OOP)**

• OOP, eylemler yerine nesne etrafında organize edilmiş bir programlama dili modelidir;

o Geliştirme ve bakımı kolaylaştırır - Veri gizleme sağlar - Gerçek dünyayı simüle etme yeteneği sağlar.

• **OOP dili 4 ilkeyi takip eder** :

o **Encapsulation** : Private key kullanarak verilere doğrudan erişimi gizleyebiliriz ve getter kullanarak özel verilere erişebiliriz.

ve ayarlayıcı yöntemi.

o **Soyutlama** : Uygulama detaylarını gizleme ve kullanıcıya sadece işlevselliği gösterme işlemidir. Soyutlama

nesnenin nasıl yaptığına değil ne yaptığına odaklanmanızı sağlar.

o **Kalıtım** : İki sınıf arasındaki ilişkiyi tanımlamak için kullanılır. Bir alt sınıf tüm özellikleri edindiğinde ve

kalıtım olarak bilinen ebeveyn sınıfının davranışları. Alt sınıf, ebeveyn sınıfta yazılan tüm kodları yeniden kullanabilir. sağlar

kodun yeniden kullanılabilirliği.

o **Polimorfizm** : Nesnenin çoklu biçimde davranma yeteneğidir. Polimorfizmin en yaygın kullanımı Java'dır.

bir alt sınıf nesnesine başvurmak için bir üst sınıf referans türü değişken kullanıldığında.

§ Örn: WebDriver sürücüsü = yeni ChromeDriver(); Elde etmek için yöntem aşırı yükleme ve geçersiz kılma kullanıyoruz

polimorfizm.

**6. Soyutlama kavramı nedir?**

• OOP'de soyutlama, uygulama ayrıntılarını kullanıcıdan gizleme işlemidir, yalnızca işlevsellik sağlanacaktır.

kullanıcıya.

• Başka bir deyişle, kullanıcı nesnenin nasıl yaptığından çok ne yaptığı hakkında bilgi sahibi olacaktır.

• Java'da soyutlama, Soyut sınıflar ve arabirimler kullanılarak sağlanır.

• Örneğin: çevrimiçi olarak banka hesabınıza giriş yaptığınızda, kullanıcı kimliğinizi ve şifrenizi girip giriş tuşuna basarsınız. Ne

o zaman, sunucuya gönderilen girdi verilerinin nasıl doğrulandığı, bunların tümü sizden soyutlanır.

- 36 -

**7. Soyutlama ve Kapsülleme Arasındaki Fark ?**

• **Soyutlama** , nesnenin **nasıl yaptığına değil, ne** yaptığına odaklanmanızı sağlar .

o **Kapsülleme** , nesnenin bir şeyi nasıl yaptığının dahili ayrıntılarını gizlemek anlamına gelir.

• **Soyutlama , istenmeyen verileri** gizlemek ve ilgili verileri vermek için kullanılır.

o **Kapsülleme** , kodun ve verilerin gizlenmesi ve verilerin dışarıdan korunması anlamına gelir.

• **Soyutlama** , Soyut sınıf ve Arayüzler kullanılarak gerçekleştirilebilir

o **Kapsülleme** , “private” anahtar sözcüğü kullanılarak gerçekleştirilebilir.

**8. Kapsülleme nedir ve nasıl kullandınız?**

• Veri: değişkenleri özel yaparak ve genel alıcı ve ayarlayıcı yöntemleri sağlayarak gizleme.

• Projemde test verilerini ve gerçek verileri yönetmek için birden fazla POJO/BEAN sınıfı oluşturdum.

§ ÖRNEK: API yanıtından JSON alıyorum ve POJO sınıfımın nesnesine dönüştürüyorum, tüm değişkenler alıcılarla özeldir ve

pasör.

**9. Soyut Sınıf ve Arayüz Arasındaki Fark?**

• Abstract anahtar sözcüğü ile tanımlanan bir sınıf, soyut sınıf olarak bilinir. Soyut ve soyut olmayan yöntemlere sahip olabilir.

• Arayüz, bir sınıfın planıdır. Bu bir şablondur ve interface anahtar sözcüğü ile bildirilir. Soyut yöntemlere sahip olabilir,

varsayılan yöntemler, statik yöntemler ve genel nihai statik değişkenler

• Abstract sınıfını kullanmak istediğimizde “extend” anahtar sözcüğünü kullanırız. Arayüz kullanmak istediğimizde “implement” kullanıyoruz.

anahtar kelime.

• Soyut sınıf ve arabirimin her ikisi de soyutlamaya ulaşmak için kullanılır Her ikisi de somutlaştırılamaz; bir nesne oluşturamayız.

**10. Değişmez nedir?**

• Immutable, bir nesnenin yapıcısının yürütmeyi tamamladığında o örneğin değiştirilemeyeceği anlamına gelir.

• Bu kullanışlıdır, çünkü başka birinin geleceğinden endişe etmeden etraftaki nesneye referanslar iletebilirsiniz.

içeriğini değiştir.

Özellikle eşzamanlılık ile uğraşırken, asla değişmeyen nesnelerle kilitleme sorunu yoktur.

sınıf Foo {

özel final String myvar ;

public Foo (son Dize initialValue) {

bu . *myvar* = initialValue ;

}

Genel Dize getValue () {

iade et . *myvar* ;

}

**11. Polimorfizm Nedir?**

• Polimorfizm OOP'de çok önemli bir kavramdır çünkü;

o Çağrının yapıldığı nesneye bağlı olarak uygulamaların çalışma zamanındaki davranışını değiştirmeyi sağlar.

olur.

o Polimorfizm ile; bir nesnenin farklı biçimleri olabilir

• Statik olan **Derleme Zamanı** ve çocuk ve ebeveyn sınıfı ile ilgili **Çalışma Zamanı** Polimorfizmi olmak üzere iki tür .

• Çok biçimlilik, Yöntem aşırı yüklemesi ve yöntemi geçersiz kılma kavramı kullanılarak uygulanır. Bu sadece olabilir

sınıflar kalıtım kullanarak ebeveyn ve çocuk ilişkisi altındayken.

- 37 -

**12. Polimorfizm ve Kalıtım Arasındaki Fark**

• Gerçek dünyada olduğu gibi Kalıtım, iki sınıf arasındaki ilişkiyi tanımlamak için kullanılır. Baba-Oğul benzer

ilişki. Java'da, Ana sınıfımız (süper sınıf olarak da bilinir) ve alt sınıfımız (alt sınıf olarak da bilinir) vardır. Benzer

gerçek dünyada, Çocuk, Ebeveynlerin niteliklerini, yöntemlerini ve kodlarını devralır.

o Bir alt sınıf, Üst sınıfta yazılan tüm kodları yeniden kullanabilir ve yalnızca, davranıştan farklı olan davranış için kod yazabilir.

Ebeveyn.

o Kalıtım aslında kodun yeniden kullanımı içindir.

• Öte yandan, Polimorfizm, nesnenin çoklu biçimde davranma yeteneğidir.

o Aşırı yükleme ve geçersiz kılma olarak sınıflandırılır.

• Bu arada, aslında birbirleriyle ilişkilidirler, çünkü Polimorfizmi mümkün kılan kalıtımı,

iki sınıf arasındaki herhangi bir ilişki. Polimorfik kod yazmak mümkün değildir.

o Dinamik Polimorfizm à Geçersiz Kılma

o Statik Polimorfizm à Aşırı Yükleme

**13. Statik bağlama ile dinamik/çalışma zamanı bağlaması nedir?**

• Statik bağlama aşırı yüklemedir ve dinamik bağlama, yöntem aşırı yüklemesidir

**14. Erişim değiştiricisi nedir ve farklı erişim değiştiricileri nelerdir?**

• Java, sınıflar, değişkenler, yöntemler ve kurucular için erişim düzeylerini ayarlamak için bir dizi erişim değiştiricisi sağlar.

o Pakete görünür, varsayılan. Değiştiricilere gerek yoktur.

o Yalnızca sınıf tarafından görülebilir (özel).

o Dünya tarafından görülebilir (genel).

o Pakete ve tüm alt sınıflara görünür (korumalı).

**15. Java'da Genel, Özel ve Korumalı değiştirici arasındaki fark?**

• Java'da, sınıfın, yöntemlerin ve değişkenlerin erişilebilirliğini belirten erişim değiştiricisi. Dört erişim değiştiricisi vardır

Java yani Genel, Özel, Korumalı ve Varsayılan.

• Bu erişim değişiklikleri arasındaki fark şudur;

o En önemlisi erişilebilirlik düzeyidir.

o Herkese açık her yerden erişilebilir

o Private'a yalnızca belirtilen sınıftan erişilebilir

o Varsayılana yalnızca aynı paket içinde erişilebilir

o Protected'a aynı paket içinde ve paketin dışından erişilebilir, ancak yalnızca alt sınıflar.

• Özel veya korumalı değiştiriciyi üst düzey bir sınıfla kullanamayız.

• Erişim değiştiricisinin Java'da public, private veya protected yerel değişkeni için uygulanamayacağını da unutmamalıyız.

**16. Bir Yapıcı ve Yöntem Arasındaki Fark?**

• Yapıcının bir dönüş tipi yoktur ve yapıcının adı sınıf adıyla aynı olmalıdır.

o Oluşturucu, yeni bir nesne oluşturulduğunda otomatik olarak çağrılır. Yapıcı dolaylı olarak çağrılır.

o Herhangi bir kurucumuz yoksa Java derleyicisi varsayılan bir kurucu sağlar.

o Yapıcılar alt sınıflar tarafından miras alınmaz

• Yöntemin bir dönüşü vardır ve yöntemin adı sınıf adıyla aynı olabilir veya olmayabilir

o Yöntem açıkça çağrılır.

o Yöntem hiçbir durumda derleyici tarafından sağlanmaz.

o Yöntemler, alt sınıflar tarafından miras alınır.

- 38 -

**17. Yöntem Aşırı Yükleme ve Yöntem Geçersiz Kılma arasındaki fark?**

• Aşırı yükleme ve geçersiz kılma arasındaki ilk ve en önemli fark şudur:

o Aşırı yükleme durumunda metot adı aynı olmalı, ancak parametreler farklı olmalıdır;

o geçersiz kılma durumunda, yöntem adı ve parametreler aynı olmalıdır

• Yöntem aşırı yükleme ve geçersiz kılma arasındaki ikinci büyük fark şudur;

o Yöntemi aynı sınıfta aşırı yükleyebiliriz ancak kalıtım olan iki sınıfta yöntem geçersiz kılma gerçekleşir

ilişki.

• Java'da statik, nihai ve özel yöntemi geçersiz kılamayız, ancak Java'da statik, nihai ve özel yöntemi aşırı yükleyebiliriz.

• Metod aşırı yüklemesinde dönüş tipi aynı veya farklı olabilir. Yöntem geçersiz kılmada, dönüş türü aynı veya kovaryant olmalıdır

tip.

**18. Java'da Küme, Liste ve Harita Arasındaki Fark?**

• Set, List ve Map, Java toplama çerçevesinin 3 önemli arayüzüdür.

o Liste, *çoğaltma içerebilecek sıralı* ve dizine alınmış koleksiyon sağlar .

o Set, benzersiz nesnelerin *sırasız* koleksiyonunu sağlar. Küme *, çoğaltmaya izin vermiyor* . Liste ve Set'in her ikisi de genişletilir

toplama arayüzü

o Harita, Anahtar Değere dayalı bir veri yapısı sağlar. Anahtar her zaman benzersizdir, değer çift olabilir.

**19. Liste, Ayar ve Harita ne zaman kullanılır?**

• İndeks kullanarak öğelere sık sık erişmemiz gerekiyorsa, o zaman List, gitmenin bir yoludur ArrayList, eğer istersek daha hızlı erişim sağlar.

indeksi bil.

• Öğeleri depolamak istiyorsak ve bir sırayı korumalarını istiyorsak, o zaman tekrar Liste'ye gidin. Liste sıralı bir koleksiyondur ve

düzeni sağlamak.

• Benzersiz öğeler koleksiyonu oluşturmak istiyorsak ve herhangi bir Set uygulamasını seçmek yerine herhangi bir kopya istemiyorsak.

(HashSet...)

• Verileri Anahtar ve Değer biçiminde depolamak istiyorsak, Harita'dan gidilecek yol budur. HashMap, Hashtable arasından seçim yapabiliriz...

**20. ArrayList'in kopya içerip içermediğini nasıl anlarsınız?**

• Birkaç yol mevcuttur. En kısa olanı .stream().distinct().count() yöntemidir .

list.size() != list.stream().distinct().count()

• Diğer yöntemler:

// YÖNTEM 1

public static <T> boolean includeUnique( List <T> list){ Set <T> set = new HashSet <>();

dönüş list.stream().allMatch(t -> set.add(t));

}

//YÖNTEM 2

public static <T> boolean includeUnique ( List <T> list){ return list.stream().allMatch( new

HashSet <>()::add);

} // yalnızca saf akışları işleyebildiği için değil, aynı zamanda durduğu için de en iyisi gibi görünüyor

ilk kopya (1 ve 2 numara her zaman sonuna kadar yinelenir)

//YÖNTEM 3

genel statik <T> boolean içerirUnique( Liste <T> listesi){

Set <T> set = new HashSet <>();

için (T t: liste){

if (!set.add(t))

yanlış döndür ; }

- 39 -

**21. Dizi Nedir?**

• Dizi, tek tipte sabit sayıda değer tutan bir kapsayıcı nesnedir. Bir dizinin uzunluğu belirlenir

dizi oluşturulduğunda. Oluşturulduktan sonra uzunluğu sabittir. Zaten ana dizilerde bir dizi örneğini gördünüz.

"Merhaba Dünya!" yöntemi başvuru. Bu bölümde diziler daha ayrıntılı olarak ele alınmaktadır.

• Bir dizideki her öğeye öğe denir ve her öğeye sayısal indeksi ile erişilir. Daha önce gösterildiği gibi

resimde, numaralandırma 0 ile başlar. Örneğin 9. öğeye bu nedenle dizin 8'den erişilebilir.

• **Java Dizisinin Avantajı**

o Kod Optimizasyonu: Kodu optimize eder, verileri kolayca alabilir veya sıralayabiliriz.

o Rastgele erişim: Herhangi bir indeks pozisyonunda bulunan herhangi bir veriyi alabiliriz.

• **Java Dizisinin Dezavantajı**

o Boyut Sınırı: Dizide yalnızca sabit boyuttaki öğeleri saklayabiliriz. Çalışma zamanında boyutunu büyütmez. Bunu çözmek için

sorun, java'da toplama çerçevesi kullanılıyor.

**22. Java'da Arrays ve ArrayList arasındaki fark?**

• Dizi, temel Java programlamanın bir parçasıdır ve özel sözdizimine sahiptir ArrayList, toplama çerçevesinin ve uygulamanın bir parçasıdır

Liste arayüzü

• En büyük fark şudur; Dizi sabit uzunluklu bir veri yapısıdır, bu nedenle oluşturulan Array'in uzunluğunu değiştirebiliriz, ArrayList

yeniden boyutlandırılabilir.

• Diğer önemli olanı, Array'in hem ilkelleri hem de nesneleri içerebilmesidir. ArrayList yalnızca nesneleri içerebilir. Olamaz

ilkel türleri içerir.

• Ayrıca, Array uzunluğunun veya ArrayList boyutunun nasıl hesaplanacağı konusunda Array ve ArrayList'i karşılaştırabiliriz. uzunluk için kullanıyoruz

Array, bir ArrayList için size() yöntemini kullanıyoruz.

**23. İş parçacığı güvenli veya Senkronize nedir?**

• İplik güvenliği çok önemlidir ve programımızın çoklu iş parçacıklı ortamlarda güvenle kullanılmasını sağlayan işlemdir,

program dizimizi güvenli hale getirmenin farklı yolları vardır.

• **Senkronizasyon** , iplik güvenliği için en kolay ve en yaygın kullanılan araçtır.

• JVM, senkronize edilmiş kodun bir seferde yalnızca bir iş parçacığı tarafından yürütüleceğini garanti eder.

• JAVA **synchronized anahtar sözcüğü** , senkronize kod oluşturmak için kullanılır ve dahili olarak, yapmak için Nesne veya Sınıf üzerindeki kilitleri kullanır.

yalnızca bir iş parçacığının eşitlenmiş kodu yürüttüğünden emin olun.

• Demek istediğim, Java senkronizasyonu kaynağın kilitlenmesi ve kilidinin açılması üzerinde çalışır, bu nedenle senkronize edilmiş hiçbir iş parçacığı girmez.

kod.

• Synchronized anahtar sözcüğünü iki şekilde kullanabiliriz, biri tam bir yöntemi senkronize hale getirmek, diğeri ise

senkronize blok oluşturun.

- 40 -

**24. Yarattığınız bir nesneyi nasıl sıralarsınız?**

• Sırala bunu sıralayabilecektir.

• Ayrıca, nesnelerimi bir TreeSet veya TreeMap à Örn: SONRAKİ SAYFA içinde saklayabilirim

• Java, bir listeyi sıralamak için çeşitli yollar sunar.

o KARŞILAŞTIRILABİLİR - KARŞILAŞTIRICI arayüzleri sıralama için kullanılabilir. Bu durumlarda, karşılaştırmayı geçersiz kılmalıyız

yöntem.

• Diğer bir yol, bir karşılaştırıcı kullanabilen Liste arabirimi sıralama yöntemidir. Bu yöntemle artan veya

Azalan.

users.sort(Comparator.comparing(User::getUserID));

• Orijinal listeyi değiştirmek istemiyorsak, ancak yeni bir sıralanmış liste döndürüyorsak; sonra sort() yöntemini kullanabiliriz.

Akış arayüzü…

List<User> sortedUsers = users.stream()

.sorted(Comparator.comparing(User::getUserID))

.collect(Toplayıcılar .toList());

**25. Java'da Hashtable ve HashMap arasındaki fark?**

Java'da HashMap ve Hashtable arasında birkaç fark vardır:

• Hashtable senkronizedir, HashMap ise senkronize değildir. Bu, HashMap'i iş parçacıklı olmayan uygulamalar için daha iyi hale getirir.

senkronize edilmemiş Nesneler, genellikle senkronize olanlardan daha iyi performans gösterir.

• Hashtable, boş anahtarlara veya değerlere izin vermez. HashMap, bir boş anahtara ve herhangi bir sayıda boş değere izin verir.

• Örneğin; HashMap'in alt sınıflarından biri LinkedHashMap'tir, bu nedenle tahmin edilebilir yineleme sırası istemeniz durumunda

(varsayılan olarak ekleme siparişidir), bir LinkedHashMap için HashMap'i kolayca değiştirebilirsiniz. Bu şöyle olmazdı

Hashtable kullanıyorsanız kolay.

Senkronizasyon benim için sorun değilse, HashMap kullanmayı tercih ederim. Eğer bir sorun haline gelirse, o zaman tercih ederim

Collections.synchronizedMap() veya ConcurrentHashMap.

• Hem Hashtable hem de HashMap, Harita arabirimini uygular ve her ikisi de Anahtar ve Değerdir.

• HashMap iş parçacığı için güvenli değilken, Hashtable iş parçacığı için güvenli bir koleksiyondur.

• İkinci önemli fark, HashMap senkronize olmadığı için performanstır.

Hashtable'dan daha iyi performans gösterdi. à Collections.synchronizedMap(...Harita...);

**26. İstisna ile nasıl başa çıkarsınız?**

İstisnayı işlemek için try-catch-finally yaklaşımını kullanırdım

• 1- İstisna oluşturabilecek kodumu bir try-catch bloğunun içine koyardım. try-catch bloğu ile bir

istisna veya kurtarma adımlarımı gerçekleştirmeyi deneyin. Ayrıca, gerekirse çoklu veya Union Catch bloklarını kullanabilirim

• 2- Ayrıca throws anahtar sözcüğünü de kullanabilirim. AMA bu, benim yöntemimi çağıran herkesin artık onunla da ilgilenmesi gerektiği anlamına geliyor!

• 3- Diğer bir yol da Otomatik Kapanabilir: try bildiriminde Otomatik Kapanabilir olan referansları yerleştirdiğimizde,

kaynağı kendimiz kapatmamız gerekiyor. Yine de, istediğimiz başka bir tür temizleme yapmak için bir nihayet bloğu kullanabiliriz. denemek

**27. TreeSet vs TreeMap**

• TreeSet: Yalnızca benzersiz değerler içerebilir - artan düzende sıralanır

• TreeMap: yalnızca benzersiz anahtarlar içerebilir. - tuşlar artan düzende sıralanır

- 41 -

**28. final mi final mi yoksa nihayet mi?**

• **final** à bir anahtar kelimedir ve sınıf, yöntem ve değişkene kısıtlamalar uygulamak için kullanılır.

o son Sınıf Devralınamaz

o son Yöntem Geçersiz Kılılamıyor

o nihai Değişken değeri DEĞİŞTİRİLEMEZ.

• **nihayet** à bir bloktur ve önemli kodu yerleştirmek için kullanılır, istisna işlenip işlenmediğine bakılmaksızın yürütülür

• **finalize** à bir yöntemdir ve Object is Garbage toplanmadan önce temizleme işlemini gerçekleştirmek için kullanılır.

**29. Java'da Hata ve İstisna Arasındaki Fark?**

• Hem Hata hem de İstisna, Java'daki Throwable'dan türetilmiştir.

• Hata, genellikle ele alınamayan hataları temsil eder.

Örnekler için: OutOfMemoryError, NoClassDefFoundError

• Öte yandan, İstisna, yakalanabilen ve çözülebilen hataları temsil eder.

Örnekler için> IOException, NullPointerException

• İstisna, kontrol edilen ve kontrol edilmeyen İstisna olarak iki kategoriye ayrılır. Kontrol Edilen İstisna zorunlu gerektirir

işlemek için kod bloğunu deneyin. İşaretlenmemiş İstisna çoğunlukla programlama hatalarını temsil eder (NullPointerException veya

Çalışma zamanı istisnası)

• Hatalar denetlenmeyen istisnalardır ve geliştiricinin bunlarla herhangi bir şey yapması gerekmez

• **Tüm Hatalar İstisnadır, ancak tersi doğru değildir.**

• Genel olarak Hatalar, kimsenin kontrol edemediği veya ne zaman olduğunu tahmin edemediği hatalardır, öte yandan İstisna tahmin edilebilir.

ve işlenebilir

**30. Java'da RuntimeException ve CheckedException arasındaki fark?**

• İstisna, Çalışma Zamanı (işaretlenmemiş) İstisna ve CheckedException olmak üzere iki kategoriye ayrılır.

• RuntimeException ve CheckedException arasındaki temel fark, işlemek için try-catch sağlanmasının zorunlu olmasıdır.

CheckedException, RuntimeException durumunda zorunlu değildir.

• NullPointerExceptio, ArrayIndexOutOfBound, ClassNotFoundException gibi en yaygın İstisnalardan bazıları,

IOİstisna.

Öncelikle Java İstisnalarının RuntimeException'ın kontrolsüz olarak da bilinen iki kategoriye ayrıldığını hatırlatmak istiyorum.

İstisna ve kontrol edilen (derleme zamanı) İstisna.

RuntimeException ile kontrol edilen İstisna arasındaki temel fark şudur: Try catch veya try nihayet sağlamak zorunludur

kontrol edilen İstisnayı işlemek için blok ve bunun yapılmaması derleme zamanı hatasına neden olurken, RuntimeException durumunda bu

zorunlu değildir.

- 42 -

NullPointerException, ArrayIndexOutOfBoundException gibi en yaygın İstisnalardan bazıları işaretli değildir ve bunlar

java.lang.RuntimeException'dan türemiştir.

Kontrol edilen İstisnaların popüler örneği ClassNotFoundException ve IOException'dır ve bu nedenle yapmanız gereken budur.

Java'da dosya işlemleri gerçekleştirirken, birçoğu IOException'ı attığından, nihayet bir try catch bloğu sağlayın.

Kişisel fikrimi soracak olursam, Kontrol Edilen İstisnaların for içine kazan plakası kodunu ekleyerek kod kodumuzu ÇİGİN kıldığını düşünüyorum.

try-catch nihayet blok.

**31. Java'da fırlatma ve fırlatma arasındaki fark?**

• fırlatma ve fırlatma, Java programlama dilinin İstisna özelliği ile ilgili iki anahtar kelimedir.

• throw anahtar sözcüğü açıkça bir istisna oluşturmak için kullanılır, diğer yandan, bir istisna bildirmek için throws anahtar sözcüğü kullanılır

bu, try--catch bloğuna benzer şekilde çalıştığı anlamına gelir.

• Eğer sözdizimini akıllıca görürsek, throw'u bir İstisna sınıfı örneği takip eder ve onu istisna sınıfı isimleri izler.

• yeni ArithmeticException ("Aritmetik İstisnası"); ArithmeticException'ı atar;

• yöntem gövdesi için throw anahtar sözcüğü kullanılırken, istisnayı bildirmek için yöntem imzasında throws kullanılır.

Her ikisi de Java'nın İstisna özelliği ile ilgili iki anahtar kelimedir. Fırlatmak ve atmak arasındaki temel farkı hatırladığım kadarıyla

atar, kullanımlarında ve işlevselliklerindedir.

• herhangi bir yöntem tarafından atılan İstisna'yı bildirmek için yöntem imzasında throws kullanılır, örneğin

genel geçersiz kapatma(), IOException{'ı atar

yeni IOException ("Kapatılamadı");

}

Ancak throw aslında Java kodunda İstisna atmak için kullanılır.

Yeni İstisna atın ("başlatılamaz");

Diğer bir deyişle; throws anahtar sözcüğü hiçbir yerde istisna yöntemi imzası kullanılamazken, throw anahtar sözcüğü içeride kullanılabilir

yöntem veya statik başlatıcı bloğu, yeterli özel durum işleme sağladı.

Oh, throw ile ilgili bir şey daha hatırlıyorum, throw anahtar sözcüğü, break kullanmadan bir switch ifadesini kırmak için de kullanılabilir.

anahtar kelime

**32. Nesne ve Sınıf Arasındaki Fark?**

• Sınıf, istediğiniz kadar nesne oluşturabileceğiniz bir plan veya şablondur. Nesne, bir sınıfın üyesi veya örneğidir

• Class, class anahtar sözcüğü kullanılarak bildirilir, Object, esas olarak yeni anahtar sözcük aracılığıyla oluşturulur.

Sınıf, nesneler için bir şablondur. Bir sınıf, geçerli bir değer aralığı ve varsayılan bir değer dahil olmak üzere nesne özelliklerini tanımlar. Bir sınıf

ayrıca nesne davranışını tanımlar. Bir nesne, bir sınıfın bir üyesi veya bir "örneğidir" ve tüm öğelerinin içinde bulunduğu durumlara ve davranışlara sahiptir.

özellikler, sizin açıkça tanımladığınız veya varsayılan ayarlarla tanımlanan değerlere sahiptir.

Sınıf - Bir sınıf, türünün nesnesinin desteklediği davranışı/durumu tanımlayan bir şablon/plan olarak tanımlanabilir.

Bunları karşılaştırırsak çok fark var ama bilmeniz gereken bazılarını söyleyeyim;

• Java'da nesne oluşturmanın new anahtar sözcüğü, newInstance() yöntemi, klon() yöntemi, fabrika gibi birçok yolu vardır.

yöntem ve seri kaldırma. Java'da class anahtar sözcüğünü kullanarak sınıfı tanımlamanın tek bir yolu vardır.

• Nesne, ihtiyaca göre birçok kez oluşturulur. Sınıf bir kez ilan edilir.

• Object bir sınıfın örneğidir. Sınıf, nesnelerin oluşturulduğu bir plan veya şablondur.

• Nesne fiziksel bir varlıktır. Sınıf mantıksal bir varlıktır.

**Örnek** : Sınıf: İnsan Obje: Erkek, Kadın

Sınıf: Meyve Nesne: Elma, Muz, Mango, Guava wtc.

Sınıf: Cep telefonu Nesne: iPhone, Samsung, Moto

Sınıf: Yiyecek Nesnesi: Pizza, Burger, Samosa

- 43 -

**33. StringBuffer ve StringBuilder?**

• Ana fark, StringBuffer'ın senkronize olması ve StringBuilder'ın senkronize olmamasıdır. Yani, StringBuilder olabilir

aynı anda denir. Bu da StringBuilder'ı daha verimli hale getirir.

• StringBuffer senkronize, StringBuilder senkronize değil

• StringBuilder, StringBuffer'dan daha verimlidir

• Yapıcı;

o StringBuilder() à başlangıç **kapasitesi 16** olan boş bir dize oluşturdu .

o StringBuilder(str str) à belirtilen dizede bir StringBuilder oluşturdu.

o StringBuilder(int length) à uzunluk olarak belirtilen kapasitede boş bir dize oluşturdu.

● Yöntem;

o StringBuilder str = new StringBuilder(“Merhaba”);

o str.append(“Java”); à //Merhaba Java

o str.insert(1,“Java”); à //HJavaello

o str.replace(1,3,“Java”); à //HJavalo

o str.delete(1,3); à //Merhaba

o str.ters(); à //olleH

string str = “Merhaba”;

ters çevrilmiş dizi = “ ”;

for (int i = str.length()-1; i>=0 ; i--){

ters += str.charAt(i);

}

sysout(ters);

**34. finalize() nedir?**

• finalize() yöntemi, Java.lang.Object sınıfının korumalı ve statik olmayan bir yöntemidir.

• Java'da oluşturduğumuz tüm nesnelerde bu yöntem mevcuttur.

• Bu yöntem, bir nesne üzerinde kaldırılmadan önce bazı son işlemleri veya temizleme işlemlerini gerçekleştirmek için kullanılır.

hafıza.

• Bir nesne yok edilmeden önce gerçekleştirmek istediğimiz işlemleri tutmak için finalize() yöntemini de geçersiz kılabiliriz.

Bu çağrılabilir. nesne.finalize();

**35. system.gc() nedir?**

• JVM'ye, belleği boşaltmak için Çöp toplayıcıyı çalıştırma isteği

• Her zaman çalışmıyor

Java.lang.System.gc() yöntemi çöp toplayıcıyı çalıştırır. Bunu çağırmak, Java Sanal Makinesinin

Hâlihazırda işgal ettikleri belleği hızlı bir şekilde yeniden kullanım için kullanılabilir hale getirmek için kullanılmayan nesneleri geri dönüştürmeye yönelik çaba.

Bu bir emir değil, bir ricadır. Bu isteği yerine getirmek çöp toplayıcının görevidir.

**36. Yöntem nedir?**

• Bir işlemi gerçekleştirmek için birlikte gruplandırılmış ifadelerin toplanması. **System.out.println()'** i çağırdığınızda

Örneğin, sistem konsolda bir mesaj görüntülemek için aslında birkaç ifadeyi yürütür.

• Yöntem, adıyla anılan ve bir programın herhangi bir noktasında basitçe şu şekilde çağrılabilen (çağrılan) bir kod kümesidir.

yöntemin adını kullanarak. Bir yöntemi, veriler üzerinde hareket eden ve genellikle bir değer döndüren bir alt program olarak düşünün. Her yöntem

kendi adı var.

- 44 -

**37. Önemli String Yöntemleri?**

**38. IS-A ve HAS-A ilişkisi arasındaki fark nedir?**

• **IS-A** kalıtım üzerine kuruludur à Bu şey o şeyin bir türüdür

• **HAS-A** ilişkileri kullanıma dayalıdır

o Ör: A sınıfı HAS -AB, eğer A Sınıfındaki kodun B sınıfının bir örneğine referansı varsa

halk atı

özel Halter myHalter;

genel boşluk atlama(){

Sysout "atlıyorum"

• Horse sınıfından gelen atlama yöntemini kullanmak için bir Halter örnek değişkeni çağırıyorsunuz - bunun yaptığı şey,

Horse HAS-A Halter anlamına gelir

• Horse sınıfının bir Halter'i vardır, çünkü Horse, Halter tipinde bir örnek değişkeni bildirir. Kod, üzerinde tie() işlevini çağırdığında

Horse nesnesinin Halter örneği değişkeni -}

• Soyut sınıfın yapıcıları varken arabirimin bir yapıcısı yok

**39. Yineleyici nedir ve her döngü için arasındaki fark nedir?**

• Yineleyici, diziyle değil ArrayList ile çalışır.

• Öğeleri yinelememize yardımcı olacaktır.

• Fark, yineleyici ile yineleme sırasında listede değişiklik (öğeyi kaldır) yapabilirsiniz.

• her döngü için listemizde değişiklik yapamayız

- 45 -

**40. Java Koleksiyon Çerçevesi**

İki çeşit Koleksiyon (Karıştırmamaya dikkat edin)

μ **java.util.Collection** - Set ve List'ten arabirim genişletilir (uygulanmaz)

μ **Set** *(Benzersiz şeyler)* - TEKRARLAMALARA İZİN VERMEZ. Kümeyi Uygulayan Sınıflar;

¨ **HashSet** à Herhangi bir kopya istemediğinizde ve yineleme yaptığınızda sırayı umursamadığınızda kullanın

vasıtasıyla

o Sırasız ve Sırasız

¨ **LinkedHashSet** à HashSet'in sıralı versiyonu ve yineleme sırasını önemsediğinizde HashSet üzerinden kullanın

¨ **Sıralı Küme**

¨ **TreeSet** à Öğeler, öğelerin doğal sırasına göre artan sırada olacaktır.

o Doğal düzenin kendi kurallarınızı uygulamak için yapıcıyı da özelleştirebilir

μ **Liste** *(nesnelerin listesi)* - dizine önem verir. List uygulayan sınıflar;

¨ **LinkedList** à İndeks pozisyonuna göre sıralanır ve elemanlar birbirine çift bağlantılıdır

o Yığın ve kuyruk uygulamak için iyi bir seçimdir

o ArrayList'ten daha yavaş yinelenir ancak hızlı ekleme ve silme

¨ **Vektör** à ArrayList ile aynı AMA vektör yöntemleri senkronize edilir (iş parçacığı için güvenli)

¨ **ArrayList** à Hızlı yineleme ve Hızlı rastgele erişim ve sıralı (dizine göre)

o Ayrıca sıralanmamış (ancak sıralamak için Collections.sort()'u çağırabilir)

μ **java.util.Collections** - koleksiyonlarla kullanım için statik yardımcı yöntemler içeren bir sınıf; Ekle, kaldır, içerir,

boyut ve yineleyici vb.

• **Harita** *(benzersiz kimliğe sahip şeyler)* à Önemli: Harita ile ilgili sınıfların ve arayüzlerin hiçbiri Koleksiyon formunu genişletmez.

Map'in uygulama sınıfları Koleksiyon değil, "koleksiyonlar" olarak düşünülür. Map uygulayan sınıflar;

¨ **Hashtable**

o HashMap ile aynı ANCAK HashTable yöntemleri senkronize edilir (Unutmayın. YALNIZCA YÖNTEMLER

SENKRONİZE, SINIFLAR VEYA DEĞİŞKENLER DEĞİL)

o Hashtable, NULL bir şeye sahip olmanıza izin vermez (HİÇBİR BOŞ YOK)

¨ **LinkedHashMap**

o Kampanya siparişini (veya isteğe bağlı olarak erişim sırasını) korur

o Eleman eklemek/kaldırmak için HashMap'ten daha yavaş ama DAHA HIZLI İTERASYON

¨ **HashMap** à Sıralanmamış ve Sıralanmamış & Bir koleksiyonda bir boş ANAHTAR ve birden çok boş değere izin verir

o Tuş Takımı()

o Map.keySet() - bir dizi Anahtar döndürür

o Map.keySet().size - tuşların sayısını döndürür

¨ **SortedMap** ve TreeMap

• Set,List ve Map'in uygulama sınıfları HİÇBİR ZAMAN hem sıralı hem de sırasız olamaz, diğer tüm kombinasyonlar olabilir.

**41. Float nasıl String'e dönüştürülür?**

float f = Float.parseFloat("25");

String s = Float.toString(25.0f);

**42. Diyelim ki "int b=3; ve int a=4;" onları nasıl değiştirebilirsin?**

// tek satırlık yöntemler

a = a ^ b ^ (b = a);

b = (a + b) – (a = b);

a += b – (b = a);

int sıcaklık = a; // geçici değişken

a = b; b = sıcaklık;

- 46 -

**43. Daktilo yazmayı biliyor musunuz? döküm nedir?**

• **Otomatik kutulama** à , ilkel bir değer alıp sarmalayıcı sınıf nesnesine atadığınız bir süreçtir int i=10;

tamsayı n=i;

Tam sayı=200;

Tamsayı num2=yeni Tamsayı(400);//KUTULAMA YOK

• **Kutudan çıkarma** à , Wrapper sınıfı nesnesini alıp ilkel nesneye dönüştürdüğünüz bir süreçtir.

Tamsayı num2=yeni Tamsayı(400);

Tam sayı=200;

int i=sayı2;

• Bir türün değerini başka bir türdeki bir değişkene atamak, Tür Dökümü olarak bilinir.

**44. Bu programın çıktısı nedir?**

for (int i = 0; i < 3; i++) {

for (int j = 3; j >= 0; j--) {

eğer (i == j)

devam et;

System.out.println(i + " " + j);

}

}

Çıktı: 1 0 2 3 2 1 2 0

**45. Projenizde soyut bir sınıfı nasıl kullanıyorsunuz bana bir örnek verin?**

• Bu kavramlar, çerçeve geliştirmede yaygın olarak kullanılmaktadır. Soyut sınıf, ortak bir süper sınıfın tanımlanmasında kullanılır

çerçevenin Sayfa Nesne Modeli katmanını yazarken. Genellikle, hepsine sahip olmak için BasePage adında soyut bir sınıf yaratırız.

bu sınıfta yazılan her sayfa için ortak üyeler örnek **getPageTitle() .**

• Ardından her Sayfa sınıfı (Ana Sayfa, Giriş Sayfası, Gösterge Sayfası vb.) BasePage'den devralır. Bazen birinin ihtiyacı olabilir

üst sınıfta uygulanan yöntemlerin davranışını değiştirin. Bu nedenle, alt sınıf, bu yöntemi geçersiz kılma özgürlüğüne sahiptir.

polimorfizm kullanır. Soyut sınıfını gerçek projelerde bu şekilde kullanıyoruz.

**46. Değere göre geçiş ve referansa göre geçiş arasındaki fark nedir? değere göre geçmek ve referansa göre geçmek?**

• Değere göre geçmek, fonksiyon parametresinin değerinin hafızanızın başka bir yerine kopyalanması anlamına gelir ve

işleviniz içindeki değişkene erişirken veya değiştirirken, yalnızca kopyaya erişilir/değiştirilir ve orijinal değer

dokunulmadan bırakılır. Değere göre geçmek, değerlerinizin çoğu zaman nasıl aktarıldığıdır.

• Referans yoluyla geçmek, değişkenin bellek adresinin (bellek konumuna bir işaretçi) iletildiği anlamına gelir.

işlev. Bu, bir değişkenin değerinin iletildiği değere göre geçişten farklıdır. Örneklerde, bellek adresi

myAge'in değeri 106'dır. myAge, artırmaAgeByRef işlevine aktarılırken, işlev içinde kullanılan değişken (yaş

bu örnek) hala orijinal myAge değişkeniyle aynı bellek adresine işaret ediyor (İpucu:

function parametresi, bir değişkenin referansını/işaretçisini almak için birçok programlama dilinde kullanılır).

- 47 -

**SELENYUM**

**1. Selenyum nedir ve nelerden oluşur?**

• Selenium, otomatik web testi için bir araç takımıdır. Tarafından bestelendi;

o Selenium IDE(Entegre Geliştirme Ortamı); kayıt ve oynatma için çalışan bir Firefox eklentisi.

o Selenyum RC(Uzaktan Kumanda) (1.0) ; bir test aracıdır ve web uygulamasını otomatikleştirmek için JS üzerinde çalışmak için kullanılır. (2004)

o Web Sürücüsü (2.0); bir web otomasyon çerçevesidir ve testlerinizi farklı tarayıcılarda gerçekleştirmenizi sağlar. (2011)

o Selenyum Izgarası; testlerin birden fazla makinede paralel olarak çalışmasına izin verir.

**2. Selenyumun avantajları nelerdir?**

• Selenium açık kaynaktır ve herhangi bir lisans maliyeti olmadan kullanımı ücretsizdir

• Java, Ruby, Python, C# gibi birden çok dili destekler...

• Çoklu tarayıcı testini destekler

• İyi miktarda kaynağa ve topluma yardım etmeye sahiptir.

• Windows, Mac, Linux gibi bir çok işletim sistemini destekler...

• Web uygulamasıyla etkileşim kurun

**3. Selenyumun dezavantajları nelerdir?**

• Selenium yalnızca web tabanlı uygulamaları destekler, Windows tabanlı uygulamaları desteklemez

• Yerleşik raporlama aracı yoktur, rapor oluşturma etkinliği için üçüncü taraf araçlara ihtiyaç duyar

• Grafikler, captcha'lar, barkodlar, şekillerle çalışamaz

• Dosya yükleme özelliğini desteklemez.

• Ustalaşması zor, geliştirici düzeyinde bilgi gerektirir

• İyi konumlandırıcılar yazmak zor

• Senkronize etmek zor

**4. Selenyum'un sınırlamaları nelerdir?**

• Masaüstü uygulamasını test edemiyoruz

• Web hizmetlerini test edemiyoruz

• Ewe, test çerçevesi (TestNG, JUnit), kaynaktan okuma gibi görevleri gerçekleştirmek için harici kütüphaneler ve araçlar kullanmalıyız.

harici dosyalar (Excel için Apache POI)

• Selenium kullanılarak Captcha'yı otomatikleştirmek mümkün değil

• Dosya yükleme özelliğini desteklemez.

**5. Selenium ile ne tür testleri otomatikleştiriyorsunuz?**

• fonksiyonel testler (pozitif/negatif, UI)

• duman testleri

• regresyon testleri

• entegrasyon testleri

• uçtan uca test

• veri odaklı

**6. Selenyum ile ne yapmıyoruz?**

• Performans, yük, stres testi, manuel özel testler, (Bu testler, bu araçlar konusunda eğitimli uzmanlar tarafından yapılır)

• Saf veritabanı testi (sadece DB'nin kendisini test edersek),

• Birim testleri..., görünüm ve his temelli testler (renk, şekiller, vb.),

• statik test

- 48 -

**7. Selenyum araç setinde neler var?**

• Selenium IDE à bir Chrome ve Firefox uzantısı olarak uygulandı ve testleri kaydetmenize, düzenlemenize ve hata ayıklamanıza izin veriyor.

• Selenium RC à herhangi bir programlama dilinde otomatik web uygulaması UI testleri yazmak için

• Selenium WebDriver à testlerinizi farklı tarayıcılarda yürütün

• Selenium GRID à testlerinizi farklı makinelerde farklı tarayıcılara karşı paralel olarak çalıştırın.

**8. Şu anda Selenium'un hangi sürümünü kullanıyorsunuz?**

• JDK (JAVA) - 1.8 à Lambda deneyimi nedeniyle hoşuma gidiyor . ve, Yakalama hatası işlemeyi deneyin, birden fazla yakalama ekleyebilirsiniz.

• IntelliJ - 2018.03.04

• Selenyum - 3.141.59

• TestNG - 6.14.3

• Salatalık – 4.2.6

• Maven - 3.6.0

• GIT - 2.17.2

**9. Örtülü Bekleme ve Açık Bekleme?**

• **Örtülü bekleme** , fırlatmadan önce bir öğeyi bulurken belirli bir süre bekleyen bir beklemedir.

"NoSuchElementException". Varsayılan olarak selenyum, öğeleri beklemeden hemen bulmaya çalışır. Yani, iyi

örtük beklemeyi kullanmak için. Bu bekleme, geçerli sürücü örneğinin tüm öğelerine uygulanır.

• **Açık bekleme** , belirtilen BeklenenKoşul karşılanana kadar belirli bir web öğesine uygulanan bir beklemedir.

• Örtük bekleme basitçe; Koşul zaman aşımından önce karşılanırsa bir sonraki adıma geçilir, koşul süre içinde karşılanmazsa bir sonraki adıma geçilir.

zaman aşımı "Böyle Bir Öğe Yok" istisnası atar.

• Açık bekleme bazen öğenin görünür olması, tıklanabilir olması, etkin olması gibi belirli bir olay/koşul için beklememiz gerekir....

driver.manage().timeouts().implicitlyWait(5,TimeUnit.SECONDS);

webDriverWait bekle = yeni WebDriverWait (sürücü, 5);

wait.until (ExpectedConditions.visibilityOf(element);

**10. FluentWait nedir?**

• Diyelim ki bazen sadece 1 saniyede görünen ve bazen de görünmesi dakikalar süren bir öğeniz var. Şöyle

durumda, akıcı beklemeyi kullanmak daha iyidir, çünkü bu, öğeyi bulana kadar veya son zamanlayıcı çalışana kadar tekrar tekrar bulmaya çalışacaktır.

dışarı. Örnek AJAX veya JQuery'dir

• Açık beklemenin alt türü ancak koşulları geçersiz kılabilirsiniz

Bekle<WebDriver>bekle=yeni

FluentWait<Webdriver>(driver).withTimout(5,timeUnit.seconds).pollingEvery(100,timeunit.

milisaniye).ignoring(NoSuchElementException.class);

**11. Selenium'da bir elementi bulmanın çeşitli yolları nelerdir?**

• Kimliği ve adı olan Selenyum Konumlandırıcılar

• Selenium'da yer belirleyici, html'de bir öğe bulmanın bir yoludur:

• Id, name, className, xpath, css, linkText, kısmiLinkText, tagName

**12. Neden eleman bulamıyorum?**

• Konumlandırıcı değişti

• Bir iframe var

• Bekleme süresi:: sayfa yavaş yükleniyor veya Öğe dinamik:: konumlandırıcı

• Sayfa tam olarak yüklenmemiş/açılmamış

• Sayfa değişiklikleri ve o öğe artık mevcut değil

- 49 -

**13. Bir öğe nasıl vurgulanır?**

• Selenium WebDriver'ın vurgulama eylemi yoktur.

• Ancak bunu yapmak için JavaScript kullanabiliriz

JavascriptExecutor js = ((JavascriptExecutor) sürücüsü);

String bgcolor = element.getCssValue("backgroundColor");

for(inti=0;i< 10;i++){

changeColor("rgb(0,200,0)",

eleman,sürücü);//1

changeColor(bgcolor,

eleman,sürücü);//2

}

**14. Xpath nedir?**

• Xpath, html yapısını kullanarak bir web sayfasındaki herhangi bir öğenin konumunu bulmak için kullanılır.

• Metin kutusu gibi web Öğelerini bulmak için bir XML belgesindeki öğeler ve nitelikler arasında gezinebiliriz.

• düğmesi. onay kutusu, Web Sayfasında Resim uzantısı...

**15. Mutlak (/) ve Göreli (//) Xpath?**

• Sözdizimi à //etiket[@attribute=`değer`]

• Mutlak xpath, kök öğeden başlayıp öğeye kadar uzanan tek bir eğik çizgiyle ( / ) başlar.

• Göreli xpath, çift eğik çizgiyle ( // ) başlar ve belgenin herhangi bir yerindeki seçim eşleşmesini başlatır.

**16. Dinamik unsurları nasıl ele alıyorsunuz?**

• Kimliğin statik kısmını bulun ve bir konumlandırıcı (xpath veya css) yazın à Ardından Startswith, include, EndsWith kullanın

• içerir( ) à //\*[içerir(@name=`btn`)]

• startwith( ) à //label[startwith(@id, `message`)]

• text() à //td[text() = `usedId`]

• veya & ve à //input[@type = `submit` AND @name = `login`]

**17. Dinamik web sayfası nasıl test edilir?**

• Bu sorunun tek bir çözümü yoktur. Uygulamayı çok iyi anlamalıyız

o Gerektiğinde açık beklemeler kullanın.

o Özel xpath'ler ve css bulucuları kullanın

§ Xpath: içerir, ile başlar, ile biter, metin içerir.

§ Ebeveyn, çocuk, kardeş ilişkilerini kullanarak öğeyi başka bir kararlı öğeye göre bularak

**18. Dinamik tablo nasıl test edilir?**

• Özel xpath'ler ve css bulucuları kullanın

o Xpath: içerir, ile başlar, ile biter, metin içerir.

o Öğeyi ebeveyn, çocuk, kardeş ilişkilerini kullanarak başka bir kararlı öğeye göre bularak

• Tablo ile çalışan fayda yöntemlerim var. Bir tablo web elemanı alan ve tüm sütunu döndüren bir yöntemim var.

isimler. Bir tablo, sayı alan ve o satırdaki tüm verileri döndüren bir yöntemim var.

**19. xpath kullanarak ana öğeye nasıl geçebiliriz?**

• xpath'de (..) ifadesini kullanarak üst öğeye geçebiliriz

**20. Close() ve Quit() komutu arasındaki fark?**

• driver.close() à mevcut tarayıcıyı kapatmak için kullanılır

• driver.quit() à tüm tarayıcı örneklerini kapatmak için kullanılır

- 50 -

**21. xpath kullanarak n'inci alt öğeye nasıl geçebiliriz?**

• **İki yol vardır:**

o dizin konumu ile köşeli parantez kullanma

Örneğin: div[2] ikinci div öğesini bulacaktır.

o pozisyon ( ) yöntemini kullanarak

Örneğin: div[position()=2] ikinci div öğesini bulur

**22. xpath ve css seçici arasındaki fark?**

• xpath ile öğeleri geriye veya ileriye doğru arayabiliriz...

css yalnızca ileri yönde çalışırken

• Xpath metinle çalışabilir, css çalışamaz

• Xpath daha fazla kombinasyona sahiptir ve dizine göre arama yapabilir

css dizine göre arama yapamaz, ancak css, xpath'ten daha hızlı çalışır.

**23. Çerçeve nedir?**

• Test otomasyonunda çerçeve, test otomasyonunun planıdır.

• Klasör yapılarınızı, fonksiyon kitaplığınızı, test sonuçlarını, test verilerini, kaynakları nereye kaydedeceğinizi içerir.

• Bu çok önemlidir, çünkü bir otomasyon projesi üzerinde çalışırken herkesin takip etmesi gereken bir kılavuz olacaktır ve bizim

betiğin bakımı daha kolay olacaktır.

**24. Görüşme sırasında HTML raporlama hakkında mı konuşuyorsunuz?**

• Çerçevemde birden fazla raporlama yöntemi kullanıyorum, sürücü komut dosyası, test senaryoları excel sayfasına başarılı/başarısız yazıyor,

• Reporter yardımcı nesnesi UFT raporuna yazar, ayrıca özel bir HTML raporlama motoru geliştirdim.

• Not Defteri'ne HTML kodu gönderir ve teknik bilgisi olmayan kişilerin kolayca yapabileceği güzel bir HTML rapor belgesi oluşturur.

anlayın ve kullanın.

**25. Bir web sayfası nasıl büyütülür?**

driver.manage().window().maximize();

**26. Bazı durumlarda, maksimuma çıkarma() çalışmayacak > peki bunun yolu ne olacak?**

• Eylemler veya sürüm değişikliği.

ChromeOptions seçenekleri = yeni ChromeOptions();

options.addArguments("startmaximized");

**27. Selenium'daki anahtar sınıf nedir?**

• Klavyeden tuşlara basma seçeneği sunar

• Anahtar.ENTER

• SendKeys() yöntemine GEÇİLMELİDİR

• Ör; . sendkeys(“şarj cihazı” + tuşlar.ENTER)

**28. Rastgele gelen dinamik bir açılır pencere varsa ne olur?**

• Uyarı ile dene/yakala özelliğini kullanın

**29. Thread.sleep() nedir?**

• Yakalamak için selenyumu yavaşlatır

• İstisna atar, bu nedenle işlemeli veya atmalıdır

- 51 -

**30. Selenyum Çerçevesi Nedir?**

• Kod bakımını, kodun okunabilirliğini ve kodun yeniden kullanımını kolaylaştıran bir kod yapısıdır.

• Test senaryolarını otomatikleştirmek için Selenium WebDriver tarafından oluşturulan başlıca 3 tür çerçeve vardır:

**Veriye Dayalı Çerçeve**

• Piyasadaki en popüler otomasyon çerçevelerinden biridir.

• Tüm test verilerimiz bazı harici dosyalardan oluşturulur;

o mükemmel

o veya özellik dosyasında senaryo taslağı

o veya TestNG Veri Sağlayıcı

• Selenium WebDriver, web tabanlı uygulamaları otomatikleştirmek için harika bir araçtır. Ancak okuma ve yazmayı desteklemiyor

excel dosyaları üzerinde işlemler. Bu nedenle, **Apache POI** gibi üçüncü taraf API'ler kullanıyoruz

**Anahtar Kelimeye Dayalı Çerçeve**

• Anahtar kelimeye dayalı test, uygulamayla ilgili anahtar kelimeleri içermek için veri dosyalarını kullanan bir komut dosyası oluşturma tekniğidir.

test edilmek.

• Anahtar kelimeler excel dosyası gibi bazı harici dosyalara yazılır ve Java kodu bu dosyayı çağırır ve test senaryolarını yürütür.

**HybridDriven Çerçevesi**

• DDF ve KDF'nin bir kombinasyonunun yaygın olarak HDF olduğu söylenir.

• Hem test verileri hem de test eylemi harici dosyalarda tutulur.

**31. Selenium'da aşırı yüklenmiş Yöntemleri nasıl kullandınız?**

• İki değerin eşit olup olmadığını belirtirken TestNG'den à Assert.assertEquals (actual, Beklenen) kullanıyorum

• String, Objects, int, boolean değerleri parametrelerini girebilirsiniz.

**32. Neden NoSuchElementException alıyoruz?**

• Konumlandırıcının doğru olup olmadığını kontrol edin

• Zamanlamanın doğru olup olmadığını kontrol edin

• Öğenin bir iframe içinde gizlenip gizlenmediğini kontrol edin

**33. js uyarılarını nasıl ele alırsınız?**

• Tarayıcıdaki uyarı JavaScript'ten geliyorsa, Alert sınıfını kullanırız.

Uyarı uyarısı = driver.switchTo.alert();

uyarı.kabul();

alarm.dismiss();

alarm.sendKeys();

uyarı.getText()

**34. Birden çok çerçeve nasıl işlenir?**

• 4 kare varsa, belirli kareye ulaşmak için her birinden art arda geçmeniz gerekir. 3. kareye atlayamıyorum

1. kareden.

**35. driver.get() ve driver.navigateto() arasındaki fark nedir?**

• driver.get() à Bir URL açmak için tüm sayfa yüklenene kadar bekleyecektir

• driver.navigateto() à Bir URL'ye gitmek için ve tüm sayfa yüklenene kadar beklemeyecektir

- 52 -

**36. Selenium'da çerçeveler nasıl işlenir?**

• Bir html sayfasını başka bir sayfaya gömmek için kullanılan çerçeveler

• Adımlar

o iframe'i bulun

o driver.switchTo().frame() ile başka bir iframe'e geçin;

.frame() à doğrudan parametre olarak string, Integer, webElement, name veya id alır

driver.switchTo().frame(webElement);

driver.switchTo().frame();

• 2. karenin dışında (şu anda bulunduğunuz) bir öğe bulmak istiyorsanız, şimdi 2. karedesiniz.

NosuchElementException'ı atar

• Bir önceki kareye geri dönmeniz gerekirse

o driver.switchTo().parentFrame() à Bir seviye yukarı gider

o driver.switchTo().defaultcontent() à En üste gider

• Sayımı kullanarak geçiş yapabilir

o driver.switchTo(0) à Varsayılan çerçeve olmayan her şeyi sayar

*Bu yöntemler, kullandığınız tarayıcıya bağlı olarak size farklı sonuçlar verebilir.*

**37. Tarayıcı açılır pencerelerini nasıl yönetirsiniz?**

• **void işten çıkarma()** à açılır pencere belirir belirmez “İptal” düğmesine tıklar.

• **void accept()** à açılır pencere belirir belirmez “Tamam” düğmesine tıklar.

• **String getText()** à , uyarı kutusunda görüntülenen metni döndürür.

• **void sendKeys(String stringToSend)** à belirtilen dize modelini uyarı kutusuna girer.

**38. Windows/OS açılır pencerelerini nasıl yönetiyorsunuz?**

• Selenium, Windows tabanlı uygulamaları desteklemez, sadece web uygulama testlerini destekleyen bir otomasyon test aracıdır.

• AutoIT, Robot sınıfı gibi bazı üçüncü taraf araçları kullanarak Selenium'daki pencere tabanlı açılır pencereleri işleyebiliriz

• **driver.getWindowHandle();** Bu, bu sürücü örneğinde onu benzersiz olarak tanımlayan geçerli pencereyi işleyecektir.

• **driver.getWindowHandles();** Tüm açık pencereleri işlemek için

**39. Başsız tarayıcı nasıl kullanılır**

• Başsız tarayıcı: Açılmayan tarayıcı, arka plan hizmeti/programı olarak çalışır.

• Örnek, selenyumdan htmlunitdriver'dır

o WebDriver = yeni htmlunitdriver()

o Çok kararlı değil

• Phantonjstarayıcı

o Daha kararlı

o tarayıcı = yeni phantomjsbrowser()

**40. findElement vs findElements?**

• FindElement > bu yöntem ilk WebElement öğesini döndürür!

o eleman bulunmazsa İstisna verir

• FindElements > <WebElement> Listesini döndürür;

o Sonuç listesinde bulunmayan öğe boş değerlere sahipse İstisna vermez

- 53 -

**41. Birden çok pencere/sekme nasıl işlenir?**

• Selenyum tek pencerede kalır

• Bir pencere açarsanız ve ardından 5 sekme açılırsa, selenyum ilk pencereye odaklanır

• Yeni bir penceredeyseniz ve Selenium'a varsayılan pencerede bir öğe yazdırmasını söylerseniz, yine de çalışacaktır.

kullanıcının odağı yeni pencerede

• Yeni pencereye geçilmelidir

o windowHandle() kullanın

o Driver.getWindowHandle()

§ Selenium bir tarayıcıyı her açtığında, Handles adlı sayfa için bir dizin kimliği verecektir.

§ Geçerli sayfanın tanıtıcısını/kimliğini döndürür (dize olarak)

o driver.switchTo().window(dize tutamacı)

o birden çok pencere için driver.getWindowHandles()

§ Bir dizi pencere tutamacı döndürür

o Başlıkları kullanarak geçiş yapın

for(string tanıtıcı: driver.getWindowHandles()){

driver.switchTo().Window(tutamaç)

if(driver.getTitle().equals(targetTitle){

kırmak;

}

}

**42. Sayfadaki tüm bağlantılar nasıl bulunur?**

List<WebElement> list = driver.findElements(By.tagName(“a”));

**43. isDisplayed() ile isEnabled() arasındaki fark. Ve Selenium WebDriver'da Selected() yöntemi mi?**

• isDispalyed() à web sayfası içinde bir web öğesinin varlığını doğrular. à doğru bulunursa , bulunmazsa à yanlış

• isDisplayed() à mevcut her türlü web öğesinin varlığını kontrol edin

• isEnabled() à web öğesinin web sayfası içinde etkinleştirildiğini veya devre dışı bırakıldığını doğrulayın.

• isEnabled() à öncelikle düğmelerle kullanılır

• isSelected() à web öğesinin seçili olup olmadığını doğrular

• isSelected() à radyo düğmeleri, açılır menüler ve onay kutuları ile kullanılır.

**44. Sürükle Bırak Nasıl Yapılır?**

Eylemler eylemi = yeni Eylemler(sürücü);

action.clickAndHold(driver.findElement(By.id("item")))

.moveToElement(driver.findElement(By.id("hedef")))

.release().build()

.rol yapmak();

**45. Aşağı Kaydır için:**

WebDriver sürücüsü = yeni ChromeDriver(); JavascriptExecutor jse = (JavascriptExecutor)sürücü;

jse.executeScript("window.scrollBy(0,250)", "");

• VEYA, aşağıdaki gibi yapabiliriz:

jse.executeScript("scroll(0, 250);");

**46. Yukarı kaydırmak için:**

jse.executeScript("window.scrollBy(0,-250)", ""); VEYA, jse.executeScript("scroll(0,-250);");

- 54 -

**47. Öğenin mevcut/görünür/etkin olup olmadığı ve mevcut metin nasıl kontrol edilir?**

• Öğe Mevcut'u kontrol etmek için:

if(driver.findElements(By.xpath("değer")).size() != 0){

System.out.println("Element Mevcut");

}başka{

System.out.println("Eleman Yok");}

• veya

if(driver.findElement(By.xpath("değer"))!= null){

System.out.println("Element Mevcut");

}başka{

System.out.println("Element Yok"); }

• Görünür'ü kontrol etmek için:

if(driver.findElement(By.cssSelector("a > yazı tipi")).isDisplayed()){

System.out.println("Eleman Görünür");

}başka{

System.out.println("Eleman Görünmez"); }

• Etkinleştir'i kontrol etmek için:

if(driver.findElement(By.cssSelector("a > font")).isEnabled()){

System.out.println("Eleman Etkinleştirildi");

}başka{

System.out.println("Element Devre Dışı"); }

• Mevcut metni kontrol etmek için

if(driver.getPageSource().contains("Kontrol edilecek metin")){

System.out.println("Metin var");

}başka{

System.out.println("Metin yok"); }

**48. JavaScriptExecutor kullanıyor musunuz?**

• Bu, kendi JavaScript'imi yazmama yardımcı oluyor. JS, selenyumdan çok daha fazla kontrole sahiptir.

• bu sınıfı kullanarak tarayıcıya JS komutları gönderebiliriz

JavaScriptExecutor jsExecutor=(JavaScriptExecutor)sürücü;

o executeScript(); komutu yerine getirir

o Parametrenin içinde JS kodunu koyduğunuz yer

• jsExecutor.executeScript(“alert('UYARI: Bu gereksiz bir mesajdır');” à Bu kod bir JS açılır penceresi açar

• Ayrıca 2 parametre de koyabilirsiniz: .executeScript(“js code”,element);

o Kaydırma için kullanılır (selenyum kaydırma ile iyi değildir, terimler üzerinde çalışırken bir zorluk olduğunu söyleyebilirsiniz

ve devam düğmesine tıklamadan önce sayfayı okumanız gereken koşul sayfası.

o Selenium ve action sınıfını kullanmayı denediğimde işe yaramadı, bu yüzden javaexecutor kullandım ve bir öğeye tıkladım;

**49. Açılır menüden seçilen çoklu değerler nasıl kontrol edilir?**

• Arabaları seçinList = new Select(el)

• carList.getSelectedOptions(): //seçilen seçenekleri bir listeye döndürür ( List<webelement>)

• her biri için : carList.getSelectedOptions()

- 55 -

**50. Eylemler sınıfı nasıl kullanılır?**

• Actions sınıfı, gelişmiş fare ve klavye işlemlerini yapmamızı sağlar:

• Fareyi kontrol edin

• Gelişmiş kullanıcı etkileşimleri için yöntemler sağlayan sınıf

o Gezinme

o Kaydır

o Çift tıklama

o Sürükle ve bırak

o Sağ tıklama

o karıştırma/eşleştirme operatörleri

• Eylemler eylemi=yeni Eylemler(sürücü)

• Eylem yöntemleri

o tıklama()

o gerçekleştirmek()

o bekle()

o tuşa basma() inşa()

o moveToElement(eleman)

o dragAndDrop(kaynak,hedef).perform()

o sendKeys() genellikle kullandığımızdan farklı

§ Sendkeys işlemini farklı elementler üzerinde yapalım

§ Web elemanından gelen normal gönderme tuşları, metin girişi olmayan bir şeye istisna atar.

§ Uzun yol;

action.moveToElement(source).clickAndHold().moveToElement(target).release().perform();

• perform() kullanılmadığı sürece eylemler çalışmaz

• Yöntemleri zincirleme yapıyorsanız, perform()'dan önce build()'i kullanmalısınız.

**51. Çift tıklama eyleminin sözdizimi nedir?**

• Eylemler sınıfını kullanarak web öğesine karşı herhangi bir eylem gerçekleştirmek için önce öğeyi bulmamız gerekir:

WebElement el = driver.findElement

Eylemler eylemler = yeni Eylemler (sürücü).perform eylemleri.doubleClick(el).perform()

eylemler.moveTo(el).perform eylemleri.doubleClick.perform

action.moveTo(el).doubleClick().build.perform()

**52. Dosya indirme ve yükleme**

• **İndir**

o Selenium'un kendisi dosya indirmelerini doğrulayamaz, indirme bağlantısını tıklayabilir ancak tarayıcının dışına çıkamaz ve

indirilmiş dosya

o Robot ve AutoIT için başka araçlar kullanılmalıdır

• **Yükle**

o Selenium yüklemeyi gerçekleştirir, ancak bunu gerçek kullanıcıdan farklı şekilde yapar

o Adımlar

- Yükleme penceresini tetikleyen öğeyi bulun

- Yüklemek istediğiniz dosyanın yolunu bulun

o Bir Dizgede Depolayın

• Ör: à file=”C:\\Users\\Andy\\Desktop\\folder1\\file.key”;

• Ardından driver.findElement(yükleme düğmesi).sendkeys(dosya);

**53. Açılır menüden seçilen değer nasıl kontrol edilir?**

Arabaları seçinList = new Select(el)

carList.getFirstSelectedOption()

assertequals(“bir metin”,carList.getFirstSelectedOption().getText() )

- 56 -

**54. Select etiketi olmadan açılır menü ile nasıl çalışılır?**

• Açılır listede select etiketi yoksa, select sınıfını kullanamayız

• Açılır listeyi ve seçeneklerini ayrı öğeler olarak ele alın, her öğeyi ayrı ayrı bulun

• Bir seçeneği seçmek için:

o 1. Listeyi bulun ve tıklayın

o 2. Seçeneği bulun ve tıklayın

**55. Seçim etiketi yoksa ne olur?**

• Açılır liste için etiketi bir web elemanı olarak ayrıca seçmelisiniz.

• Ardından tıklama yöntemini manuel olarak kullanın

**56. Çerçeveyi değiştirmek için sözdizimi nedir?**

• Çerçeve, başka bir html belgesinin içindeki bir html belgesidir.

• Web sürücüsü, bir seferde bir sayfa/html belgesini işler. Başka bir çerçeveyi kontrol etmek için her zaman geçiş yapmamız gerekir.

• Driver.switchTo.frame(webelement) à iframe'i bulun ve bir param olarak iletin

• Driver.switchTo.frame(string) à iframe'in kimliğini veya adını bulun ve bir parametre olarak iletin

• Driver.switchTo.frame(int) à dizini bulun ve bir param olarak iletin

**57. Bazen sendKeys çalışmıyor**

• Robot veya AutoIT

• kitaplık==jar dosyası==bağımlılık

**58. Pencereleri değiştirmek için sözdizimi nedir?**

• Ayrı sekmeleri/pencereleri işlemek için o sekmeye geçmeliyiz

• Web sürücüsü, bir seferde bir sayfa/html belgesini işler.

• Başka bir sekmeyi kontrol etmek için her zaman geçiş yapmamız gerekir

• Geçiş yapabilmek için önce pencere tutamağını almamız gerekir.

getWindowHandles() yöntemi driver.switchTo.window(String) // à pencere tanıtıcısı

//her döngü için : driverGetWindowHandles:

Driver.switchTo.window(“tutamak”)

Ifdriver.getTitle==beklenenbaşlık;

Kırmak;

**59. Dosya yüklemek için sözdizimi nedir?**

Public void fileUpload(Stirng yolu){

WebELEment yüklemesi = driver.findELement; Upload.sendKeys(yol)

}

• html'de upload butonunu bulmamız gerekiyor.

• Elemanın etiket girişi olacaktır.

• Ardından, yüklemek istediğimiz dosyanın yolunu ileterek SendKeys yaparız.

**60. Bazen sendKeys/path çalışmıyor**

• Projemizin içindeki bir dosya için dinamik bir yol oluşturma Proje konumuna giden yol:

String projectDir= System.getProperty(“user.dir”) // proje dizini

String file= “src/test/resources/test\_data/myfile.txt”;

Element.sendKeys(projectDir+dosya);

- 57 -

**61. SendKeys()'i çağırmadan metin kutusuna metin nasıl girilir?**

//javascriptExecutor kullan

JavascriptExecutor JS = (JavascriptExecutor)web sürücüsü;

// kullanıcı adını girmek için

JS.executeScript(“document.getElementById('User').value= 'www.google.com'”);

//Şifre girmek için

JS.executeScript(“document.getElementById('pass').value='tester'”);

**62. Selenium WebDriver'da metin kutusundaki ENTER tuşuna nasıl basılır?**

• Selenium WebDriver kullanarak Enter tuşuna basmak için,

• Selenium Enum anahtarlarını sabit Enter ile kullanmamız gerekiyor

• Driver.findElement(By.xpath(“xpath”)).sendKeys(Keys.ENTER);

**63. Herhangi bir çapraz tarayıcı testi yaptınız mı? çapraz tarayıcı testi**

• Tarayıcı türü, ana url, kullanıcı adı, şifre, ortam gibi anahtar kelimeler için bir kontrol dosyanız olduğunu her zaman belirtin.

**64. Sertifika sorununu nasıl çözersiniz?**

• CHROME, IE à DesiredCapabilities yeteneği = DesiredCapabilities.chrome ();

• Jenkins'te à eklememiz gerekiyor . rahatHTTPSdoğrulama

Yanıt yanıtı=RestAssured.given()

.contentType(ContentType.JSON)

.relaxedHTTPSValidation()

.get("https://api.got.show/api/continents");

System.out.println(response.asString());

**65. Web Öğesinin sayfadaki konumunu nasıl doğrularsınız?**

• element.getLocation();

• WebElement sınıfı, öğenin sol üst köşesini döndüren bir get Location yöntemine sahiptir

**66. Sayfa Fabrikası sınıfı?**

• Page Factory sınıfı Selenium ile birlikte gelir.

• Ve sayfa nesne sınıfları oluşturduğumuzda kullanılır.

• Amacı, sınıfta tanımlanan webElement öğelerini başlatmaktır.

**67. Bana salatalık ile test yürütme akışınızı açıklayın.**

• Çalıştırıcı > Özellik dosyası > Senaryo > Adımlar > Adım def > POM kullanarak Selenium kodu

**68. UX ve Restful webServices'i test etmek için hangi araçları kullanıyorsunuz?**

• UX à Kullanıcı Deneyimi. Önce UX'in manuel olarak kabul edilebilir olduğundan emin olun.

• Bundan sonra UI testi olduğu için otomatikleştirmek için Selenium WebDriver kullanıyorum.

• RESTFul API Otomasyonu > RestAssured Library, manuel testler için PostMan

**69. Selenium WebDriver Kullanılarak Tarayıcı Penceresi Nasıl Yeniden Boyutlandırılır?**

• Tarayıcı penceresini belirli boyutlara yeniden boyutlandırmak için, tarayıcı penceresini yeniden boyutlandırmak için 'Dimension' sınıfını kullanıyoruz.

• //Dimensions sınıfının nesnesini oluştur

Boyut d = yeni Boyut(480,620);

• //Geçerli pencereyi verilen n boyutuna göre yeniden boyutlandır

driver.manage().window().setSize(d);

- 58 -

**70. Selenyum'da hangi istisnaları biliyorsunuz?**

• Sık sık **NoSuchElementException yaşıyorum**

• **StaleElementException**

o Öğe tamamen silindi.

o Öğe artık DOM'a bağlı değil.

o StaleElementException'ı nasıl ele alıyoruz;

§ Öğe, 'for döngüsü' içinde DOM à 'deneyin yakalama bloğu'na eklenmemiş

Veya

§ 1. Sayfayı yenileyin ve aynı öğe için tekrar deneyin.

§ 2. Eleman kullanılabilir hale gelene kadar bekleyin

• **TimeOutException**

**71. ASSERT(sert iddia) VS VERIFY(yazılımdan onaylama)**

• Hard assert, bir assert deyimi başarısız olduğunda hemen bir AssertException oluşturur ve test paketi sonraki ile devam eder

@Ölçek. Assert adımları başarısız olursa, testin yürütülmesi o noktada durur! ve varsa bir sonraki teste gidecek!

o (Örnek: sadece basit Assert.assertTrue(boolean);)

• Soft assert, @Test sırasında hataları toplar Soft Assert, bir assert başarısız olduğunda ve devam ettiğinde bir istisna oluşturmaz

assert ifadesinden sonraki adımla. Doğrulama adımları başarısız olursa, başarısız olduğunu bildirir ancak yürütmeye devam eder!

o Örnek: SoftAssert soft=new SoftAssert(); // yumuşak nesne oluşturma için

o soft.assertTrue(boolean);

o soft.assertAll(); // sonuna koy, neyin başarısız olduğunu bildirecek!

**72. Selenium'da bulunan doğrulama noktası nedir?**

• Selenium IDE'de Doğrulama noktaları olarak Selenium Verify ve Assert Commands kullanıyoruz

• Selenium WebDriver'da, doğrulama noktaları için yerleşik özellikler yoktur, tamamen kodlama stilimize bağlıdır. Bazı

Doğrulama noktaları

o sayfa başlığını kontrol etmek için

o belirli bir metni kontrol etmek için

o Belirli bir öğeyi kontrol etmek için (metin kutusu, düğme, açılır menü, vb.)

**73. Metnin var olduğu doğrulansın mı?**

• VerifyTextPresent à , belirtilen metin dizesi sayfanın herhangi bir yerinde BULUNDUysa DOĞRU döndürür; Aksi takdirde YANLIŞ.

• Belirtilen metin dizesi sayfanın herhangi bir yerinde BULUNAMADIysa, VerifyTextNotPresent à DOĞRU döndürür; YANLIŞ olsaydı

bulundu.

**74. Bir web sayfasındaki bir metni nasıl bulursunuz?**

• //tagname[içerir(metin(),'metin')] belirli testler içeriyor

• //tagname[.='text'] tam metni içeriyor bazen Selenyum çalışmıyor

**75. Apple'ın önceki tüm kardeşleri nasıl edinilir?**

• Xpath: "//ul/li[içerir(metin(),'Apple Mobiles')]/precedingsibling::li"

• Bu, "Samsung Cep Telefonları"nı verecektir

**76. Aşağıdaki tüm Apple kardeşlerini nasıl edinebilirim?**

• Xpath: "//ul/li[içerir(metin(),'Apple Mobiles')]/followingsibling::li"

• Bu, önceki tüm kardeşlere (Nokia Mobiles, HTC Mobiles, Sony Mobiles, Micromax cep telefonları) verecektir.

- 59 -

**77. Web Tabloları/ızgaraları nasıl ele alınır?**

• Tablo verileri için kullanılan tablo etiketi ızgara biçiminde düzenlenmiştir

o th etiketi Örnek –

<tr>

<th>FirstName</th> sütun adları en üst satırda

<th>Soyadı</th>

<th>Yaş</th>

</tr>

o </tr> tr etiketi bir satırı belirtmek için kullanılır, tüm sütuna uygulanır td etiketi bir satırdaki bir sütunu belirtmek için Örnek

<tr>

<td>Danny</td> real\_data\_on\_the\_very\_first\_row

<td>Smith</td>

<td>29</td>

</tr>

• Bazı tablolarda tbody bulunur Tablonun verilerini belirtmek için kullanılır, genellikle sütun adlarını içermez ( th )

**78. Excel nasıl kullanılır?**

FileInputStream ExcelFile = new FileInputStream(path);

excelWBook = yeni XSSFWorkbook(ExcelFile);

excelWSheet = excelWBook.getSheet(sayfaAdı);

hücre = excelWSheet.getRow(rowNum).getCell(colNum);

**79. Selenium sürüm 3'ü nasıl buldunuz? Selenium 3, Selenium 2'den büyük ölçüde farklı mı?**

• Selenium 3, Selenium 2'den gelen hata düzeltmelerine sahiptir, ayrıca daha mobil otomasyon odaklıdır.

• Selenium 3'ün “mobil ve web uygulamalarının kullanıcı odaklı otomasyonu için bir araç” olmasını hedefliyoruz.

• İşte değişikliğin özeti.

o WebDriver kullanıcıları için, daha çok hata düzeltmeleri ve 2.

o Selenium Grid hata düzeltmeleri de yapıldı.

o Selenium projesi yalnızca WebDriver API'sini aktif olarak desteklemeyecektir.

o Mozilla, bir zamanlama tuhaflığıyla, Firefox'ta, Firefox 48'den kendi özelliklerini kullanmanız gerektiği anlamına gelen değişiklikler yaptı.

Selenium 2 veya kullandığınızdan bağımsız olarak bu tarayıcıyı kullanmak için geckodriver

o Bildiğimiz gibi Selenium 3.0, Selenium Jar'ın en son sürümüdür.

- 60 -

**UZMAN**

**1. Maven nedir?**

• Koşucu sınıfımı çağıran ve bağımlılıklarımı yöneten POM xml dosyasını çağıran bir oluşturma aracı ve komut istemi aracı

• Komut İstemi mvn arketipi; üretmek

o Proje oluşturur

• Bir # seçin enter'a basın

• Bir # seçin enter'a basın

• Grup Kimliği; com.nameOfProject (genellikle com.example.foo gibi ters çevrilmiş bir alan adı)

• ArtifactID; testmaven projesi

o Sürüm girişi

o Paket girişi

o Y; girmek

**2. Neden Maven? Projenizi etkili bir şekilde geliştirmenize nasıl yardımcı olur?**

• cevap

**3. Maven Eseri Nedir?**

• Yapı, bir Maven deposuna dağıtılan bir dosyadır, genellikle JAR'dır.

• Bir Maven yapısı, derlenmiş bir JAR ve bir "kaynak" JAR gibi bir veya daha fazla yapı üretir.

• Her yapıtın bir grup kimliği (genellikle com.example.foo gibi ters çevrilmiş bir etki alanı adı), bir yapı kimliği (yalnızca bir ad) ve bir

sürüm dizesi. Üçü birlikte eseri benzersiz bir şekilde tanımlar. Örnek:

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>seleniumjava</artifactId>

<sürüm>3.11.0</sürüm>

• Bir projenin bağımlılıkları yapıtlar olarak belirtilir.

**4. Bana maven yaşam döngüsünü açıklar mısınız?**

• Komutlar yalnızca belirli **pom xml** dosyasının bulunduğu dizinde çalışabilir

• 3 yerleşik yapı yaşam döngüsü

o Varsayılan à Proje dağıtımınızı yönetir

o Temizle à Proje temizliğini yönetir

o Site à Projenin site belgelerinin oluşturulmasını yönetir

**5. Bir yapı yaşam döngüsü aşamalardan oluşur**

• Onaylayın à Projenin doğru olduğunu ve gerekli tüm bilgilerin mevcut olduğunu onaylayın

• Derle à Projenin kaynak kodunu çalıştırın (hata olup olmadığını kontrol edin, değilse → başarı inşa et)

à Hedef klasör oluşturulur ve Raporlar burada saklanır

• Test

o Derlenmiş kaynak kodunu uygun bir birim test çerçevesi kullanarak test edin.

o Kodun paketlenmesini veya dağıtılmasını gerektirmemelidir

o Mvn D(VariableName) = testname à Parametreye dayalı olarak belirli testleri çalıştırın

• Paket → Derlenmiş kodu alın ve JAR gibi dağıtılmış bir biçimde paketleyin

• Doğrula → Kalite kriterlerinin karşılandığından emin olmak için entegrasyon testlerinin sonuçlarını kontrol eder

• Yükle → Bağımlılık olarak kullanmak için paketi yerel depoya yükleyin

• Dağıt → Yapı ortamında yapılır, son paketi diğer geliştiricilerle paylaşmak için uzak depoya kopyalar

ve projeler

**6. Maven projesini Eclipse projesine nasıl dönüştürürsünüz?**

- 61 -

• Mvn tutulması

**7. Java projeleri nasıl yapılır?**

1. Klasörler/paketler oluşturun

2. Kitaplıklar/bağımlılıklar ekleyin

3. Sınıf dosyaları oluşturun

4. Derleme

5. Testleri çalıştırın

6. Dağıt

**8. Bağımlılıklarınızı/kütüphanelerinizi nerede bulabilirsiniz?**

• Mvnrepository.com

• Maven çalışmıyorsa projeyi güncelleyin

o Pom dosyanızın içinde bağımlılıklar olduğunda ve güncellemeyi kullandığınızda, maven JAR dosyalarını internetten çeker ve

projenize ekleyin

**9. .m2 klasörü nedir?**

• Jar dosyalarınızın/depolarınızın bilgisayarınızda kaydedildiği yer

**10. POM xml dosyası nedir?**

• Tüm projeyi yöneten bir dosya

• Bir maven komutu çalıştırdığınızda, her şey pom.xml üzerinden yapılmalıdır.

**11. Araçların sürümleri?**

• RestAssured 3.3.0 çıkış tarihi: 2019-01-11

**12. Log4j?**

• Herhangi bir uygulama tarafından kullanılır

• Örnek: LOG4J2 à Apache'den

• Aktiviteyi kaydeder

• Dev, günlüklere bakacak, saate bakacak, IP adresine gidecek ve bir hata varsa ne olduğunu görecek.

• Kaydediciler uygulamaların çok önemli bir parçasıdır ve gerçekleşen her adımı/olayları zaman damgasıyla tutar

• Normalde günlükler programlı olarak .log dosyasına yazılır

• Herhangi bir çerçeveye veya uygulamaya eklemek için hazır araçlar/kütüphaneler vardır.

• Java'da en ünlü kayıt kitaplığı/çerçevesi apache'den LOG4J'dir.

**13. Günlüklerin amacı?**

• Uygulamada karşılaşabileceğiniz sorunları ayıklamamıza yardımcı olun.

• Bazen uygulamada bir hata bulunduğunda geliştiriciler önce logları kontrol eder. Kullanıcının hangi adımları attığını görmek için

alma ve uygulama beklendiği gibi davranmadı.

• Günlükler, sorunun kaynağını bulmanıza yardımcı OLABİLİR (test açısından değil, uygulama açısından)

**14. Test otomasyonunda günlüklerin rolü nedir?**

• ● Test çalıştırmalarımızın durumunu görmek için konsol veya html raporuna bakıyoruz. Bir şey başarısız olursa, oradan buluruz.

• ● Çerçevemizde oturum açmayı uygularsak, otomasyon yürütme adımlarına bakmanın başka bir yolu olacak ve

testimiz başarısız olduğunda sorunu bulmamıza yardımcı olun

- 62 -

**TEST & JUNIT**

**1. TestNG nedir?**

• 500 test durumunuz var → Her test senaryosu için bir Java Paketi ve 500 Sınıfı oluşturuyoruz

Müşteri, duman testi için sadece 40 tanesini çalıştırmanızı istedi → Jasmine'de blokları ve raporlama mekanizması ile hallediyoruz.

• TestNG bir test çerçevesidir

• Merkezi kontrolör: farklı test senaryolarını yönetin ve ardından raporlar, günlükler oluşturun

• Toplu yürütme: 100 test senaryosu ve bunları tek tek çalıştırın

• İsteğe bağlı yürütme: bazı test senaryolarını atlayabiliriz

**2. TestNG'deki iddialar nedir?**

• Testi çalıştırdık ve başlık testi başarısız oldu. Diğer test senaryolarını etkilemeyecek, bu yüzden betiğimizin durmasını istemiyoruz.

o Kritik à durma/başarısızlık İddiası

§ Bir boole argümanı ve String mesajı alır. Bir koşulun doğru olduğunu iddia eder. Değilse, AssertionError,

verilen mesajla birlikte atılır.

o Kritik değil à başarısızlık/devam SoftAssert

§ Soft Assert, bir onaylama başarısız olduğunda bir istisna oluşturmaz ve bir sonraki adımla devam eder.

beyanı ileri sürmek.

**3. JUnit ve TestNG arasındaki fark**

• Açıklamalar; **JUnit:** @Test, @BeforeClass, @AfterClass, @Before, @After, @Ignore

**TestNG** : @Test, @BeforeTest, @BeforeClass, @BeforeSuite, @BeforeMethod,

@AfterTest, @AfterClass, @AfterSuite, @AfterMethod

• Her ikisi de bize yardımcı olacak test çerçevesidir

çalışan otomasyon komut dosyaları.

• TestNG, html raporu sağlar

• TestNG'de @Dataprovider bulunur

salatalık ile aynı açıklama

Veriye Dayalı Senaryo Anahattı

Test yapmak.

• TestNG'de paralel test yapabiliriz,

ancak JUnit paralelliği desteklemiyor

test edin, bu yüzden bunun için soapLab kullanıyoruz.

• TestNG destek grubu testi ancak JUnit

desteklemiyor

• TestNG ve JUnit'in her ikisi de

testi parametreleştir, ancak TestNG

parametreli test konfigürasyonu

yapılandırması çok kolay. İki tane

Parametrelendirmeyi başarmanın yolları

TestNG;

o @Parameters ve TestNG xml dosyası

o @DataPrivider

- 63 -

**4. Çapraz Tarayıcı ve Paralel Test**

• Şu anki projemde, tarayıcılar arası testler için soapLab kullanıyoruz. Ama önceki projemde testng.xml dosyasını kullanmıştım.

• Temel olarak, paketin içinde 3 anahtar var (isim, iş parçacığı sayısı, paralel) ve 2 farklı test oluşturdum, bunlardan biri

• Chrome ve diğeri Firefox içindir.

• Parametre açıklaması da vardır ve ad ve değer içerir, ad tarayıcıdır ve değer Chrome'dur.

<? xml sürümü = "1.0" kodlaması = "UTF 8" ?>

<! DOCTYPE süit SİSTEMİ ...>

< süit ... >

< test adı = "ChromeTest" ... >

< parametre adı = "tarayıcı" değeri = "krom" />

< sınıflar >

< sınıf adı = "deneme takımı..." />

</ sınıfları >

</ testi > <! İlk Test >

< test adı = "FireFox" ... >

< parametre adı = "tarayıcı" değeri = "FireFox" />

< sınıflar >

< sınıf adı = "test süiti…." />

</ sınıfları >

</ testi > <! İkinci Test >

</ süiti > <! süit >

- 64 -

**salatalık ve kornişon**

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

• Son birkaç yılda, gitgide daha fazla BT ekibi, geliştirme süreçlerinde Çevik metodolojiyi takip ediyor.

pazarın hızlı değişimleri. Bu aynı zamanda test ekibi için test senaryolarını ve test komut dosyalarını yönetmede bir zorluktur.

gereksinimler aylık olarak güncellendiğinde değişir. Baştan itibaren uygun bir test yöntemi bulmak,

Çevik bir yazılım projesinin başarısının anahtarı.

• Birçok Çevik ekip, aşağıdakileri kullanarak test sürecinde Davranış Odaklı Geliştirme (veya BDD) yaklaşımını başarıyla uygulamıştır.

Salatalık aracı. Peki, Salatalık nedir? Ve neden Çevik projelerde birlikte kullanılan iyi yaklaşımlardan biridir?

BDD?

• Salatalık, davranışa dayalı bir geliştirme tarzında yazılmış otomatik kabul testlerini çalıştırmak için bir araçtır. onun bir

harika ana özellikler, düz metin işlevsel açıklamasını (Gherkin adlı dilde yazılmış) aşağıdaki gibi yürütme yeteneğidir.

otomatik testler. İşte bir örnek:

**Özellik** : Şifreyi güncelle

**Senaryo** : Yönetici kullanıcı kullanıcı şifresini güncelleyebilir

Yönetici hesabıyla İK sisteminde olduğum **göz önüne alındığında**

**Başka bir** "kullanıcının" şifresini güncellediğimde

**Sonra** şifreyi başarıyla güncellemek için bir mesaj alıyorum

**Ve** kullanıcının şifresi yeni şifreye güncellenir

• Bu harika özellik, aşağıdaki **avantajlarla** BDD yaklaşımını desteklemede birincil bir rol oynamıştır :

o Alan modeli etrafında yapılandırılmış ve tüm ekip tarafından kullanılan bir dil olan Ubiquitous dilinde BDD testleri yazmak

geliştiriciler, testçiler, BA'lar vb. dahil olmak üzere üyeler.

o Bir yazılım ekibinin teknik ve teknik olmayan üyeleri arasında köprüler kurmak

o Geliştiricilerin koduyla doğrudan etkileşime izin verir, ancak iş paydaşlarının anlayabileceği bir dilde yazılır

anlamak

o Son olarak, Salatalık, davranış odaklı bir şekilde yazılmış testleri çalıştıran bir Otomatik Kabul Test Aracıdır.

geliştirme (BDD) stili.

• **Salatalık Aracı, bir projedeki teknik ve teknik olmayan üyeler arasındaki iletişimi geliştirmeye yardımcı olur.**

**2. Salatalıktaki en önemli şeyin ne olduğunu söyleyin, onu benzersiz yapan nedir?**

• Özellikler dosyası, Adım Tanımlamaları, Koşucu Sınıfları, Kanca Sınıfı, Etiketler

**3. Raporlarınızı salatalıkta nasıl görebilirsiniz?**

• Çerçevem, raporları içeren hedef klasörde salatalık raporları klasörü oluşturur.

• Jenkins üzerinde testler yaptığımızda, Jenkins her çalıştırmanın raporunu kaydeder.

• Jenkins işinin ana sayfası her zaman son çalıştırma raporlarına işaret eder.

• Önceki çalıştırmaların tüm raporları yapı numarası altında bulunabilir.

• Hedef klasöre git

• Sistem gezgini ile aç

• Hedefe git>salatalık raporu>dizin, çalıştırdığınız testleri gösterir

**4. Kornişon nedir?**

• Özellik dosyaları tarafından kullanılan dil

• Özellik, Senaryo, Verilen, Sonra, Ne Zaman, Ve, Ama, Arka Plan, Senaryo Anahattı

- 65 -

**5. Salatalık BDD çerçevesinin bileşenleri nelerdir?**

**1. Özellik dosyaları**

o Belirli bir özelliği veya işlevi test eden senaryolardan oluşur

o Özellik ana hikayedir, senaryolar ise hikayenin test durumlarıdır(özellik)

**2. Cukes Runner**

o Testleri sıkı bir şekilde çalıştıran, adım tanımı için kodlar üreten bir sınıf

o @smoketest

o Cukesrunner → CUCKESRUNNER'DA ÖZELLİĞİMİN NEREDE OLDUĞUNU GÖSTEREN BİR ÖZELLİK KONUMUM VAR

BULUNDU

**3. Adım tanımı**

o Gherkin 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

• BAĞIMLILIK BDD BAĞIMLIDIR

• POM.XML DOSYASINDAKİ MVN DEPOSU

• CUCUMBER.IO'DAN SALATALIK BDD

• TDD teknolojilerini birleştirin

• Davranış odaklı

• Akış müşteri davranışını ifade edin → Öğelere odaklanmayın

**6. @CucumberOptions ne yapar?**

• Salatalık testlerinin çalışmasını özelleştirmek için kullanılan etiket

• @CucumberOptions içine şunları ekleyebilirsiniz:

o kuru çalışma

o Eklenti

§ “ **Güzel** ”

• Konsola daha fazla bilgi ekler à Size etiket, senaryo, yöntem bilgisi verir.

• "html:hedef/salatalık raporu" à Hedef/salatalık raporu klasöründe bulunan html raporu oluşturur

"json:hedef/salatalık.json

§ **Etiketler**

• Etiketler, özellik yolunda bulunmalıdır

• Birden fazla etiket ekleyebilir...tags= “@Köpek, @Cat”

§ **Özellik** dosyalarının bulunduğu yerin özellikleri

§ Adım tanımlama adımlarının aranacağı yeri **yapıştırın .** kanca sınıfı da yapıştırıcının 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 testi NG bağımlılığı ekleyin

• CukesRunner'ı AbstractTestNG CucumberTests'e genişletin

**9. Koşucu sınıfınızı etiketsiz çalıştırırsanız ne olur?**

• Tüm özellik dosyaları yukarıdan aşağıya doğru çalışacaktır, ancak yalnızca @CucumberOptions içinde bulunan özellik dosyaları

“özellikler=”

- 66 -

**10. Salatalıkta 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 @Önce → her salatalık senaryosundan önce çalışır

o @Sonra → her senaryodan sonra çalışır (Senaryo geçer veya başarısız olursa her zaman çalışır)

• Class, stepdefinition ile aynı pakette olmalıdır

• Hook sınıfı içinde ekran görüntüleri uyguladım

• DryRun=true ise Hook Class çalışmayacaktır.

• Senaryoyu önce/sonra yöntemimde parametre olarak kullanıyorum

**11. Salatalıkta ekran görüntüsü nasıl alınır?**

• Aftermethod'umda bir kod kullanıyorum:

• TakeScreenShot arayüzünü kullanıyorum

• Ekran görüntüsünü bayt veya dosya olarak saklayabilirsiniz

o @Sonra

public void yırtma(Senaryo senaryosu) {

if(scenario.isFailed()) {

// ekran görüntüsü alma

final byte[] ekran görüntüsü = ((Screenshot Alır)

Driver.getDriver()).getScreenshotAs(OutputType.BYTES);

//ekran görüntüsünü rapora ekleme

senaryo.embed(ekran görüntüsü, "image/png"); }

**12. DDT ile Salatalık nasıl çalıştırılır?**

• Salatalık tabloları kullanıyorum:

**| Ana Sayfa | e-postalar | Belgeler | Projeler |**

• Yöntemi (DataTable arg1) ile elde edersiniz

• DataTable parametresinde bunu şu şekilde değiştirebilirsiniz:

List< YourType > , List<List< E >> , List<Map< K , V >> , ve Map< K , V >

• Liste için sırayla yazdırır

• Harita siparişi yok

**13. Arka Plan Nedir?**

• Salatalığın kendine has bir öncesi yöntemi vardır.

• Kancalı olan java içindir

• Özellik dosyası içinde bir senaryodan ÖNCE çalışan bir adım

• Tüm senaryolardan önce sadece üstüne koyabilir

• İşlem hatları arka plana yerleştirilemez (Yalnızca senaryo ana hatlarında)

**14. Senaryo Anahattı Nedir? vs Senaryo?**

• Salatalıkta senaryo bir kez çalışır.

• Veriye dayalı testler için kullanılır

• Aynı salatalık adımlarına sahip olun, ancak senaryodan sonra anahtar kelime örneklerini kullanarak tablo olarak veri sağlıyoruz

- 67 -

**15. Geçebileceğim değişken türlerini nasıl sınırlayabilirim?**

• Kornişon parantezine ekleyebilirsiniz (İşbirliği | Satış | İşaretleme vb.)

• Ör: @When("^ İşbirliği | Satış | Pazarlama | Etkinlikler | Tümü ) menüsünün üzerine geldim$")

public void i\_hover\_over\_the\_Collaboration\_menu(Dize menüsü) {

geçiş(menü) {

durum "Satış":

BrowserUtils.hover(dashboard.sales); kırmak;

durum "Pazarlama":

BrowserUtils.hover(dashboard.marketing); kırmak;

durum "İşbirliği":

BrowserUtils.hover(dashboard.collaboration); kırmak;

durum "Etkinlikler":

BrowserUtils.hover(dashboard.activities); kırmak;

durum "Tümü":

BrowserUtils.hover(dashboard.all); kırmak;};

**16. Ya iki parametreli bir senaryonuz varsa (sınırlayıcı parametre, tablo parametresi)?**

• Örnek:

o Senaryo: İşbirliği menü seçeneklerini doğrulayın

o SuiteCRM'de oturum açtığıma göre

o İşbirliği menüsünün üzerine geldiğimde

o Ardından, İşbirliği için aşağıdaki menü seçenekleri görünür olmalıdır:

**| Ana Sayfa | e-postalar | Belgeler | Projeler |**

o Bu senaryoda bir masam var, işbirliğini sadece işbirliği ve diğer menü kategorileriyle sınırlamak istiyorum

• Çözüm:

o @Then("^aşağıdaki menü seçenekleri şunlar için görünür olmalıdır:

( İşbirliği | Satış | Pazarlama | Faaliyetler | Tümü ):$")

o public void follow\_menu\_options\_should\_be\_visisble\_for\_Collaboration(Dize menüsü, Liste<Dize> seçenekleri) {

o Dize menüsü 5 menü seçeneğini temsil eder ( İşbirliği | Satış | Pazarlama | Faaliyetler | Tümü )

< String>options, tabloları temsil eder; Ana Sayfa | e-postalar | Belgeler | Projeler |

**17. DDT için salatalık senaryosunu nasıl kullanırım?**

• Mevcut projemde Örneklerle Senaryo Anahattı kullanıyorum

• Senaryo özellik dosyamda, veriye dayalı olarak bir değişken kullandığımda “<variable>” kullanırım

• Ardından Örneklerde:

**| değişken |** sütun adı

| veri1. | 1. satır

| veri 2 | 2. sıra

| veri3 | satır3

**20. Veri odaklı**

● Test verileri koddan ayrılır ve harici kaynaklarda depolanır: Salatalık Örnekleri tablosu, Excel dosyaları, CSV

dosyalar, Veritabanı.

● Veri miktarı o kadar büyük değilse, Örnekler tablosuyla birlikte Salatalık Senaryosu ana hatlarını kullanırım.

● 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'da JDBC kitaplığı ile birlikte SQL sorguları kullanırım.

- 68 -

**18. Haritalar salatalıkta nasıl kullanılır?**

• Senaryo Dışı Bir Anahat Kullanmak

• Senaryo: Bir harita kullanarak kişi oluşturun

o SuiteCRM'de oturum açtığıma göre

o Yeni bir kişi oluşturduğumda:

| ad\_adı | John |

| soyadı | Smith |

| cep telefonu | 801 888 8889 |

o Sonra "John Smith" için iletişim bilgilerini görmeliyim

o Sol taraf anahtardır ve sağ yalnızca 2 sütun değerindedir

• Bir Senaryo Anahattı Kullanma

o Senaryo Anahattı: Bir harita kullanarak kişi oluşturun

o SuiteCRM'de oturum açtığıma göre

o Yeni bir kişi oluşturduğumda:

| ad\_adı | <first\_name> |

| soyadı | <lname> |

| cep telefonu | <cep\_phone> |

| office\_phone |<office\_phone> |

o Sonra "<first\_name> <lname>" için iletişim bilgilerini görmeliyim

o Örnekler: **| ad\_adı | ad | cep telefonu | office\_phone |**

| Michael | Jackson | 1234567890 | 2345678891 |

| bonnie | Garcia | 4569871234 | 4567890987 |

• def adımında yazıyorum;

@When("^Yeni bir kişi oluşturuyorum:$")

public void i\_create\_a\_new\_contact(Map<String,String>contact) {

// kişi oluştur iletişim kutusunu aç

**21. POJO salatalıkta nasıl kullanılır?**

• **contactBean** sınıfı oluşturun

o Tüm değişkenleri ekle

o Alıcı/ayarlayıcıları ekleyin

• Fasulye özellik dosyası oluşturun

• ContactBean sınıfındaki değişkenleri içeren ilk satırı içeren bir tablo oluşturun

o Tablonun altına değerler ekleyin

o Parametreli uygulama yöntemi (List<ContactBean>contacts)

• Senaryo: Kişi oluştur

o SuiteCRM'de oturum açtığıma göre

o Yeni bir kişiyi kaydettiğimde:

| **ad | soyadı** | **ofisTelefon** | **cep telefonu** | **e-posta** |

| Steve | kapılar | 3456758888 | 1234329999 | SteveGates123@gmail.com |

o Sonra "Steve Gates" için iletişim bilgilerini görmeliyim

**22. TestNG kullanarak bir grup test senaryosu nasıl çalıştırılır?**

@Test (gruplar={“smokeTest”,”FonksiyonelTest”})

public void loginTest(){

System.out.println(“Başarıyla Giriş Yapıldı”);

}

- 69 -

**23. Veriye Dayalı Test**

● ***NE ZAMAN*** *:* Bir uygulamadaki bir işlevsellik veya modül, birden fazla veri seti ile test edilmesini gerektirdiğinde (Parametriizasyon),

Çoklu girdiler daha sonra veriye dayalı test ve otomasyon gerçekleştirmemiz gerekiyor.

● Bu senaryolar, otomatikleştirilmesi gereken şeylerden biridir.

● ***NASIL*** *:* Test verileri koddan ayrılır ve harici kaynaklarda saklanır: Salatalık Örnekleri tablosu, Excel dosyaları, CSV

dosyalar, Veritabanı.

● ***FAYDA*** *:* Daha organize, Veri merkezi, Test verileri üzerinde işbirliği - BA, MT'lerden vb. gelebilir

**24. TestNG kullanarak nasıl veri odaklı çerçeve oluşturabiliriz?**

● @DataProvider ek açıklamasını kullanarak bir Veriye Dayalı Çerçeve oluşturabiliriz

@DataProvider(name=”getData”) Public Object[][] getData(){ Object [][] data = new Object[2][2];

Data[0][0] = “firstUid”; Data[0][1] = “FirstPWD”;

Veri[1][0] = “İkinciUid”;

Veri[1][1] = “İkinciPWD”; Veri döndür; }

**25. TestNG'de Gruplar Grubu nasıl oluşturulur?**

● Bu gruplara metagruplar denir.

● Örnek: SmokeTest ve FunctionalTest'i içeren bir grup tanımlamak isteyebilirsiniz.

Test.xml dosyamızı değiştirelim:

<gruplar>

<define name="all">

<include name="duman Testi"/>

<include name = "işlevsel Test"/>

</tanımla>

<koşmak>

<include name = "tümü"/>

</run>

</gruplar>

**26. TestNG kullanarak paralel olarak test senaryoları nasıl çalıştırılır?**

● TestNG'de paralel test yürütmeyi gerçekleştirmek için testng.xml'de “parallel” özniteliğini kullanabiliriz

● suite etiketinin parallel özelliği dört değeri kabul edebilir:

○ Sınıflar → Bir java sınıfı içindeki tüm test durumları paralel çalışacak

○ Yöntemler → @Test ek açıklamalı tüm yöntemler paralel olarak yürütülür

○ Örnekler → Aynı örnekteki test senaryoları paralel yürütülecek ancak iki farklı örnekten oluşan iki yöntem

farklı iş parçacığında çalıştırın. <suite name=”softwaretestingmaterial” parallel=”methods”>

**27. testNG'de bir test senaryosu nasıl görmezden gelinir?**

● Test durumunu yok saymak için, etkin = false parametresini kullanırız.

● @Test notu @Test(enabled=false)

**28. Belirli bir test yöntemi, bir test senaryosu yürütmesinden nasıl hariç tutulur?**

● Test.xml dosyasına hariç tutma etiketi ekleyerek

<sınıflar>

<class name="TestCaseName">

<yöntemler>

<exclude name="TestMethodNameToExclude"/>

</yöntemler>

</sınıf>

</sınıflar>

- 70 -

**29. Belirli bir test grubu, bir test senaryosu yürütmesinden nasıl hariç tutulur?**

● Test.xml dosyasına hariç tutma etiketi ekleyerek

<gruplar>

<koşmak>

<exclude name="TestGroupNameToExclude"/>

</run>

</gruplar>

**30. TestNG sonuçları için raporlar üretmenin farklı yolları nelerdir?**

● TestNG, bir rapor oluşturmak için iki yol sunar

o Dinleyiciler **org.testng** arabirimini uygular . **testListener** ve bir test başladığında gerçek zamanlı olarak bilgilendirilir,

geçer, başarısız olur vs...

o Muhabirler **org.testng.reporter** arabirimini uygular ve tüm süitler tarafından çalıştırıldığında bilgilendirilir.

TestNG.

● IReporter örneği, tüm test çalıştırmasını açıklayan nesnelerin bir listesini alır.

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

● raporları ve günlüğe kaydetmeyi yapılandırın.

● yaygın olarak kullanılan dinleyiciler: ITestListener arayüzü.

● onTestStart, onTestSucess gibi metodları vardır. onTestFailure, TestSkipped'de...

● bu arayüzü kendimize ait bir dinleyici sınıfı oluşturarak uygulamalıyız,

● Ardından, sınıfa dinleyici notunu (@Listeners) eklemeliyiz.

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

● Normal ifade, bir arama düzenini açıklamak için özel bir metin dizesidir.

● Normal ifadeleri steroidlerdeki joker karakterler olarak düşünebilirsiniz.

● Bir dosya yöneticisinde tüm metin dosyalarını bulmak için \*.txt gibi joker karakter gösterimlerini muhtemelen biliyorsunuzdur.

● Normal ifade eşdeğeri.\*\.txt'dir.

**33. "Smoke" anahtar sözcüğünü içeren @Test yöntemlerinde arama yapmak için test.xml dosyasında düzenli ifade nasıl yazılır?**

● “Smoke” anahtar sözcüğünü içeren @Test yöntemini bulmak için normal ifade aşağıda belirtilmiştir.

<yöntemler>

<include name=".\*smoke.\*"/>

</yöntemler>

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

● Test takımlarında zaman birimini belirtiriz ve test senaryoları milisaniye cinsindendir.

**35. @Test(invocationCount=someInteger) ne işe yarar?**

@Test(invocationCount=10)

Genel geçersiz test çantası(){}

● //çağırma sayısı özelliği, TestNG'nin bir test yöntemini kaç kez çalıştırması gerektiğini söyler

**36. @Test(threadPoolSize=someInteger) ne işe yarar?**

● threadPoolSize özniteliği, bir iş parçacığı havuzundan test yöntemini birden çok iş parçacığı aracılığıyla çalıştırmasını söyler

● Not: Çağırma sayısı BELİRTİLMEMİŞSE bu öznitelik yok sayılır

- 71 -

**37. Testte test zaman aşımı ne anlama geliyor?**

● Bir test senaryosunun alması gereken maksimum milisaniye sayısı

@Test1(threadPoolSize=3,invocationCount=10,timeOut=10000)

genel geçersizlik testi() {}

● : // bu örnekte: test1 işlevi üç farklı iş parçacığından on kez çağrılacak, Ek olarak, bir

on saniyelik zaman aşımı, iş parçacıklarının hiçbirinin bu iş parçacığında sonsuza kadar engellenmeyeceğini garanti eder.

**38. @Factory ve @DataProvider ek açıklaması nedir?**

● @Fabrika → ile sınıfın ayrı bir örneğini kullanarak bir test sınıfında bulunan tüm test yöntemlerini yürütür.

farklı veri seti

● @DataSağlayıcı → dataProvider kullanan bir test yöntemi, belirli yöntemlerde birden fazla sayıda yürütülecektir.

dataProvider tarafından sağlanan verilere dayalı olarak süreler.

**39. ek açıklamalar - öncelik**

● Hangi numaradan başladığınız önemli değil Örn: @Test(priority=0)

● DependsOnMethods = “test yöntemi adı” Birden fazla test adı ekleyebilirsiniz

● İlki başarısız olursa, 2. test hiç çalışmayacaktır.

● İlk yöntem başarısız olursa, raporunuz 2. testin atlanacağını gösterecektir.

**40. testNG'de paralel yürütme**

● xml dosyasına yazın.

○ parallel=”testler”thread-count=”4”

● Thread-count, aynı anda kaç tane tarayıcı açmak istediğinizdir.

● xml dosyasında her şeyi çalıştırmak için .\* ekleyebilirsiniz

○ Ör:<paket adı=”.\*”></paket>

● TestNG'nin kendi raporları vardır -xml çalıştırdığınızda size test-output klasöründe raporu verir.

● Html olarak test raporunu içerir

**41. Çerçeve Araçları: Salatalık BDD çerçevesi**

● Junit, Salatalık Java, Maven

● Selenyum, Log4j ekran görüntüleri ile HTML raporlama,

● JDBC, İçiniz Rahat, Apache POI, Git, Jenkins

**42. Çerçeve Araçları: TestNG + Selenyum**

● Java, Maven, TestNG,

● Selenium, Log4J ekran görüntüleri ile Raporları Genişletin,

● JDBC, İçiniz Rahat, Apache POI, Git, Jenkins

**43. Çerçeveniz raporları nasıl oluşturuyor?**

● Salatalık BDD çerçevemiz HTML raporları oluşturur.

● Rapor, özellik dosyaları, etiketler ve adımlar için başarılı/başarısız kapsamını gösterir

● Rapor, her test için tüm adımları içerir Raporda, arızalar için ekran görüntüleri bulunur

**44. Seçici salatalık testleri nasıl yapılır?**

● Etiketler anahtar kelime cukesrunner

● özellik anahtar kelimesi cukesrunner

● etiketler ve özellikler komut satırı kullanılarak da iletilebilir

● mvn testi -Dcucumber.options="--etiket @duman"

- 72 -

**45. Kayıt için ne kullanıyorsunuz?**

● Kayıt için Log4J kullanıyorum. Test yürütmede her zaman önemli adımları günlüğe kaydederim. Bu, bir hata olduğunda hata ayıklamama yardımcı olur.

arıza.

● Log4J, HTML raporlarının yerini almaz.

<bağımlılık>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j- çekirdek</artifactId>

<sürüm>2.11.0</sürüm>

</bağımlılık>

**46. ÖZELLİK DOSYA NASIL ÇALIŞIR?**

● **Özellik** → @tags test edilen şeyin açıklaması. Örnek özellik dosyası;

○ Özellik: oturum açma işlevi → Arka fon:

○ Giriş sayfasında olduğum göz önüne alındığında → Senaryo: 1, Senaryo: 2

○ Arka plan her iki senaryodan da önce çalışır

● **Senaryo** → test edilen senaryonun açıklaması

○ Giriş sayfasında olduğum göz önüne alındığında

○ Ve kullanıcı adı ve şifreyi giriyorum

○ Gönder düğmesine tıkladığımda

○ O zaman profil resmini görebilmeliyim

○ Ancak gönder düğmesi görüntülenmemelidir

● **Verilen →** bir ön koşul

● **ne zaman** → beklenen sonucu tetikleyen koşul Sonra —> beklenen koşul

**47. Test temel Sınıfı nedir? ve Çerçevenizde nasıl uyguluyorsunuz?**

● Test Base sınıfı, testlerimde en çok metot kullandığım sınıftır.

● Test sınıflarım , Test Base sınıfını genişletir ve böylece bu yöntemlere erişim sağlar. Bu , benim

kod yeniden kullanılabilir

● Test yöntemleri öncesi/sonrası bekleme/senkronizasyon yardımcı program yöntemleri.

○ SwitchToWindow(başlık)

○ Web Sürücüsü sürücüsü;

**48. TestNG'de başarısız testler nasıl yeniden çalıştırılır?**

● TestNG çerçevemde, hedef klasördeki testng\_failed\_.xml dosyasında başarısız testler rapor edilir .

● Bu dosyayı pom dosyasına ekleyebiliriz, böylece maven her seferinde başarısız olan testleri çalıştırmayı deneyecektir.

● If, yalnızca testte hatalar olduğunda çalışır.

**49. Hıyarda başarısız olan testler tekrar nasıl çalıştırılır?**

● CukesRunner'da yeniden çalıştırma seçeneğini kullanıyoruz.

● Tekrarı cukes runner'a ekleyin.

● Bu seçenek, başarısız testlerin listesini içeren bir dosya oluşturacaktır.

● Başarısız testlerin bir listesini içeren dosyaya işaret eden ikinci bir koşucu sınıfı oluşturun

● İkinci koşucuyu pom dosyasına ekleyin

**50. Jenkins'de başarısız testler nasıl yeniden yapılır?**

● Jenkins'de, başarısız olan Unit case testlerini yeniden çalıştıran bir eklenti var.

● Böylece, şu seçeneği kullanarak Maven derleme yürütmenizi Jenkins'te yapılandırabilirsiniz:

● Dsurefire.rerunFailingTestsCount=2

- 73 -

**51. SALATALIK TESTLERİNİN PARALEL OLARAK YAPILMASI?**

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

**1.** Bir **salatalık-jvm-paralel-eklenti eklentisi var**

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

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

• Yapılandırmaya bağlı olarak, bu eklenti özellik dosyası başına bir cukes koşucusu oluşturur.

• Her koşucu bir özellik dosyasına işaret edecektir. ve bu cukes koşucuları paralel olarak koşacak.

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

aynı anda kaç testin çalıştığını belirtebiliriz

**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.

<inşa>

<eklentiler>

<eklenti>

<artifactId>maven-surefire eklentisi</artifactId>

<yapılandırma>

<parallel>her ikisi</parallel>

<threadCount>4</threadCount>

</configuration>

</eklenti>

</plugins>

</inşa>

Ancak benim özel projemde, bazı testlerin devam etmesine rağmen testlerin devam etmesini sağlamak için maven failsafe eklentisi ekledik.

başarısız. Bu eklenti, testlerin çalışmaya devam etmesini sağlar

<eklentiler>

<eklenti>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-failsafe-eklentisi</artifactId>

<sürüm>2.18</sürüm>

<yapılandırma>

<testFailureIgnore>doğru</testFailureIgnore>

<skipTests>yanlış</skipTests>

<içerir>

<include>\*\*/runners/\*TestRunner.java</include>

</içerir>

</configuration>

</eklenti>

<eklentiler>

- 74 -

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

bu eklenti testleri paralel olarak yürütür. Bu eklenti yapılandırmasında hangi koşucu dosyalarını çalıştırmak istediğimizi belirtiyoruz. yapabiliriz

aynı zamanda nasıl eş zamanlı testler yapmak istediğimizi de belirtiriz.

<include>\*\*/runners/\*TestRunner\*.java</include>. à eklentisi bu dosyaları çalıştıracak

<threadCount>10</threadCount> à bu, aynı anda kaç tarayıcıya sahip olmak istediğimizi gösterir.

<parallel>classes</parallel> à bu satır, cukes koşucu sınıflarının paralel olarak çalışması gerektiğini söyler

Kaç test çalıştırmak istediğimize ve testleri nasıl parçalamak istediğimize göre cukes koşucu dosyaları oluşturduk.

Her cukes koşucusu belirli kurulum senaryolarına/özellik dosyalarına işaret edecektir.

**Nasıl çalıştırılır?**

o Testleri yalnızca maven komutu olarak çalıştırarak çerçevemizde paralel olarak testler yürütebiliriz

o **mvn doğrulama** à bu testleri çalıştırır ve raporlar oluşturur

o **mvn temiz doğrulama** à önce hedef klasörü siler, ardından testleri çalıştırır, ardından raporlar oluşturur

**Paralelleştirmenin faydaları:**

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

**Paralelleştirmenin zorlukları?**

o uygulaması zor --> yapması kolay değil.

o load --> aynı makinede çok fazla instance açarsak makineyi aşırı yükleyebilir. testlerin çalışmasına neden olacak

yavaşlatır ve başarısızlık oranını artırır.

o bu, GRID kullanılarak farklı makinelerde testler yapılarak çözülebilir.

o Projemde bazı testler paralel olarak çalışmadı.

- 75 -

**API**

**1. API nedir?**

● Bağlantı anlamına gelir. API, istekleri alan ve bir sisteme ne yapmak istediğinizi söyleyen ve

sonra yanıtı size geri döndürür.

● API, **Uygulama Programlama Arayüzü'nün** (yazılım aracısı olan) kısaltmasıdır.

birbirleriyle konuşmak için uygulamalar.

**2. API, Web Servisleri mi?**

● API = *tarayıcı* : Selenium WebDriver, *veritabanı* : JDBC, *MsOffice* : Apache POI

● Web servisleri = bir API iletişim için interneti kullanıyorsa, bu bir web servisidir. \*Tüm web servisleri API'dir.

● UI ile web uygulaması gibi UI (kullanıcı arayüzü) yok ve Selenium Webdriver kullanıyoruz

● Kullanırız:

○ SABUN → XML

○ DİNLENME → JSON, XML, METİN

○ Postacı, İçiniz Rahat Edin Kütüphane

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

● SOAP UI, önde gelen açık kaynaklı çapraz platform API Test aracıdır

● SOAPUI, test uzmanlarının farklı Web API'lerinde otomatikleştirilmiş işlevsellik, regresyon, uyumluluk ve yük testleri yürütmesine olanak tanır.

● SOAPUI, her türlü 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 verili sunucudan veri alır)

● Güncelle → PUT (Hedef kaynağın tüm mevcut temsillerini yüklenen içerikle değiştirir)

● Sil → DELETE (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 → yeniden 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-Kötü Ağ Geçidi, 501-Uygulanmadı, 503-Hizmet Kullanılamıyor)

**6. Yanıt aldığınızda ilk kontrol ettiğiniz şey nedir?**

● Durum alıntısı (200 her zaman Tamam anlamına gelir)

● 404'ün bulunamadığını her zaman kontrol ederiz

● rest-assured.io==> ECS makinesini uzak Masaüstü arama türünde bulma otomasyonu için

**7. Http yöntemleri ve istek türleri**

● Al vücut gerektirmez

● Put , gövde anlamına gelir GÜNCELLEME bilgisi gerektirir

● Gönderi , gövde anlamına gelir CREATE bilgisi gerektirir

● Sil gövde gerektirmez

● GET -> READ , POST -> OLUŞTUR, PUT -> GÜNCELLE, SİL -> SİL

● POST VS PUT

- 76 -

**8. Parametreler API'si**

● 2 TÜR:

○ YOL PARAMETRE(DEĞER URL'NİN PARÇASI OLACAKTIR) SORGU/İSTEK

○ PARAMETRELER (ANAHTAR+ DEĞER FORMATI)

**9. Hamcrest Matcher ne işe yarar?**

• Hamcrest, 'eşleşme' kurallarının bildirimsel olarak tanımlanmasına izin veren eşleştirici nesneler yazmak için bir çerçevedir.

org.junit.jupiter.api.Test'i içe aktarın ;

statik org'u içe aktarın . hamcrest . MatcherAssert . iddia ki ;

statik org'u içe aktarın . hamcrest . Eşleşenler .\*;

public class BiscuitTest {

@Ölçek

public void testEquals () {

Bisküvi theBiscuit = yeni Bisküvi ( "Zencefil" );

Bisküvi myBiscuit = yeni Bisküvi ( "Zencefil" );

assertThat ( theBiscuit , equalTo ( myBiscuit ));

**assertThat** ( **"çikolatalı cips"** , **theBiscuit** . **getChocolateChipCount** (), **equalTo** ( **10** ));

assertThat ( "fındık" , theBiscuit . getHazelnutCount (), equalTo (3));

}

}

// ilk argümanın ikinciye eşit olup olmadığını doğrulayın

*assertThat* ( str1 , *is* ( "Kunkka" ) ) ) ;

*assertThat* ( str1 , *is* (str2)) ;

// ilk argümanın ikinciye eşit OLMADIĞINI doğrulayın

*assertThat* ( str1 , *is* ( *not* ( "Tidehunter" ) ))) ;

// yoksayılan durumu karşılaştır

*assertThat* ( str1 , *equalToIgnoringCase* ( "kunkka" ) ) ) ;

// önce ve sonra boşluğu yok sayarak karşılaştır

*assertThat* ( str1 , *equalToIgnoringWhiteSpace* ( " Kunkka " ) ) ;

// sayıları karşılaştır

*assertThat* ( 10 , *büyüktür* ( 9 ) ) ;

*assertThat* ( 10 , *lessThan* ( 11 ) ) ;

*assertThat* ( 10 , *lessThanOrEqualTo* ( 11 ) ) ;

// boş olmadığını doğrula

*assertThat* ( str1 , *notNullValue* ()) ;

Liste<Dize> liste = Diziler. *asList* ( "bir" , "çok" , "ağaç" ) ;

*assertThat* (liste , *hasSize* ( 3 )) ;

*assertThat* (liste , *includeInAnyOrder* ( "too" , "ağaç" , "bir" ) ) ;

*assertThat* (liste , *hasItems* ( "one" , "too" ) ) ;

Liste<Tamsayı> sayılar = Diziler. *asList* ( 11 , 12 , 13 ) ;

*assertThat* (sayılar , *herItem* ( *büyüktür* ( 9 ))) ;

- 77 -

**10. RestAssured Günlük Kayıt Günlükleri**

● **Günlüğe Kaydetme İsteği**

Given().log().all() // Parametreler, başlıklar ve gövde dahil olmak üzere tüm istek spesifikasyon ayrıntılarını günlüğe kaydet

verilen().log().params() // Sadece isteğin parametrelerini günlüğe kaydet

verilen().log().body() // Yalnızca istek gövdesini günlüğe kaydet

Given().log().headers() // Yalnızca istek başlıklarını günlüğe kaydet

verilen().log().cookies() // Yalnızca istek çerezlerini günlüğe kaydet

verilen().log().method() // Yalnızca istek yöntemini günlüğe kaydet

verilen().log().path() // Yalnızca istek yolunu günlüğe kaydet

• **Yanıt Kaydı**

get("/x").then().log().body()

get("/x").then().log().ifError()

get("/x").then().log().all()

get("/x").then().log().statusLine() // Sadece durum satırını günlüğe kaydet

get("/x").then().log().headers() // Yalnızca yanıt başlıklarını günlüğe kaydet

get("/x").then().log().cookies() // Yalnızca yanıt çerezlerini günlüğe kaydet

get("/x").then().log().ifStatusCodeIsEqualTo(302)

// Yalnızca durum kodu 302'ye eşitse günlüğe kaydet

get("/x").then().log().ifStatusCodeMatches(eşleştirici)

// Yalnızca durum kodu sağlanan Hamcrest eşleştiriciyle eşleşirse günlüğe kaydet

**11. Serileştirme ve Seriyi Kaldırma**

● Serileştirme; bir Java nesnesini API JSON formatına eşlediğimizde (JAVA OBJECT TO JSON);

○ Java nesnesi (POJO(Plain Old Java Object), BEANS) → API JSON/XML ile eşleyin

○ Bir sınıftan bir nesnemiz olduğunda ve onu RESTful API'mizde bir JSON formatına eşlediğinde

{yap: “Toyota”,

Model: “Camry” }

Araba araba = yeni Araba();

car.setMake(“Toyota”);

car.setModel(“Camry”);

verilen().body(car).When().post(uri)

● Seri durumdan çıkarma; API JSON/XML → Java Nesnesine eşleyin (JSON TO JAVA OBJECT)

Araba araba2 = yeni Araba();

car2=ne zaman().get(uri).body.as(car.class);

car.setMake(“Toyota”);

car.setModel(“Camry”);

- 78 -

**12. RestAssured Library ile API/Web Servisleri?**

statik içe aktar io.restassured.RestAssured.\* ;

URI uri = yeni URI(“ ... / yöntemler(get,post)”)

● ALMAK;

Yanıt yanıtı = verilen().accept(ContentType.JSON).When().get(URI);

tepki.then().assertThat().statusCode(200).

and().assertThat().ContentType(ContentType.JSON);

● İLETİ;

Yanıt yanıtı = verilen().ContentType(ContentType.JSON).with().accept(ContentType.JSON)

.and().body(JSONbody).When().post(URI);

tepki.then().assertThat().statusCode(200);

•

statik org.hamcrest.Matchers'ı içe aktarın.\* ;

then().assertThat().body(“Id”,Matchers.equalTo(123));

●

JsonPath json = JsonPath(JSONbody);

json.getString(“anahtar”);

json.getInt(“anahtar”);

json.getList(“key1.key2”);

**13. EndPoint nedir?**

● <protocol>://<hizmet-adı>/<ResourceType>/ResourceID à URI (Tekdüzen Kaynak Tanımlayıcı)

Temel URI / kaynak? parametreler

( http://www.google.com/search?source=book… ) à ? à sorgu parametreleri

**14. Yetkilendirme ve Kimlik Doğrulama**

● kimlik doğrulama --> sen kimsin

● yetki --> hangi haklara sahipsiniz

● Kimlik doğrulama kullanıcı ve şifredir

● Yetkilendirmenin türleri vardır:

○ Yetki yok

○ Temel Yetkilendirme

○ Taşıyıcı Simgesi

○ Yetkilendirmeyi ebeveynden devral

**15. RESTful Web Hizmeti / API**

● REST Temsili Devlet Transferi anlamına gelir

● RESTful, REST mimari konsepti uygulanarak yazılan web servislerine başvurulur.

○ RESTful'da CRUD işlemlerini gerçekleştirmek için GET, POST, PUT, DELETE gibi web servis http yöntemleri kullanılabilir.

○ CRUD = Oluştur à Oku à Güncelle à Sil

- 79 -

**16. Response gövdenizdeki bir değeri nasıl doğrularsınız?**

● Deneyim için: kimliğin doğru numarayı içerdiğini doğrulayın

o *Hamster Matcher* , onay kitaplığıdır.

then().assertThat().body(“Id”,Matchers.equalTo(123));

o JsonPath'e ayrıştırın ve Id değerini okumak için getInt(), getList(), getString() yöntemlerini kullanın.

Ve JUnit Assertion'ı kullanabilirim:

Dize gövdesi = ...thenReturn().body().asString();

JsonPath json = yeni JsonPath(gövde);

assertEquals(123,json.getInt(“Id”));

o Seri hale getirme (POJO) nesnesine (veya Nesne Eşleme)

POJO myPojo = … while().post(url).thenReturn().body().as(Pojo.class);

assertEquals(123,myPojo.getId() );

Ve JUnit Assertion'ı kullanabilirim.

**17. API Kimlik Doğrulama Türleri**

• Temel

**○** önleyici

■ Bir hizmet önleyici olacak şekilde yapılandırılmışsa, bir istemciden kimlik bilgilerini istemeyecektir.

bunu gerektirir.

■ Bir istek, kimlik bilgilerini içermiyorsa, **401 Yetkisiz** durum kodunu döndürür.

○ meydan okundu

■ İstek API'ye ulaştığında API, kimlik bilgileri gerektirdiğini söyler ve ardından müşteri sağlar

kimlik bilgileri.

○ oauth --> kimlik doğrulaması için 3. taraf anahtarların ve belirteçlerin kullanıldığı kimlik doğrulama türleri. Var

2 tür yemin:

■ oauth1 à uygulaması zor

■ oauth2 à daha güvenli

• Özet

○ Temelden daha şifrelidir. https...

**18. SABUN kullanmanın avantajı nedir?**

● REST, çok çeşitli veri biçimlerine izin verirken, SOAP yalnızca XML'e izin verir.

● JSON (genellikle verilerle daha iyi çalışır ve daha hızlı ayrıştırma sunar) ile birleştiğinde, REST genellikle kabul edilir

ile çalışmak daha kolay.

● JSON sayesinde REST, tarayıcı istemcileri için daha iyi destek sunar.

● REST, özellikle değiştirilmemiş ve dinamik olmayan bilgiler için önbelleğe alma yoluyla üstün performans sağlar.

● Yahoo, Ebay, Amazon ve hatta Google gibi büyük hizmetler için en sık kullanılan protokoldür.

● REST genellikle daha hızlıdır ve daha az bant genişliği kullanır. Ayrıca mevcut web sitelerine entegre etmek de daha kolaydır.

refactor site altyapısı. Bu, geliştiricilerin bir siteyi sıfırdan yeniden yazmak için zaman harcamak yerine daha hızlı çalışmasını sağlar.

Bunun yerine, yalnızca ek işlevler ekleyebilirler.

- 80 -

**19. SOAP ve RESTful web servisleri arasındaki fark?**

● RESTful JSON, XML, TEXT'i destekler, ancak SOAP yalnızca XML'i destekler

● REST, SOAP tabanlı web servislerinden daha hızlıdır

**20. URI, amaç ve format nedir?**

● URI, Tekdüzen Kaynak Tanımlayıcı anlamına gelir

● URI'nin amacı, web hizmetini barındıran sunucudaki bir kaynağı bulmaktır.

● Bir URI aşağıdaki biçimdedir:

○ <protokol>://<hizmet-adı>/<KaynakTürü>/<KaynakKimliği>

**21. Projenizde hangi Web Servislerini kullanıyorsunuz?**

● Temsili Aktarım Durumu olan Restful kullanıyorum ve XML ve JSON ile iletişim kuruyorum, ancak şu anki durumum

proje JSON kullanıyor

**22. XML nedir?**

● Bilgi işlemde, Genişletilebilir İşaretleme Dili (XML), kodlama için bir dizi kural tanımlayan bir işaretleme dilidir.

belgeler hem insan tarafından okunabilen hem de makine tarafından okunabilen bir biçimde.

**23. JSON nedir?**

● JavaScript Nesne Gösterimidir (verileri yapılandırmak için minimal, okunabilir bir formattır.)

● XML'e alternatif olarak öncelikle bir sunucu ve web uygulaması arasında veri iletmek için kullanılır.

● Temel olarak, hafif bir XML sürümü

● Anahtarda: Değer biçimi

● Anahtar her zaman çift tırnak içindedir ve eğer dizi çift tırnak içindeyse ve sayılar tırnak içermiyorsa değer

● Tamamen http protokolüne dayalıdır, bu nedenle tarayıcıdaki bağlantıya ulaşır ve sonuçları görür

**24. Swagger'ı biliyor musunuz? havalı nedir**

● Swagger, geliştiricilerin tasarım yapmasına yardımcı olan geniş bir araç ekosistemi tarafından desteklenen açık kaynaklı bir yazılım çerçevesidir.

RESTful Web servislerini oluşturun, belgeleyin ve tüketin.

● Swagger, makinelerin okuyabilmesi için API'lerinizin yapısını tanımlamanıza olanak tanır.

● API'lerin kendi yapılarını tanımlama yeteneği, Swagger'daki tüm mükemmelliğin köküdür.

● xml şemasına benzer ancak Json için

**25. json vs gson**

● JSON, anahtarı ve değerleri olan bir formattır.

● GSON bir dönüştürme işlemidir

○ Java'dan json'a (seri hale getirme),

○ json'dan Java'ya (seri hale getirme)

**26. Nasıl ve nereye istek gönderiyorsunuz?**

● Rest kullandığım için uç noktaları var. Geliştiricilerim genel URL'ler oluşturuyor ve istekler bu URL'ye gönderiliyor

**27. Web hizmetleri dışında herhangi bir API kullanıyor musunuz?**

● - Tarayıcı için Selenium API, veritabanı için JDBC ve API için RestAssured kullanıyorum

**28. API'niz için API dokümantasyon web siteniz var mı?**

● Evet, API belgelerimiz için swagger kullanıyoruz ve API uç noktalarının açıklaması ve yönergeleri buradadır.

- 81 -

**29. Projenizde API'yi nasıl test edersiniz?**

● Mevcut projemde sadece şirketlerimizin api'sini değil, diğer harici api'leri de test ediyoruz.

○ Örneğin, yetkili son kullanıcının bilgilerini veritabanımıza kolayca aktarmak için LinkedIn api kullanıyoruz.

● Bir test kullanıcısı olarak bir API isteği gönderir ve durum kodunu, yanıt gövdesini ve api'nin uç noktalarını kontrol ederiz.

URL beklendiği gibi çalışıyor

○ Örneğin projemde API 57'nin Pozitif/Negatif testini de yapıyorum

● Olumlu - Geçerli istekler, başlıklar, parametreler ve Json gövdesi gönderiyorum ve yanıtın 200/201 olduğunu doğrulıyorum

● Negatif- Durumun 200 olmamasını bekleyen geçersiz istekler, başlıklar, parametreler ve gövde gönderiyorum

**30. Dinlenme API'sini nasıl test edersiniz?**

● Her REST API uç noktasının beklendiği gibi çalışıp çalışmadığını doğrularım.

● Manuel API testi için POSTMAN kullanıyorum ve otomasyon için Java'da RESTASURED kitaplığını kullanıyorum.

● POST,PUT,GET, DELETE türünde istekler gönderirim ve yanıt durum kodunu ve yanıt gövdesini, başlığını doğrularım.

● Ayrıca API'nin pozitif ve negatif testlerini yapıyorum.

● Olumlu testler yaptığımda, geçerli istek parametreleri, geçerli başlıklar, geçerli istek json gövdesi gönderir ve doğrularım

yanıt durum kodu 200 başarılı ve Json yanıt gövdesi verileri de beklenenle eşleşiyor.

● Negatif test yaptığımda, geçersiz istek parametreleri veya geçersiz başlıklar veya geçersiz istek json gövdesi ve

bu yanıtı doğrula

● durum kodu 200 değil ve Json yanıt gövdesi hata mesajı içeriyor.

**31. Tüm API uç noktaları, tüm Http protokollerini kullanabilir mi?**

● Bu URL'nin GET, POST, PUT veya DELETE istekleriyle çalışıp çalışmadığına API geliştiricim karar verir.

**32. API'nizi manuel olarak nasıl test edersiniz?**

● Postacı kullanıyorum → REST API URL'sini test eden bir REST API istemci aracıdır.

**33. API testi için hangi araçları kullanıyorsunuz?**

● Manuel test için postacı

● Emin olun kitaplık

**34. Rest API'deki İstek türleri nelerdir?**

● Al, Gönder, Koy ve Sil istekleri var

○ Okuma verilerini alın

○ Gönderi veri oluşturur

○ Güncelleme verilerini koy

○ Sil, verileri siler

**35. REST API'deki başlıklar nelerdir?**

● Kabul Et.(Content Type.JSON) türünü kullanıyorum - aldığım şeyin JSON veya XML biçiminde olması gerektiğini kontrol ediyor

● Ve ContentType.(Contenttype.Json) - gönderdiğim şeyin JSON formatında olması gerektiğini kontrol ediyor

**36. RestAssured Kitaplığı Nedir?**

● BDD biçiminde olan ve seri durumdan çıkarma ve serileştirme kullanarak Java kodunu entegre etmeye yardımcı olan web hizmeti olmayan bir api.

verileri depolamak, doğrulamak ve doğrulamak için Json'dan veri çıkarın ve bir Java nesnesine dönüştürün.

beklenen biri.

**37. Projenizde Enum'u nasıl kullanıyorsunuz?**

● Yanıt türümün JSON biçimi olduğundan emin olmak için içerik Türü kullanıyorum

- 82 -

**38. JsonPath nedir?**

● Yanıt gövdesini doğrulamanın başka bir yolu

● JsonPath j=response.jsonpath;

**39. Yanıt verilerinin boyutunu doğrulamak için hangi yöntemleri kullanıyorsunuz?**

● Hamcrest'ten Matchers kullanıyorum

○ hasitems()

○ eşittir()

**40. Response arayüzünü nasıl kullanırdım?**

μ Raporlama

● Mvn Verify, başarısız olsa bile testleri çalıştırır (arızayı yok sayar)

○ Tüm testin bitmesini bekler

○ Yapı yapılandırmamızda buna sahip olduğumuz için başarısızlığı yok sayar

<testFailureIgnore>doğru</testFailureIgnore>

○ Verify, testten sonra gelen bir Maven yaşam döngüsüdür

● Bir şey başarısız olursa Mvn testi testi çalıştırmayı durduracak

● Aldığımız orijinal html raporu o kadar iyi değil, istatistiksel verilere ihtiyacımız var

● Ör; "html:hedef/salatalık raporu" → Kaç testin başarılı/başarısız olduğu gibi yüzdeler

● Daha fazla istatistiksel veri raporlaması için *Salatalık Sandviçini* kullanacağız (bu, pom xml'de bir bağımlılık dosyasıdır)

● cukesrunner'da şunu ekleyin; "json:target/cucumber.json"

○ JSON dosyasından bir html raporudur → Bu raporun nasıl çalıştığı, raporu oluşturmak için Json dosyasının kullanılmasıdır.

○ Sürüm 3.15 (videodan)

○ pom'a yeni bir yapı xml ekleyin (zaten pom dosyanızda, TestProject adındaki dosyada)

■ Yalnızca bu json raporlamasını alacaksınız (grafikler ve istatistikler içeren salatalık raporu. Bu rapor,

sadece SİZİN görmeniz için yerel olun, Jenkins için değil) YALNIZCA MVN Verify'ı çalıştırırsanız

■ AMA ÇALIŞTIKTAN SONRA HER ZAMAN BİR JSON DOSYASI (salatalık raporundan farklı) ALACAKSINIZ

TEST, BİLE DOĞRULANMIŞTIR

● Bu JSON dosyası, Jenkins için çok önemlidir - salatalık raporu eklentisi için

● TestProje yapısı:

○ <id>yürütme</id>

○ <phase>verify</phase> - bu nedenle html (json) raporu yalnızca doğrulama kullanılırken oluşturulur

○ <hedefler>

○ <hedef>oluştur</goal> </hedefler>

○ Rapor ayrıca size bir json dosyası da verecektir.

○ Doğrulamayı kullanarak testleri çalıştırmak için, pom dosyasına sağ tıklayın ve maven derlemesine tıklayın…

■ Ayrıca parametreler de ekleyebilirsiniz (koşucu değişkeni ve xml dosyası olan değer gibi) - Hedefleri yazın: doğrulayın

● Bunu komut satırında çalıştırmak için

○ pom dosyasının konumuna gidin ve mvn doğrulama yazın

○ Sözdizimi mvn<lifecycle/hedef> şeklindedir

● mvn doğrulama kullanarak Yürütme Sırası

1. Pom dosyasına karşı çalıştırın

2. Pom dosyası xml dosyasını çalıştırır

3. Xml, cukesrunner dosyasını çalıştırır

4. Cukesrunner salatalık özellik dosyasını/testini çalıştırır

● json salatalık raporu ekran görüntüsü gösteriyor mu?

- 83 -

**41. 100 limitli parametreye ve çalışan id = 100 yol parametresine ihtiyacım olan bir yöntemi nasıl yazardım?**

● Yazardım;

○ .and().params("sınır",100)

○ .and().pathParams("çalışan\_kimliği", 110)

**42. Backend-API nedir?**

● Uygulama mantık kodunun olduğu yerdir. Koşullarınız vb.

● Nasıl test edilir?

1) Manuel olarak → Postacı vb. araçları kullanarak İstek göndererek ve yanıtları doğrulayarak.

2) Otomasyon → Java + RestAssured Kitaplığı

- 84 -

**HTML ve CSS**

**1. HTML kodunda kullanılan <div> etiketi nedir?**

● <div> öğeleri ve < div > etiketi, diğer sayfa öğelerini kapsayana kadar bir kapsayıcıdan başka bir şey değildir ve

HTML belgesini bölümlere ayırır.

● Web geliştiricileri, HTML öğelerini bir arada gruplandırmak ve birçok öğeye CSS stilleri uygulamak için <dib> öğelerini kullanır.

bir Zamanlar.

- 85 -

**SQL**

**1. SQL biliyor musunuz?**

● Evet, SQL Sorguları ve DDL ve DML komutları yazma konusunda çok rahatım.

● Şu anda AMAZON CLOUD SERVER'da çalışan Oracle veritabanı ile çalışmaktadır.

● DDL (Veri tanımlama dili): CREATE , ALTER, DROP, TRUNCATE..

● DML(Veri işleme dili): SELECT, DELETE, INSERT, UPDATE

**2. SQL?**

● Yapılandırılmış sorgu dili. Db'deki verileri yönetmek ve işlemek için kullanılır.

● Çeşitli görevler için ifadeler sağlayın

○ Veri sorgulama

○ Tabloya satır ekleme, güncelleme, silme

○ Nesneleri oluşturma, değiştirme, değiştirme ve bırakma

○ Veritabanına ve nesnelerine erişimi denetleme

○ Veritabanı tutarlılığı ve bütünlüğü

**3. SQL deyimlerinin kategorileri nelerdir?**

**4. Bana TCL'den bahseder misiniz?**

● SQL dili dört tür birincil dil ifadesine ayrılmıştır: DML, DDL, DCL ve TCL.

● Bu ifadeleri kullanarak, veritabanı nesneleri oluşturarak ve değiştirerek bir veritabanının yapısını tanımlayabiliriz ve

güncellemeler veya silmeler yoluyla bir tablodaki verileri değiştirin.

● Ayrıca, tek bir iş birimi oluşturmak için hangi kullanıcının veri okuyabileceğini/yazabileceğini veya işlemleri yönetebileceğini de kontrol edebiliriz.

**5. Sürümler**

● Java 8 → 2014 mevcut Java 7 → 2011 - 2014 Java 6 → 2006 - 2011

Selenyum 3.5.3

**6. Veritabanı Şeması?**

● Tüm tabloları ve sütun adlarını, veri türlerini ve PK, FK'yi ve tabloların birbirleriyle nasıl ilişkili olduğunu içeren bir diyagram gibidir.

**7. SQL yan tümcesi?**

● SEÇ ve KİMDEN

**i. DML (Veri Manipülasyon Dili)**

● DML ifadeleri bir tablodaki kayıtları etkiler. Bunlar temel

birkaç kayıt seçme gibi veriler üzerinde yaptığımız işlemler

tablodan, yeni kayıtlar ekleme, gereksiz silme

kayıtlar ve mevcut kayıtları güncelleme/değiştirme.

**ii. DDL (Veri Tanımlama Dili)**

● DDL deyimleri, bir veritabanını veya tabloyu değiştirmek/değiştirmek için kullanılır

yapı ve şema. Bu ifadeler tasarımı ele alır ve

veritabanı nesnelerinin depolanması.

**iii. DCL (Veri Kontrol Dili)**

● DCL ifadeleri, kullanıcıların erişim düzeyini kontrol eder.

veritabanı nesnelerinde var.

**iv. TCL (İşlem Kontrol Dili)**

● TCL ifadeleri kontrol etmenize ve yönetmenize izin verir

SQL içindeki verilerin bütünlüğünü korumak için işlemler

ifadeler.

- 86 -

**8. Ne tür bir Veritabanı testi yapıyorsunuz?**

● Çoğunlukla Veritabanı doğrulamaları yapıyorum.

● Ön uçta değişiklik yapıyorum veya veri ekliyorum (ödünç oluştur) ve veritabanında doğruluyorum. Ön uçtaki veriler,

DB

● Ayrıca RESTapi kullanarak değişiklikler yapıyorum ve değişikliklerin Veritabanında da başarılı olduğunu doğruluyorum.

● Ayrıca DB geçiş sürecini de destekliyorum. Kodum, JDBC kullanarak Sybase'e (eski veritabanı) bağlanır ve ardından Oracle'a bağlanır

(YENİ DB) ardından verilerin taşındığından emin olmak için kayıtları karşılaştırın

**9. RDBMS**

● İlişkisel veritabanı yönetim sistemi

● Veriler birbiriyle ilişkili tablolar halinde düzenlenir

○ Nasıl ilişkilidirler?

■ Birincil Anahtar (benzersiz ve NULL değil) ve Yabancı Anahtar (yinelenen ve NULL)

○ Ne tür bir veritabanı sistemi konusunda uzmanlığınız var?

■ SQL ve Oracle gibi RDBMS

**10. Kısıtlamalar nelerdir?**

● Tablo sütununun uyması gereken özellikler.

● Sütunların, verilerin nasıl depolanabileceğini tanımlayan kısıtlamaları vardır.

○ Birincil Anahtar: benzersiz ve NULL DEĞİL

○ Yabancı Anahtar: yinelenen ve NULL ve PK'de olmayan verileri ekleyemez

○ Benzersiz Anahtar: yalnızca benzersiz değer

○ Null: null olabilir

○ Null değil: null olamaz

**11. SQL'deki veri türleri?**

● Sayı

● tamsayılar

● karakter → char(20): 20 yıllık boşluklar bellekten alınır

● varchar → varchar(30): varchar2 belleğinden 5 boşluk

● boole

● tarih

● para birimi

**12. SQL select deyimleri için yetenekler**

● Projeksiyon → Bir sorgu tarafından döndürülen tablodaki sütunları seçin

● Seçim → Bir sorgu tarafından döndürülen tablodaki satırları seçer

● Katıl → Aralarındaki bağlantıyı belirterek farklı tablolarda depolanan verileri bir araya getirir

- 87 -

**13. DML (Veri Manipülasyon Dili) vs DDL (Veri Tanımlama Dili)**

DML komut eylemleri geri yüklenebilir. DDL komut eylemleri geri yüklenemez / geri **alınamaz** .

**Komutlar** :

● tablo adından **SEÇ** ; ( okuyun )

● tablo adı değerlerine **INSERT (...); (** ekle )

● **GÜNCELLEME** tablo adı SET değeri NEREDE konum;

● Tablo adı WHERE konumundan **DELETE ; (** satırlar )

● **BİRLEŞTİRMEK**

**Komutlar** :

● **CREATE** tablo tablo adı (sütun1, sütun2 …);

● **ALTER** tablo tablo adı değeri değiştir;

● **TRUNCATE** tablo tablo adı; ( tüm tablo verilerini sil )

● **DROP TABLE** ; ( yapı ile tüm tabloyu sil )

● **ADINI DEĞİŞTİRMEK**

● **YORUM**

**14. Oracle'da PL / SQL fonksiyonları ve Tetikleyiciler?**

● Bir DML ifadesi yürütüldüğünde bir tetikleyici otomatik olarak tetiklenir.

● 2 tür işlev vardır:

○ Prosedür (Java'daki void yöntemi gibi)

○ İşlev (Java'daki geçersiz olmayan (dönüş) bir yöntem gibi)

**15. SQL JOIN JOINS**

.

**16. BİRLİK**

● Union, iki sorgunun sonuç kümelerini birleştirir ((1) tablo adından sütun adlarından seç {UNION} (2) seç

tablo adından sütun adı)

**17. En yüksek ücretli 3 çalışan nasıl bulunur?**

● MySQL'de \*\*oracle'da ROWNUM var

● Çalışanlardan maaş, ad, soyadı SEÇİN

Maaş TARAFINDAN SİPARİŞ AÇIKLAMA SINIR 3;

**JOIN (INNER) JOIN**

birden çok tablodan veri alınırken kullanılır

ve yalnızca eşleşen verileri döndürür

**SOL DIŞ KATILMA**

birden çok tablodan veri alınırken kullanılır

ve soldaki tabloyu ve eşleşen herhangi bir hakkı döndürür

tablo kayıtları.

**SAĞ (DIŞ) BİRLEŞTİRME**

birden çok tablodan veri alınırken kullanılır

ve sağdaki tabloyu ve eşleşen soldaki tabloyu döndürür

tablo kayıtları.

**TAM (DIŞ) BİRLEŞTİRME**

birden çok tablodan veri alınırken kullanılır

ve her iki tablo kaydını da döndürür, eşleştirir ve

eşleşmeyen

- 88 -

**18. Çalışanlarda mükerrer isimler nasıl bulunur?**

**●** Çalışanlardan ad, COUNT (ad\_adı) SEÇİN

ada GÖRE GRUPLA

HAVING (COUNT(ad\_name>1);

**19. Maaşları ortalamanın altında olan çalışanlar nasıl bulunur?**

● ADI SEÇİN, çalışanlardan maaş

NEREDE maaş<= (çalışanlardan AVG(maaş) SEÇİN);

**20. Her departmanda maksimum maaş nasıl bulunur?**

● AD SEÇ, MAKS(maaş)

d departmanından SOL DIŞ KATIL çalışan e AÇIK (d.department\_id = e.department\_id)

GROUP BY departmanı\_kimliği;

**21. En düşük maaşlar nasıl bulunur?**

● Çalışanlardan ad, soyad, maaş, iş\_kimliği SEÇ

● NEREDE maaş = (Çalışanlardan MIN(maaş) SEÇİN);

**22. Çalışanların en yüksek ikinci maaşı nasıl bulunur?**

Çalışanlardan MAKS(maaş) SEÇİN

NEREDE maaş IN DEĞİL (çalışanlardan MAKS(maaş) SEÇİN);

**23. SQL Geliştirici**

● Geliştirme ortamı (sorguları kullanarak veritabanını manuel olarak test etme)

○ Sürüm 2.1 -2009 - 3.0 - 2011

○ Sürüm 4.0 - 2013(en son)

● Geçiş sürümüne sahiptir(1.2) → kullanıcılara üçüncü taraf DB'deki verilere göz atmak ve bunlardan geçiş yapmak için tek bir nokta sağlar

DB'den Oracle'a

● Windows, Linux ve Mac OS x'i destekler

**24. SQL İfadeleri Yazma**

● Anahtar kelimeler büyük harfle, sütunlar ve tablo adları küçük harfle yazılır

● İfadeler büyük/küçük harfe duyarlı değildir

● Cümleler genellikle ayrı satırlara yerleştirilir

● Anahtar kelimeler kısaltılamaz veya satırlara bölünemez

**25. Aritmetik İfadeler**

● Operatörleri herhangi bir tümcede kullanırsınız (From tümcesi dışında)

● Tarih ve Zaman Damgası ile - yalnızca toplama ve çıkarma kullanılabilir

● Toplama (+), Çıkarma (-), Çarpma (\*), Bölme (/)

**26. Tarihlerle Çalışmak**

● Varsayılan tarih görüntüleme biçimi: DD-MON-RR

sistem tarihi işlevi

Tarih ve saati döndürür

İkiliden sysdate'i seçin ;

**27. SQL ile deneyiminiz var mı?**

**●** Evet, ilişkisel veritabanlarıyla çalıştım ve DDL ve DML komutlarında çok rahatım

- 89 -

**28. Arka uç test çerçevesi**

● Seri durumdan çıkarma ve serileştirme işleminin bu şekilde gerçekleşmesi için çerçevemde RESTASURED kullanmak

Json yanıtını bir java toplama veri yapısında depolayın ve verileri beklenen değerle belirtin (ayrıca Java'da depolanır)

veri yapısı)

● ÇERÇEVEMİ ÇAĞRILAMADAN ÖNCE manuel test için postacı kullanıyorum

**29. Arka Uç Testi**

**30. Herhangi bir arka uç/veritabanı testi yaptınız mı?**

● Evet, veritabanlarıyla çalışma konusunda çok deneyimim var.

● Ve SQL sorguları yazmakta çok rahatım.

● Oracle, MySQL, SQL Server gibi İlişkisel Veritabanları üzerinde çalışma deneyimim var.

● İlişkisel olmayan veritabanlarıyla çalıştınız mı?

● Tecrübem yok ama JSON formatı gibi olduğunu biliyorum.

● Veritabanı ve ben JSON dosyalarıyla çalışma konusunda iyi bir deneyimim var.

● Ve çabuk öğrenen biriyim

**31. Veritabanı test çerçevesi**

● Manuel test için SQL sorguları üretmek için SQL geliştiricisi kullanıyorum

● OTOMASYON İÇİN; Oracle veritabanından bir CONNECTION alıp sonra oluşturarak Java'yı entegre etmek için JDBC kitaplığını kullanıyorum

STATEMENTS SQL sorgularını kullanır ve ardından verileri bir RESULTSET nesnesine depolar.

**32. Meta veri nedir?**

● Verilerle ilgili MetaData Verileri

ResultSetMetaData rsmd = rs.getMetaData();

int columNum=rsmd.getColumnCount();

**33. Veri Yapıları ve Neden İhtiyacımız Var?**

● Veri yapıları, verileri verimli bir şekilde organize etmenin bir yoludur.

● manipülasyon: Ekleme, arama, okuma, veri silme.

● Java veri yapılarını her zaman uygulamamızdan, veritabanımızdan veya API'den veri okumak ve veri depolamak için kullanırım.

- 90 -

**34. db'yi nasıl bağlayabiliriz?**

Bağlantı bağlantısı = DriverManager.getConnection(URL, kullanıcı, parola);

İfade ifadesi = bağlantı.createStatement();

ResultSet sonuç kümesi = deyim.executeQuery(“sorgu”);

sonuç kümesi.close();

deyim.close();

bağlantı.kapat();

// Bağlantıdan sonra;

DatabaseMetaData db = bağlantı.getMetaData();

// sonuç kümesinden sonra;

ResultSetMetaData rs = sonuçSet.getMetaData();

● Bağlanamamak bir istisna atarsa:

- SQLException (hatalı URL veya kimlik bilgileri)

- ClassNotFoundException (JDB sürücüsü sınıf yolunda değil)

**35. JDBC için Bağımlılık**

<bağımlılık>

<groupId>oracle</groupId>

<artifactId>ojdbc6</artifactId>

<sürüm>11.2.0.3</sürüm>

</bağımlılık>

**36. Prosedür Nedir?**

● Saklı yordam, veritabanında oluşturulan ve depolanan bir grup SQL ifadesidir.

● Bir saklı yordam giriş parametrelerini kabul eder, böylece tek bir yordam ağ üzerinden birkaç kişi tarafından kullanılabilir.

farklı girdi verileri kullanan istemciler.

● Bir saklı yordam, ağ trafiğini azaltacak ve performansı artıracaktır. Bir saklı yordamı değiştirirsek, tüm

istemciler güncellenmiş saklı yordamı alır. Saklı yordam oluşturma örneği

CREATE PROSEDURE test\_display AS

tb\_test'ten Adı, Soyadı SEÇİN ;

EXEC test\_display;

91

**Git ve GitHub**

**1. GitHub nedir?**

● Sürüm kontrol sistemi

● Belgelerin yeni/eski sürümünün kaydını tutar

● Dosya kümesini yönetir/depolar

**2. Depo nedir?**

● Dosyaların kaydedildiği klasör ve

● Tek, dosya koleksiyonları veya tek projeler içerebilir.

**3. Uzak ve Yerel Depo Nedir?**

● *Uzak Depo:* Sunucuda barındır (GITHUB) Değişikliklerimiz yerelden uzak depoya gidiyor

*● Yerel Depo:* Tipik olarak bilgisayarınızda -Değişikliklerimiz burada Çalışma Dizini, dizin ve HEAD'den oluşur.

**4. Git komutları nelerdir?**

● **Ekle** : hazırlama alanına ekle

● **Taahhüt** : çalışma dizininden ve yerel depodan ekleyin

● **Push** : uzak depoya ekle

● **Çekme** : değişiklikleri uzaktan çalışma dizinine alın

● url ile klonla: url'yi dizine klonlar

● Git sürümü: size git sürümünü verin

● Git durumu: hangi şubede olduğunuzu, izlenmeyen değiştirilmiş dosyaları gösterir

○ Menşei: uzaktan kumandanın adı ○ usta: şubenin adı

● Git ekle:

○ Hazırlama alanına ekleme ○ Özyinelemeli ekleme ○ Her şeyi ekler

● **git commit -m** : "mesaj tüm dosyalar için geçerli olacak"

● **git push** : köken nameOfBranch

● **git yoksay** :

○ Notepad.gitignore à Hazırlama alanına eklemek istemediğiniz dosyaları not defterine ekleyin

○ GÖZDEN GEÇİRMEK İSTEDİĞİNİZ DOSYALAR İÇİN .GITIGNORE DOSYASINI REPO YAPMAK ZORUNDASINIZ

GIT'TE GÖSTERİLDİ

○ Bazı dosyalar önemli değil ve git'e itilmemeli

● file-git à GIT'i kaldır ADD REMOVE POM à COMMIT à à AND PUSH Kendi şubesini oluşturma

● checkout branch -git à Git checkout -b nameOfBranch master

**5. Git'i terminalde nasıl kullanırım?**

● yeni repo-git oluştur

● echo "# SqlMentor" >> README.md

git init

git README.md ekle

git commit -m "ilk taahhüt"

git uzaktan kaynak ekle

https://github.com/Andylam224/SqlMentor.git git push -u kaynağı

usta

● mevcut bir repo-git'i itin

○ git uzaktan kaynak ekle https://github.com/Andylam224/SqlMentor.git

git push -u kökenli usta

● Varsayılan düzenleyici

92

**6. GIT Komutları?**

git init

git ekle.

git commit -m "yorumum"

-------------------------------------------------- ----------------

git bilgisi

git günlüğü

git push -u kökenli usta

git itme

-------------------------------------------------- ---------------

git init

git remote Origin URL'sini ekle // https://url adresini URL yerine kopyala yapıştır

git add src/ // sadece bu klasörü eklemek istersem

git commit -m "yorumum"

git günlüğü

git push -u kökenli usta

**7. En son sürüme mi dönüyorsunuz?**

// ikisini de yazmamız gerekiyor

git getirme kaynağı

git reset --hard Origin/master

**8. Bir seferde çift dosya ekleme?**

git dosya1 dosya2 dosya3 ekle //

**9. GİT Şube şubeleri?**

git rm dosyası.java

git commit -m "kaldırılıyor"

git push orijin ustası

git Branch BranchName : - Şube oluşturma

git şubesi: - şube yöneticisini kontrol etme

git checkout BranchName :- name, geçiş yapmak istediğiniz bir şube adıdır

git branch -d BranchName : - yerelde brach siliniyor

git Push Origin : deleteBranchName : - Uzaktan (gitHub WebSite) yerel (intellij) üzerinde silinen Şube siliniyor

git branch -a : - Yerel'de silinen tüm dalları kontrol etme (ancak uzaktan değil)

git checkout -b BranchName : - Şube oluşturma ve yeni şubeye geçiş

git merge BranchName : - Şube birleştirme

git push --set-upstream Origin BranchName : - Şubeyi yerelden (intellij) uzaktan uzağa (gitHub WebSite) itme

git fetch Origin BranchName : - Şubeyi uzaktan (github WebSite) yerele (intellij) çekmek

git push Origin branch1:branch2

git çekme kökenli şube1:dal2

93

**10. Şubeyi master ile birleştirme**

-- ikinci şubenize gidin sonraki adımları uygulayın

git ekle.

git commit -m "yorumunuz"

-- ana şubenize gidin

git birleştirme "şubeAdı"

--eğer birleşmiyorsa ana daldan git commit -m "yorum" yapmalıyız

**11. Yeni Şubeyi GitHub deposundan Yerel yöneticiye, kod GIT Komutlarında Değişiklikler ile birleştirme?**

git getirme Origin BranchName

git checkout ŞubeAdı

git şube // yeni şubede olmalısın

git ödeme ustası

git branch // ana dalda olmalısın

git merge BranchName // birleşecek. herhangi bir çakışma varsa düzeltmeniz gerekir, birleştirme çakışması yoksa geçer

git branch -a // hem yerel hem de uzak dalları görebiliyoruz

git klonu klonlamak istediklerinizin URL'si // ve bağlantıyı kopyaladıktan sonra

git getirme // ctobi obnovit obnovleniya v glavnom

git birleştirme

git log --graph // taahhüt edilenleri ve olanları gösteren

git log --graph -- çevrimiçi // neler olduğunu tek satırda gösteriyor

--eğer çakışma varsa projeye gidin sağ tıklayın -> git -> çakışmayı çöz -> merge :

**wq ve kaçış**

**12. GITHUB URL'SİNİ KONTROL EDİN?**

git uzak -v

git config --get remote.origin.url

git uzaktan gösteri kökeni

git config --get remote.origin.url

**13. Çekme talebi nedir?**

git birleştirme fetch\_head --allow-ilgisiz-geçmişler

● Hızlı ileri sarma olmayan bir sorunu çekme sorunu çözüldü

● Ardından kaçışa basın

○ Shift “:x!” düğmesine basın. → Kaydet ve çık

○ ":q!" → Kaydet ve çık yok

**14. Çekme isteği nedir?**

● Git birleştirme fetch\_head --allow-ilgisiz-geçmişler

○ Hızlı ileri sarma olmayan bir sorunu çekme sorunu çözüldü

**15. Git'te çatışmayı nasıl çözersiniz?**

● deponuz → cd ~/<repo\_dizini>

● Son sürüm deposunu çekin → git çek

● Kaynak şubeye göz atın → git checkout <feature\_branch>

● Hedef şubeyi kaynak şubeye çekin → git çekme kaynağı <destination\_branch>

● Çakışmaları düzeltin ve ardından sonucu taahhüt edin.

94

**JIRA**

**1. JIRA nedir?**

● Proje yönetimi aracı ve kusurların izlenmesine yardımcı olur

○ Planlama ve zaman yönetimine izin verir

○ Teslim tarihlerini/atamaları takip eder

● Yalnızca biriktirme listesinde ve aktif sprintlerde test edici

**2. Hata takibi için hangi araçları kullanıyorsunuz?**

● JIRA, içindeki tüm işleri bir Sorun olarak ele alır

● Bu nedenle, JIRA'da bir kusur yaratmak, "Bug" türünde bir sorun yaratmak olacaktır.

● Arıza bildirimi:

○ kusur kimliği

○ Kusur başlığı

○ Kusur açıklaması (yeniden üretme adımları)

○ Çevre bilgisi

○ Ekran görüntüsü(ek)

○ önem

○ Geliştiriciye atayın

**3. Aktif Sprint Tahtası Nedir?**

● *İş Akışı:* Yapılacak>Devam Ediyor(burada da engellenmiş olabilir)>İncele(Teknik lider, şuraya geçmeden önce kodu inceleyin

bitti>Bitti

*● Engellendi:* Hikayeye gidin ve seçeneklere tıklayın ve daha fazla seçeneğe tıklayın> engellendi

○ Neden engellendiği hakkında bir yorum yazın

○ Scrum master en kısa sürede uğraşmak zorunda kalacak

○ Blokta bir günden fazla hiçbir şey olmamalıdır

○ Düzeltildiğinde, engelleyici çözüldü olarak değiştirebilirsiniz

**4. Jira terimleri nelerdir?**

● Sorun → Yapmanız ve düzeltmeniz gerekiyor

● Sorun Türleri

○ Hikaye

○ Görev

○ Böcek

○ Epik

**5. Epic ve biletlerin farkı nedir?**

● Epic BA tarafından yazılır, biletler testçiler tarafından oluşturulur

● Açıklama kutusu

○ Hata bildirme örneği

○ sen kutuya yaz

■ Hata ne hakkında

■ Hangi işlevsellik bozuluyor

■ Hatayı yeniden oluşturma adımları nelerdir (gerekli verilerle)

■ Hata raporunu ve ekran görüntüsünü ekleyin

■ Beklenen sonuçlar

■ Fiili sonuç

95

**6. JIRA'dan Kullanıcı Hikayelerini nasıl otomatikleştirirsiniz?**

● Açıklamaya bakın - Çevik hikaye

● Özellik dosyası oluşturun ve dosyayı Jira story.feature olarak kaydedin

○ Kabul kriterlerinde yer alan senaryo ekle

● cukesRunner'ı dryRun=true ile çalıştırın

● Yöntemleri uygula

● JIRA'DA TEST VAKALARINI OTOMATİKLEMEDEN ÖNCE ÖNCE ÖNCE DAİMA MANUEL TEST EDİN

**7. Selenium'u Jira ile nasıl entegre edersiniz?**

● Selenium'un Jira ile yerleşik bir entegrasyonu yoktur.

● Ancak selenyum test çerçevesini Jira ile entegre eden eklentiler var.

○ Xray (Jira eklentisi, Jenkins eklentisi)

○ Zephyr (Jira eklentisi)

96

**JENKINS**

**1. JENKINS nedir?**

● Sürekli Entegrasyon ve Dağıtım aracı. Jenkins'in 3 bileşeni

○ 1. Kod değişikliği à Devs, uygulama kodunda değişiklik yapar

○ 2. Test à CI aracı, değişiklikleri otomatik olarak alır ve uygulamayı test eder

○ 3. Deploy à CI aracı, uygulamayı değişikliklerle birlikte dağıtır

**2. Jenkins işi nedir?**

● Her şey bir iş yaratarak yapılır

● Jenkins'in programına göre gerçekleştirdiği bir görev

● Birkaç adımdan oluşan

● Ne zaman çalışacağını belirleyen bir tetikleyiciye sahip olabilir

● Çalıştırmanın sonuçlarını otomatik olarak raporlar

**3. Sürekli Entegrasyon nedir?**

● CI, geliştiricilerin kodu günde birkaç kez paylaşılan bir havuza entegre etmesini gerektiren bir geliştirme uygulamasıdır.

● Yazılımın kodu her değiştirildiğinde, otomatik olarak oluşturulur ve test edilir.

**4. Sürekli dağıtım nedir?**

● Kod değişiklikleri otomatik olarak oluşturulur, test edilir, dağıtılır ve üretime sunulmak üzere hazırlanır

● Her check-in daha sonra otomatik bir yapı tarafından doğrulanır ve ekiplerin sorunları erken tespit etmesine olanak tanır.

**5. Jenkins'i koruyor musunuz?**

● Özel Ekip, çevre ekibi, mimari ekip tarafından yapılır.

● Ancak onlara testlerim ve yapılandırma bilgilerimin yanı sıra bildirim göndermeleri için e-postalar da sağlıyorum.

○ Git yolu

○ Mvn kodu ; hedefler - derleyin veya doğrulayın -drunner=xml, vb.

○ Belirli testler için zaman çizelgesi

● Regresyon testinizi jenkins üzerinde çalıştırırsanız, otomatik olarak çalışırsa nasıl çalıştırırsınız?

**6. Regresyon testinde neler oluyor?**

● Takıma, test planına, kapsama ve iş değerine bağlıdır

**7. Jenkins'te kaç iş var?**

● Otomatik testler için kişisel olarak 2-3 iş ayarladım

○ 1 duman için → Duman günde 2,3 kez çalışıyor ve tüm ortamların çalışır durumda olduğundan emin oluyor

○ Tam regresyon (manuel ve otomatik testler)?

■ Her üretim sürümünden önce (3 sprintten sonra)

■ Tam regresyonda yalnızca çok kararlı test durumları saklanır

■ İşlevselliği güncelleme

○ küçük regresyon

■ Sprint sonunda çalışır

■ Belirli modüller ve işlevlerle ilgili testler

■ Hangi modülün çalıştırılacağını belirtmek için etiketleri kullanıyorum

97

**8. Kod ortamınıza nasıl dağıtılır?**

● Geliştiriciler kodu yazar, test eder, ardından dev'den test ortamına jenkins'te dağıtılır

● **Ya olmazsa?**

○ Geliştiricinizle konuşun ve dağıtmasını isteyin

**9. Etiketlere göre nasıl arama yaparsınız?**

● Eclipse'de arama aracını açmak için **ctrl+H'yi** kullanabilir ve etiket adını buraya koyabilirsiniz ve

**10. Jenkins'te bir yapı nasıl planlanır?**

● Jenkins'de iş konfigürasyonu altında çeşitli inşa tetikleyicileri tanımlayabiliriz.

● Basitçe 'Derleme Tetikleyicileri' bölümünü bulun ve 'Periyodik Olarak Oluştur' onay kutusunu işaretleyin.

● Periyodik olarak derlemeyle, derleme tanımını haftanın tarihine veya gününe ve yürütme zamanına göre planlayabilirsiniz.

yapı.

● 'Schedule' metin kutusunun formatı aşağıdaki gibidir:

○ DAKİKA (0-59), SAAT (0-23), GÜN (1-31), AY (1-12), HAFTANIN GÜNÜ (0-7)

● Jenkins'te zamanlamayı nasıl yapıyorsunuz? Testi her 3 saatte bir yürütülecek şekilde nasıl planlayacaksınız?

○ H 3\*\*\* Yapınızı her gün saat 7:00'de planlamak istiyorsanız, bu işinizi görecektir : 0 7 \* \* \*

**11. Jenkins'te boru hattı inşa etmek nedir?**

● Jenkins'te iş zincirleme, bir işin yürütülmesinden sonra otomatik olarak diğer iş(ler)e başlama sürecidir.

● Bu yaklaşım, çok adımlı yapı ardışık düzenleri oluşturmanıza veya bir projenin bağımlılıklarından birinin

güncellenmiş.

**12. Jenkins'te duman testleri işinizi nasıl sürdürüyorsunuz?**

● İşi git'ten kodu alacak şekilde yapılandırdım, testleri bir mvn komutu ileterek çalıştırdım, raporlar oluşturdum ve gönderdim.

e-posta. Ben gidip iş konfigürasyonunu değiştirmiyorum. Bir kez kurulduktan sonra her zaman çalışır.

**13. JENKINS SALATALIK RAPORU**

● Jenkins, Kullanıcı Dostu rapor oluşturabilen Salatalık raporu eklentisine sahiptir

● Salatalık raporu eklentisinin ihtiyacı olan tek veri Json format raporudur

● Salatalık raporu oluşturulduktan sonra, raporu isteyen herkese raporun URL'sini gönderin

● Otomasyon testlerinin tüm geçmişi Jenkins'te saklanır. Tarihleri, saatleri ve diğer ayrıntıları gösterir.

● Geçmişten gelen otomasyon raporunu görmeniz gerekiyorsa, onu Jenkins'te bulabilirsiniz.

**14. Jenkins İşini Planlamak**

● CRON İşi à Zamanlanmış otomatik görev

○ Jenkins kullanıcılarının CRON zaman biçimini kullanmalarının nedeni budur.

- 98 -

**SELENIUM GRID & SauceLabs**

**1. Selenyum Izgara Nedir?**

● Otomatik testlerinizi farklı tarayıcılarda (ve bunların farklı sürümlerinde) ve platformlarda çalıştırma olanağı sağlar

(temel olarak İşletim sistemleri ve sürümleri. Window, Linux, Mac) VISGRID

● Bu araç, çok sayıda Testiniz varsa (500'den fazla) kullanışlıdır.

● yapmak yerine

WebDriver sürücüsü=yeni Chromedriver()

● Yapmak

WebDriver sürücüsü=yeni RemoteWebDriver(url,yetenekler) //Yapıcılarda 2 parametre içerir

**2. Selenium Grid'i ne zaman kullanıyorsunuz?**

● Selenium Grid, aynı veya farklı test komut dosyalarını birden çok platformda ve tarayıcıda aynı anda yürütmek için kullanılabilir.

dağıtılmış test yürütme elde etmek için

**3. Selenyum Izgarası nasıl çalışır?**

● Grid, Hub ve düğümlerden oluşan bir kurulumdur.

● Hub, tüm düğümlerin bağlandığı merkezi bir makinedir.

○ IP adresi ve bağlantı noktası numarası var, Ardından Hub'ı Düğümlere bağlarsınız

● Hub olarak adlandırılan bir ana makine ve birden çok düğüm (testlerinizi gerçekten çalıştıran makineler) vardır.

● Yürütme sırası ise;

○ Kodunuz > uzak sürücü > Selenium Hub > Selenium düğümleri (birden çok olabilir)

■ Selenium Hub'ınızdan testlerinizi çalıştırmasını istediniz

● Ardından Selenium hub, hub'a bağlı bir düğüm bulacak ve testinizi oradan çalıştıracaktır.

■ İstediğiniz kadar düğüme sahip olabilirsiniz, ancak yalnızca bir hub'a sahip olabilirsiniz.

**4. Selenyum ızgaranızı nasıl kurarsınız?**

● Hub farklı bir sunucu makinesidir ve her düğüm ayrı bir sunucu makinesidir

● Merkezimiz ve düğümlerimiz Amazon AWS Ec2 makinelerinde kurulacak (ideal olarak)

● Hub'ınızdan testler yapmasını isteyebilirsiniz, ardından hub bir düğüm bulur ve testinizi oradan çalıştırır.

● Aynı makinede veya sunucuda hub ve birden fazla düğüme de sahip olabiliriz.

● Her düğüm, belirli bir yapılandırma ile HUB'a kaydolur ve HUB, düğümde bulunan tarayıcının farkındadır.

● HUB'a belirli bir tarayıcı için (İstenen yetenekler nesnesi ile) bir istek geldiğinde, HUB için bir eşleşme bulunursa

istenen tarayıcı, aramayı \*o\* belirli GRID Düğümüne yönlendirir ve ardından çift yönlü olarak bir oturum kurulur

ve yürütme başlar

**5. Çapraz tarayıcı/platform nerede çalışıyor?**

● Jenkins'in kurulu olduğu Amazon AWS makinesinde çalışmaktadır. Ancak normalde, jenkins şirketi geliştiriciler tarafından kullanılır.

ekip, geliştiriciler, dağıtım ekibi ve KG otomasyon ekibi

○ orda tavsiye edilmez

● İdeal olarak Jenkins > GitHub > Maven > Runner class>Selenium Hub'dır ve farklı olarak yapılandırılmış düğümlerden birinde çalıştırın.

sunucu

○ Hooks sınıfınızda, webDriver'ınızı yerel bir sürücüye yönlendirmek yerine, onu RemoteDriver(); ile değiştirin ve sahip olun

hub'a sahip bulut makinesini işaret edin

- 99 -

**6. Çoklu tarayıcı testindeki zorluklar nelerdir?**

● Bir şey tıklamıyor

● Görünmez

● Bazı öğeler bir tarayıcıda diğer tarayıcıdan farklı görünüyor

**7. Hangi tarayıcıları test ediyorsunuz?**

● Chrome - Firefox - IE/Edge - Safari - Opera

**8. Selenium Grid'de hub nedir?**

● Hub, farklı makinelerde test yürütmelerini kontrol eden sunucu veya merkezi bir noktadır.

**9. Selenium Grid'deki düğüm nedir?**

● Düğüm, hub'a bağlı olan makinedir, Selenium Grid'de birden fazla düğüm olabilir.

**10. Çoklu tarayıcı testini nasıl otomatikleştirirsiniz?**

● Çerçevemdeki özellikler dosyamda tarayıcıyı başka bir şeyle değiştir

○ Testlerimi farklı tarayıcıda çalıştırmak istediğimde

○ Ör; “Tarayıcı=chrome”dan “Internet Explorer”a

○ Bu yöntem, testleriniz 500 testten azsa işe yarar

● Çerçevemde Selenium Grid uyguladım ve farklı tarayıcılar kullanarak farklı bulut makinelerinde testler çalıştırabilirim.

**11. Başarısız testlerle ne yapmalı?**

● Otomasyon yürütme raporuna bakın

● Başarısızlığın nedenini öğrenin

● Adımları manuel olarak yapmayı deneyin,

○ Manuel geçiyorsa, otomasyon sorunu → böylece düzeltirsiniz, yeniden çalıştırın ve geçip geçmediğini görün

○ Eğer uygulama sorunu

■ Bir kusur oluştur

● Kusur düzeltilirken, Ad-hoc testi kullanarak manuel olarak test ediyorum

○ Kusur bir gösterici değilse, o zaman başka testler yapar ve otomatikleştirirsiniz.

■ Eğer öyleyse beklemeniz gerekiyor, daha fazla test yapamazsınız.

○ Yeniden çalıştırırken, yalnızca yeniden çalıştırmak istediğiniz testi test etmek için @ReRun etiketini kullanıyorum

**12. Ad-Hoc testi nedir?**

● Uygun planlama ve dokümantasyon olmadan gerçekleştirildi

● Bu senaryolar için hizalanmış test senaryoları olmadığından, bu yöntem kullanılarak bulunan kusurların çoğaltılması zordur.

● Resmi test uygulamasından sonra gerçekleştirilir

**13. SauceLabs - bulut Izgara hizmeti. Birden çok tarayıcıyla birden çok Platforma erişim**

● Bir sürü çalıştırabilmeniz için bulut makinesi sağlar

● Paralel testi iyi yapıyor

**14. SauceLabs kullanarak nasıl rapor veriyorsunuz?**

● JIRA sunucuma bağlanıyorum

- 100 -

**15. SauceLabs ile nasıl test yapılır?**

● Genellikle “WebDriver driver=new FireFoxDriver();

● Şimdi yapacağımız;

DesiredCapabilities caps=DesiredCapabilities.firefox(); caps.setCapability(“platform”,”Windows 7”);

caps.setCapability(“sürüm”,”38.0”);

WebDriver sürücüsü=yeni RemoteWebDriver(yeni

URL(“http://KULLANICI ADI:YOUR\_ACCESS\_@ondemand.saucelabds.com:80/wd/hub”,ca ps)

**16. Testlerinizi soslabs/herhangi bir hazır selenyum ızgarasında çalıştırma adımları**

**●** İstenenCapabilities nesnesini oluşturun ve testlerinizin Selenium Grid ile çalışmasını istediğiniz işletim sistemi, tarayıcı türü ile belirtin.

DesiredCapabilities caps=DesiredCapabilities.firefox(); caps.setCapability(“platform”,”Windows 7”);

caps.setCapability(“sürüm”,”38.0”);

**●** HUB url'si ile RemoteWebDriver oluşturun:

WebDriver sürücüsü=yeni RemoteWebDriver(yeni URL(URLOFHub),büyük harfler);

**●** SosLabsDemo sınıfı oluşturun

// @BeforeTest'te

// DesiredCapabilities(Selenium'dan geliyor)

caps=DesiredCapabilities.firefox(); //(hangi tarayıcıyı seçer)

caps.setCapability(“platform”,”Windows 7”);

caps.setCapability(“sürüm”,”38.0”);

● Dize URL=” http://YOUR\_USERNAME:YOUR\_ACCESS\_@ondemand.saucelabds.com:80/wd/hub ”

○ Bu Selenium Hub adresi

○ **URI** = birleşik kaynak tanımlayıcısı

○ **URL** = birleşik kaynak bulucu

**17. Testlerinizi birden çok iş parçacığında paralel olarak nasıl çalıştırıyorsunuz?**

● İş parçacığı bir işlem veya uygulama çalıştırma örneği gibidir

● 4 yol

○ 1. Farklı etiketlerle birden fazla cukesrunner oluşturabiliriz

■ Eski. cukerunner'da “@Test” etiketi var

■ Smokerunner'da @smoke var

■ Regresyon koşucusu @Regression'a sahiptir

○ 2. Testng xml oluşturun ve bu koşucu sınıfını tek bir testin altına ekleyin

■ Tüm 3 koşucuyu bir xml'ye ekleyin

■ Sonra verbose=2'nin yanına(xml dosyasının üstüne) “parallel=”classes” thread-count=”10”> ekleyin

○ 3. Ardından, sürücümüzün hub'a giden uzak bir Web Sürücüsü açtığından emin olun.

■ Kodu sürücü sınıfına ekleyin

○ 4. Testng xml'yi kendi başına veya maven kullanarak çalıştırın.

- 101 -

**18. AWS'de Selenium Grid'i nasıl kurarsınız?**

● (1) 2 bulut sunucusu (Ec2)

○ 1 HUB olacak

○ 1 düğüm olacak

● (2) 1. makinede Selenium StandAloneServer'ı indirin

○ Remote Selenium WebDriver'ı çalıştırmak için gereklidir.

■ Çok fazla yapılandırma

■ Komut satırını kullan

● HUB'ı kurun;

*java -jar Selenium-server-standaline-3.5.3.jar - rol merkezi http://localhost:444/grid/console*

● 1. düğümü ayarla

*Java -jar Selenium-server-standalone-3.5.3.jar -rol düğümü - hub http://locahost:4444/grid/register*

● Düğüm2

*java -jar Selenium-server-standaline-3.5.3.jar -rol düğümü -hub http://localhost:4444/gird/register -port 7777*

● (3) Visgrid'i de kullanabilirsiniz

○ 1. makinede indir

○ jar dosyasını aç

■ Maksimum oturumu 10'a ayarla

■ Hub'ı başlat

■ Port 4444 (hub'ınızın olduğu yer. Değiştirebilirsiniz ama hatırlayın)

○ Düğüm oluştur'u tıklayın

■ Düğüm için bir tarayıcı seçin

■ Örnek sayısını yazın

■ Ekle'yi tıklayın

● Tarayıcıyı ec2 makinesinde açın

● Şunu yazın: localhost:4444 (bu SeleniumGrid hub'ıdır) (Şimdi HUB'ımıza güç verdik)

● konsola tıklayın

● Sayfayı yenile

● Şimdi düğümleri göster (hepsi aynı makinede)

● Başka bir düğüm ekle (şimdi 2 düğüm gösterecek)

○ Şimdi başka bir makineye gidin ve oradan düğümler oluşturun ve onu HUB'a bağlayın

○ 2. makineye geçmeden önce

■ aws konsoluna git

■ Hub'ınızı tutan örneğe gidin

● Güvenlik gruplarına tıklayın = sihirbazı başlat

● Gelene git

○ Hub'ın bağlantı noktası numarasını (4444) ekleyin ve kaydedin

○ 2. makineye git

■ 1. makine için kullandığınız Visgrid'in aynı jar dosyasını indirin

■ jdk'yi indirin (dosyayı açmak için)

■ Kavanozu açın> hub'ı başlat> düğüm oluştur> HUB'ı Geçersiz Kıl'a tıklayın (üzerinde başka bir hub'ımız olduğundan bu hub'a ihtiyacımız yok

ilk makine)>makine 1 ip adresi ve bağlantı noktası numarası ekle: 4444>ekle

- 102 -

**19. Linux Komutları (büyük/küçük harfe duyarlı)**

● yeniden başlat → sistemi yeniden başlatır

● adam → size komutun talimatını verir - Ör: "man yeniden başlatma"

● mikdir → Dizin (klasör) oluşturur

● cd → Dizini değiştir

● Ls → Dizin içeriğini listele

● şifre → Geçerli çalışma dizininin adını yazdırın. Size tam konumu verir; Ör: /home/Andy/Desktop

● ll → Uzun liste formatı

● ls-la → Dosyaları ve gizli dosyayı yazdırır

● temizle → Temiz ekran

● cd.. → Üst dosyaya gider (kök dosyaya değil)

● cd/ → Ana kök dosyaya gider

● cd~ → Kullanıcı dosyasının ana sayfasına gider

● grep → Bir desenle eşleşen bir çizgi yazdırır

● df-h → Disk alanı kullanımını yazdırır

● üst → Linux görevlerini görüntüler (görev yöneticisi gibi)

● **Hesap nasıl oluşturulur?**

○ kullanıcı → kullanıcı ekle Andy

○ Grup → grup ekle Cybertek

**● Gruba kullanıcı ekleme**

○ useradd -G Cybertek Andy

○ kimlik Andy → bu bireyin ayrıntılarını yazdırır (Andy'nin cybertek'e sahip olduğunu gösterir)

**● Ağı yapılandırma/değiştirme**

○ vi /etc/sysconfig/network

○ crontab → Dosyanızın çalışması için bir zamanlayıcı ayarlar (jenkins gibi bir program oluşturun)

**● İzinleri ayarlama**

○ chmod → Dosya modu bitlerini değiştir

■ Sipariş sahibi, grubu, diğerleri

● Dosya klasör ise, d önde

● Klasör değilse d yoktur

○ chmod 777 → Sahip, grup ve diğerlerine erişim sağlar; Çok tehlikeli; KULLANMAYIN

■ r-oku

■ W-yazma

■ X-yürütmek

■ rwxrwxrwx (777)

○ chmod 644 → Sahibine (okuma ve yazma), gruba (salt okunur), diğerlerine (salt okunur) erişim ver

■ -rw-r--r--

■ Varsayılan erişim ve Standart

● Grep kullanarak bir dosya nasıl bulunur (eğer dosyanın adını biliyorsanız)

○ grep 'başarısız test dosyasının adı' /home/Andy/Test1/TestScenario (konum) > /home/Alex/AutomationFile

■ Şimdi dosya bu konumun içinde; /home/Alex/AutomationFile

○ 'Adı' olan herhangi bir dosyayı bulun

○ grep 'başarısızlık senaryosu' \*

103

**AWS**

**1. AWS ile mi çalışıyorsunuz?**

● EC2 örnekleriyle çalışıyorum.

● Temel olarak, bu benim sanal makinelerim.

● Selenium Grid'e sahip olduğumda farklı sanal makinelerim oluyor ve her makineyi ayrı ayrı çalıştırıyorum.

● Örneğin regresyon testleri için harcanan zamanı minimuma indirmek gerçekten verimli, firmamıza çok zaman kazandırıyor.

**2. AWS nedir?**

● AWS, bulut VM sağlıyor. Bir EC2 örneği oluşturun.

● Bu örneği uzak masaüstü ile kullanabilirim. Aslında, öğle yemeğinden sonra örneğimi normal bir bilgisayar gibi kullanıyorum.

**3. Temel sayfa nedir?**

● Ortak işlevlerimizi bir temel sınıfta saklarız ve daha sonra bu temel sınıfı genişletir ve diğer sınıfta kullanırız.

104

**Çerçevenizi sıfırdan nasıl oluşturabilirsiniz?**

- Test Çerçevesi - test senaryoları oluşturmak ve tasarlamak için kullanılan yönergeler ve kurallar

**1. Ortamı kurun; JDK, MAVEN, ECLIPSE IDE'yi kurun**

**2. Maven projesi oluşturun**

● ArtifactID - projenizin adı

● GroupID - projenizi tüm projelerde benzersiz bir şekilde tanımlar

**3. Maven Deposundan Bağımlılıklar Ekleyin**

● Selenyum Java

● salatalıklı sandviç

● <özellikler> içinde JRE Sistem kitaplığı 1.8

● TestNG

● WebDriverManager\_BoniGarcia

● salatalık Java

● Salatalık TestiNG

● Apache POI.XML

● APACHE POI

● JDBC

● Emin olabilirsiniz

● Gson

● Log4j

**4. Çerçeve Yapısı (paketleri) oluşturun**

● Sayfalar

i. Web öğeleri ve yöntemler

● Fasulye

i. Özel sınıflar

● koşucu

i. Cuckesrunner - kodlar oluşturur ve hedeflenen html raporunu depolar

ii. Sigara içmek

iii. regresyon

● Adım Tanımları

i. Gerçek kodlar ve kanca sınıfı

● testler

i. Veri odaklı testler

● JDBC

● API

● Araçlar

i. Yapılandırma Okuyucu

ii. Sürücü sınıfı (Singleton)

iii. tarayıcıUtils

iv. DBUtils

v. APIUtils

● Configuration.properties

● Testng\_runner.xml

i. Koşucu sınıflarından biri Paketleri com.app.utilities ile çağırın

105

**5. Kaynaklarda özellik dosyası (.feature) içeren özellik klasörü oluşturun**

● Bir kornişon dilinde yazılmış dosya yürütülebilir belirtimi

**6. Senaryonuzu yazmayı bitirdikten sonra** , cukesRunner'ınızı dryRun=false ile çalıştırın, bu size uygulama kodunu verecektir.

StepDefinition sınıfında depolayacaksınız

**7. Kodunuzu nasıl çalıştırırsınız?**

● Koşucu sınıfını kullanın - kodları çalıştırın ve salatalık raporu ve html raporları oluşturun

**8. Çerçeveyi GitHub veya SVN'ye gönderin**

● Yeni depo oluştur

● Git URL'sini kopyalayın ve Eclipse'e gidin

● Git deposunu yapılandırın ve url'yi ekleyin

● Projeye sağ tıklayın, ekibi tıklayın, taahhüt edin ve git evrelemesinde olacaksınız

● ŞİMDİ KODUNUZ GITHUB'DA

**9. Sıradaki JENKINS ENTEGRASYONU**

● Jenkins

i. Açık kaynak otomasyon sunucusu

ii. Yazılım geliştirme sürecinin insan dışı kısmını otomatikleştirmeye yardımcı olur

iii. Sürekli entegrasyona izin verir

iv. Geliştiricinin, kodu düzenli aralıklarla paylaşılan bir depoya entegre etmesini gerektiren geliştirme uygulaması

v. Bağlantı noktası 8081 yerel ana bilgisayardır

● Projenizi jenkins'te çalıştırmak için

i. jenkins hesabına giriş yap

ii. Proje oluştur - serbest stil

iii. Eklentileri yükleyin -salatalık raporu ve git

iv. Kaynak kodu yönetimi altında git ve geçmiş git url'yi seçin

v. Oluşturma tetikleyicisi - periyodik olarak oluşturmayı seçin

vi. Üst düzey maven'i çağır

1. Maven versiyonu ; MAVEN\_HOME

2. Hedefler ; temiz doğrulama -Drunner=smoke\_runnerxml

vii. Derleme sonrası eylemler altında

1. Salatalık raporlarını seçin

2. Düzenlenebilir e-posta bildirimini seçin

viii. Düzenlenebilir e-posta bildirimi

1. Derleme günlüğü ekleyin; derleme günlüğünü seçin

2. Gelişmiş ayarlara tıklayın

ix. Arıza-Herhangi biri

1. Gelişmiş'i tıklayın

2. Alıcı listesi - raporu alacak olan e-posta adresi. Birden fazla ise virgül ekleyin

3. Tetik ekle'yi tıklayın - her zaman başarısızlık gibi

4. Derleme günlüğü ekle; derleme günlüğü ekle'yi seçin

5. Kaydet

x. Son adım

1. Şimdi oluştur'a tıklayın, test çalışacaktır ve salatalık raporunuzu verecektir.

106

**İnşadan sonra salatalık raporu hakkında konuşun**

**1. Sayfa Nesne Modeli Nedir?**

a. Kod fazlalığını azaltır ve kodu düzenler

b. Öğeleri tanımlamaya ve onu bir sayfa nesnesi değişkeni olarak saklamaya yardımcı olur

c. Saklandığı yere bağlayabilirsiniz

d. PageFactory tasarım deseni eklendi

**2. Otomasyon aracım olarak Selenium WebDriver**

e. Önce manuel olarak test edin:

i. Başlangıç aşaması

● Fonksiyonel test

ii. arka uç

● Veritabanı - SQL Geliştirici IDE

● API - Postacı

f. Selenium'u şunlarla entegre edin:

i. Uzman

● Test paketi

● yardımcı program paketi

a. kullanıcı arayüzü

b. DB

c. API

● Yapılandırma dosyası

a. Özellikleri

● Sürücü sınıfı

a. Tek tonlu tasarım deseni

a. Özel bir kurucuya sahip olun

ii. salatalık

● BDD sürecinde işbirliğini kolaylaştırmak

● Öykünün ve kabul kriterlerinin kolay bir dille anlatılmasını sağlar.

iii. Git - kaynak kontrolü

iv. Jenkins

v. Java

● Koleksiyonlar Çerçevesi

● Apaçi İÇN

● JDBC

● Emin olabilirsiniz

**3. Davranış Odaklı Gelişim**

● Müşteri için doğru standartları karşıladığından emin olmak için müşteri ile birlikte gelişmek

**4. Veriye Dayalı Geliştirme**

● Farklı veri kümelerine karşı aynı test senaryosunu yürütme

● Test akışı verilere göre değişmemelidir

**5. Salatalık raporlaması**

● Hedef klasör

● Jenkins

107

**Çerçevenizi ve Testlerinizi Tanımlayın**

ÇERÇEVEMİ ve görüşmeciye nasıl açıklanacağı - Andy Lam ve biraz Alex

// **Veriye Dayalı ve Davranışa Dayalı** - Hibrit çerçeveye dayalı

// **Maven** - oluşturma aracı ve yalnızca bağımlılık yönetimi için değil, aynı zamanda pom xml dosyasını kullanan bir komut istemi aracı olarak, I

çalıştıran belirli bir **xml dosyası var,**

// regresyon ve işlevsellik testleri

// Programlama dili olarak **Java** - çalışan ön uç, arka uç (api) ve veritabanı Depolamak için **Java Koleksiyon çerçevesini kullanıyorum**

veri ve karşılaştırma

// hassas/yeniden kullanılabilir verileri depolayan bir **özellikler dosyam da var - URL, şifre, tarayıcı**

// ve kusurları bulmak için verileri java'ya biçimlendirdikten sonra akışı kontrol etmek ve verileri onaylamak için **TestNG test aracını kullanıyorum**

Ayrıca , yalnızca bir evrensel web sürücüsü oluşturmak ve kullanmak için **tekli desen** kullanan yardımcı program paketinde **Sürücü sınıfını oluşturdum.**

**FRONT END** ; **selenyum web sürücüsü** ve çerçevemde tasarım desenim olarak **Sayfa nesnesi modelini kullanıyorum** ; - sayfa oluştur

nesneler; webElements'i tanımlayın ve bir webelement değişkeni olarak saklayın,

POM = ELEMAN/YÖNTEM TABANLI SAYFA NESNELERİNİN YENİDEN KULLANILABİLİRLİĞİ

// @FindBy kullanarak web öğelerimi başlatmak için **Sayfa Fabrikası tasarım desenini de kullanıyorum - daha kolay/uygun**

// yardımcı program; tarayıcı kullanımı - burada hayatınızı kolaylaştıran statik yeniden kullanılabilir kod, tarayıcıyı otomatikleştirme kodlarını daha kolay hale getirir;

**ARKA UÇ (API)**

// Seri durumdan **çıkarma** ve **serileştirme** işleminin gerçekleşmesi için çerçevenizde **RESTASSURED kullanma**

// bu şekilde **Json yanıtını** bir Java toplama veri yapısında saklarsınız (/ yüksek seviyeli Pojos ve harita nesneleri üretirim) ve onaylarsınız

beklenen değere sahip veriler (ayrıca Java veri yapısında saklanır)

// ayrıca bir api yardımcı program sınıfım var - yeniden kullanılabilir kodlar - bir satırın bir Pojo oluşturduğu yöntem

// ÇERÇEVEMİ ÇAĞRILAMADAN ÖNCE manuel test için **postacı kullanıyorum**

**VERİTABANI testi**

// SQL sorguları üretmek için manuel I **sql geliştiricisi**

//OTOMASYON İÇİN ; Java'yı entegre etmek için Oracle veritabanından bir **BAĞLANTI alarak JDBC kitaplığını kullanıyorum**

//sonra SQL sorgularını kullanarak **STATEMENTS oluşturma** ve ardından verileri bir **RESULTSET** nesnesine depolama.

// Java veri yapılarını içerideki verileri depolamak ve karşılaştırmak için kullanıyorum

//ve **DATA DRIVEN ve CUCUMBER BDD çerçevesini kullandığım için tüm bu testler özellik dosyalarının** içinde saklanıyor

// **FEATURE FILE'dan** kodlar oluşturmaya ve bunları b adlı bir dosyaya uygulamaya yardımcı olan **RUNNER sınıflarım var**

// ayrıca tüm testlerimden önce ve sonra çalışan kodlarımı uygulayan **HOOK sınıfına** da sahibim - burası benim

arayüzünü kullandığımda tetiklenen **TAKECREENSHOT arayüzü (senaryo başarısız olduğunda)**

//başarısız olan adımdayken bir resim çek

**kornişon dilinin** beklenen değerine dayalı kodlarımı burada sakladım

**DDT**

//Eğer küçük miktarlarda test verisi ile çalışıyorsam **senaryo anahatlarıyla çalışacağım** , burada **örnekler oluşturacağım** ve

ardışık düzen kullanarak veri depolamak

//çok miktarda test verisi varsa, bu genellikle harici bir dosyada (excel) bulunur, bu yüzden **DDT EXCEL'i ÇIKARMAK için Apache POI** kullanırım

**OTOMASYON** ve excel dosyasından okuma ve verileri saklama

//java veri yapısına

108

// Temelde yüksek riskli olan kodlarımı günlüğe kaydetmek için **log4j2 adında bir günlük aracım da var**

// ve son olarak raporlarım için, çerçevemde , salatalık tarafından oluşturulan "rerun:target/rerun.txt" salatalığında **Rerun.txt kodunu kullanıyorum**

sandviç kitaplığı

//bu benim başarısız salatalık özellik dosyalarımı saklayacak

//sonra , başarısız senaryoların bulunduğu yere sahip **failScenario runner sınıfım da var (rerun.txt)**

// başarısız bir Senaryo xml dosyası oluşturuyorum

//bu yüzden ne zaman özellik dosyalarında başarısız olsam mvn komutunu kullanırım; **mvn -Drunner=failedScenarios.** başarısız testlerimi çalıştırmak için **xml dosyası**

//raporlama - Salatalık raporları adı verilen hedef klasörde bulunan html raporunu kullandım -"html:target/cucumber-report"

**Paralel test** - koşucular oluşturmak için **salatalık jvm-paralel eklentisi** ve testleri çalıştırmak için **maven hatasız eklenti kullanıldı**

**Sürekli entegrasyon** için (jenkins)

//devops yapılandırmayla ilgilenir

// github yolu var

//ama araç benim bir mvn komutum çağrıldı - mvn doğrulama -drunner=smoketest.xml test cihazı tarafından sağlanan - xml dosyası

//raporlar için her derlemede, test ve ekran görüntüsünün grafik bilgilerini veren bir salatalık raporu olacaktır

**Kaç test senaryosu/senaryo/özellik dosyası:** Sprint başına 23 test senaryosu, 150 özellik dosyası, 400 senaryo Duman testim

2 özellik dosyası, her biri 2 senaryo, günde bir kez 5 dakikalık duman testi yapılır.

gerilemem

150 özellik dosyası

400 senaryo

3 yıl ; 1 testçi 3 yıl çalıştı, ikincisi 2 yıl çalıştı

**Duman testiniz ne sıklıkla, ne kadar sürüyor ve neyi test ediyor?** Her sabah saat 5'te, yaklaşık 5 dakika, 5 öğrenci oluşturur

gerekli alanlara sahip iki tarayıcıda, hem kullanıcı arayüzünden hem de veritabanından doğru şekilde kaydedilip kaydedilmediğini kontrol eder, ardından raporu

tarayıcı ve ayrıca bir excel dosyası olarak indirin ve ardından karşılaştırın.

**Regresyon testiniz ne sıklıkla, ne kadar sürüyor ve neyi test ediyor?** Haftada bir gün 01:00 yarım saat, tüm ilçeler her gün,

sadece SIMS. Hafta sonları validasyon regresyon testleri, 5 ayrı EC2 makinesi, her okul için paralel test, teslim,

doğrulama, çapraz doğrulama ve sertifikasyon. Testimiz SCS ile doğrulanır.

**Örnek Senaryo:** SIMS raporunun indirildiği ve dosya türünün bir Excel Dosyası türü olduğu göz önüne alındığında, ilk sütun başlığı şu şekilde olmalıdır:

“LASİD” ve ilk sütundaki her hücre alfanümerik olmalı ve ilk sütundaki her hücre 9 basamaklı olmalı ve tüm LASID

sayılar benzersiz olmalıdır.

**Örnek Test Vakası:** Kullanıcı oturum açtığında, SIMS raporu yürütüldüğünde, ilk sütundaki her hücre 9 olmalıdır.

rakamlar ve ayrıca alfanümerik.

Olması Gerekenler: ön koşul, adımlar, test verileri, beklenen sonuç, gerçek sonuç

**Uç Durum Senaryosu:** boş, negatif sayılar, boş liste/dize, yinelenen kontrol, limitleri kontrol etme, aşırı durumlar (uzunluk, boyut)

**Riske dayalı test:** Tüm regresyon testini yapmak için zaman olmadığında, yalnızca önemli olan, ilgili olan kısımları test edersiniz.

**Gereksinim olmadan test etme:** Üretim kusurlarının genellikle herhangi bir gereksinimi yoktur ve bunu anlamak için geliştiriciyle konuşurum.

durumu daha iyi ve sonra test edin.

109

**Aşırı Yükleme yöntemi örneği** : Beklemeler için BrowserUtils'de birkaç aşırı yüklenmiş yöntem. Konumlandırıcıya göre açık beklemeler veya

Web Öğesi.

**Geçersiz Kılma yöntemi örneği:** Aşağıda

**Kalıtım/soyutlama örneği** : Üst çubuk ve kenar çubuğu sayfaları soyut sınıflardır ve bazı yöntemler soyuttur.

çünkü nerede olduklarına ve nereye tıkladıklarına göre farklı uygulamalarımız var. Okul/Bölge işlevleri

açılır menülerde farklı şekilde doldurulur.

- 110 -

**MARUFJAN Sınıflarından**

**jenkin nedir?**

● Sürekli entegrasyon aracı.

● Geliştiriciler ve testçiler tarafından kullanılan derlemeleri, dağıtımları vb. planlamak ve otomatikleştirmek için kullanılır.

**Jenkins kullanılarak ne tür süreçler otomatikleştirilebilir?**

● çalışan birim testleri

● uygulamayı oluşturma

● otomatik kullanıcı arayüzü testleri çalıştırma

● farklı ortamlara dağıtma

**Şirketinizde test için jenkins nasıl kullanılıyor?**

● Projemde, otomatik duman testlerini planlamak ve yürütmek için jenkins kullanıyoruz.

● duman testlerimiz her gün yapılır.

**Jenkins üzerinde hangi testlerin var?**

● Projemde jenkins için duman ve regresyon testlerimiz var.

**Jenkins'te ne tür testler yapılabilir?**

● Böylece, jenkins herhangi bir otomatik testi çalıştırabilir.

● Örneğin birim testleri, duman testleri, entegrasyon testleri, regresyon testleri, akıl sağlığımız olabilir.

**Jenkins kullanılarak hangi test katmanları test edilebilir?**

● Uygulamanın farklı katmanlarının test edilmesi test kodumuz ile yapılmaktadır.

● Jenkins, kullanıcı arayüzünü, veritabanını veya api'yi test etmemizi umursamıyor. sadece testleri başlatır ve raporlar gönderir.

● Bu nedenle, otomatik testim bir UI testiyse, bu, jenkins'in UI testleri çalıştırdığı anlamına gelir. veya otomatik testim bir API testiyse, bunun anlamı

jenkins API testleri çalıştırıyor.

**Jenkins'i kim kurdu?**

● Şirketimde jenkins'in bakımından sorumlu olan DevOps/Operations Support ekibimiz var. kurulu olanlar ve

yapılandırılmış jenkins.

● Dağıtımlar vb. için uygulamaları oluşturmak ve dağıtmak için işler yaratmak üzere geliştiricilerle birlikte çalışırlar.

● Duman testleri oluşturmak için otomasyon mühendisleri ile birlikte çalışırlar.

**Jenkins'te duman testlerini kim yapıyor?**

● Bir otomasyon mühendisi olarak duman testlerimi oluşturmak için devops/operasyon destekleriyle çalışıyorum.

● Testlerimi çalıştırmak için Jenkins'te kurulu belirli eklentilere ihtiyacım var, ayrıca jenkins'te yapılandırılmış java, maven'e ihtiyacım var, ayrıca ihtiyacım var

Testlerin çalışacağı sunucuda yüklü olan tarayıcılar.

● Projemde sadece devops/operasyon destek ekibi üyeleri yukarıdaki konfigürasyonları yapma hakkına sahiptir. bu yüzden bende var

onlarla çalışmak için.

● Jenkins'in konfigürasyonu tamamlandıktan sonra duman testi işini oluşturabilir ve çalıştırabilirim.

**Duman testleriniz jenkins'te nasıl yapılandırılıyor?**

● Jenkins işimiz GitHub'dan otomatik testlerimi alacak ve maven kullanarak her gün çalışacak şekilde yapılandırıldı.

● konfigürasyonda önce yeni bir iş oluşturdum ve "duman testleri" adını verdim. sonra Kaynak Kodunda git seçeneğini seçtim

Yönetim bölümü ve GitHub'da çerçeveme giden yola girdim. ve ayrıca git kimlik bilgilerini girdi.

● sonraki yapılandırma, periyodik olarak oluştur seçeneğini seçtiğim ve ne sıklıkta yapmak istediğimi zamanı girdiğim derleme tetikleyicileri ile ilgilidir.

testlerimi çalıştır.

- 111 -

**Build bölümünde** , otomasyon çerçevem maven kullanılarak oluşturulduğundan, üst düzey maven komutunu çağır seçeneğini seçiyorum.

● bölümde maven komutunu giriyorum (mvn kısmı olmadan). so fi normal komutu, testimi terminal kullanarak çalıştırmak için

'mvn testi', jenkins'te sadece 'test' giriyorum. Burada ayrıca çalıştırmak istediğim etiketi komut satırını kullanarak giriyorum.

**test -Dcucumber.options="--etiketler @duman"**

● regresyon testleri yapmak istersem komut şudur:

**test -Dcucumber.options="--tags @regresyon"**

**Post-build Actions'da** çift konfigürasyonu yapıyorum,

● Salatalık raporları seçeneği ekliyorum. Bu, jenkins'imize salatalık html raporları eklentisi yüklediğimiz için kullanılabilir.

Salatalık raporları eklentisi, her yapı için html raporu oluşturur. Jenkins üzerinde her duman testi yaptığımızda, yeni bir

bildiri. ve tüm yapı için tüm raporlar kaydedilir.

● Bir sonraki Oluşturma Sonrası Eylemler olarak e-posta seçeneği ekliyorum. Burada, çevik ekibimdeki herkesin bundan haberdar olması için yapılandırıyorum.

test sonuçları.

**Testler için e-posta raporlarını kim alır?**

● Agile ekibimdeki herkes test sonuçları hakkında bilgilendirilir.

**"DevOps boru hattı üzerinde çalıştınız mı?"**

● Devops boru hattının bir parçası olan jenkins üzerinde duman testleri yaptım. devops boru hattı tarafından oluşturulur ve yönetilir

DevOps/Operasyon Desteği. Ancak duman testimiz boru hattının bir parçası.

● Bu yüzden jenkins üzerinde duman testleri oluşturarak ve yapılandırarak katıldım.

**Entegrasyon testi ile uçtan uca test arasındaki fark nedir?**

• **Google genellikle 70/20/10'luk bir bölünme önerir: %70 birim testleri, %20 entegrasyon testleri ve %10 uçtan uca testler.**

• **Birim Testleri**

o En küçük işlevsellik birimini, tipik olarak bir yöntem/fonksiyonu test eder (örneğin, belirli bir duruma sahip bir sınıf verilir, x'i çağırır)

sınıftaki yöntem y'nin olmasına neden olmalıdır). Birim testleri belirli bir özelliğe odaklanmalıdır

• **Entegrasyon Testleri**

o Entegrasyon testleri, kod birimlerinin birleştirilmesi ve elde edilen kombinasyonun test edilmesi yoluyla birim testleri üzerine kurulur.

düzgün çalışır. Bu, bir sistemin iç kısımları olabilir veya yapmak için birden fazla sistemi bir araya getirebilir.

kullanışlı bir şey. Beyaz kutu test yaklaşımıdır. Geliştirici, tek üniteyi tek tek geliştirir ve

birbirinize ve test edin.

• **Fonksiyonel Testler**

o İşlevsel testler, belirli bir girdi için sonuçları,

Şartname. Fonksiyonel testler, ara sonuçlar veya yan etkilerle ilgilenmezler, sadece sonuçlarla (bunlar) ilgilenirler.

x yaptıktan sonra, y nesnesinin z durumuna sahip olması umrumda değil).

• **Kabul Testi**

o Yazılım müşteriye teslim edilmeden önce yapılan son testtir. Bunu sağlamak için gerçekleştirilir

geliştirilen yazılım tüm müşteri gereksinimlerini karşılamaktadır. İki tür kabul testi vardır - biri

Dahili kabul testi (Alfa testi) olarak bilinen geliştirme ekibi üyeleri tarafından gerçekleştirilir ve

(Beta testi) olarak bilinen müşteri veya son kullanıcı tarafından gerçekleştirilen diğer

• **Uçtan Uca test**

o , a noktasından z'ye gittiğiniz ve oraya giderken çeşitli farklı noktalara dokunduğunuz yerdir. Tek kişilik olabilir

bir e-posta gönderme işlemi gibi bir sistem veya birden fazla sistemin dahil olduğu durumlarda kullanılabilir, örneğin bir

bir sınava kayıt olan, sınava giren ve sonunda puanlarını alan öğrenci.

o Gerçek dünya sistem testi. Uygulama tüm entegre donanım, veri tabanı, ağ ve diğer sistemlerle test edilir.

arayüzler.

- 112 -

**Onları Jenkins'te nasıl çalıştırıyorsunuz?**

Uzun hikaye…

**Çerçevenizden bahseder misiniz?**

• Son Book-IT otomasyon çerçevemde, otomatik testler için Cucumber BDD çerçevesini kullandık. bu çerçeve

çok esnek bir çerçeve. Birçok farklı otomasyon test konseptini entegre eden hibrit bir çerçevedir.

• Çerçeve MAVEN kullanılarak oluşturulmuştur. maven çerçeveyi oluşturmak, bağımlılıkları ve eklentileri yönetmek, çalıştırmak için kullanılır.

maven yaşam döngüsü olarak testler.

• Java dili kullanılarak yazılmıştır.

• Bir Salatalık BDD çerçevesidir. Salatalık, otomatikleştirilmiş testi teknik olmayanlar için anlaşılır kılmak için kullanılan bir araçtır.

takım üyeleri. Salatalık, otomasyon mühendisleri ile teknik olmayan ekibi birbirine bağlamada bir köprü görevi görür.

üyeler. salatalık sürüm 4 kullanıyoruz

• Test aracı olarak junit kullanıyoruz. Junit, salatalık testlerini başlatmak ve ayrıca iddialar yapmak için kullanılır.

• Tarayıcıları otomatikleştirmek için Selenium WebDriver kullanıyoruz. selenyum ile farklı tarayıcılarda testler yapabiliriz.

• Çerçevemiz, maven-cucumber-report eklentisini kullanarak ekran görüntüleriyle adım adım HTML raporları oluşturur.

• Çerçevemiz veriye dayalı testleri destekler. Salatalık, senaryo ana hatlarını kullanarak veriye dayalı testleri doğal olarak destekler. bu

çerçeve ayrıca apache POI kitaplıklarını kullanarak Excel'den Veriye dayalı testler yapabilir.

• Çerçevem sayfa nesne modeline dayalıdır. sayfa nesne modeli, sayfadaki uygulamadan sayfaları temsil ettiğimiz zamandır.

nesne sınıfları.

• bir web sürücüsü oluşturmak için fabrika modelini kullanıyoruz. web sürücüsü sınıfımız, ne tür bir sürücüye sahip olduğumuza bağlı olarak bir web sürücüsü oluşturabilir.

istek. web sürücüsü nesnesi için tekil deseni kullanır

• URL, tarayıcı türü, oturum açma bilgileri gibi çerçevemiz hakkındaki önemli bilgileri depolamak için özellikler dosyasını kullanırız.

• Şirketimizde IntelliJ kullanıyoruz ama tutulma konusunda da oldukça iyiyim.

**Hangi uygulama katmanını test ediyor?**

Çerçevemizi kullanarak, uygulamanın kullanıcı arayüzünü, veritabanını ve API'sini test ediyoruz.

• **UI** à UI'yi test etmek için selenyum web sürücüsü kullanıyoruz

• **Veritabanı** à JDBC kitaplıklarını kullandığımız veritabanını test etmek için

• **API** à , API'yi test etmek için REST Assured kitaplıklarını kullanıyoruz

**Test türleri:**

• **fonksiyonel** , **kabul testleri** , **duman testleri** , **regresyon testleri** , **entegrasyon testleri** yapıyoruz .

• Projemde sürüm kontrol aracı olarak git kullanıyoruz. (SVN)

• Otomatik duman ve regresyon testlerimizi planlamak ve çalıştırmak ve test sonuçlarını e-posta ile göndermek için Jenkins kullanıyoruz. kullanma

jenkins, Xray eklentisini kullanarak testler yapabilir ve JIRA'yı test sonuçlarıyla güncelleyebiliriz.

• Çerçevemizde oturum açmak için Log4J kullanıyoruz.

• kod ve testlerin organizasyonu

• Java kodumuzu düzenlemek için paketler kullanıyoruz, sayfa nesneleri, yardımcı programlar, StepDefinition, pojos, koşucular için paketlerimiz var

farklı yardımcı programlar kullanıyoruz, WebDriver, tarayıcı yardımcı programları, excel yardımcı programı, yapılandırma yardımcı programı, Veritabanı için yardımcı programlarımız var

yardımcı program, api ile ilgili yardımcı programlar, tarih yardımcı programı (takvimle ilgili uygulamayı test ettiğimizden, tarihle ilgili birçok şey yapıyoruz).

• özellikler:

birçok özellik dosyamız var ve bunları düzenlemek için klasörler ve etiketler kullanıyoruz. her ana bileşen için klasörlerimiz var

uygulama : rezervasyonlar, harita, hesap bilgileri....

Jira ile eşleyebilmemiz için Jira'dan gelen sayıyı özellik dosyasındaki etiketlerden biri olarak kullanıyoruz.

• özellikler

hesap

rezervasyon

harita

- 113 -

Özellik: Kullanıcı rolleri

# *verilenlerde herhangi bir müsait yer olup olmadığını öğrenmenin bir yolunu bulmanız gerekiyor*

@giriş @BRIT\_3521 @duman

Senaryo Anahattı: <user> tipi olarak oturum açın

Kullanıcının hafif bir <user> olarak oturum açması göz önüne alındığında

Ve zamanlama için uygun noktalar var

Kullanıcı bir nokta aradığında

Sonra kitap düğmesi <beklenen> görüntülenecektir

Örnekler:

| kullanıcı | beklenen |

| ekip üyesi | yapmamalı |

| takım lideri | gerekir |

| öğretmen | gerekir |

**Paralelleştirme** à maven-surefire eklentisini kullanarak testlerimizi paralel olarak çalıştırıyoruz. paralel testi destekleyen salatalık 4 kullanıyoruz

doğal olarak.

**Test verileri** à özellik dosyalarını kullanın, test verilerimizi test/kaynaklar paketinde depolamak için excel dosyalarını kullanın

**Yürütme akışı veya testleri nasıl çalıştırdığımız**

1. **cukes koşucusundan koşmak**

cukes runner'a tag koyuyoruz ve sağ tıklayıp dosyayı çalıştırıyoruz. cukes runner eşleşen tüm etiketleri çalıştırır.

2. **terminalden maven komutu olarak çalıştırın**

testleri başlatmak için mvn doğrulama veya mvn test komutunu giriyoruz. ve maven, pom'da gösterilen cukes runner dosyalarını çalıştıracak

dosya. cukes runner dosyası, sahip olduğu etiketlerle eşleşen özellikleri çalıştırır.

**GIT dallanma**

**git dallanmayı şimdi nasıl yapıyorsun?**

Şu anda GRUP PROJELERİNDE NASIL YAPILIR:

Her ekip üyesi için ana dal ve ayrı dallar vardır. Birisi işini bitirdiğinde, iterler

kendi şubeleri, daha sonra gözden geçirildikten sonra master ile birleştirilir.

**NASIL YAPTIK?**

projemde usta, geliştirme ve kişiye özel dallarımız vardı. yani, 2 otomasyon test cihazımız varsa,

usta

geliştirmek

test cihazı1

test cihazı2

her test kullanıcısı kendi şubelerine giriş yapar. daha sonra gözden geçirdikten sonra şube geliştirmek için birleştirilir. ustayı birleştiriyoruz ve

sprintte yalnızca bir kez geliştirin.

μ röportajda dallanma hakkında soru sorduklarında, otomasyon projesi dallanma stratejinizden bahsedin.

projenizde, kodunuz uygulama kodu deposundan ayrı bir depodur. Otomasyon çerçevesi daha küçük bir kod tabanına sahiptir

ve daha az insan dahil oldu. Böylece, daha az karmaşık dallanma politikasına sahip olabiliriz.

- 114 -

**ÇERÇEVE HAKKINDA KONUŞMA**

**Araçlar** :

**Java** : Çerçevem Java dili kullanılarak yazılmıştır

**Maven** : Bir maven projesi olarak oluşturulan çerçevem, maven bağımlılıkları yönetmek ve testlerimizi mvn hedefleri olarak çalıştırmak için kullanılıyor

terminalden

**Selenium WebDriver** : tarayıcıyı otomatikleştirmek için kullanılan bir kitaplık/araç/api, tarayıcı ile etkileşime girer.

**TestNG** : xml dosyalarını kullanarak testleri gruplamak, yumuşak ve sert iddialar yapmak, test yöntemleri oluşturmak, belirli bir sırada çalıştırmak için kullanılır

**Kapsam** : çerçevem, teknik olmayan ekip üyeleri için okunması ve anlaşılması kolay ayrıntılı HTML raporları oluşturur.

Raporlarım, oluşabilecek arızalar için ayrıntılı test adımları ve ekran görüntüleri içerir. Ayrıca yüzde kaç olduğuna dair metrikler de yapabilir.

geçiyor, başarısız, atlanıyor vb.

**IDE** : Mevcut çerçevemde IntelliJ kullanıyorum, ancak daha önce kullandığım Eclipse ile de oldukça rahatım

**Tasarım** :

**Sayfa Nesne modeli** : çerçevem, sayfalarımın sayfaları için ayrı bir sınıf oluşturduğuma göre sayfa nesnesi modelini kullandı.

başvuru.

**PageFactory tasarımı** : sayfa nesne sınıfına erişmeyi kolaylaştıran bir tasarım.

bu Sayfa fabrika tasarımı değil. PageFactory tasarımıyla aynı adı taşıyan sınıftır:

PageFactory.initElements(sürücü, bu)

**Singleton Driver** : Çerçevelerim, web sürücüsü örneğini farklı sınıflar arasında paylaşmak için tek bir desen kullanır

**TestBase** : Çerçevem, testlerimin kapsamına giren bir testbase sınıfına sahip. testbase sınıfı, tüm testlerim için ortak adımlara sahiptir.

**yapılandırma dosyası** : önemli test verilerini depolamak için kullanılır

**yardımcı programlar** : çerçevemin farklı sınıflarında kullanılabilecek yeniden kullanılabilir yardımcı programlara sahip

**Faydaları** :

**bakımı kolay** :

, bakımı kolaylaştıran **sayfa nesne modelini** kullanıyor . örneğin, herhangi bir konumlandırıcıyı güncellemem gerekirse,

sadece bir kod değişikliği yapmanız gerekir.

Testlerimi birbirinden bağımsız hale getirmeye çalışırım. bu, bir testi güncellersem diğerlerini etkilemeyeceği anlamına gelir ve ayrıca bir

başarısız olursa, diğerleri etkilenmeyecektir.

test 1

2. test

test 3

**uzatmak kolay;** çerçeveme yeni test senaryoları eklemek kolaydır.

**yeniden kullanımı kolay;** Herhangi bir test için yeniden kullanabileceğim sayfa nesne modeli, yardımcı programlar var.

**Çoklu tarayıcı testi** : çerçevem, minimum kod değişikliği ile aynı testleri farklı tarayıcılara karşı çalıştırabilir

**türleri** : Çerçevem, uygulamanın kullanıcı arayüzünü, veritabanını ve API'sini test edebilir.

**Paketleme** : Farklı sınıf ve mantık türleri için farklı paketler oluşturdum. Her sayfa paketi yalnızca sınıfları içerir

aynı işlevselliğe sahip.

**Adlandırma kuralları** : Ekibimde kodlama standartlarına, özellikle de adlandırma kurallarına çok dikkat ederiz. sınıflar,

metotlar değişkeni yaptıklarına göre isimlendirilir ve bir standardı takip eder.

**Sayfa nesne sınıfı** :

ana sayfa, giriş sayfası

**Değişkenler** : loginButton, signOutLink

**Yöntemler** : login(): Bu yöntem sadece oturum açmak için kullanılır, diğer işlevler için kullanılmaz.

- 115 -

**Veriye Dayalı Çerçeve**

testlerin veri kümesine dayalı olarak yürütüldüğü bir çerçeve, çerçeve, excel gibi dış kaynaklardan gelen verileri okumak için tasarlanmıştır

ve verilere dayalı testler çalıştırın. Veri odaklı çerçevede, aynı testi farklı kümelere karşı birden çok kez çalıştırabiliriz.

veri.

**Sayfa Nesne Modeli Çerçevesi**

uygulamanın her sayfası için ayrı bir Java sınıfı oluşturduğumuz sayfa nesnesi tasarım desenini kullanır

Davranış Odaklı Çerçeve

bunun hakkında daha sonra konuşacağız

Anahtar Kelimeye Dayalı Çerçeve

KDF'de testlerimizi çalıştırmak için dış kaynakta (excel. csv) anahtar kelimeler kullanırız. Çerçeve, verileri ve adımları okumak için tasarlanmıştır

Excel'den ve buna dayalı eylemler yürütün.

KDF kurulduktan sonra, teknik olmayan testçiler bile otomatik testler yazabilir ve yürütebilir.

**Hibrit Çerçeve**

hibrit çerçeve, yukarıda verilen türlerden en az ikisini kullanan bir çerçevedir.

- 116 -

**JAVA Mülakat Teknik Soruları**

**Ters Dize – Döngü İçin**

genel statik String reverseString(String str) {

Ters dizi = "";

for(int i = str.length()-1; i >=0; i--) {

ters += str.charAt(i);

}

geri dönüş;

}

**EN İYİ YOL:**

String name= "Alper Aslan";

Ters dizi = new StringBuilder(name).reverse().toString();

System.out.println(ters);

**Ters Dize – StringBuilder**

**public static void main(String[] args) {**

**String str = "Merhaba Dünya";**

**StringBuilder sb = yeni StringBuilder(str);**

**Sistem. *out* .println(sb.reverse()); }**

**Ters Dize – Karakter Dizisi**

**public void printReverse(char[] harfleri, int boyutu){**

**for (int i = letter.length-1; i >= 0 ; i--){**

**System.out.print(harfler[i]);**

**}**

**}**

**Asal sayı**

**genel statik boolean checkPrime(int n) {**

**if (n <= 1) {**

**yanlış döndür;**

**}**

**for (int i = 2; i < Matematik. *sqrt* (n); i++) {**

**if (n % ben == 0) {**

**yanlış döndür;**

**}**

**}**

**true döndür; }**

- 117 -

**palindrom**

genel statik boolean isPalindrome(String str) {

if(str == boş)

yanlış döndür;

StringBuilder sb = yeni StringBuilder(str);

dönüş sb.reverse().toString().equals(str); }

**Palindrom – charAt() ile**

genel statik boolean isPalindrome(String s) {

int kafa = 0;

int kuyruk = s.uzunluk() - 1;

while (kafa <kuyruk) {

if (s.charAt(kafa) != s.charAt(kuyruk)) {

yanlış döndür;

}

kafa++;

kuyruk--;

}

true döndür; }

**Ters Dize – ArrayList – İtiratör**

public static void main(String[] args){

String input = "Geeks For Geeks";

List<Character> arrList = new ArrayList<>();

for (char c: input.toCharArray()){

arrList.add(c);

}

Koleksiyonlar. *ters* (arrList);

Object[] arr2 = arrList.toArray();

for(int i = 0; i <dizi2.uzunluk; i++) {

ters += dizi2[i];

}

Sistem. *dışarı* .println(ters);

}

DİĞER YOL

String ters = arrList.stream() // Akış<Karakter>

.map(String::valueOf) // Akış<Dize>

.collect(Collectors.joining());

- 118 -

**faktöriyel**

int sayı = 10;

int faktöriyelToplam = 1;

for(int i = 1 ; ben <=sayı; i++) {

factorialSum = factorialSum \* i;

}

Sistem. *out* .println("" + sayı + " faktöriyelinin " + faktör toplamı); }

**Rakamların Toplamı**

**int sayı** = 1346;

**int toplam** = 0;

**while** (sayı > 0) {

toplam += sayı %10;

sayı = sayı/10;

}

**Sistem** . ***çıktı*** .println(toplam); }

**Armstrong**

int toplam = 0;

int rakamı;

int sıcaklık;

int sayı = 370;

sıcaklık = sayı;

süre(sıcaklık >0){

rakam = temp %10;

toplam = toplam + (rakam\*rakam\*rakam);

sıcaklık = sıcaklık /10;

}

if(sayı==toplam)

System.out.println(sayı + " bir armstrong sayısıdır");

}başka

**merdiven**

for(int x = 1; x<=5; x++) {

for(int y=1; y<=x; y++) {

System.out.print(y+"");

}

System.out.println(); }

- 119 -

**Fibonacci**

int a = 0;

int b=1;

System.out.print(a + " " + b + " ");

for(int i = 2; ben <=10; i++) {

int c = a+b;

a = b;

b = c;

System.out.print(c + " + ");

}

**Geçici Değişken Kullanmadan İki Sayının Değiştirilmesi**

// tek satır yöntemleri

a = a ^ b ^ (b = a); //Yöntem 1

b = (a + b) – (a = b); //yöntem 2

a += b – (b = a); //yöntem 3

// geçici değişken

int sıcaklık = a;

a = b;

b = sıcaklık;

**Dizileri Kullanan Dizideki En Büyük Sayı**

int [] dizi = {5, 6, 76, 31, 43, 1};

Diziler.sort(dizi);

System.out.println(dizi[dizi.uzunluk-1]);

Koleksiyonları Kullanan Dizideki En Büyük Sayı

public static int returnLargest(Tamsayı[]b, int toplam) {

Liste<Tamsayı> list1 = Diziler. *asList* (b);

Koleksiyonlar. *sırala* (liste1);

int en büyük = liste1.get(toplam-1);

en büyük dönüş; }

public static void main(String args[]){

Tamsayı x[] = {4,3,2,12,54,34,88};

Sistem. *out* .println( *returnLargest* (x,7));

- 120 -

Koleksiyonları Kullanan Dizideki En Büyük Sayı

public static int getLargest(int[] a, int toplam){

int sıcaklık;

for (int i = 0; i < toplam; i++) {

for (int j = i + 1; j < toplam; j++) {

if (a[i] > a[j]) {

sıcaklık = a[i];

a[i] = a[j];

a[j] = sıcaklık; }}}

a[toplam-1] döndür; }

Ters Sayı

genel statik int ters(int sayı){

int ters = 0;

int kalan = 0;

yapmak{

kalan = sayı %10;

ters = ters\*10 + kalan;

sayı = sayı/10;

}while(sayı > 0);

geri dönüş; }

**Dizideki İlk İki Maks Sayıyı Bulun**

public void GetTwoMaxValues(int[] nums) {

int maxBir = 0;

int maxİki = 0;

Diziler.sort(sayılar);

System.out.println("Maks1 - " + (sayı[sayı.uzunluk-1]));

System.out.println("Maks2 - " + (sayı[sayı.uzunluk-2]));

}

**Divide veya Mod Operator kullanmadan bölme**

public static String bölümü(int bölen, int bölen) {

int bölüm = 0;

int kalan = 0;

while(bölen >= bölen) {

bölen = bölen - bölen;

bölüm++;

kalan = bölen; }

return "quotient = " + quotient + " kalan = " + kalan;}

- 121 -

İkili Arama (Doğrusal Aramadan Daha Hızlı) –

Sıralanmış bir dizi içindeki hedef değerin konumunu bulur. İkili arama karşılaştırır

dizinin orta elemanına hedef değer. Eşleşiyorsa, değilse doğru döndürür

ulaşana kadar tekrar bölersiniz.

public static void ikiliSearch(int dizi[], int ilk, int son, int anahtar){

int orta = (ilk + son)/2;

while( ilk <= son ){

if ( dizi[orta] < anahtar ){

ilk = orta + 1;

}else if ( dizi[mid] == anahtar ){

System.out.println("Öğe dizinde bulundu: "

+ orta);

kırmak;

}başka{

son = orta - 1;

}

orta = (ilk + son)/2;

}

if (ilk > son){

System.out.println("Eleman bulunamadı!"); }

}

public static void main(String args[]){

int dizi[] = {10,20,30,40,50};

int anahtarı = 30;

int son=dizi.uzunluk-1;

ikiliArama(dizi,0,son,anahtar); }

Kabarcık Sıralaması

Mevcut eleman bir sonraki eleman ile karşılaştırılır. Mevcut eleman şu değerden büyükse

sonraki eleman, değiştirilir.

public static void bubbleSort(int[] arr) {

int n = dizi.uzunluk;

int sıcaklık = 0;

for(int i=0; ben < n; i++){

for(int j=1; j < (ni); j++){

if(dizi[j-1] > dizi[j]){

sıcaklık = dizi[j-1]; // öğeleri değiş tokuş et

dizi[j-1] = dizi[j];

arr[j] = sıcaklık;

} } } }

public static void main(String[] args) {

int dizi[] ={3,60,35,2,45,320,5};

bubbleSort(arr);//bubble sort kullanarak dizi öğelerini sıralama

System.out.println("Kabarcık Sıralamasından Sonra Dizi");

for(int i=0; i < dizi.uzunluk; i++){

System.out.print(dizi[i] + " ");

- 122 -

**Dizideki Çoğaltmaları Filtrele**

public static void main(String[] args) {

ArrayList<String> list = new ArrayList<String>();

// 0-9 arasındaki sayıların bir listesini oluşturun.

for (int i = 0; i < 10; i++) {

list.add(String.valueOf(i));

}

// 0-5 arasında yeni bir sayı kümesi girin.

for (int i = 0; i < 5; i++) {

list.add(String.valueOf(i));

}

System.out.println("Giriş listesi : " + liste);

System.out.println("\nFiltrelenen kopyalar : " + processList(list)); }

public static Set<String> processList(List<String> listContainingDuplicates) {

final Set<String> resultSet = new HashSet<String>();

final Set<String> tempSet = new HashSet<String>();

for (String yourInt : listContainingDuplicates) {

if (!tempSet.add(yourInt)) {

sonuçSet.add(yourInt);

}

}

sonuç kümesini döndür; }

**en büyük palindrom**

public static String en büyükPalindrome(String str) {

str = str.toLowerCase();

Dize en büyükPalindrome = "";

String[] dizi = str.split(" ");

for(Her biri dize: arr) {

Ters dizi = "";

for(int i = her.length() - 1; i >= 0; i--) {

ters += her.charAt(i);

}

if(each.equals(ters) && each.length() >

en büyükPalindrome.length()){

en büyükPalindrom = her biri; }

}

en büyük dönüşPalindrome; }

- 123 -

**Bir Dizeden Fazladan Boşlukları Kaldırma**

public class removeExtraSpaces {

public static void main(String args[]){

String input = "Fazladan boşlukları kaldırmayı deneyin.";

StringTokenizer substr = new StringTokenizer(input, " ");

StringBuffer sb = yeni StringBuffer();

while(substr.hasMoreElements()){

sb.append(substr.nextElement()).append(" "); }

System.out.println("Gerçek dizi: " + girdi);

System.out.println("İşlenen dizi: " + sb.toString().trim()); }

DİĞER YOL

String input = "Fazladan boşlukları kaldırmayı deneyin.";

String inputNew = input;

while (inputNew.contains(" ")) {

inputNew = inputNew.replace(" ", " ");

};

System.out.println(giriş);

System.out.println(inputNew);

**Ünlülerin ve Ünsüzlerin Sayısı**

genel statik Dize numaraları(Dize sözcüğü) {

int sayÜnlüler = 0;

int sayıÜnsüzler = 0;

Liste<Karakter> ünlüler = Diziler. *asList* ('a', 'e', 'i', 'o', 'u');

List<Character> ünsüzleri = new ArrayList<>();

for(char i = 'a'; i <= 'z'; i++) {

if(!vowels.contains(i)) {

ünsüzler.add(i); }

}

kelime = kelime.toLowerCase();

for( int i = 0; i < kelime.uzunluk(); i++) {

if(vowels.contains(word.charAt(i))) {

sayÜnlüler++;

}başka {

sayÜnsüzler++; }

}

dönüş "sesli harf sayısı " + countVowels + " ve ünsüz sayısı " +

sayÜnsüzler; }

- 124 -

**Dinamik – Çalışma Zamanı Polimorfizmi**

sınıf Hayvan {

geçersiz yemek() {

System.out.println("yemek"); }

}

sınıf Köpek, Animal {

geçersiz yemek() {

System.out.println("meyve yemek");}

}

sınıf BabyDog, Dog'u genişletir {

geçersiz yemek() {

System.out.println("süt içmek");

}

public static void main(String args[]) {

Hayvan a1, a2, a3;

a1 = yeni Hayvan();

a2 = yeni Köpek();

a3 = yeni BabyDog();

a1.eat(); // "yemek" yazdırır

a2.eat(); // "meyve yemek" yazdırır

a3.eat(); } } // “içme sütü” yazdırır

**Çok Düzeyli Kalıtım ile Dinamik Polimorfizm**

sınıf Hayvan {

geçersiz yemek() {

System.out.println("hayvan yiyor..."); }

}

sınıf Köpek, Animal {

geçersiz yemek() {

System.out.println("köpek yiyor..."); }

}

BabyDog1 sınıfı, Dog'u genişletir {

public static void main(String args[]) {

Animal a = new BabyDog1();

a.eat(); }

}

**Xpath – Kardeşi Takip Etmek**

Aşağıdaki kardeş::siblingName[1];

Ebeveyn::ebeveynEtiket

//etiket[başlar-ile(@id,'mesaj')]

- 125 -

**Derleme Zamanı Polimorfizmi**

sınıf Aşırı Yükü {

geçersiz demo (int a) {

System.out.println ("a: " + a); }

geçersiz demo (int a, int b) {

System.out.println ("a ve b: " + a + "," + b);

}

double demo(double a) {

System.out.println("çift a:" + a);

a\*a döndür; }

}

class MethodOverloading {

public static void main (String args []) {

Aşırı Yük Obj = yeni Aşırı Yük();

çift sonuç;

Nesne .demo(10);

Obj .demo(10, 20);

sonuç = Nesne .demo(5.5);

System.out.println("O/P : " + sonuç); } }

1-100 arasındaki tüm çift sayılar ( i % 2 == 0)

Tümü tek sayılar (i%2 !=1)

Çalışma zamanı polimorfizmi, veri üyeleri tarafından elde edilemez.

**String'i Double'a dönüştürmek için**

Double doubleString = Double.parseDouble(toBeDouble);

Double doubleStr = Double.valueOf(toBeDouble);

Double'ı String'e dönüştürmek için

String strDouble = String.valueOf(toBeString);

String stringDouble = toBeString.toString();

genel statik karakter returnFirstChar(String str) {

dönüş str.charAt(0);

}

genel statik karakter returnLastChar(String str) {

dönüş str.charAt((str.length()-1));

}

- 126 -

**uyarılar**

Uyarı uyarısı = driver.switchTo.alert();

uyarı.kabul(); alarm.dismiss(); alert.sendKeys() alert.getText()

**örtük bekle**

driver.manage.timeouts.implicitlyBekle(5 saniye)

**açık bekleme**

WebDriverWait bekle = yeni WebDriverWait(sürücü,30);

WebElement öğesi = wait.until(

BeklenenKoşullar.elementToBeClickable(By.id(item)));

görünürlükOf(), alertIsPresent()

**Excel – Apachi POI**

**public void** readExcel(), IOException { 'yi **atar**

String **FileP** a ***th*** = "D:\\sampledoc.xls";

**FileI** n Stream ***koy*** fis = **new** FileInputStream(FilePath);

Çalışma kitabı wb = WorkbookFactory.create(fis);

Sayfa sh = wb.getSheet("Sayfa1");//getSheetAt(1)

// Sayfada bulunan satır sayısını almak için

**int** totalNoOfRows = sh.getRows();

// Sayfada bulunan sütun sayısını almak için

**int** totalNoOfCols = sh.getColumns();

**for** ( **int** satır = 0; satır < toplamNoOfRows; satır++) {

**for** ( **int** sütun = 0; sütun < toplamNoOfCols; sütun++) {

System.out.print(sh.getCell(col, row).getContents() + "\t");}

System.out.println();}

Satır satır= workSheet.getRow(0);

Hücre hücresi=satır.getCell(0);

System.out.println(cell.toString());

Hücre hücresi=satır.getCell(0);

cell.setCellValue("kahve");

FileInputStream outStream = new FileInputStream(filePath);

workbook.write(outStream);

- 127 -

**JDBC – Veritabanı Bağlantısı**

**Bağlantı** bağlantısı=DriverManager.getConnection(Url, Kullanıcı, Geçiş);

**İfade** ifadesi=connection.createStatement

(ResultSet.TYPE\_SCROLL\_INSENSITIVE,ResultSet.CONCUR\_READ\_ONLY);

**ResultSet** resultSet = deyim.executeQuery("ülkelerden \* seçin");

sonuçSet.last(); // sonuç kümesinde kaç kayıt olduğunu bul

int rowsCount = sonuçSet.getRow();

System.out.println("Satır sayısı:" + rowsCount);

sonuçSet.first();

while(resultSet.next()) {

System.out.println(resultSet.getString(1)+"-"+resultSet.getString("ülke\_adı")+"-

"+resultSet.getInt("region\_id"));}

**Sınıf Seç**

Arabaları seçinList = new Select(el) **;**

carList.getFirstSelectedOption();

**assertequals** (“bir metin”,carList.getFirstSelectedOption().getText())

**iframe**

Driver.switchTo.frame(webElement veya Dize veya dizin olarak kimlik)

**Pencere tutacağı**

getWindowHandles()

for(Dize tanıtıcısı: driver.get.getWindowHandles())

driver.switchTo.window(“handle”):: bir dize alır

if (driver.getTitle().equals(beklenen başlık)

kırmak;

**Eylem Sınıfı**

WebElement el = driver.findElement

Eylemler eylemler = yeni Eylemler (sürücü).perform

eylemler.doubleClick(el).perform

**Dosya yükleme**

Public void fileUpload(Dize yolu){

WebELEment yüklemesi = driver.findELement;

Upload.sendKeys(yol)

- 128 -

**SQL - İç Birleştirme**

e.İLK\_NAME, e.SALARY, j.JOB\_TITLE **SEÇ**

ÇALIŞANLARDAN e **INNER JOIN** JOBS j **ON e.JOB\_ID** = j.JOB\_ID;

**İstenen Yetenekler – Sertifika Sorunu**

**DesiredCapabilities** yeteneği = DesiredCapabilities.chrome();

// SSL sertifikasını kabul etmek için

yetenek.setCapability(CapabilityType.ACCEPT\_SSL\_CERTS, true);

System.setProperty("webdriver.chrome.driver", "E:/chromedriver.exe");

// Google Chrome örneği oluştur ve onu büyüt

sürücü = yeni ChromeDriver(yetenek);

**Tablodan 5. Satırı göster**

(ÇALIŞAN\_KİMLİĞİNİ SEÇİN,FIRST\_NAME, ROWNUM AS RN'den SEÇ \* SEÇİN)

ÇALIŞANLARDAN) NEREDE RN = 5;

- 129 -

**B10 – GERÇEK HAYAT RÖPORTAJ SORULARI:**

Şahin G.

• Bize projenizden bahsedin.

• Java'yı nerede kullanıyorsunuz?

• String ve Stringbuilder arasındaki fark nedir?

• Yığın ve yığın arasındaki fark nedir?

• Birleştirmeler (SQL) nedir?

• Senaryo Anahattı ne işe yarar?

İbrahim S.

• Projenizdeki Arka Uç Testinde hangi araçları kullanıyorsunuz ve Arka Uç Testinde neler test ediyorsunuz?

• Selenyum tanımlayıcısı nedir?

• Açık ve örtük bekleme arasındaki fark, bana bir örnek verir misiniz?

• java senkronize veri yapıları nedir ve ne zaman kullanıyorsunuz?

• JENKINS kullanıyor musunuz, (evet), Jenkins ile hangi SRC araçlarını kullanıyorsunuz?

• Test Stratejisi ve Test Planı oluşturma yaklaşımınız nedir?

• Tarayıcı uyumluluğu testi nedir?

• Test otomasyon çerçevesinin oluşturulmasıyla ilgili adımlar nelerdir?

Enes S.

• AWS sertifikanız olduğunu görüyorum,

o Biraz bahsedebilir misin?

o EC2'de jenkins nasıl yapılandırılır

• Günlük aktiviteler

• bir google doc belgesi açtı ve harita için bir liste ve anahtar değeri olarak bir değer yazmamı istedi--> nasıl koyarsınız

iç harita

• selenyumda eleman görünür değilse nasıl aşağı kaydırılır--> benden jsexecuter aşağı kaydırma yöntemini yazmamı istedi

• Java'da akış nedir?

Ferik Ç.

• Mevcut ve son iş projenizi açıklayabilir misiniz?

• Mevcut projenizde kaç API kullanıyorsunuz?

• Bulut sistemine aşina mısınız ve bunu açıklayabilir misiniz?

• AWS ile deneyiminiz var mı, lütfen açıklayın?

• Ekip yapınızı açıklayabilir misiniz?

• Bu pozisyonla neden ilgileniyorsunuz ve sizi neden işe alalım?

• Hiç sıfırdan bir çerçeve oluşturdunuz mu?

• BDD çerçevenizi ve mevcut projenizde veriye dayalı çerçeveye ihtiyaç duymanızın sebebini açıklayabilir misiniz?

ve lütfen açıklar mısınız?

• Mevcut projenizde kullandığınız araçları açıklayabilir misiniz?

• JMeter ile PostMan arasındaki fark?

• Entegrasyon testinin nasıl yapıldığını biliyor musunuz?

Merve Ö.

• Çerçevemi detaylı olarak anlattım.

• Ve bana duman ve regresyon testi hakkında sorular sordu.

o Duman mı yoksa regresyon testi mi olduğunu nasıl anlarsınız?

• OOP'yi sordu

• Özet ve arayüz?

• Kalıtım (Daha çok insan dilinde bir örnek veriyorum)

• Açık bekleme ve Kapalı bekleme

• Jira ve bitbucket hakkında konuştu

- 130 -

Toufiq N.

• Bana günlük aktivitelerinizden bahseder misiniz?

• Java açısından Webdriver nedir?

• Java'da bir dizgiyi nasıl tersine çevirirsiniz?

• Junit ek açıklamaları ile testNG ek açıklamaları mı?

• Selenium ile tarayıcıda açılır pencereleri nasıl yönetirsiniz?

• Bir tarayıcıda bir açılır menüyü nasıl yönetirsiniz?

• Diyelim ki bir tarayıcıdayız ve bizi farklı bir bağlantıya yönlendiren 5 farklı unsur var. Bir kez tıkladığımızda

element ve yeni bir pencere açılır, o pencereye nasıl geçilir?

• Selenium'da onay kutusu düğmesinin seçili olup olmamasını sağlayan yöntem nedir?

• XSSF ve HSSF arasındaki fark nedir?

• Örtük ve açık bekleme arasındaki fark nedir?

• Küme ve Liste arasındaki fark nedir?

• Haritalar çerçevesinde HashMap nedir?

• PUT ve POST http protokolü ve API arasındaki fark nedir?

• API'de sorgu parametresi nedir?

• Çerçevenizden bahseder misiniz? (çerçevenizde hangi araçları kullandığınızdan bahsetmeyin, NASIL kullandığınızı açıklayın.

çerçevenizde bu araçları kullandınız ve bu araçları NEDEN kullandınız...)

• Çerçevenizde POM.xml'nizi tanımlayın? (ne için kullanılır, çerçeveniz için neden önemlidir ve nasıl etkiler…

)

• Java'da Aşırı Yükleme ve Geçersiz Kılma arasındaki fark nedir?

• Bir alt sınıf sınıfı, Java'da final olarak başlatılan bir üst sınıfın işlevlerini çağırabilir mi?

• FindElement() ve FindElements() arasındaki fark nedir?

• Jenkins'e aşina mısınız? Ne için kullanıyorsun?

• Java'da çöp toplayıcı nedir?

• SQL'de iç birleşim nedir?

• SQL'de sol birleşim nedir?

• SQL'de sağ birleştirme nedir?

• SQL'de Birleşim nedir?

• SQL'deki kısıtlamalar nelerdir?

• URI nedir ve API açısından amacı nedir?

• API'de ne tür Kimlik Doğrulama vardır? OAuth(Oauthenticaion) nedir?

• Java'da İstisnaları nasıl ele alırsınız?

• Java'da bana farklı türde koşullu ifadeler verir misiniz? (if-else, üçlü, switch deyimi…)

• Selenium'da bir elementi bulmanın farklı yolları nelerdir?

• Bir öğenin yerini belirleme açısından, tek eğik çizgi(/) ve çift eğik çizgi(//) nedir?

• OOP'de Kapsülleme nedir?

• SQL'de Birincil ve Yabancı anahtar arasındaki fark nedir?

• Dizi ve Vektör (Koleksiyonlar çerçevesi) arasındaki fark nedir?

Tarık K.

**WEBEX RÖPORTAJI**

• Son projenizden ve özellikle Selenium ve Salatalık ile ilgili deneyiminizden kısaca bahseder misiniz?

• Otomasyon çerçeveniz hakkında biraz bilgi verebilir misiniz?

• Ekip yapınızdan bahseder misiniz?

• Diyelim ki size bazı test senaryoları verildi; bunların otomatik olup olmayacağına nasıl karar veriyorsunuz? bana belirli bir şey söyleyebilir misin

faktörler?

• Regresyon takımınız ne kadar büyük?

• Bunun için Q Takip -> Bunların tümü (test senaryoları) daha çok UI tabanlı mı yoksa REST tabanlı mı?

• Regresyon paketinin bir yürütme döngüsünü tamamlamak ne kadar sürer?

• Bunları yürüttükten sonra herhangi bir hata görüyor musunuz veya kusursuz çalışıyorlar mı?

• Karşılaştığınız bir otomasyon sorununu ve nasıl çözdüğünüzü anlatır mısınız?

• Bir web sayfasındaki dinamik öğeleri nasıl ele alırsınız?

• API dışında arka uç doğrulamaları yapıyor musunuz? (Veritabanı doğrulamasını ve nasıl yaptığımı sordu)

• Jira ve ALM gibi test yönetim araçlarıyla ilgili herhangi bir deneyiminiz var mı?

• Java kodunuz Jira'daki kusurları otomatik olarak tanımlıyor ve günlüğe kaydediyor mu yoksa bunu manuel olarak mı yapıyorsunuz?

- 131 -

• Raporlama yapınızdan bahseder misiniz? Raporlarınızı nasıl düzenlersiniz?

• UFT, İletki gibi diğer test otomasyon araçlarıyla ilgili herhangi bir deneyiminiz var mı?

• Kaynak kontrol yapınızdan bahseder misiniz?

**SKYPE RÖPORTAJI**

• Bana kendinden bahseder misin?

• Projeniz, çerçeveniz ve kullandığınız araçlar/teknolojiler hakkında bilgi verir misiniz?

• DDD ile de çalıştığınızı söylediniz, test verilerini nereden alıyorsunuz? Örnekler tablosundan mı yoksa dışarıdan mı

kaynak?

• API kullanırken, aldığınız farklı durum kodları nelerdir?

• Salatalık özellik dosyasındaki bileşenler nelerdir?

• Arka planı yalnızca bir senaryo için kullanabilir miyiz?

• UI otomasyonu için ne tür tarayıcılar kullandınız?

• Duman testlerinizi ne sıklıkla yapıyorsunuz? Nasıl bir CI/CD süreci izliyorsunuz?

• Regresyon takımınızda kaç tane test vakanız var?

• Regresyon takımınızı ne sıklıkla çalıştırıyorsunuz?

• Sprint'iniz ne kadar sürüyor?

• Çerçevenizde OOP'yi nasıl uygularsınız?

• Maven'i nasıl kullanıyorsunuz? (Ona testlerimi çalıştırması için maven komutlarından bahsettim ve bu komutların ne olduğunu sordu)

• Projenizde Collections çerçevesini kullandınız mı?

• Otomasyon yaparken bana bir meydan okuma veya senaryo söyleyebilir misiniz? (Zorluğu açıkladıktan sonra, daha fazlasını sordu.

bu zorlukla ilgili ayrıntılar)

• Tarayıcı açılır pencerelerini nasıl yönetiyorsunuz?

**TELEKONFERANS RÖPORTAJI**

• X'in (SDET ekip üyesi) görüşmeye katılmasını beklerken kendinizden bahseder misiniz?

• Bana şu anki Scrum Takımınızdan bahsedin, rolünüz nedir, orada kaç kişi var?

• Bir SDET olarak ekibe katkınız nedir?

• SDET ekip üyesi çağrıya katıldı ve sordu: Çerçevenizde bana rehberlik edebilir misiniz? teknolojiler nelerdir

ve kullandığınız araçlar? Test yürütmeniz nasıl oluyor?

• Ayrıca duman ve regresyon süitleri tuttuğunuzdan da bahsettiniz, bunları nasıl organize ediyorsunuz?

• Bir iş yaratmanın tüm adımlarında bana rehberlik edebilir misiniz? (Jenkins)

• Jenkins'iniz raporları nasıl gönderiyor? Raporları nasıl saklıyorsunuz? Ne tür raporlar kullanıyorsunuz?

• Rapor e-postasını kim alır?

• Salatalığın teknik olarak nasıl çalıştığını anlatır mısınız? Salatalık testlerini yürütme adımlarını açıklayın?

• Runner sınıfında bana rehberlik edebilir misin? Sınıfın içinde ne var?

• Sınıfınızda Page Object Model kullandığınızdan da bahsetmişsiniz, sayfanızın yapısından bahseder misiniz?

nesne sınıfları?

• Credit Union ve Bank arasındaki farkı anlatır mısınız?

• Kimler Kredi Birliği'ne üye olabilir?

• Üzerinde çalıştığınız uygulamanın bir veritabanı var mı?

• Kendi sorgularınızı mı yürütüyorsunuz, yoksa birileri sizin için mi yapıyor?

• Bana Inner Join ve Outer Join arasındaki farkı söyleyebilir misiniz?

• Bu teknik olabilir veya teknik olmayabilir, bana X'te çalışırken öğrendiğin bir şey söyle.

çalıştı)?

• Çerçevenizde bir API testi uygulamanız var mı? Nasıl kullandınız ve hangi araçları kullandınız?

• POJO'larınızdan bahseder misiniz?

• Günlük olarak üzerinde çalıştığınız HTTP durum kodları nelerdir?

Hazar M.

• Regresyon takımınızda kaç tane test vakanız var?

• Regresyon takımınızı ne sıklıkla çalıştırıyorsunuz?

• Bana kendinden bahset.

- 132 -

o A: BT kariyerime 2013 yılında manuel test uzmanı olarak başladım. Sonra Java'yı öğrendim ve bir yıl sonra aynı şirkette

Java kullanarak otomasyon yapmaya başladı. Otomasyon konusunda uzman olmama rağmen, şimdiye kadar Sağlık Hizmetlerinde QA gerçekleştirdim,

Hem manuel hem de otomasyon tekniklerini kullanan Sigorta ve Finans alanları. Java'da çok rahatım ve

OOPS kavramları. Şimdiye kadar, POM tasarımını kullanarak Maven, Gradle yerleşik BDD, DDF ve Hybrid çerçeveleriyle çalıştım

TestNG, JUnit, Cucumber, Selenium WebDriver gibi desen ve kullanılan test araçları ve Apache gibi kitaplıklar

POI, Rest- Assured, Jackson, JDBC içinde. SQL konusunda deneyimli, RDBS gibi RDBS'ler ile Veri Bütünlüğü Testi gerçekleştiren

Oracle, PostgreSQL ve MySQL. Farklı tipte çerçeveler oluşturma konusunda çok rahatım ve kolayca uyum sağlayabilirim

ve mevcut olanları korumak. Smoke, Regresyon, Fonksiyonel ve Back-End testlerinde deneyimliyim. şimdiye kadar, ben

Waterfall ve Agile/Scrum metodolojilerinde çalıştı ve SDLC, STLS ve Bug Life'ın tüm aşamalarında deneyimli

Döngü. Tüm scrum törenlerine yakından katıldı ve çapraz işlevli bir ekip üyesi olarak önemli bir rol oynadı. ben

çok pozitifim, sonuç odaklıyım, uyum sağlayabilen takım oyuncusuyum ve arkadaşlarım harika bir kişilerarası ve

iletişim becerileri, son teslim tarihine yetişmek için baskı altında çalışabilirim. Hızlı öğrenen ve esnek insan.

• 1. Test senaryoları, testler, süre, sürat koşusu, sürüm:

o Duman: Yaklaşık 20 test vakası ve 16 dakika +-20 saniye boyunca yürütülür, her gün sabah 7'de yürütülür

o Regresyon: 500'den fazla test senaryosu, 200'den fazla özellik dosyası, her biri 2-4 senaryo ve yaklaşık 5,5-6 saatlik yürütmeler sırasında yürütülür

her sprint, her sürüm, yeni işlevler eklendikten sonra, büyük hata düzeltildikten sonra

o Her sprint 3-5 kullanıcı hikayesi

o Günde 1-2 test vakası, sprint başına 15-20 test vakası

o Her 2 ayda bir (4 sprint), yaklaşık 14-18 kullanıcı hikayesi, sürüm başına 60-70 test vakası.

• Çevik sürecinizden bahsedin:

o A: Çevik süreçler genellikle, sık incelemeyi teşvik eden disiplinli bir proje yönetimi sürecini destekler.

ve takım çalışmasını, kendi kendini organize etmeyi ve hesap verebilirliği teşvik eden bir liderlik felsefesi olan adaptasyon, bir dizi

yüksek kaliteli ürünün hızlı teslimatına izin vermeyi amaçlayan mühendislik en iyi uygulamaları. Sprintim 2 hafta. Yaparız

tüm scrum törenleri ve ayrıca başka toplantılarımız da var: sprint ortası (2. Salı), önce

sürüm (2 ayda bir), bakım ve bilgi aktarımı (her ay) toplantıları. Sprint'imize sprint ile başlıyoruz

planlama toplantısı ve geliştireceğimiz ve test edeceğimiz uygulamanın bir kısmını öğreniyoruz. biz aldıktan sonra

proje hakkında genel fikir, hikayeler için bazı tahmin noktaları ve zaman vermek için sprint tımar yapıyoruz. Bizim

sürüm 2 ayda bir. Sprint başladığında, her sabah günlük stand-up toplantısı yaparız ve tartışırız.

dün ne yaptık, bugün ne yapacağız ve herhangi bir engelleyici varsa. Sprint sonu genellikle sprint yaparız

Demo. SDET olarak ekibimde sunumlar yaptım ve işlevleri gözden geçirdim. Müşteriler, paydaşlar veya

iş adamları anlamadıkları teknik kısım hakkında sorular soracaklar. Sprint demosundan sonra sprint yapıyoruz

geriye dönük toplantı. sprint retro'da, son sprint'te neyin iyi gittiğinden bahsederiz. ne tür hatalar yaptık hepimiz

iyi ve kötü! ve onların üzerinden geçip aynı hataları tekrar yapmamaya özen gösteriyoruz. iyi kısımlar hakkında

Devam edecek ve önümüzdeki sprint için daha iyi olmaya çalışacağız.

• Bana ekibinizden bahsedin:

o C: Ekibim, yüksek motivasyona sahip, uyumlu, çapraz işlevli ve kendi kendini organize eden bireylerden oluşur.

bilgili. Biz 4 geliştiriciyiz, 3 test uzmanıyız - 1 kılavuz, 2 otomasyon, 1 BA, 1 scrum master.

• Mevcut projenizdeki rol ve sorumluluklarınız nelerdir?

o C: Bir otomasyon mühendisi olarak, POM tasarım modeline dayalı "test çerçevemi" geliştiriyor/değiştiriyorum.

o Fonksiyonel test, duman testi, regresyon testi, arka uç testi gibi çeşitli testler yapıyorum.

o Uygulamada veya sprint sonunda yeni işlevsellik olduğunda regresyon testi yapmaktan veya

herhangi bir büyük hata düzeltildikten sonra.

o Ortamın çalışır durumda olduğundan emin olmak için duman testi raporlarını kontrol etmekten de sorumluyum.

sabah. Herhangi bir sorun varsa, onları analiz edeceğim. sunucu sorunuysa, hemen geliştiricilerle iletişime geçeceğim. Eğer o

komut dosyalarım hakkında, onları ayıklayacağım. eğer bir kusursa, onu yeniden üreteceğim ve kusuru kaydedeceğim.

o Şimdi, otomatik test senaryolarının yaklaşık %80'ine sahibiz. 50/50'den önce ana başarımdı.

o Ayrıca Back-end otomasyon faaliyetlerinde bulundum, bu yüzden otomasyon için emin kitaplığı kullanıyordum, Postman-

Manuel API testi için istemci. PostgreSQL DB'leri ile API testi yapıyordum

o Hata izleme aracı olarak Jira kullanıyorum. Hata geliştiriciler tarafından düzeltildikten sonra tekrar test ediyorum ve geçerse kapatıyorum. Eğer

Arıza giderilmedi biletimi tekrar açacağım. Ayrıca, Agile-Scrum ekibinin bir parçası olarak, çeşitli organizasyonlara katılıyorum.

gereksinim incelemeleri için gözden geçirme toplantıları yapın ve BA'ya değerli geri bildirim sağlayın. Son olarak, ben bir çapraz fonksiyonluyum

Sprint hedefimize ulaşmak için takımıma herhangi bir şekilde yardım etmeye her zaman istekli olan takım üyesi. Bu guzel

şu anki projemde otomasyon mühendisi olarak rolüm hakkında çok şey var.

• Bana çerçevenizden bahsedin:

- 133 -

o Mevcut projemdeki çerçevem Maven tarafından oluşturulmuş, içinde DDT bulunan Selenium Java BDD çerçevesi. Çerçeve (şimdiki değeri)

POM ve Singleton WebDriver tasarım deseni kullanılarak geliştirilmiştir. Geliştirmeye aktif olarak katılıyorum ve

OOPS'ye dayanan yeni sayfa nesneleri, genel ve işlevsel yöntemler ekleyerek çerçeveyi korumak.

o Çerçeve JUnit ile Salatalık kullanır ve JDBC ve API Testini kullanarak Veri bütünlüğü Testini yürütebilir.

RestAssured kütüphaneleri. Framework, tüm testlerin yaklaşık %80'ini otomasyon komut dosyalarını yönetir. Çerçeve bağlı

Jenkins tarafından kaynak kontrolü ve Sürekli Entegrasyon için uzaktan GitHub deposuna. Salatalık Json üretir ve

Hata ekran görüntüleri ile html raporları. İyi organize edilmiş ve anlaşılması ve uyarlanması kolaydır.

• Toplantılar:

o Sprint planlama - 1. hafta 1. gün - Salı

o Sprint demosu - 2. Pazartesi sabahı

o Sprint retro - 2. Pazartesi öğleden sonra

o Her gün saat 10.00'da günlük stand-up

• Bana uygulamanız ve şu anda üzerinde çalıştığınız işlevsellik hakkında bilgi verin

o Uygulamam web uygulamasıdır, geniş bankacılık, yatırım ve aracılık hizmetleri vermekteyiz. Şu anda, biz

ipotek taleplerinde Bireysel yaklaşım işlevi üzerinde çalışmak, bu nedenle müşteri ipotek için başvurduğunda,

onay sonucu sadece kredi raporuna ve gelirine değil, aynı zamanda üyelik geçmişine ve varlıklarının oynamasına da bağlıdır.

karar vermede önemli rol oynar. İpoteğinizi varlıklarınızla sigortalama veya bunları dahil etme fırsatı verir. Son

Hata bulundu şartı, kredi raporundan sonra bu seçeneği seçebileceğinizi söylüyor. Sistem otomatik olarak size gösterir

Gerekirse kredinizi sigortalamak için önerilen minimum miktar. Ayrıca manuel olarak girmenizi sağlar ve gösterir

minimum sınır değeri. Miktar kutusundan çıktıktan sonra daha az girdiğinizde hata mesajı görüntülenmelidir.

kutunun yanında. Ancak asıl sonuç, yatırım türleriyle ilgili farklı bir mesajdı.

• 2. Penceresiz, bir kapısı kapalı, içeride 3 ampul, dışarıda 3 anahtar bulunan bir odanız olduğunu hayal edin. Sen

kapıyı bir kez açmasına izin verildi. Hangi anahtarın hangi ampule bağlı olduğunu nasıl belirleyebilirsiniz.

• 3. Rest Api hakkında bir saatlik güzel bir soru. Neredeyse her şeyi sordular, Pojo da dahil.

• 4. Eleman listeniz varsa, belirli bir elemanın görünür olup olmadığını kontrol etmek için nasıl seçebilirsiniz?

• 5. Geliştiricilerle nasıl bütünleşirsiniz?

• 6. Test planı, test senaryosu oluşturma yaklaşımınız nedir?

• 7. Bildiğiniz gibi Api statik yöntemler kullanıyor, test senaryolarımızı yürütmek uzun zaman alıyor, paralel çalıştıramıyoruz,

Statik yöntemler kullandığı için, sizce bu yürütme süresini nasıl azaltabilir ve kısaltabiliriz?

• 8. Kütüphanedeki herhangi bir yöntemin düzgün çalışıp çalışmadığını nasıl kontrol edebilirsiniz?

• 9. Lütfen çerçevenizi çerçevenizde POM'u nasıl kullandığınızı çizin

• 10. Stres testi, birim testi ve entegrasyon testini nasıl gerçekleştiriyorsunuz?

• 11. Paralel test yaparken statik nesneler kullandığımızda çakışmalar oluyor, daha iyi önerileriniz var mı?

• 12. Ne tür API'ler biliyorsunuz, hangisini tercih ediyorsunuz, neden?

• 13. xpath ve css arasındaki fark nedir? hangisini tercih edersin? Neden? zorluklar nelerdir?

• 14. Sunucuya istek gönderiyoruz ve yanıtı bir java nesnesi olarak geri alıyoruz, yanıtı nasıl test edebilirsiniz?

• 15. Junit'i neden Salatalık ile kullanıyorsunuz? TestNG değil mi?

• 16. Salatalığın artılarını ve eksilerini anlatır mısınız?

• 17. API kitaplığınızı nasıl yapılandırırsınız? Bu lib'den yöntemleri nasıl çağırırsınız?

• 18. Bana kendinizden, çerçevenizden, rolünüzden bahsedin

• 19. Bu şirkete neden başvurdunuz, kültürümüzden ve ne yaptığımızdan bahsettiniz.

• 20. Bir ünite test görevi, bir java görevi (kelimenin Palindrome olup olmadığını kontrol edin)

• 21. Şu anda yük dengelerimizle ilgili bir performans sorunu yaşıyoruz (tüm veri akışıyla birlikte grafik sorunu gösteriliyor)

bu zorluğu nasıl çözebilirsin?

• 22. Arkadaşım şirketinizde QA müdürü, George'u tanıyor musunuz? J

• 23. Postacı hakkında konuşun, Postacı kullanarak nasıl test yaparsınız?

• 24. İsteğinize (API) gövde nasıl eklenir?

• 25. Test verilerinizi nasıl tanımlarsınız? nereden buluyorsun?

• 26. Geliştirme ortamında test yapıyor musunuz?

• 27. Bir çerçeve oluştururken neleri dikkate alıyorsunuz?

• 28. Zaman aşımı zorluklarını nasıl çözersiniz? Selenium'un sağladığı bekleme yöntemlerine güvenmiyorum? Nasıl buldun

zaman aşımı sorunu olduğunda uygun bekleme yöntemi?

• 29. API kitaplığımızda, bir parametreyi kabul eden ve sunucuya istek gönderen ve geri dönen bir arama yöntemi vardır.

Java nesnesi, bu yanıtı nasıl test edersiniz? (düzenlendi)

• 30. aaaabbcccdda==> a4b2c3d2a için bir metod yazdınız, güzel.! Yönteminizin işe yarayıp yaramadığını kontrol etmek için hangi verileri kullanacaksınız?

dogru ya da değil? *(cevap bu soruların sonundadır)*

• 31. Boş zamanlarınızda ne yaparsınız?

- 134 -

• 32. İşyerinde boş olduğunuzda ne yaparsınız? sadece telefonunu al ve oyun oyna?

• 33. Hangi test senaryolarının otomatikleştirileceğine ve nasıl tanımlanacağına kim karar veriyor?

• 34. Ne sıklıkla uzaktan çalışıyorsunuz ve bu, iş verimliliğini azaltıyor mu?

• 35. Bir kalem aldı ve benzersiz bir şekil çizdi ve içine nokta koydu, bu bir harita dedi ve sen buradasın, hangi yolları yapacaksın?

Bu haritadaki benzin istasyonlarını tanımlamayı mı seçiyorsunuz? Telefonumu toparlayıp google'a aratacağımı söyledim)

*dize almak ve aşağıdaki gibi döndürmek için bir yöntem yazın:*

*aaaabbcccdda==> a4b2c3d2a*

public class returnWord {

public static void main(String[] args) {

System.out.println(getword(“aaaabbcccdda”));

}

public static String getword(String word) {

int sayı=1;

ret=“”;

for (int i = 0; i < word.length()-1; i++) {

if(word.charAt(i)==word.charAt(i+1)) {

say++;

devam et;

}başka {

ret=ret+word.charAt(i)+count;

}

say=1;

}

ret=ret+kelime.charAt(kelime.uzunluk()-1);

geri dönüş;

}

}

Görüşme soruları

\*JVM:

- java sanal makinesidir, jvm aslında java bayt kodunu çalıştırır

.class dosyaları

- bir kez yazın ve her yerde çalıştırın -> bayt kodu

\* JRE:

- java çalışma zamanı ortamı, bir java'yı çalıştırmak için ihtiyacınız olan şey mi?

program

ve diğer şeylerin yanı sıra bir jvm içerir.

\*JDK:

- java geliştirici kitidir -- derlemek için ihtiyacınız olan şey jdk'dir

java kaynak kodu. java'dan .class dosyasına

- Java'da programlamak için jdk'ye ihtiyacınız olacak, JDK'yı şuradan yükleyebilirsiniz:

oracle sitesi.

-------------------------------------------------- ---------------

\* Path ve classPath arasındaki fark?

- Yol:

yürütülebilir dosyaların .exe dosyalarının nerede olduğunu tanımlamak için kullanılır

java.exe, javac.exe vb.

- Sınıf yolu:

Java.class dosyalarının konumunu belirtmek için kullanılır

- java kaynak kodu -> derleme (javac) -> bytecode .class uzantısı

-------------------------------------------------- ---------------

\* birlikte çalıştığınız java sürümleri?

\* Java 7 ve 8 arasındaki fark nedir?

- Sürüm Yayın bitiş genel tarihleri

- Java SE 7 2011

- Java SE 8 2014 Ocak 2019, Aralık 2020

- Java SE 9 2017 Mart 2018

- Java SE 10 2018 Eylül 2018

JAVA 7 vs JAVA 8 }

- İçeri dize - Lambda İfadesi

Switch deyimi - işlem hatları ve Akışlar

- Çoklu İstisna - tarih ve saat API'si

İşleme - Java 8 arabirim değişiklikleri

Statik Yöntem, Varsayılan

Yöntem

-------------------------------------------------- ---------------

\* Çöp toplayıcı nedir ve kaç çeşidi vardır?

- Otomatik çöp toplama, yığına bakma işlemidir.

Hangi nesnelerin kullanımda olduğunu ve hangilerinin kullanılmadığını belirleyen bellek

ve kullanılmayan nesneleri silme

- Kullanımdaki bir nesne veya başvurulan bir nesne, bazılarının

programınızın bir kısmı hala buna bir işaretçi tutuyor

nesne

\* Çöp Toplayıcılar

- seri : tek bir iş parçacığı ile çalışır, tek bir iş parçacığı için çalışır

dişli uygulamalar

- paralel: yığın alanını yönetmek için birden çok iş parçacığı kullanır

- CSM: (geçerli işaret taraması) birden çok GC iş parçacığı kullanır

- G1: çok işlemcili uygulamalar için tasarlanmıştır

büyük bellek alanına sahip makineler. yığını daha küçük parçalara ayırır

bölgeler

\* Çöp Toplayıcı nasıl aranır?

- System.gc();

-------------------------------------------------- ---------------

\* Public Static void main'i açıklayın (String args[])

- halka açık:

public, kimin kim olduğunu belirtmek için kullanılan bir erişim değişikliğidir.

bu yönteme erişebilir. Public, bu yöntemin uygulanacağı anlamına gelir.

olmak

herhangi bir Sınıf tarafından erişilebilir.

- statik:

Java'da sınıf temelleri olduğunu tanımlayan bir anahtar kelimedir.

yani

örneğini oluşturmadan erişilebilir

sınıf

- geçersiz:

void metodunun dönüş tipidir.

hangi yöntem

herhangi bir değer döndürmez

- ana:

olarak JVM tarafından aranan yöntemin adıdır.

belirli bir uygulama için başlangıç noktası

sadece imza.

ana yürütmelerin gerçekleştiği yöntemdir

- Dize argümanları:

ana yönteme iletilen parametredir

-------------------------------------------------- ---------------

\* bir sınıf başka bir sınıfın ana yöntemini çağırabilir mi?

- evet, başka bir sınıf ana yönteminde bir ana yöntem çağrılabilir

- kod ana yöntem olmadan derlenecek ancak çalışmayacak

-------------------------------------------------- ---------------

\* ilkel ve sarmalayıcı sınıfları nelerdir?

- her ilkel veri türünün kendisine ayrılmış bir sınıfı vardır

- bunlar sarmalayıcı sınıflar olarak bilinirler çünkü

ilkel veri türü

o sınıfın bir nesnesine.

- döküm kullanarak bir türden diğerine dönüştürebiliriz

\* İLKEL \* SARICI SINIF \* YAPICI ARGUMENTİ

boole Boole boole veya String

bayt Bayt bayt veya Dize

karakter karakter

int Tamsayı int veya Dize

şamandıra şamandıra şamandıra çift veya

Sicim

double Double double veya String

uzun Uzun uzun veya String

kısa Kısa kısa veya Dize

-------------------------------------------------- ---------------

\* örtük döküm vs açık döküm?

- Örtülü Döküm:

int ben = 100;

çift d = ben;

- Açık Döküm:

int n = 12;

bayt b = (bayt)n;

- Otomatik - Boks

Tam sayı = n;

- Un - Boks

int j = sayı;

ÇALIŞMA:

Tamsayı i = yeni Tamsayı(100);

çift d = ben ; // çalışmayacak

-------------------------------------------------- ---------------

\* Örnek Değişken ve statik Değişken arasındaki fark?

- Statik Değişkenler:

nesnelere değil sınıfın kendisine aittir

sınıfın farklı nesneleri olamaz

farklı

statik bir değişken için değerler. sadece bir kopyası

statik değişken

herkes için. Bir sınıfın her nesne örneğinin sahip olduğu

kendi kopyası

sınıfta bildirilen her örnek değişkenin

-------------------------------------------------- ---------------

\* Java'da yapıcı nedir

\* yapıcı ve yöntem arasındaki fark nedir?

- Java'daki bir yapıcı, bir yönteme benzer bir kod bloğudur

buna denir

bir nesnenin örneği oluşturulduğunda. arasındaki fark

yapıcı ve

yöntem:

- Bir kurucunun dönüş tipi yoktur

- yapıcının adı, kurucunun adıyla aynı olmalıdır.

sınıf

- yöntemlerden farklı olarak yapıcılar, bir

sınıf

- yeni bir örneği olduğunda bir kurucu otomatik olarak çağrılır

nesne

yaratıldı

- bir kurucunun amacı, nesnesini başlatmaktır.

Bir sınıf

bir yöntemin amacı bir görevi yerine getirmek iken

java kodunu yürütme

- programcı eklemediyse varsayılan kurucu eklenir

yapıcı.

Java Yapıcısı Türleri

|

|

------------------------------------------

| |

varsayılan kurucu parametreli

yapıcı

-------------------------------------------------- ---------------

\* Bir super() ve this() anahtar sözcükleri aynı kurucuda olabilir mi?

- genel bilgisayar(){

Bu();

Süper();

//kod

}

- HAYIR her ikisinin de kurucu içinde ilk satırda olması gerekir.

-------------------------------------------------- ---------------

\* Süper. VS Süper()

- Süper.

ebeveyn/süper sınıf üyelerine erişmek için kullanılır değişkenler, yöntemler

- Süper()

süper sınıf yapıcısını çağırmak için kullanılır

\* Bu. buna karşı()

- Bu.

bunu kullanarak bu nesneyi. örneğe erişebiliriz

değişkenler ve

yöntem. örnek ve argüman arasında ayrım yapmak

değişken

- Bu()

aynı sınıftaki başka bir kurucudan bir kurucu çağırın

Örnek:

genel sınıf bilgisayar{

özel String markası;

ortak bilgisayar(){

bu elma");

}

halka açık bilgisayar (String markası){

this.marka = marka;

}

}

-------------------------------------------------- ---------------

\* Java'da erişim değiştiricileri arasındaki fark nedir?

Değiştirici Sınıf Paketi Alt Sınıf Dünya

- genel YYYY

- korumalı YYYN

- değiştirici YYNN yok

- özel YNNN

-------------------------------------------------- ---------------

\* Java'da statik anahtar kelime nedir?

- Statik anahtar kelime, değişkenin veya yöntemin ait olduğu anlamına gelir.

sınıf

ve tüm örnekler arasında paylaşılır.

- sınıf adını veya nesnesini kullanarak statik üyeleri çağırabiliriz

- statik yöntemler, Statik Olmayan üyeleri çağıramaz/başvuru yapamaz

- Statik şunlar olabilir: değişkenler, yöntem, blok, iç sınıf

genel sınıf Bilgisayar{

özel String markası;

genel Bilgisayar(){

bu elma");

}

genel Bilgisayar(Dize markası) {

this.marka = marka; }

genel statik sınıf Bellek{

özel int ramsize;

}

}

Bellek c = yeni Bilgisayar().yeni Bellek();

-------------------------------------------------- ---------------

\* Statik blok ve örnek başlatıcı bloğu?

- statik başlatıcı bloğu, yüklendiğinde çağrılacak

sınıf ve kod

başlangıçta sadece bir kez çalışacak

- her oluşturduğunuzda örnek başlatıcı bloğu yürütülür

nesne

bir sınıf için. herhangi bir kurucudan önce.

Örnek:

genel sınıf programcısı{

System.out.println("kurucu")

}

statik{

System.out.println("statik")

}

{

System.out.println("örnek")

}

-------------------------------------------------- ---------------

\* değere göre mi yoksa referansa göre mi geçiyorsunuz?

- Java bir "değere göre geçiş" dilidir. Bu, bir kopyanın

değişken yapılır ve themethod bu kopyayı alır. Ödevler

yönteminde yapılan arayan etkilemez.

-------------------------------------------------- ---------------

\* final vs nihayet vs finalize()

- Son:

- final bir anahtar kelimedir

- final, sınıf yöntemine kısıtlama uygulamak için kullanılır ve

değişken

- bir sınıf final olarak işaretlenmişse, bu sınıf olamaz

miras

başka bir sınıf tarafından

- en sonunda:

- nihayet bir blok

- nihayet istisna işleme için kullanılan bir bloktur

ile birlikte

dene ve blokları yakala

- Sonuçlandırmak:

- finalize() yöntemi, korumalı bir yöntemdir

java.lang.nesne sınıfı

java'da oluşturduğunuz her sınıfa miras alınır

- bazı temizlik işlemlerini gerçekleştirmek için kullanılan finalize() metot kimliği

operasyonlar

bellekten kaldırılmadan önce bir nesne üzerinde

-------------------------------------------------- ---------------

\* çerçevenizde statik nerede kullandınız?

- Normalde daha iyi yeniden kullanılabilirlik için yardımcı program sınıfları yazarım ve

yardımcı programlarımdaki tüm yöntemler statik yöntemler olarak

kolayca

onları sınıf adıyla çağırın.

- dataseUtil sınıfı, sorguları bağlamak ve çalıştırmak için statik yöntemlere sahiptir

ve

veri al

-------------------------------------------------- ---------------

\* Java'da eşittir yöntemi ile "==" operatörü arasındaki fark?

- "==":

referans ve adres karşılaştırması

== ilkel olması durumunda değeri kontrol ediyor. olarak

ilkeller

herhangi bir davranış olmadan tek değer.

- eşittir()

içerik karşılaştırması

-------------------------------------------------- ---------------

\* Dize havuzu nedir?

- Java'daki String Pool, Java Heap'te depolanan bir String havuzudur

Hafıza. Eskiden

bellekte yerden tasarruf edin.

-------------------------------------------------- ---------------

\* Java'da String nasıl tersine çevrilir?

1:

String string = "ne olursa olsun";

Dize ters =

newStringBuilder(string).reverse().toString();

System.out.println(ters);

2:

Dize sonucu = "";

for(int i = string.length()-1; i>=0; i--){

sonuç = sonuç+ string.charAt(i);

}

System.out.println(sonuç);

-------------------------------------------------- ---------------

\* Bir cümleyi kelime kelime nasıl tersine çevirebilirim?

StringBuilder stringBuilder = new StringBuilder();

String [] kelimeler = string.split(" ");

for(int j = kelimeler.uzunluk-1; j>=0; j--){

stringBuilder.append(kelimeler[j].append("");

}

System.out.println("ters Sözcükler"+

stringBuilder.toString().trim());

-------------------------------------------------- ---------------

\* String vs StringBuilder vs StringBuffer?

STRING STRINGBUFFER STRINGBUILDER

Depolama sabiti HEAP Heap

Alan Dize havuzu

Değiştirilebilir hayır(değişmez) EVET(değişebilir) EVET(değişebilir)

İplik Güvenli EVET EVET HAYIR

Performans HIZLI ÇOK YAVAŞ HIZLI

Değişmez, değerlerin oluşturulduktan sonra değiştirilemeyeceği anlamına gelir:

-------------------------------------------------- ---------------

\* Sırt güvenli veya Senkronize nedir?

- sırt nedir: Sıralı veya tek iş parçacıklı bir program

tek akış

- senkronize: İki iş parçacığının yöntemi çalıştıramayacağı anlamına gelir

veya değişkenlere aynı anda erişin ve JVM ilgilenir

nın-nin

bunu zorlamak. iplik güvenliğini sağlamak için kullanılır

-------------------------------------------------- ---------------

\* String'in palindrom olup olmadığı nasıl kontrol edilir?

- civic = civic <- bir palindromdur

- genel statik boolean isPalindrome(String str){

if(str==null)

yanlış döndür;

StringBuilder strBuilder = new StringBuilder(str);

strBuilder.reverse();

strBuilder.toString().equals(str) döndür;

}

-------------------------------------------------- ---------------

\* Singleton class nedir ve nasıl singleton class yapabiliriz?

- Bir sınıfın yalnızca bir örneğe sahip olmasını ve genel bir

puan

ona erişim

genel sınıf SingletonExample{

// statik üye, singleton sınıfının yalnızca bir örneğini tutar

özel statik singletonÖrnek singletonInstance;

// singleton, başka herhangi bir sınıfın başlatılmasını engeller

özel SingletonExample(){

}

}

Örnek:

- Java'da Singleton Örneği

genel sınıf dbaseConnector{

özel statik nihai dbaseConnector örneği = yeni

DbaseConnect();

özel dbaseConnector(){

//yapı kodu ....

}

genel statik DbaseConnector getInstance(){

dönüş(örnek);

}

}

-------------------------------------------------- ---------------

\* armtron nedir?

-------------------------------------------------- ---------------

\* Bir sınıfın Örneklenmesi nasıl engellenir?

- özel kurucu

- soyut sınıf

-------------------------------------------------- ---------------

\* Java'da OOP kavramları nelerdir?

- kapsülleme

- miras

- soyutlama

- polimorfizm

-------------------------------------------------- ---------------

\* Kapsülleme nedir ve nasıl kullandınız?

- veri: değişkenleri özel yaparak ve genel sağlayarak gizleme

alıcı ve ayarlayıcı yöntemleri.

- projemde birden fazla POJO/BEAN sınıfı oluşturdum.

üstesinden gelmek

test verileri ve gerçek veriler.

EX: API yanıtından JSON alıyorum ve POJO'mun nesnesine dönüştürüyorum

sınıf tüm değişkenler alıcılar ve ayarlayıcı ile özeldir.

-------------------------------------------------- ---------------

\* Miras nedir ve faydaları nelerdir?

- kalıtım, bir alt sınıfın üyeleri miras aldığı bir süreçtir.

süper sınıf.

Tüm genel ve korumalı ve bazen varsayılan üyeler

miras kalan

alt sınıf.

- onunla kod yeniden kullanılabilirliği ve kod organizasyonu elde edeceğiz.

ayrıca

polimorfik programlama için çok kullanışlıdır.

- test tabanı süper sınıftır ve diğer test sınıfları alt sınıflardır.

EX: bir sayfa için bir sayfa nesnemiz vardı ve başka bir sayfa olarak

aynı içeriyor

tanımlanan öğeleri yeniden kullanmak için kalıtım kullandım

süper sayfada

selenyum web sürücüsünde nesne sınıfı.

-------------------------------------------------- ---------------

\* Java'da soyutlama ve amaç nedir?

- Genel olarak programlamada soyutlamayı kullanarak bir temel oluşturabiliriz/

taslak

kodumuz için ve alt sınıfların gerektiği gibi uygulanmasına izin verin

Gereksinimler

uygulamayı gizleme ve yalnızca davranışı gösterme

soyut sınıfın bir yapıcısı olabilir

JAVA'da soyutlama elde etmenin 2 yolu:

1) Soyut Sınıflar

2) arayüzler

-------------------------------------------------- ---------------

\* Soyut sınıf ve arayüz arasındaki fark?

- java'da soyut sınıf ve arayüz arasındaki fark

- yöntem türleri: arabirimde yalnızca soyut yöntemler olabilir.

Öz

sınıf soyut ve soyut olmayan yöntemlere sahip olabilir. java 8'den

varsayılan olabilir

ve ayrıca statik yöntemler.

- son değişkenler: ajva arayüzünde bildirilen değişkenler

varsayılan son.

soyut bir sınıf, nihai olmayan değişkenler içerebilir

- bir sınıf birden çok arabirim uygulayabilir, ancak yalnızca genişletilebilir

bekar

soyut sınıf. java çoklu kalıtımı desteklemiyor

- yöntemler

- arayüzde: soyut statik varsayılanlar,

- soyut sınıfta: soyut soyut olmayan statik

- soyut sınıfın bir kurucusu olabilir

arayüzün yapıcısı olamaz

- arayüz diğer arayüzleri genişletebilir (çoklu)

-------------------------------------------------- ---------------

\* java neden çoklu kalıtımı desteklemiyor?

-

-------------------------------------------------- ---------------

\* Aşırı yükleme ve geçersiz kılma arasındaki fark?

Aşırı yükleme, farklı yöntemlerin aynı ada sahip olmasına, ancak farklı yöntemlere sahip olmasına izin verir.

imzanın giriş parametrelerinin sayısına göre farklılık gösterebileceği veya

giriş parametrelerinin türü veya her ikisi. Aşırı yükleme derleme ile ilgilidir

zaman

(veya statik) polimorfizm.

1: overlaoding = aynı yöntem adı ancak farklı parametreler

aşırı yükleme = dönüş tipi farklı olabilir

2: geçersiz kılma = aynı tür veya alt tür

geçersiz kılma = aynı yöntem adı ve aynı parametreler

-------------------------------------------------- ---------------

\* Statik bir yöntemi geçersiz kılabilir misiniz?

- Hiçbir statik yöntem geçersiz kılınamaz, yalnızca

gizlenmiş.

-------------------------------------------------- ---------------

\* Statik bağlama ile dinamik/çalışma zamanı bağlaması nedir?

- Statik bağlama aşırı yükleniyor ve

- dinamik bağlama, yöntemi geçersiz kılar.

-------------------------------------------------- ---------------

\* Polimorfizm nedir? çerçeve üzerinde nerede kullandık?

- Polimorfizm, bir nesnenin birçok form alma yeteneğidir.

OOP'de polimorfizmin en yaygın kullanımı, bir ebeveyn

sınıf referansı, bir alt sınıf nesnesine atıfta bulunmak için kullanılır.

- DİNAMİK POLİMORİİZM - GEÇERLİ KIRMA

- STATİK POLİMORİİZM - AŞIRI YÜKLEME

-------------------------------------------------- ---------------

\* Veri Yapıları ve Neden İhtiyacımız Var?

- Veri yapıları, verileri verimli bir şekilde organize etmenin bir yoludur.

manipülasyon: Ekleme, arama, okuma, silme

veri.

- Verileri okumak ve depolamak için her zaman Java veri yapılarını kullanırım

uygulamamızdan, veritabanımızdan veya API'den gelen veriler.

-------------------------------------------------- ---------------

\* Dizi veya dinamik diziyi nasıl oluşturursunuz?

- int [] sayılar = yeni int [7];

sayılar [0] = 100;

sayılar [1] = 100;

sayılar [2] = 100;

sayılar [3] = 100;

sayılar [4] = 100;

sayılar [6] = 100;

- Java dinamik Dizileri desteklemiyor

Dinamizm için ArrayList kullanmanız gerekir.

-------------------------------------------------- ---------------

\* Array'den benzersiz değerler nasıl okunur?

- int [] sayılar = yeni int [7];

sayılar [0] = 44;

sayılar [1] = 32;

sayılar [2] = 100;

sayılar [3] = 100;

sayılar [4] = 7;

sayılar [5] = 7;

public static void main (String[] args){

// 1. yol

int [] sayılar = yeni int [6];

sayılar [0] = 44;

sayılar [1] = 32;

sayılar [2] = 100;

sayılar [3] = 100;

sayılar [4] = 7;

sayılar [5] = 7;

Set<Tamsayı> set = new HashSet<>();

for (int i = 0; i < sayı.uzunluk; i++) {

set.add(sayı[i]);

}

System.out.println(set);

// 2. yol

String uniqNums="";

for (int n:nums) {

if (!uniqNums.contains(String.valueOf(n)))

uniqNums = uniqNums+n+" ";

}

System.out.println(uniqNums);

}

-------------------------------------------------- ---------------

\* String'den benzersiz değerler nasıl yazdırılır?

-

public static void main (String[] args){

String str = "aaabbcccdddeereeegggg";

// 1. yalnızca dizeden benzersiz değerleri yazdır

Set<Karakter> setC = new HashSet<>();

for (int i = 0; i < str.length(); i++) {

setC.add(str.charAt(i));

}

System.out.println(setC);

// dizeden yalnızca benzersiz değerleri 2. yazdır

Benzersiz dize ="";

for (int i = 0; i < str.length(); i++) {

if(!unique.contains(str.charAt(i)+"")) {

benzersiz += str.charAt(i);

} }

System.out.println(benzersiz);

}

}

-------------------------------------------------- ---------------

\* dizi ve dizi listesi arasındaki fark?

- Dizi sabit boyuttadır ArrayList dinamiktir

- Dizi, ilkel ArrayList türünü yalnızca sarmalayıcı türlerini depolayabilir ve

nesneler

- Dizi çok boyutlu olabilir ArrayList değil

- Dizi daha hızlı ve ArrayList daha yavaş

-------------------------------------------------- ---------------

\* Her döngü için yineleyici ve arasındaki fark nedir?

- Yineleyici, diziyle değil, arrayList ile çalışır. Bize yardımcı olacak

Öğeler arasında yineleyin.

- Fark, yineleyici ile değişiklik yapabilirsiniz (öğeyi kaldırın)

yineleme sırasında listeye.

her döngü için listemizde değişiklik yapamayız

// her döngü için:

int dizi[] = {23,54,6,56,76,878,9};

for(int n : dizi) {

System.out.println(n);

}

System.out.println("=================");

Liste<Tamsayı> liste = new ArrayList<>();

liste.add(23);

liste.add(2323);

liste.add(2312);

liste.add(2343);

liste.add(2553);

// Yineleyici döngüsü:

Yineleyici<Tamsayı> o = list.iterator();

while(it.hasNext()) {

Tamsayı n = it.next();

System.out.println(n);

it.remove();

}

System.out.println(liste);

System.out.println("=================");

for(Tamsayı n: liste) {

System.out.println(n);

//liste.kaldır(n);

}

-------------------------------------------------- ---------------

\* 2 diziyi tek dizide nasıl birleştirebilirim?

- Eminim bu çözümü çözmenin farklı yolları vardır.

bu noktada geliyorum:

1. boyutunda bir dizi oluştururdum

2. diğer diziler ve her iki diziden de her değeri atayın

bir döngü kullanarak 3. diziye.

3. Bir dizi listesi oluşturabilir ve içindeki her bir Öğe arasında dolaşabilirim.

2 dizi ve bunları dizi listesine ekleyin,

Sonra gerekirse Arraylist'i diziye dönüştürebilirim.

-------------------------------------------------- ---------------

\* Liste vs Set vs Harita arayüzleri

- LİSTE: - yinelenen değerleri saklayabilir,

- ekleme sırasını korur / tutar

- liste herhangi bir sayıda boş değere izin verir

- Koleksiyon arayüzünden gelir

- listeden belirli bir değeri dizine göre okuyabiliriz

- SET : - sadece benzersiz değerleri saklayabilir.

- set düzeni korumaz

- set yalnızca 1 boş değere izin verir

- Koleksiyon arayüzünden gelir

- set buna izin vermiyor

- MAP : bir anahtar+değer biçimidir ve anahtarlar her zaman benzersizdir

- anahtarı geçerek harita okuma değeri

- haritanın tek bir boş anahtarı olabilir, ancak birden çok boş değeri olabilir

-------------------------------------------------- ---------------

\* hashcode ve equals yöntemi ne için kullanılır?

- Birden çok POJO (Düz eski Java nesnesi) sınıfı oluşturdum

farklı kaynaklardan gelen verileri depolamak için.

Veritabanı, API JSon formatı, Excel, CSV dosyası, Webtable gibi.

- Ve ne zaman benzersiz bir pojos setine ihtiyaç duysam, geçersiz kılmak zorunda kaldım

hashCode ve equals yöntemleri, HASHSET içine depolayarak,

Yalnızca benzersiz nesnelerin saklandığından emindim.

- Karşılaştığınız zorluklar ve bunları nasıl çözdünüz?

hashCode ve equals yöntemlerini ne zaman geçersiz kılmak zorunda kaldınız?

Projemde web tablolarından, veritabanından verileri doğrulamak zorunda kaldım,

JSON ,

Exceller vb. Hepsi farklı formatlarda geldiği için,

zorlayıcı

onlarla çalışmak ve doğru olduklarını iddia etmek ve

bekleniyor.

Yaptığım şey, uygulama verilerime uyması için birden fazla POJO oluşturmaktı.

Ve bazen, benzersiz POJO nesnelerine ihtiyacım var, hashCode'u geçersiz kılıyorum

ve yöntemlere eşittir.

- Otomasyonda veri yapılarını nasıl kullandınız?

Projemde web tablolarından, veritabanından verileri doğrulamak zorunda kaldım,

JSON ,

Exceller vb. Hepsi farklı formatlarda geldiği için,

zorlu

onlarla birlikte çalışın ve doğru ve beklendik olduklarını iddia edin.

Bu yüzden farklı kaynaklardan gelen tüm verileri Java Verilerine sakladım.

yapılar.

Listeler, Haritalar vb. gibi ve beklenenle karşılaştırıldı/onaylandı.

= Örnek:

Çalışan emp1 = yeni Çalışan(100);

Çalışan emp2 = yeni Çalışan(100);

Çalışan emp3 = yeni Çalışan(110);

System.out.println(emp1.equals(emp2));

Tamsayı i1 = yeni Tamsayı(100); Tamsayı i2 = yeni Tamsayı(100);

System.out.println(i1.hashCode());

System.out.println(i2.hashCode());

Set<Çalışan> set = new HashSet<>(); set.add(emp1);

set.add(emp2);

set.add(emp3);

System.out.println(set.size());

sınıf Çalışanı {

özel int kimliği;

public Çalışan(int id) { this.id=id;

}

public int getId() {

dönüş kimliği;

public void setId(int id) {

this.id = kimlik;

} }

@Geçersiz kıl

public int hashCode() { dönüş kimliği;

}

@Geçersiz kıl

public boole eşittir(diğer nesne) {

this.id'yi döndür == ((Çalışan)diğer).id;

}

-------------------------------------------------- ---------------

\* Oluşturduğunuz bir nesneyi nasıl sıralarsınız?

- Karşılaştırılabilir arabirimi uyguluyorum ve karşılaştırma yöntemini geçersiz kılıyorum.

Sonra nesnelerimi bir Listede sakladığımda

Koleksiyonlar.Sırala

sıralamak mümkün olacak. Ayrıca nesnelerimi bir

TreeSet veya TreeMap.

public static void main (String[] args){

Çalışan emp1 = yeni Çalışan(500);

Çalışan emp2 = yeni Çalışan(100);

Çalışan emp3 = yeni Çalışan(110);

Liste listesi = new ArrayList<>();

liste.add(emp1);

list.add(emp2);

list.add(emp3);

Collections.sort(liste);

for (Nesne nesnesi: liste) {

System.out.println(nesne);

}

}

}

class Çalışan, Karşılaştırılabilir<Çalışan>{ uygular

özel int kimliği;

public Çalışan(int id) {

this.id=id;

}

public int getId() {

dönüş kimliği; }

public void setId(int id) {

this.id = kimlik;

}

@Geçersiz kıl

public int hashCode() {

dönüş kimliği;

}

@Geçersiz kıl

public boole eşittir(diğer nesne) {

this.id'yi döndür == ((Çalışan)diğer).id;

}

@Geçersiz kıl

public int CompareTo(Çalışan emper) {

if(this.id > emp.id) {

dönüş 1;

}else if(this.id < emp.id) {

dönüş -1;

}

0 döndür; }

@Geçersiz kıl

public String toString() {

dönüş kimliği+"";

}

}

-------------------------------------------------- ---------------

\* ArrayList vs LinkedList

- Arraylist, dizi gibi koleksiyon sıralanır.

- bağlantılıListe düğümlerden oluşur:

1: Tek Bağlantılı Bağlantılı Liste

değer + sonraki düğüme işaretçi

2: doubleLinked LinkedList:

Değer + önceki düğüme işaretçi ve sonraki düğüme işaretçi

-------------------------------------------------- ---------------

\* ArrayList vs Vektör?

- Her ikisi de Liste Arayüzünü uygular ve ekleme sırasını korur

- Dizi Listesi

- senkronize değil

- hızlı çünkü senkronize değil

- Vektör

- senkronize edildi

- senkronize olduğu için yavaş

-------------------------------------------------- ---------------

\* HashSet ve HashMap karşılaştırması

- HashSet benzersiz öğeleri depolar

- HashSet, Set'i uygular

- HashSet, Tek Nesneleri saklar

- HashMap benzersiz anahtarları ve yinelenen değerleri depolar

- HashMap, Haritayı uygular

- HashMap, Anahtar, Değer eşleştirilmiş nesneleri depolar

-------------------------------------------------- ---------------

\* Hashtable ve HashMap karşılaştırması

- Hem anahtar + değer

- her ikisi de MAP uygular

- HashMap: Senkronize edilmedi, yalnızca 1 boş anahtar ve birden çok boş

değerler

- Hashtable: senkronize, boş anahtar veya değer yok

-------------------------------------------------- ---------------

\* Yığın vs Queve

- queve: ilk giren ilk çıkar

- Yığın: son giren ilk çıkar

-------------------------------------------------- ------------

\* treeset vs TreeMap

TreeSet ve TreeMap arasındaki en büyük fark, TreeSet'in

TreeMap Harita arayüzünü uygularken Set arayüzünü uygular

Java'da. ... TreeMap kullanırken TreeSet yalnızca bir nesne depolar

anahtar ve Değer adlı iki nesne. TreeSet'teki nesneler sıralanır

TreeMap'teki anahtarlar sıralı Sırada kalırken.

- TreeSet: Yalnızca benzersiz değerler içerebilir

- artan düzende sıralanır

- TreeMap: yalnızca benzersiz anahtarlar içerebilir.

- tuşlar artan düzende sıralanır

-------------------------------------------------- ---------------

\* Java'da istisna işleme?

- Duruma göre try catch nihayet bloklarını kullanabiliriz.

TRY : Bazı istisnalar oluşturabilecek kod

CATCH: Yakalanacak istisna tipini tanımlarız

Ve eğer TRY blok kodunda istisna olursa ne yapmalı?

NİHAİ: Varsa, her zaman çalışan kod

İstisna veya hayır.

SONUNDA blok: JVM çökerse veya System.exit (0) ÇALIŞMAZ

mevcut

deneyin veya blokları yakalayın. Bazılarını kapatmak için kodu temizle

bağlantılar, yakın

dosyaları aç vb.

örnek:

genel statik boşluk testCatch() {

denemek {

int ben = 4/0;

}yakala(İstisna e) {

dönüş;

}en sonunda {

System.out.println(“SONUNDA!”);//çalışır

}

}

-------------------------------------------------- ------------

\* Tüm istisnaların ebeveyni nedir?

- Atılabilir sınıf ebeveyndir

-------------------------------------------------- ---------------

\* Ne tür İstisnalar biliyorsunuz?

- KONTROL EDİLDİ -> Siz işleyin veya beyan edin, aksi takdirde derlemezsiniz.

- İŞARETSİZ -> RUNTIMEEXCEPTION'ın Alt Sınıfları. ve isteğe bağlı

kolu veya hayır.

-------------------------------------------------- ------------

\* fırlatma vs fırlatma nedir

- ATIŞLARA KARŞI ATIŞLAR:

ATIŞ : -> BİR İSTİSNA OBJESİ OLUŞTURUR

yeni RuntimeException() atın;

THROWS -> yöntem imzasına gider ve bir yöntemin

atabilir Ve istisna

-------------------------------------------------- ---------------

\* Projenizde karşılaştığınız istisna türleri

-

JAVA : NULLPOINTER İSTİSNA, YASAL UYARI, CLASSNOTFOUND,

FILENOTFOUND, CLASSCAST, ARİTMETİK, İNDEKSOUTOFBOUNDS,..

SQLEXCEPTION, IOEXCEPTION

SELENYUM: NOSUCHELEMENT, STALEELEMENT, TIKANABİLİR, GÖRÜNMEZ,

TIMEOUTEXEPTION, WEBDRIVE, ILLEGALSTATE EXP, GİBİ PENCERE YOK,

UYARI BİLGİSİ...

SALATALIK: BEKLEMEDE İSTİSNA, İKİLİ YÖNTEM UYGULAMA EXP,

-------------------------------------------------- ------------

================================================= ====================

====

============================ SELENIUM

==================================

================================================= ====================

====

10 Mayıs 2018

Önyükleme KAMPI

==== SELENYUM İNCELEME ====

SELENYUM IZGARA: düğüm

KODUNUZ VAR --> UZAKTAN SÜRÜCÜ -> SELENIUM HUB -> düğüm

düğüm

ÖRNEK: WebDriver sürücüsü = yeni RemoteWebDriver( , )

\* SauceLabs

- bulutta test sağlarlar

\* Selenyum Izgara

- Otomatik testlerinizi farklı ortamlarda çalıştırma olanağı sağlar.

tarayıcılar

ve sürümler ve işletim sistemleri.

- Selenium Suite'in bir parçası olan açık kaynak aracı

\* Test otomasyonu nedir ve faydaları nelerdir?

- insan etkileşimi yok

- hızlı

- güvenilir

- ölçeklenebilir

- veri odaklı test

\* ne otomatikleştirilebilir?

- Kararlı bir şey

- Duman / akıl sağlığı testi

- fonksiyonel test

- Test verileri oluşturma

- API

- Veri tabanı

\* Hangi Test durumu otomatikleştirilmelidir?

- kararlı test durumu olmalı

- tekrarlayıcı olmalı, birden fazla kullanılabilir

- herhangi bir insan etkileşimine ihtiyaç duymamalı

- bir değere sahip olmalıdır, Yüksek YG

\* Yaygın test otomasyon araçları nelerdir?

- Watir, selenyum, UFT

\* SELENYUM nedir?

- selenyum tarayıcıyı otomatikleştiren bir araçtır, kullanılır

test için

\* Selenyumun Avantajları?

- Açık kaynak

- web uygulaması ile etkileşim

- birden çok tarayıcıyı, işletim sistemini destekler

- farklı dilleri destekler

- tekrar Kullanılabilirlik

- toplum

\* Selenyumun Dezavantajı?

- yalnızca web uygulamalarını destekler

- resmi destek yok / yardım masası yok

- grafikler, capcha'lar, barkodlar, şekillerle çalışamaz

- yerleşik raporlama aracı yok

- ustalaşması zor, geliştirme düzeyinde bilgi gerektirir

- iyi konumlandırıcılar yazmak zor

- senkronize etmek zor

\* selenyumda ne yapmıyoruz?

- performans, yük, stres testi, manuel, ad hoc testi

kullanılabilirlik saf veritabanı testi (yalnızca db it'i kendi kendine test edersek)

, birim testleri .. biz yapmıyoruz. bak ve hisset tabanlı test

renk şekilleri statik test.

\* selenyumda yer belirleyici nedir?

- selenyum konumlandırıcı, html'de bir öğe bulmanın bir yoludur

- kimlik

- isim

- css

- Xpath

- etiket adı

- sınıf adı

- bağlantı metni

- kısmiLinkText

\*xpath nedir?

- selenyumdaki konum belirleyicilerden biridir

\* Mutlak ve Göreli Xpath arasındaki fark nedir?

- Mutlak:

- / ile başlar, kökten aramaya başlar

html belgesinin öğesi

- Akraba:

- // ile başlar, herhangi bir bölümünde eleman arar

sayfa

\* Xpath'i hangi durumlarda kullanacaksınız?

- kimlik, isim gibi benzersiz bilgileri kullanamadığımızda,

etiketler, sınıf ..

- dinamik öğelerle uğraşırken: kimlik her seferinde değişir

içerir, ile başlar, ile biter kullanabiliriz

- indekslerle uğraştığımızda

- metne göre arama yaptığımızda: //span[.= 'gönder']

- xpath tabloları ebeveyn çocuk kardeşini idare edebildiğinde

\* konum belirleyicileri nasıl seçersiniz? (en iyi konum belirleyici nedir)

- web sitesindeki, uygulamadaki duruma bağlıdır

- Sitede kimlik varsa evet, kimlik yoksa Ad,

- eğer bir bağlantı ise:

- bağlantı metni

- kısmi bağlantı metni

- sonra diğerleri

- eğer link değilse:

- Kimlik

- isim

- etiket adı

- sınıf

- css : okuması ve yazması kolay, hızlı

- Xpath

\* css Xpath'a karşı mı?

- css:

- hızlı okuması ve yazması kolay, her ikisi de kullanılıyorsa kullanılmalıdır

ccc ve xpath seçeneklerdir.

- dizini desteklemiyor

- metni desteklemiyor

- Yol:

- metinle çalışır

- dizini destekler

- xpath daha fazla inandırıcılığa sahiptir, bu yüzden güçlüdür ancak

çirkin.

\* örtük bekleme/açık bekleme nedir?

- Örtük bekleme:

- Sözdizimi: driver.manage.timeouts.implicitlyWait(5)

- bir kez ayarla, öğe bulamazsa her zaman çalışır

5 saniye bekler yoksa şikayet eder

findelement/s yöntemi her çalıştığında geçerlidir

- Örtük bekleme, web sürücüsüne şunu söyleyecektir:

atmadan önce belirli bir süre bekleyin

"Böyle Bir Öğe İstisnası Yok".

- Açık bekleme:

- bazen belirli bir olayı beklememiz gerekir

eleman görünür, tıklanabilir, etkin gibi..

- WebSürücüsüBekleme,

- Beklenen Koşullar

zaman aşımı istisnası

- AkıcıBekleme:

- ayrıca bir tür açık beklemedir. akıcı bekleme kullanma

webDriverWait'i özelleştirebiliriz

-

\* Assert ve Verify arasındaki fark nedir?

- İddia:

- başarısız olursa test yürütme durur

- Doğrulayın:

- başarısız olup olmadığını kontrol edin ve not alın ve devam edin

test

\* QUIT ve CLOSE arasındaki fark nedir?

- Çıkış yapmak:

- pencereleri kapatır

- Kapat:

- geçerli sekmeyi/pencereyi kapatır

\* selenyum düşüşleri nasıl ele alır?

- Select sınıfı, selenyumdaki açılır liste ile başa çıkmak için kullanılır.

bir seçme nesnesi oluşturmak için bir webElement'i şu şekilde iletmemiz gerekir:

yapıcı. bu öğenin select etiketine sahip olması gerekir

- Dizine göre seç: Bir int parametresi alır, dizine göre seçer

0 tabanlı.

- görünür metne göre seç: bir dize alır,

Metin görüntülendi.

- değere göre seç: bir dize alır parametre seçimini temel alır

seçeneğin değer özelliği

- örnek: WebElement öğesi =

driver.findelement(by.id"açılır menü")

Seçim listesi = yeni Seçim (eleman)

list.getFirstSelectedOption;

\* Açılır menüden seçilen çoklu değer nasıl kontrol edilir?

- arabaları seçList = new Select(el)

- carList.getSelectedOption(): seçilen seçeneği döndürür

bir liste(Liste<WebElement>)

- carList.getFirstSelectedOption <- döndüren yöntem

seçilen seçenek(webElement)

- asserEquals("Tahta Kaşık",

carList.getFirstSelectedOption().getText())

\* Açılır menüdeki tüm seçenekleri nasıl doğrularsınız?

- StateList'i seçin = new Select(el)

- stateList.getOptions(): mevcut tüm seçenekleri döndürür

seçili ya da değil

\* Select etiketi olmadan açılır menü ile nasıl çalışılır?

- açılır listede seçme etiketi yoksa, kullanamayız

sınıf seç

- açılır listeyi ve seçeneklerini ayrı olarak işleyin

elemanlar her elemanı ayrı ayrı konumlandırır

- bir seçenek belirleyin: 1 bulun ve tıklayın

\* Çerçeve değiştirmek için sözdizimi nedir?

- html sayfası html sayfasının içinde / onlara şu şekilde davranmalıyız

farklı sayfalar.

- Iframe/frame : çerçeveler için etiketler. Çerçeve bir html belgesidir

başka bir html belgesinin içinde.

- Web sürücüsü, başka bir çerçeveyi kontrol etmek için her seferinde bir sayfayı işler

her zaman değiştirmemiz gerekiyor

- driver.switchTo.frame() : webElement'i alır -> iframe'i bulur

ve bir param olarak iletin.

- driver.switchTo.frame() : String'i alır -> kimliği bulun veya

iframe'in adı ve param olarak geçiş

- driver.switchTo.frame() : int alır -> dizini bulur ve geçer

param olarak

\* pencereleri değiştirmek için sözdizimi nedir?

- ayrı sekmeleri/pencereleri işlemek için değiştirmemiz gerekiyor

o sekmeye

- web sürücüsü, bir seferde bir sayfa html belgesini işler. ile

her zaman değiştirmemiz gereken başka bir sekmeyi kontrol et

- geçiş yapabilmek için önce pencere tutamağını almamız gerekiyor

getWindowHandesl() yöntemini kullanarak

- driver.switchTo.window() : bir dize alır -> pencere tanıtıcısı

- her biri için: driverGetWindowHandles:

driver.switchTo.window("başlık")

if driver.getTitle == beklenen başlık

kırmak;

- selenyum, sekmeler arasında geçiş yapmak için pencere koluna ihtiyaç duymaz

başlığı kullanın. başlığı kullanarak geçiş yapmak için hala kullanarak geçiş yapıyoruz

önce pencere tutamacı, ardından başlığı kontrol edin.

\* Eylemler sınıfını kullanarak hangi işlemleri yapabilirsiniz?

- Actions sınıfı, gelişmiş fare ve klavye yapmamızı sağlar

operasyonlar.

- Öğenin üzerine gelin / öğeye gidin

- yukarı/aşağı kaydırma

- çift tıklama

- sağ tık

- sürükle ve bırak

- klavye kombinasyonları

\* çift tıklama eylemlerinin sözdizimi nedir?

- eylemler sınıfını kullanarak bir web öğesine karşı herhangi bir eylem gerçekleştirin

önce öğeyi bulmamız gerekiyor

WebElement el = driver.findElement

Eylemler eylemler = yeni Eylemler(sürücü)

eylemler.doubleClick(el);

eylemler.moveTO(el)

eylemler.doubleClick

\* bir dosya yüklemek için sözdizimi nedir?

- selenyum kullanarak dosya yüklemek için yerini bulmamız gerekiyor

html'deki yükleme düğmesi. elemanın etiketi olacak

girdikten sonra yolu dosyaya ileterek anahtarları göndeririz

Hangi

WebElement yüklemesi = driver.findElement

upload.sendKeys("C:/user/me/file.txt")

\* bazen sendKeys yolu çalışmıyor

- robotu veya autoid'i kullanmalısın

- düzeltmek için içindeki dosya için dinamik bir yol oluşturmanız gerekir

projemiz

- projedeki dosyanın yolu:

String projectDir= System.getProperty("usr.dir")

projedeki dosyanın yolu:

"src/test/resources/test\_data/myfile.txt"

- Element.sendKyes(projeDir+dosyası)

\* herhangi bir çapraz tarayıcı testi yaptınız mı?

- evet

- her zaman anahtar kelimeler için bir kontrol dosyanız olduğunu belirtin

browserType ana url şifre kullanıcı adı ortamı gibi.

\* herhangi bir başsız tarayıcı testi yaptınız mı?

- açılmayan bir tarayıcı, arka plan hizmeti olarak çalışır

veya program. Onlar hızlı

- html unitdriver: selenyumdan gelen başsız tarayıcı

web sürücüsü sürücüsü = yeni hemlUnitDRiver();

- phantomjsDriver: üçüncü taraf sürücüsü

web sürücüsü sürücüsü = yeni htmlUnitDriver();

- ekran görüntüsü almak başsız tarayıcı ile çalışmaya devam edecek

\* sertifika sorununu nasıl çözersiniz?

- CHROME kullanıyorsanız, Internet Explorer

DesiredCapabilities yeteneği = DesiredCapabilities.chrom();

yetenek.setcapability(capability.ACCEPT\_SSL\_CERTS, true );

System.setProperty("webdriver.chrome.driver","E:/

chromedriver.exe")

sürücü = yeni ChromeDriver(yetenek);

- daha fazla örnek badssl.com

\* Stress Security SQL enjeksiyon testini nasıl yapacaksınız?

- selenyum mühendisi bu testlerin hiçbirini yapmaz. bu testler

tarafından yapılır

Bu araçlar konusunda eğitimli uzmanlar.

\* Js uyarılarını nasıl ele alıyorsunuz?

- tarayıcıdaki uyarı javaScript'ten geliyorsa,

Uyarı Sınıfı.

- Uyarı uyarısı = driver.switchTo.alert();

- alarm.kabul();

- alarm.dismiss();

- alarm.sendkeys();

\* tarayıcı açılır penceresini nasıl idare ediyorsunuz?

- İptal bağlantısını bulmak için kapatmak için başka bir webElement gibi davranın

x simgesine ve tıklayın();

\* Windows/OS açılır pencerelerini nasıl idare ediyorsunuz?

- robotu veya AutoIT'i kullanma

\* Öğenin görüntülenmediğini doğrulayın?

- eleman = driver.findElement()

- element.isDisplayed();

- driver.findElement().isDisplayed() -> bu satır

eleman eğer istisna

html'de değil

denemek{

bulmak

görüntüleniyorsa geri dön

}

Yakala(öğe değil fon)

{gerçeğe dön}

\* Neden eleman bulamıyorum?

- konumlandırıcı değişti

- bir iframe var

- bekleme süresi: sayfa yavaş yükleniyor

- eleman dinamiktir: konumlandırıcı onu bulamıyor

- sayfa tam olarak yüklenmemiş/açılmamış

- sayfa değişir ve o öğe artık yok

================================================= ====================

=========

12 Mayıs 2018

selenyum eğitim kampı

-------------------------------------------------- --------------------

----

Metnin var olduğu doğrulansın mı?

\* xpath kullanarak. metin için xpath yaz

Belirli/tam metin içeren herhangi bir öğeyi bulmak için Xpath:

//\*[.='metin']

Belirli bir metni içeren herhangi bir öğeyi bulmak için Xpath:

//\*[içerir(metin(),'not')]

getPageSource() -> ..

-------------------------------------------------- --------------------

----

StaleElementExceptions'ı nasıl ele alırsınız?

- istediğimiz zaman eski eleman istisnası alabiliriz

etkileşimde olmak

daha önce bulduğumuz web öğesi yeniden yüklendi

DOM.

javascript veya AJAX işleminin bitmesini beklememiz gerekiyor

yerini bul

tekrar eleman

örnek:

sayaç = 0;

while (sayaç < 5 ){

denemek {

bul ve tıkla

kırmak;

}yakalama(eskileme istisnası)

1 saniye bekle

sayaç ++

}

================================================= ====================

====

============================== SQL VE REST

=============================

================================================= ====================

====

11 Mayıs 2018

SQL ve REST

-------------------------------------------------- --------------------

------------------

Çerçeveniz hakkında konuşurken bu çerçevelerden bahsedin:

Sayfa Nesne modeli çerçevesi

salatalık Bdd çerçevesi

TestNG tabanlı

Veriye Dayalı çerçeve

(QTP / UFT stilinden gelir) <- çok fazla bahsetme çünkü

çoğunlukla UFT

Hibrit Çerçeve

Anahtar Kelimeye Dayalı Çerçeve

\*\* Bana kendinden bahset SQL \*\*

Java kullanarak ön uç test otomasyonu konusunda rahatım ve

selenyum

WebDriver, Salatalık, Maven, Testng, Junit de iyiyim

dinlendirici API

JAVA'da postacı ve güvenilir kitaplığı kullanarak test otomasyonu.

zenginim var

sql sorguları kullanarak veritabanı test otomasyonu deneyimi

JDBC kütüphanesi ile birlikte

JAVA'da.

-------------------------------------------------- --------------------

------------------

\* veri odaklı Test?

- veri odaklı çerçeve ne zaman yapıyoruz?

- Ne zaman: Bir uygulamadaki bir işlev veya modül gerektiğinde

ile test etmek

birden fazla veri seti (parametrizasyon olarak da adlandırılır)

birden fazla girdi daha sonra veri odaklı gerçekleştirmemiz gerekiyor

test ve otomasyon.

bu senaryolar olması gereken şeylerden biri

otomatik.

- Nasıl: Test verileri koddan ayrılır ve harici olarak depolanır.

kaynaklar. Excel dosyaları,

CSV dosyaları, Veritabanı.

FAYDA: testte daha organize veri merkezi işbirliği

gelebileceği veriler

BA, MT'ler, VB

-------------------------------------------------- --------------------

------------------

\* excelden veri nasıl okunur?

- Excel'den veya herhangi bir excel ile ilgili veri okumak için Apache POI kullanıyorum

testler

ÖRNEK:

Dize dosyası = "TestData.xlsx";

FileImputStream inStream = new FileInputStream(dosya);

String filePath = "C:\\user\\alex\\Desktop";

FileInputStream akış içi = new FileInputStream(filePath);

Çalışma kitabı çalışmaKitabı = WorkbookFactory.create(inStream);

Sayfa çalışma sayfası = workBook.getsheet(0);

Satır satırı = workSheet.getRow(0);

Hücre hücre = row.getCell(0);

System.out.println(cell.toString());

VERİ YAZIN:

MERKEZİ MAVEN REPO -> KULLANIYORUZ

ŞİRKET MAVEN REPO -> İŞ YERİNDE KULLANIYORSUNUZ

-------------------------------------------------- --------------------

------------------

\* Veritabanından Veri Okumak?

- Manuel olarak: Oracle SQL geliştiricisi <- manuel olarak test etmek için

- Otomasyon: JDBC -> JAVA DATABASE BAĞLANTISI <- kullanılır

otomasyon için

Nesnelerin Hiyerarşisi:

BAĞLANTI -> AÇIKLAMA -> SONUÇ SETİ

BAĞLANTI NASIL SAĞLANIR:

Bağlantı bağlantısı =

DriverManger.getConnection("URL","kullanıcı adı","şifre")

Bildirim durumu =

bağlantı.createStatement(ResultSet.TYPE\_...)

Sonuç Kümesi özgeçmişi =

deyim.executeQuery("çalışanlardan \* seçin");

- test verileri koddan ayrılmıştır ..

- eğer veri miktarı o kadar büyük değilse o zaman salatalık kullanırım

örnekle senaryo taslağı

masa

- ve diğer zamanlarda test verilerini excel dosyalarında tutuyorum ve

apache POI kitaplığını kullan

JAVA kullanarak veri okumak ve yazmak için

- eğer veri bir veritabanından geliyorsa veya datase yapmam gerekiyorsa

doğrulama

Java'da JDBC kitaplığı ile birlikte sql sorguları kullanıyorum

-------------------------------------------------- --------------------

------------------

\* try catch bloğunda veritabanındaki bağlantıyı nereden kapatıyoruz

durumda atılan muafiyet alırız. "yakalama bloğunda"

1) nihayet blokta

2) kaynaklarla denemeyi kullanıyorum ve otomatik olarak kapanacak

Örnek 1: fanally blokta

denemek{

Bağlantı bağlantısı =

DriverManger.getConnection("URL","kullanıcı adı","şifre")

Bildirim durumu =

bağlantı.createStatement(ResultSet.TYPE\_...)

Sonuç Kümesi özgeçmişi =

deyim.executeQuery("ülkelerden \* seçiniz");

}yakala(SQLException e){

// rapor edilecek bazı kodlar

e.printStackTrace();

}en sonunda{

denemek{

sonuç.kapat();

deyim.close();

bağlantı.kapat();

}yakala(e istisnası){

// bir kod

}

}

Örnek 2: kaynaklarla deneyin

kapatılabilir veya otomatik kapatılabilir kapsamını genişleten herhangi bir sınıf veya arabirim

arayüz

denemek(

Bağlantı bağlantısı =

DriverManger.getConnection("URL","kullanıcı adı","şifre")

Bildirim durumu =

bağlantı.createStatement(ResultSet.TYPE\_...)

Sonuç Kümesi özgeçmişi =

deyim.executeQuery("ülkelerden \* seçin");)

{

// sonuç kümesinden veri okumak için bazı kodlar

}

}yakala(SQLException e){

// rapor edilecek bazı kodlar

e.printStackTrace();

}

-------------------------------------------------- --------------------

------------------

\* herhangi bir GERİ UÇ TESTİ veya VERİTABANI TESTİ yaptınız mı?(bunlar

API ve Veritabanı)

- Evet, veritabanlarıyla çalışma konusunda çok deneyimim var

ve

SQL sorguları yazma konusunda rahatım. deneyimim var

ilişkisel çalışma ile

ORACLE, MySQL, SQL Server gibi veritabanları.

- İlişkisel olmayan veritabanlarıyla çalıştınız mı?

- Tecrübem yok, öyle olduğunu biliyorum.

JSON gibi

veritabanını biçimlendirin ve iyi bir deneyimim var

JSON ile çalışmak

Dosyalar. Ve hızlı öğrenen ve çapraz fonksiyonel biriyim.

-------------------------------------------------- --------------------

------------------

\* SQL'de nasıl tablo oluşturursunuz, kısıtlamalar nelerdir?

- masa bilgisayarı oluştur(

computer\_id numarası birincil anahtarı,

marka varchar2(20) boş değil,

char(1) yazın boş değil

);

- SQL sütun kısıtlamaları:

Birincil anahtar, yabancı anahtar, boş, boş değil benzersiz.

- Bilgisayarları tanımlayın -> tablo bilgisi meta verilerini görüntülemek için

masanın

Bilgisayarları tanımlayın;

- SQL veritabanına veri ekleme

bilgisayar değerlerine ekleyin( 123 ,'HP','L');

bilgisayarlardan \* seçin

-------------------------------------------------- --------------------

------------------

\* SQL teknik görüşme

1. Kaç kayıt olduğunu bulmak için bir SQL sorgusu yazın

Kasaba tablosunda nerede

"TOWN\_NAME" içinde "ell" mi var?

- sayıyı seç(\*)

kasabadan

burada kasaba\_adı '%ell%' gibi;

2. Kaç tane geçtiğini bulmak için bir SQL sorgusu yazın.

her "COUNTRY\_CODE"

Kasaba masasında orada mı?

- ülke kodunu seçin, sayın(\*)

kasabadan

Ülke koduna göre gruplandır;

3. “TOWN\_NAME” i bulmak için bir SQL sorgusu yazın.

alfasayısal değildir

karakterler? REGEX, 10 gibi bazı formatları eşleştirmek için kullanılır.

sayı vb

seçme(\*)

kasabadan

kasaba\_adı '%[^a-zA-Z0-9]%' gibi değil;

4. İlk 5 kaydı seçmek için bir SQL sorgusu yazın.

Şehir masası?

seçme \*

kasabadan

burada satır sayısı <= 5;

-------------------------------------------------- --------------------

------------------

\* Dahili JOIN röportajı

- ihtiyacımız olan birden fazla tablodan veri sorguladığımızda

katılır ve ihtiyacımız olduğunda

eşleşen kayıtları bulmak için iç birleştirmeleri kullanırız.

- veriyi görüntülememiz gerektiğinde iç birleştirme kullanılır

çoklu tablolar.

ve eşleşen kayıtları döndürür.

1.Çalışanların maaşını bulmak için bir SQL sorgusu yazın

KG departmanında

ve İnsan Kaynakları Departmanı?

empid, isim, maaş, deptid seçin

çalışandan e departmanlara katıl d

e.empid üzerinde = d.empid

deptid in('QA','HR');

-------------------------------------------------- --------------------

------------------

\* selfJoin röportajı ( )

1.

emp.employee\_id, emp.first\_name, man.first\_name, seçin

man.last\_name

çalışanlardan emp katıl çalışanlar adam

emp.manager\_id üzerinde = man.employee\_id;

-------------------------------------------------- --------------------

------------------

\* iç birleşim vs dış birleşim

- birden fazla veriden sorgu verisine ihtiyacımız olduğunda birleştirmelere ihtiyacımız var

tablolar

1.Inner birleştirme, her iki tablodan eşleşen kayıtları döndürür

katılma durumuna göre

2.outter birleştirme, eşleşen verileri ve eşleşmeyen verileri döndürür

ya sağdan

veya sol tablolar.

Örnek: iç birleştirme

soyadını seç, departman\_adı

çalışanlardan departmanlara katılmak

çalışanlarda.department\_id =

departmanlar.department\_id;

-------------------------------------------------- --------------------

------------------

\* Union ve Union ALL arasındaki fark

- verileri görüntülememiz gerektiğinde bu küme operatörlerini kullanırız.

çoklu veya 2 sorgu

- UNION: kopyaları ve boş değerleri kaldırır ve

sonuç,

- UNION ALL: kopyaları kaldırmaz ve sıralama yapmaz

-------------------------------------------------- --------------------

------------------

\* Intersect ve Eksi arasındaki fark

- INTERSECT: her iki sorguda da ortak verileri görüntüler

- EKSİ: ilk sorgudan mevcut olmayan kayıtları döndürür

ikinci sorguda.

-------------------------------------------------- --------------------

------------------

\* JOIN VS UNION mülakat sorusu

- birleştirmeler, birden çok tablodan sorguyu görüntülemek için kullanılır

- Birleşim, birden çok sorgudan veri sorgulamak için kullanılır.

-------------------------------------------------- --------------------

------------------

\* GROUP BY ile ORDER BY arasındaki fark

- group by, grup işlevleriyle çalıştığımızda kullanılır

yardımcı olacak

bir grup içinde alt gruplar oluşturun

- order by azalan ve artan olarak sıralamak için kullanılır

-------------------------------------------------- --------------------

------------------

\* HAVING ile NEREDE arasındaki fark

Cevap: HAVING, bir grup veya bir grup için bir koşul belirtmek için kullanılır.

toplama işlevi

select deyiminde kullanılır. WHERE yan tümcesi önce seçer

gruplama. HAVING

yan tümcesi, gruplamadan sonra satırları seçer. HAVING yan tümcesinden farklı olarak,

WHERE yan tümcesi olamaz

toplu işlevleri içerir.

- durumumuz bir grup işlevi içerdiğinde

örnek: maks(maaş) > 9000;

- sonuçları filtrelemek için nerede kullanılır ve ne zaman kullanılmaz

grup işlevi

örnek: burada çalışan\_kimliği = 234;

-------------------------------------------------- --------------------

------------------

\* Tablodan 5. satırı göster ( ÇEKİRDEK İLGİLİ ALT SORGU)

- ilişkili alt sorgu kullanmamız gerekiyor

- iç sorgu, satır numarası ve dış ile birlikte sütunları alacak

sorgu görünecek

belirli satır numarası için

örnek 1:

seçme \*

from (person\_id,first\_name, rownum'u rn olarak seçin

çalışanlar)

burada n = 5;

örnek 2:

( rownum <=5 olan çalışanlardan \* seçin)

eksi

( rownum <=4 olan çalışanlardan \* seçin);

örnek 3: 5. en büyük maaş soruları

çalışanlardan maaş seç e1

burada 5= (sayıyı seçin(maaş)

çalışanlardan e2 burada e1.maaş<=e2.maaş);

-------------------------------------------------- --------------------

------------------

\* düşürme ve kesme arasındaki fark

- ikisi de ddl komutlarıdır ve geri alınamaz,

- drop, verileri ve tabloyu birlikte kaldıracak

- truncate tüm verileri kaldırır ancak tablo yapısını kaldırmaz.

- ÇALIŞANLARI BIRAK;

- ÇALIŞANLARI KESMEK;

-------------------------------------------------- --------------------

------------------

\* Komutları yerine getir

- veritabanında herhangi bir değişiklik yaptığımızda, kaydetmeyi taahhüt edebiliriz

değişiklikler

-------------------------------------------------- --------------------

------------------

\* SQL biliyor musunuz?

- evet tecrübem var ve SQL yazma konusunda rahatım

DDL ve DML'yi sorgular

komutlar. şu anda oracle veritabanı ile çalışıyor

amazon bulut sunucusunda çalışıyor.

- DDL NEDİR? veri tanımlama dili: alt değiştirme oluştur

kesmek

- DML veri işleme dilleri: ekleme güncellemesini sil'i seçin

-------------------------------------------------- --------------------

------------------

\*SQL nedir?

- SQL, eklemek ve eklemek için tasarlanmış Yapılandırılmış Sorgu Dilidir.

ilişkisel bir veritabanı sisteminde değiştirme.

-------------------------------------------------- --------------------

------------------

\*Birincil anahtar ile benzersiz kısıtlamalar arasındaki fark nedir?

- Cevap: Birincil anahtarın NULL değeri olamaz, benzersiz

kısıtlamalar NULL değerlere sahip olabilir.

Bir tabloda yalnızca bir birincil anahtar vardır, ancak şunlar olabilir:

çoklu benzersiz kısıtlamalar.

-------------------------------------------------- --------------------

------------------

-------------------------------------------------- --------------------

------------------

-------------------------------------------------- --------------------

------------------

-------------------------------------------------- --------------------

------------------

-------------------------------------------------- --------------------

------------------

-------------------------------------------------- --------------------

------------------

================================================= ====================

====

============================ API ==================== ==============

================================================= ====================

====

API ÖĞRENME KAMPI

-------------------------------------------------- --------------------

------------------

\* ne tür veri tabanı testleri yapıyorsunuz?

- Çoğunlukla veritabanı doğrulamaları yapıyorum

- Ön uçta değişiklik yapıyorum veya veri ekliyorum ve doğrulıyorum

veritabanında

Ön uçtaki veriler veritabanıyla eşleşir

- ayrıca RESTapi kullanarak değişiklikler yapıyorum ve değişiklikleri doğrulıyorum

veritabanında başarılı

ilave olarak

- Veritabanı taşıma işlemini de destekliyorum, Kodum bağlanıyor

tüm veritabanına

JDBC kullanarak Sybase (eski veritabanı) daha sonra oracle'a bağlanır

(yeni DB)

ve ardından verilerin taşındığından emin olmak için kayıtları karşılaştırın

başarıyla

-------------------------------------------------- --------------------

------------------

\* Toplu uygulama işleri?

- her gece çalışan ve bazılarını yapan bazı otomatik kodlar

durum değişiklikleri

bazı verilere

Örnek: 7 gün geçerli kupon. Toplu iş taranıyor

her kupon aracılığıyla ve eğer

kupon isabeti 7 gün, veritabanındaki durumu şu şekilde güncelleyecektir

geçersiz.

-------------------------------------------------- --------------------

------------------

\* RestfulAPI Testi ile ilgili deneyiminiz var mı?

- mevcut projedeki uygulamanın şuraya entegre edilmesi gerekiyor

diğer dahili

ve ekibimizin geliştirdiği entegrasyon için harici uygulama

dinlendirici api bu yüzden test ediyorum.

- dinlenme API bitiş noktasının beklendiği gibi çalıştığını doğrularım

manuel api testi için postacı kullanıyorum ve güvenceli kitaplığı kullanıyorum

otomasyon için java'da

Gönderiyi gönder, istek türünü sil ve yanıtları doğrula

stasus kodu ve yanıt gövdesi başlığı

Ayrıca api'nin pozitif ve negatif testlerini yapıyorum.

pozitif test yaptığımda geçerli istek parametreleri gönderirim, geçerli

üstbilgiler geçerli istek json gövdesi

ve yanıt durumu kodunu i 200 doğrulayın

-------------------------------------------------- --------------------

------------------

\* http yöntemleri/istek?

- al: verileri oku

- gönderi: oluştur

- koymak: güncelleme

- sil: sil

- post vs koymak

-------------------------------------------------- --------------------

------------------

\* Durum kodu

2×× Başarı

200 TAMAM

201 Oluşturuldu

3×× Yönlendirme

300 Çoklu Seçenek

301 Kalıcı Olarak Taşındı

302 Bulundu

4×× İstemci Hatası

400 Hatalı İstek

401 Yetkisiz

402 Ödeme Gerekli

403 yasak

404 Bulunamadı

5×× Sunucu Hatası

500 Dahili Sunucu Hatası

501 Uygulanmadı

502 sunucu hatası

-------------------------------------------------- --------------------

------------------

\* Başlıklar

- kabul, içerik türü, uygulama JSON, Uygulama XML

-------------------------------------------------- --------------------

------------------

\* parametreler

- 2 tip

- yol parametreleri (değer, URL'nin bir parçası olacaktır)

- sorgu / istek parametreleri (anahtar+değer biçimi)

-------------------------------------------------- --------------------

------------------

\* JSON gövdesini doğrulayın/kontrol edin/onaylayın

- {

"çalışan\_kimliği": 100,

"first\_name": "steven"

}

yeniden güvence altına alınmış kitaplık ile birlikte gelen hamcrest eşleştiriciyi kullanıyoruz

- 1.When().get(uri).then().body("first\_name",equalto("steven"));

jsonpath kullanarak:

- 2.JsonPath json =

When().get(uri).thenreturn().body().jsonpath();

assert.asserEquals(json.getString("last\_name"),"king");

- 3.Koleksiyonları kullanma. HashMap

Yanıt yanıtı = ne zaman().get(uri);

Harita haritası = answer.body().as(Map.class);

assert.assertEquals(map.get(job\_id),"AD\_PRES");

- 4.POJO'ları kullanma. önce bir sınıf oluştur

kamu sınıfı çalışanı{}

Yanıt yanıtı = ne zaman().get(uri);

Çalışan haritası = answer.body().as(Employee.class);

assert.assertEquals(emp.getSalary(),24000);

-------------------------------------------------- --------------------

------------------

\* Seralizasyon vs DE-Serealization

- seralizasyon: Java nesnesini Json'a dönüştür

- Serileştirme: JSON'dan Java Nesnesine

-------------------------------------------------- --------------------

------------------

\* Gson Ayrıştırıcı

- Gson ayrıştırıcısı, json'u Java nesnesine dönüştürmek için kullanılır veya tam tersi

- json tojjson'dan 2 faydalı yöntem

-------------------------------------------------- --------------------

------------------

================================================= ====================

====

=========================== FrameWork

=================================

================================================= ====================

====

ÇERÇEVE

12 Mayıs 2018

-------------------------------------------------- --------------------

----

\* Testng vs Junit

Haziran Testi

Ek açıklama desteği yy

süit Testi yy

yy Testini yoksay

istisna Testi yy

zaman aşımı yy

parametreli Test yy

bağımlılık testi N y

-------------------------------------------------- --------------------

----

\* uygulamanızı geliştirmek için hangi araçlar kullanıldı?

- Java Spring -> web sitesinin arka uç kodu

- açısal -> uygulamanın ön ucu

- Tomcat sunucusu -> geliştiriciler web sitesini barındırıyor

- Oracle Veritabanı -> Veritabanı

- jenkins -> uygulamayı dağıtmak için

- Eclipse ve IntelliJ Fikir

- SonarCube -> kod kapsamı yaş aracı

- Docker -> barındırın, dağıtın

-------------------------------------------------- --------------------

----

\* çerçevenizi tanımlayın 1 TestNG + selenyum

- JAVA

- UZMAN

- TestNG

- SELENYUM

- RAPORLARI GENİŞLETİN: raporlar için testng ve selenyum ile çalışır

- LOG4J

-JDBC

- EMİN OLABİLİRSİNİZ

- APACHE POI

-------------------------------------------------- --------------------

----

\* çerçevenizi tanımlayın 2 ( SALATALIK BDD ÇERÇEVESİ)

- TEST / JUNIT (HİÇ BİRİ DEĞİL)

- SALATALIK

- JAVA

- UZMAN

- SELENYUM

- HTML RAPORLAMA

-JDBC

- EMİN OLABİLİRSİNİZ

- APACHE POI

- GIT

-JENKINS

-------------------------------------------------- --------------------

----

\* ÇERÇEVE NASIL OLUŞUR?

- salatalık BDD çerçevemiz HTML raporları oluşturur

- raporlar, özellik dosyaları için başarılı/başarısız kapsamını gösterir

Etiketler, adımlar.

- rapor, her testin adımlarını içerir

- raporun arızalar için ekran görüntüleri var

-------------------------------------------------- --------------------

----

\* Çerçevenizde nasıl ekran görüntüsü alıyorsunuz?

- Çerçevemde ekran görüntüsü alma mantığım var

kanca sınıfında. çünkü ekran görüntüsü almak istiyorum

test başarısız olsa bile. Kanca, yöntemin

ekran görüntüsü

test başarısız olsa bile çalışacaktır.

- TakeScreenshot arabirimi, verileri almak için kullanılan yöntemdir.

ekran görüntüsü. BT

selenyum arayüzünden gelir.

- bu rapor, jenkins testimde kullanılan raporun aynısıdır. ne zaman

jenkins'te salatalık raporlarını kullanarak raporu oluşturuyorum

Eklenti

- yerel olarak çalıştırdığımda aynı rapor tarafından oluşturulur

pom'da yapılandırma

dosya.

- bu yapılandırma seçeneklerinin her ikisi de Json dosyasını kullanır.

rapor oluştur

EKLENTİ = "json...."

-------------------------------------------------- --------------------

----

\* Log için ne kullanıyorsunuz?

- log4j loglamak için kullanıyorum. her zaman önemli adımları kaydederim

test yürütme

Bu, bir başarısızlık olduğunda çıkış yapmama yardımcı oluyor.

- log4j, html raporlarının yerine geçmez.

-------------------------------------------------- --------------------

----

\* özellik dosyası nasıl çalışır? bana bir bütünün sözdizimini söyle

özellik dosyası

- özellik -> neyin test edildiğinin açıklaması

- senaryo -> test edilen senaryonun açıklaması

- Verilen -> bir ön koşul

- Ne zaman -> beklenen sonucu tetikleyen koşul

- O zaman ->

-------------------------------------------------- --------------------

----

\* salatalık senaryo örneği

- Giriş sayfasında olduğum için

- Ve kullanıcı adını ve şifreyi giriyorum

- Gönder düğmesine tıkladığımda

- O zaman profil resmini görebilmeliyim

-------------------------------------------------- --------------------

----

\* Özellik dosyası örneği

- Özellik: oturum açma işlevi

- Arka plan: "ortak olan her şey arka planda gider"

Giriş sayfasında olduğum göz önüne alındığında

- Senaryo: 1

- Senaryo: 2

-------------------------------------------------- --------------------

----

\* Senaryo Anahattı

- Senaryo taslağı: arama

- Orada olduğum için

- Ve "<item>" i arıyorum

Örnek:

|öğe|

|tahta kaşık|

|karbon fiber kaşık|

\* senaryo ilk kez iki kez çalışacak, tahta arayacaktır

kaşık

bir dahaki sefere karbon fiber kaşık arayacak

-------------------------------------------------- --------------------

----

\* test temel sınıfı nedir? ve çerçevenizde nasıl uyguluyorsunuz?

- Test temel sınıfı hakkında konuştuğumuzda çerçeve 1 hakkında konuşuruz

sadece

- test temel sınıfı, hayatımda en çok yöntemleri kullandığım sınıftır.

Ölçek

test sınıflarım test temel sınıfını genişletiyor ve bu nedenle erişime sahip

onlara

bu, kodumu yeniden kullanılabilir hale getirmemize yardımcı olan yöntemler

- Test yöntemleri öncesi/sonrası

- bekle/senkronizasyon yardımcı programı yöntemleri

- switchToWindow(başlık)

- Web Sürücüsü sürücüsü;

-------------------------------------------------- --------------------

----

\* Başarısız testler tekrar nasıl çalıştırılır?

- hangi araçların kullanıldığına bağlı olarak farklılık gösterir mi?

BU KİMLİK POM'DA YAPILMIŞTIR \*

- TESTNG'de: çerçeve (salatalık değil), başarısız testler

içinde rapor edildi

hedef klasörde testng\_failed.xml dosyası. yapabiliriz

Ekle

pom dosyasındaki bu dosya, böylece maven deneyecek

çalıştırmak için

her seferinde başarısız testler. sadece ne zaman çalışacak

başarısızlıklar var

testte.

-------------------------------------------------- --------------------

----

\* Hıyarda başarısız olan testler tekrar nasıl yapılır?

- salatalıktaki başarısız testleri yeniden çalıştırmak için yeniden çalıştırma seçeneğini kullanıyoruz

CukesRunner'da

1: Tekrar çalıştırmayı cukes runner'a ekleyin. bu seçenek bir

ile dosya

başarısız testlerin listesi.

2: bir dosyaya işaret eden ikinci bir koşucu sınıfı oluşturun

listesi

başarısız testler

3: İkinci koşucuyu pom dosyasına ekleyin

-------------------------------------------------- --------------------

----

\* Jenkins'de başarısız testler tekrar nasıl çalıştırılır?

- jenkins'de başarısız testleri çalıştıran eklentiler var. Birim

vakalar (salatalık değil)

-

-------------------------------------------------- --------------------

----

\* Salatalık seçici olarak nasıl test edilir?

- CukesRunner anahtar sözcüğünü etiketler

- CukesRunner anahtar kelimesini içerir

- etiketler ve özellikler komut satırı kullanılarak da iletilebilir

- mvn temiz testi -Dcucumber.options="--tags ~@nonexistantTag"

-------------------------------------------------- --------------------

----

\* lütfen mevcut uygulamanızdan bir kullanıcı hikayesi seçin ve yürüyün

benim aracılığımla

otomasyon için?

- // 1 veya 2 ile gel

-------------------------------------------------- --------------------

----

\* Tropi hatanız nedir?

- // 1 veya 2 ile gel

-------------------------------------------------- --------------------

----

\* Selenyum'u JIRA ile nasıl entegre edersiniz?

- selenyum, JIRA ile yerleşik bir entegrasyona sahip değildir. ancak

eklentiler var

selenyum test çerçevesini jira ile entegre edin.

- XRAY(jira eklentisi, Jenkins eklentisi)

- Zephyr (jira eklentisi)

-------------------------------------------------- --------------------

----

\* Sprint'inizde otomasyonu ne zaman yapıyorsunuz?

- Geliştiricilerin kendi rolleri bittiğinde

- Kod QA/test ortamına dağıtıldığında

- tüm manuel testler yapıldığında

- evreleme/UAT ortamında test yapıyorsanız, olmayacaksınız

test yapmak

geliştirilmekte olan hikayeler.

- Test ekibindesiniz. içine aldığın tüm hikayeler

takım

test hikayeleri ve test edilmeye hazır.

- bu konuda test ediciler ve geliştiricilerden oluşan karma bir ekibin parçasısınız

dava

istediğiniz şekilde geliştirilmekte olan özellikleri test edeceksiniz.

geliştirici

kod değişikliklerini bitirmek ve QA ortamına geçmek için

-------------------------------------------------- --------------------

----

\* Jenkings'te herhangi bir iş kurdunuz mu?

-

-------------------------------------------------- --------------------

----

\* \* Test sonuçlarını/raporlarını nerede saklıyorsunuz?

- raporların konumu, nasıl çalıştığımıza bağlıdır

onlara.

- yerel olarak çalışıyorsak, raporlar

hedef klasör. çerçevem içinde salatalık raporları klasörü oluşturuyor

hedef

raporu içeren yem

- jenkins üzerinde testler yaptığımızda jenkins raporu kaydeder

her koşunun. önceki çalıştırmalar için tüm raporlar altında bulunabilir

yapı numarası.

-------------------------------------------------- --------------------

----

\* TEST VERİLERİNİ NASIL ALABİLİRSİNİZ?

- delphix -> test verilerini sağlayan araç/veya şirket

- projemde test verileri için 2 kaynak vardı

1. Geliştiriciler, üretim veritabanındaki verileri değiştirdi

hassas

bilgi

2. Test verileri oluşturmak zorunda kaldım. işimde üretmek zorunda kaldım

doldurmak için selenyum kullanmak zorunda kaldığım park biletleri için test verileri

oluşturulacak form

test ortamında park cezası.

- java sahtekarı

- apache'nin ayrıca birkaç dips vardır

-------------------------------------------------- --------------------

----

\* sizin için hangi kodlama standartlarına sahipsiniz?

- projemde kod formatı için bir formatlayıcımız vardı. herkes vardı

aynısını kullanmak

standart biçimlendirme.

- şirketimde kod incelemesi sırasında kontrol ettiğimiz şeylerden biri

kodlama standartları

-------------------------------------------------- --------------------

----

\* Kaç çevreniz var?

- projemde aşağıdaki ortamlar vardı:

1. geliştirme ortamları

2. QA/Test ortamı - test ettiğim yer burası

3. Sahneleme ortamı

4.

-------------------------------------------------- --------------------

----

\* Kod incelemesi nasıl yapılır?

- şirketimde kod incelemeleri yapmak için kritik değer kullanıyoruz. sadece çalışır

bitbucket ile

- github'da durum farklıdır:

- kodu kontrol ediyorsun

- bir çekme isteği oluşturun

- Bazıları çekme isteğini gözden geçirecek ve birleştirecek

-------------------------------------------------- --------------------

----

\* Kodunuzu kim gözden geçirecek?

- diğer testçiler

- test ucu.

-------------------------------------------------- --------------------

----

\*API hizmetini nasıl test edersiniz?

Restassured kitaplığını kullanıyorum, doğruladığım şey durum

kod, Yanıt mesajı/hata

Json formatı gibi diğer birkaç yanıt ayrıntısını iletin.

API'DEKİ GENEL HATALAR:

1xx: Bilgilendirici - Aktarım protokolü düzeyinde iletişim kurar

bilgi

2xx: Başarılı -Müşterinin isteğinin kabul edildiğini gösterir

başarılı bir şekilde.

3xx: Yönlendirme - İstemcinin biraz alması gerektiğini gösterir.

için ek işlem

isteklerini tamamla.

4xx: İstemci Hatası - Bu hata durum kodları kategorisi,

müşterilere parmak.

5xx: Sunucu Hatası - Sunucu bunların sorumluluğunu alır

hata durum kodları.

https://nordicapis.com/best-practices-api-error-handling/

Veritabanınız ne tür?

\*JSON bedeninizi Oracles'ta mı saklıyorsunuz?

Evet, Json verilerini Oracle Veritabanında, verileri

türleri

varchar2. (ayrıca clob veya blob) hangisinin kullanılacağı seçimi

tipik olarak motive

JSON belgesinin boyutuna göre yönetmem gerekiyor:

-Jason belgem aşmıyorsa varchar2 (4000) kullanıyorum

4000 bayt (veya karakter)

-JSON'unuzun bazılarının olduğunu biliyorsanız, VARCHAR2(32767) kullanın.

belgeler 4000'den büyük

bayt (veya karakter) ve hiçbirinden emin değilsiniz

belgeler 32767 baytı aşıyor

(veya karakterler)Footref 1.

-BLOB (İKİLİ BÜYÜK NESNE) veya CLOB (KARAKTER BÜYÜK) kullanıyorum

OBJECT) eğer biliyorsanız depolama

32767'den büyük bir JSON belgeniz olduğunu

bayt veya karakter

ORACLE'a JSON Verileri için örnek: (komut + tıklama tuşlarına basın) bağlantıyı açmak için

https://docs.oracle.com/en/database/oracle/oracle-database/12.2/

adjsn/yaratma-a-tablo-with-a-json-column.html#GUIDE6CC0DCF-

3D72-41EF-ACA4-B3BF54EE3CA0

Hizmetinizi ne zaman test ediyorsunuz ve aldığınız 500 hatası ve nasıl

debug eder misin?

Her şeyden önce 500, dahili bir sunucu hatasıdır. bu değil çünkü

bu hatalar genellikle

düzeltmesi zor ama. Bunun nedeni, hatanın genellikle

belirli ve bir için oluşabilir

sayıda farklı sebep. Hatayı ayıklamak için ne yapacağıma bağlı

sunucu üzerinde ve

aslında o sırada ne çalışıyor.

\*DEBUG ORTAK PLATFORMLAR

1: Yazılım yeni bir sürüme yükseltildiyse,

büyük ihtimalle yükseltme

başarısız oldu ve yenilenmesi gerekiyor.

2: Yeni bir eklenti veya tema henüz etkinleştirildiyse,

muhtemelen bunu yuvarlamak iyi bir fikir

geri değiştirin ve tekrar deneyin. İyi test edilmiş eklentiler bile

diğer eklentilerle çakışma

kurulumda.

3: Yazılım yükseltilmişse, eski eklentiler veya

temalar uyumlu olmayabilir

yükseltme ile. Tek eylem yolu başlamaktır

kadar şeyleri devre dışı bırakmak

hata gider. Bu, sorunu doğrudan çözmez;

ancak, onu bulacak

sorun. Eklenti geliştiricisinin

bir güncelleme yayınlayın.

\*SUNUCU TARAFINDAN DESTEK BELİRTİLERİNDE HATA AYIKLAMA

1: Sunucu izni: Çok yaygın, yanlış izinler

bir dosya veya klasörde

komut dosyalarından birini içeriyorsa, bu hataya neden olur.

2: Sunucu zaman aşımı: Sunucu meşgulse,

yeniden başlatıldı veya bağlantı

kayboldu, komut dosyası yine de 500 Dahili rapor verebilir

Server hatası. Mümkün

hatanın bir dahaki sefere olmayabileceğini.

\*400 nedir?

4XX – İstemci Hatası

4XX serisi hata kodları, belki de en ünlüsüdür.

ikonik 404 Bulunamadı durumu,

olan URL'ler ve URI'ler için iyi bilinen bir işaretçidir.

yanlış oluşturulmuş. Diğer daha kullanışlı

API'ler için durum kodları ancak bu aralıkta bulunur.

414 URI Too Long, verilerin

bir GET isteğinde iletildi

çok uzun ve bir POST isteğine dönüştürülmeli. Bir diğer

ortak kod 429 Çok fazla

Bir istemciyi not etmek için hız sınırlaması için kullanılan istekler

aynı anda çok fazla istekte bulunma,

ve trafiklerinin reddedildiğini.

\* API'nizden yanıtınızı nasıl doğrularsınız?

\*Veritabanınızda herhangi bir doğrulama yapıyor musunuz?

evet ama ondan önce bir bağlantı oluşturduğumdan emin oluyorum:

1: Bağlantı bağlantısı =

DriverManager.getConnection(dbUrl,kullanıcı adı,şifre);

Veritabanına Sorgu Gönder Bağlantı yapıldıktan sonra yürütüyorum

sorgular.

2: Açıklama

deyim=connection.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE

,ResultSet.CONCUR\_READ\_ONLY);

Yürütülen sorgunun İşlem Sonuçları,

Sonuç Kümesi Nesnesi.

3: ResultSet sonuçSet = deyim.executeQuery("seçin \*

ülkeler");

\*\*\*\* Selenyum Veritabanı Testi Adımlarının Özeti \*\*\*\*

Adım 1) Yöntemi kullanarak Veritabanına bağlantı kurun.

DriverManager.getConnection(URL, "kullanıcı kimliği", "şifre")

Adım 2) İfadeyi Kullanarak Veritabanına Sorgu Oluşturun

Nesne.

İfade stmt = con.createStatement();

Adım 3) Yürütme sorgusunu kullanarak sorguyu veritabanına gönderin ve

sonuçları ResultSet nesnesinde saklayın.

ResultSet rs = stmt.executeQuery(çalışandan \* seçin;);

Veritabanı doğrulama bölümünü nasıl otomatikleştirirsiniz?

@Ölçek

public void jdbcMetadata() İstisna{'yı atar

Bağ

bağlantı=DriverManager.getConnection(oracleDbUrl, oracleDbKullanıcı Adı,

oracleDbPassword);

Beyan

deyim=connection.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE

,ResultSet.CONCUR\_READ\_ONLY);

String sql = "çalışan\_kimliği,soyadı,iş\_kimliği,maaşı seç

çalışanlardan";

ResultSet sonuçSet = deyim.executeQuery(sql);

// 1: VERİTABANI META VERİLERİ

DatabaseMetaData dbMetadata = bağlantı.getMetaData();

System.out.println("Kullanıcı:" + dbMetadata.getUserName()); //

Kullanıcı ve veritabanı türünü döndürür

System.out.println("Veritabanı türü:" +

dbMetadata.getDatabaseProductName());

// 2: RESULTSET, ResulSetMETADATA içinde saklar

ResultSetMetaData rsMedata = resultSet.getMetaData();

System.out.println("Sütun sayısı:" +

rsMedata.getColumnCount());

System.out.println(rsMedata.getColumnName(1));

// bir döngü kullanarak tüm sütun adlarını yazdır

for (int i = 1; i <= rsMedata.getColumnCount() ; i++) {

System.out.println(i +" -> " +

rsMedata.getColumnName(i));

}

//ResulSet'i Harita Listesine Atın

//Bir Harita Listesi Oluşturun

List<Map<String,Object>> list = new ArrayList<>();

ResultSetMetaData rsMdata = sonuçSet.getMetaData();

int colCount = rsMdata.getColumnCount();

while(resultSet.next()) {

Map<String,Object> rowMap = new HashMap<>();

for(int col = 1; col <= colCount; col++) {

rowMap.put(rsMdata.getColumnName(col),

sonuçSet.getObject(col));

}

list.add(rowMap);

}

// bir harita listesinden tüm Çalışan Kimliklerini yazdır

for (Harita<Dize, Nesne> em: liste) {

System.out.println(emp.get("EMPLOYEE\_ID"));

}

sonuçSet.close();

deyim.close();

bağlantı.kapat();

}

}

\* API'yi manuel olarak nasıl test edersiniz? Adımları açıklayın

Öncelikle API belgelerini gözden geçirerek başlıyorum. bu

nerede

API Kimlik Doğrulama URL'sini topluyorum. erişimimi elde etmek için

jeton.

Postacıyı manuel otomasyon aracı olarak kullanıyorum, erişimim olduğundan emin oluyorum

ile

yetkilendirmeyi anahtar olarak ve erişim belirtecimi değer olarak girmek. nın-nin

kurs

içerik türünün json olduğundan ve kabul ettiğimden emin olmalıyım

tip de

Json. Sonra çağrımı aldığımda API belgelerime dayanarak test ediyorum

bitiş noktalarım

Şu anda tercihler API'sini test ediyorum. Ben de durumdan emin

kod 200

ve bu şekilde manuel test yapıyorum. Diğer seçenek,

Ağ seçeneği

UI uygulamamı incelediğimde. Başlıkları, Önizlemeyi kontrol ediyorum,

Tepki.

======================== LINUX RÖPORTAJ SORUSU

===========================

\*UNIX veya Linux'ta bir klasördeki tüm bağlantılar nasıl bulunur?:

Daha spesifik olmak için ls -lrt kullanıyorum, sadece bağlantıları toplamak için grep kullanıyorum

kullanarak

ls - lrt | grep '^ l' " ^ <- bu sembole şapka adı verilir"

linux@nyj872:~ ls -lrt toplam 2.0K

-rw-r--r-- 1 Linux Etki Alanı Kullanıcıları 0 Aralık 6 2011 a

drwxr-xr-x+ 1 Linux Etki Alanı Kullanıcıları 0 Eylül 19 12:30 java/

lrwxrwxrwx 1 Linux Etki Alanı Kullanıcıları 4 Eylül 19 12:31 version\_1.0 ->

java/

linux@nyj872:~ ls -lrt | grep '^l'

lrwxrwxrwx 1 Linux Etki Alanı Kullanıcıları 4 Eylül 19 12:31 version\_1.0 ->

java/

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkaVVrHV

\* Bir süreç nasıl bulunur ve onu nasıl öldürürüm?:

"ps -ef" kullanarak tüm işlemlerin bir listesini alabilir ve ardından

seninkini bulmak için grep

işlem yapın ve bu işlemin PID'sini alın. PID'yi aldıktan sonra yapabilirsiniz

öldür komutunu kullan

bu süreci öldürmek

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkfGi5lc

\*UNIX veya Linux'ta arka planda bir program nasıl çalıştırılır?:

&amp; arka planda herhangi bir işlemi çalıştırmak ve ardından

işleri kullanabilirsin

bu işlem için iş kimliğini bulmak ve fg ve bg komutunu kullanabilir

bunu getirmek

ön plana ve arka plana işleme

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkfwCDNf

\*Linux'ta bir komutun çıktısı ters sırada nasıl sıralanır veya

UNIX ?:

1) UNIX sort komutunu kullanarak String'in sayısal değerine göre sıralama:

Çoğu zaman alfabetik sıralama yerine sayısal sıralamaya ihtiyaç duyarız.

Sadece

Sıralamak istiyorsak aşağıdaki Unix sıralama komutu örneğindeki gibi

dayalı bir

PID'nin sayısal değeri, sort -n'yi sort ile birlikte kullanabiliriz -

k(sütun).

Burada PID ikinci sütun olduğu için sort -nk2 işimize yarayacaktır. Bu

aynı zamanda

yapmanızı sağlayan, sütuna göre sıralamanın bir başka harika örneği

sıralamak

UNIX'teki herhangi bir sütuna dayalı veriler

unix-sort-examples@unix-tutorial:~/test ps -ef | sıralama -nk2

UID PID PPID TTY STIME KOMUTU

unix sıralama örnekleri 500 2832 0 Temmuz 18 /usr/bin/bash

unix sıralama örnekleri 1976 3556 2 Tem18 /usr/bin/ps

unix-sort-examples 2324 1 con 18 Temmuz /cthelper

unix-sort-examples 2676 1 con 18 Temmuz /cthelper

unix-sort-examples 2832 1 con 18 Temmuz /cthelper

unix sıralama örnekleri 3332 2676 1 Temmuz 18 /usr/bin/bash

unix sıralama örnekleri 3556 2324 2 Temmuz 18 /usr/bin/bash

2) UNIX sıralama komutunu kullanarak sıralamayı tersine çevirin:

Bazen ters sırada sıralamamız gerekir, örneğin azalan düzende.

sort -r seçeneği Unix'te ters sıralama yapmamızı sağlar.

unix-sort-examples@unix-tutorial:~/test ps -ef | sıralama -rnk2

unix sıralama örnekleri 3616 3556 2 11:49:43 /usr/bin/ps

unix sıralama örnekleri 3556 2324 2 Temmuz 18 /usr/bin/bash

unix sıralama örnekleri 3448 0 0 Ocak 1 /usr/bin/ps

unix sıralama örnekleri 3332 2676 1 Temmuz 18 /usr/bin/bash

unix sıralama örnekleri 500 2832 0 Temmuz 18 /usr/bin/bash

UID PID PPID TTY STIME KOMUTU

3) UNIX sütuna göre sıralama : Girişteki herhangi bir sütuna göre sıralama.

Unix'te sort komutu, çoğunlukla diğer Unix'lerle birlikte kullanılır

komutlar

find, grep, ls veya ps gibi ve bu komutların çoğu

çıktı

tablo biçiminde ve herhangi bir sütuna göre sıralamak istiyoruz. Unix sıralama

emretmek

sort -k seçeneğini kullanarak bunu yapmamızı sağlar. bir örnek görelim

veya Unix sıralama

ps komutunu kullanacağımız herhangi bir sütundaki çıktıyı sıralamak için komut

bunun için çıktı

örnek ve bu çıktıyı 2. sütunda (PID) ve sonrasında sıralayacağız

3. sütunda (PPID)

unix-sort-examples@unix-tutorial:~/test ps -ef | sıralama -nk2

UID PID PPID TTY STIME KOMUTU

unix sıralama örnekleri 500 2832 0 Temmuz 18 /usr/bin/bash

unix-sort-examples 2324 1 con 18 Temmuz /cygdrive/c/

Yazılım/puttycyg-20101029/puttycyg-20101029/cthelper

unix sıralama örnekleri 2564 0 0 Ocak 1 /usr/bin/bash

unix-sort-examples 2676 1 con 18 Temmuz /cygdrive/c/

Yazılım/puttycyg-20101029/puttycyg-20101029/cthelper

unix-sort-examples 2832 1 con 18 Temmuz /cygdrive/c/

Yazılım/puttycyg-20101029/puttycyg-20101029/cthelper

unix sıralama örnekleri 3332 2676 1 Temmuz 18 /usr/bin/bash

unix sıralama örnekleri 3556 2324 2 Temmuz 18 /usr/bin/bash

unix sıralama örnekleri 3764 3556 2 11:58:08 /usr/bin/ps

Devamını okuyun: https://javarevisited.blogspot.com/2011/08/unix-sortcommand-

example-tutorial.html#ixzz5IkhnBHek

\*UNIX veya Linux İşletim Sisteminde bir arşiv dosyası nasıl oluşturulur?:

UNIX veya Linux bilgisine dayalı başka bir mülakat sorusu

emretmek.

UNIX veya Linux'ta harika arşivler için tar komutunu kullanabilirsiniz.

yapabilirsiniz

UNIX'te sıkıştırılmış bir arşiv oluşturmak için tar ve gzip'i bile birleştirin

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkijuQg4

\* 644 izni olan bir dosyanın anlamı nedir?

644, sahip için izin 110'u temsil eder,

grup için 100 izin ve diğerleri için 100 izin, yani okuma +

sahibine yaz

bu dosyayı kim oluşturur ve grup için salt okunur izni ve

diğerleri.

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkjFOam7

\* Boş dosyaları veya dizinleri /tmp'den nasıl kaldıracaksınız?:

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5Ikls2zoo

OLUŞTURMAK:

//Bu, geçerli dizinde boş dosya oluşturacak

test@localhost:~/unix boş.txt'ye dokunun

//Bu, geçerli dizinde boş dizin oluşturacak

test@localhost:~/unix mkdir empty\_dir

//Bu komut, içindeki tüm boş dosyaları ve dizinleri bulacaktır.

Unix

test@localhost:~/unix bul . -boş

./boş.txt

./empty\_dir

YAZDIR:

//bu komut boş dosyaları yazdıracak

test@localhost:~/unix bul . -type f -boş

./boş.txt

//bu komut boş dizinleri yazdıracak

test@localhost:~/unix bul . -d tipi -boş

./empty\_dir

// Boş dosya ve dizinleri yazdırmak için grep komutu nasıl kullanılır

test@localhost:~/unix ls -ltr | grep '\<0\>'

drwxr-xr-x+ 1 test Etki Alanı Kullanıcıları 0 15 Haz 11:43 empty\_dir/

-rw-r--r-- 1 test Etki Alanı Kullanıcıları 0 15 Haziran 11:44 empty.txt

//boş dosya ve dizinleri yazdırmak için komut bul

test@localhost:~/unix bul . -maxdepth 1 -size 0 -ls

90353467524120775 0 drwxr-xr-x 1 test Etki Alanı Kullanıcıları

0 Haziran 15 11:43 .

9007199255261921 0 -rw-r--r-- 1 Test Etki Alanı Kullanıcıları

0 15 Haziran 11:44 ./empty.txt

19421773393554899 0 drwxr-xr-x 1 test Etki Alanı Kullanıcıları

0 15 Haziran 11:43 ./empty\_dir

KALDIRMAK:

// find komutunu kullanarak Boş dosya ve dizinleri kaldırma

test@localhost:~/unix bul . -boş -sil

test@localhost:~/unix bul . -boş -sil

test@localhost:~/unix ls -lrt

toplam 1.0K

-rw-r--r-- 1 test Etki Alanı Kullanıcıları 118 4 Ağustos 2011 contact.txt

//boş dosyaları kaldırmak için find ve xargs komutunu kullanarak ve

dizinler

test@localhost:~/unix bul . -boş | xargs rm -r

test@localhost:~/unix bul . -boş -tür d -exec rm -r {} \;

bul: `./empty\_dir': Bir dizin değil

Devamını okuyun: https://javarevisited.blogspot.com/2012/08/delete-emptyfiles-

dizinler-unix.html#ixzz5IkledGw5

\*Bir dizinde okuma iznim var ama giremiyorum

Niye ?:

Bir dizine girmek için izin yürütmeniz gerekir.

dizininiz varsa

izin yürütmek zorunda değil, o zaman buna giremezsin

cd komutunu kullanarak dizin

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkmKTFXi

SAYISAL DOSYA İZNİ UNIX:

0 – izin yok

1 – yalnızca yürüt

2 – sadece yazın

3 - yaz ve çalıştır

4 – salt okunur

5 - oku ve çalıştır

6 - okuma ve yazma

7 – oku, yaz ve çalıştır

SEBOLİK DOSYA İZNİ UNIX:

Referans Sınıfı Açıklama

kullanıcı dosyanın sahibi

dosya grubunun üyesi olan g grubu kullanıcıları

o dosyanın sahibi olmayan diğer kullanıcılar veya

grubun üyeleri

a Yukarıdakilerin üçü de ugo ile aynıdır

r okumak bir dosyayı okumak veya bir dizinin içeriğini listelemek

w bir dosyaya veya dizine yazma

x yürütmek bir dosyayı yürütür veya bir dizin ağacını tekrarlar

VARSAYILAN İZİN:

Dosyanın varsayılan izni -- 666

usmak -- 022

----------------------------------

Dosyadaki son izinler -- 644 (yani 110 100 100

rw- r-- r--)

kullanıcı için okuma ve yazma ve yalnızca grup ve diğerleri için okuma

Dizinin varsayılan izni -- 777

umask -- 022

-------------------------------------------

Dosyanın son izni -- 755 (111 101 101

yani rwx rx rx)

kullanıcı (sahip) için okuma, yazma ve yürütme ve bunun için okuma+yürütme

grup üyeleri ve diğerleri.

DEĞİŞİM İZNİ:

stock\_options:~/stok\_trading\_systems'e dokunun

stock\_options:~/test ls -lrt

toplam 8.0K

-rw-r--r-- 1 stock\_options Etki Alanı Kullanıcıları 0 15 Kasım 11:42

stock\_trading\_systems

stock\_options:~/test chmod 400 stock\_trading\_systems

stock\_options:~/test ls -lrt

toplam 8.0K

-r------- 1 stock\_options Etki Alanı Kullanıcıları 0 15 Kasım 11:42

stock\_trading\_systems

stock\_options:~/test vim stock\_trading\_systems

stock\_options:~/test chmod 777 stock\_trading\_systems

stock\_options:~/test ls -lrt

toplam 8.0K

-rwxrwxrwx 1 stock\_options Etki Alanı Kullanıcıları 0 15 Kasım 11:42

stock\_trading\_systems\*

Devamını okuyun: https://javarevisited.blogspot.com/2011/11/filepermissions-

in-unix-linux-example.html#ixzz5IknXmjRM

\*Dosya/dizin/belgenin boyutu nasıl bulunur:?

Okunabilir hale getirmek için du -h "dosya adı" kullanıyorum, sonuç

bayt/Megabit vb.

alexs-MacBook-Pro:Masaüstü arod$ du -h ALEX.pdf

6.9M ALEX.pdf

alexs-MacBook-Pro:Masaüstü ve $

\*Belirli bir bağlantı noktasında dinleyen tüm süreci nasıl buluyorsunuz?

Linux?:

Belirli bir bağlantı noktası örneğinde bir işlem bulmak için netstat kullanıyorum

$ netstat -nap | grep 8084

(Tüm süreçler tanımlanamadı, sahip olunmayan süreç bilgisi

gösterilmeyecek, hepsini görmek için root olmanız gerekir.)

tcp 0 0 0.0.0.0:8084 0.0.0.0:\* LISTEN 25314/java

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5Ikp8kt62

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5Ikp3c6W9

\*Linux'ta bir dosya açan tüm işlemleri nasıl bulursunuz?:

lsof (açık dosyaları listele) komutunu kullanabilirsiniz.

dosyası olan işlem

belirli dosya üzerinde işlemek. Kontrol etmek için çok kullanışlı bir komut

hangi işlem bir dosyayı okuyor

\*Bir kullanıcı tarafından açılan tüm dosyalar nasıl bulunur

$ lsof -u kullanıcı adı

alexs-MacBook-Pro:Masaüstü arod$ lsof | grep java

Devamını okuyun: https://javarevisited.blogspot.com/2016/06/10-example-oflsof-

komut-in-unix-linux.html#ixzz5Ikr9eUQV

Bir Linux sunucusundan HTTP isteği nasıl gönderilir?:

HTTP isteği göndermek için wget veya curl komutunu kullanabilirsiniz.

Aşağıda gösterildiği gibi bir Linux makinesinden GET ve POST

alexs-MacBook-Pro:Masaüstü arod$ curl http://

34.223.219.142:1212/ords/sa/çalışanlar/100

{"Çalışan kimliği":

100,"first\_name":"Steven","last\_name":"Kral","email":"SKING","phone\_nu

ber"

:"515.123.4567","hire\_date":"2003-06-17T04:00:00Z","job\_id":"A

D\_PRES","maaş":24000,

"commission\_pct":null,"manager\_id":null,"department\_id":

90,"bağlantılar":

[{"rel":"self","href":"http://34.223.219.142:1212/ords/hr/

çalışan/100"},

{"rel":"edit","href":"http://34.223.219.142:1212/ords/hr/

çalışan/100"},

{"rel":"describedby","href":"http://34.223.219.142:1212/ords/

hr/meta veri-katalog/çalışanlar/öğe"},

{"rel":"koleksiyon",href":"http://34.223.219.142:1212/ords/

sa/çalışanlar/"}]}alexs-MacBook-Pro:Masaüstü arod$

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5IkrjrAE2

\*Tam bir dizin yapısını nasıl oluşturursunuz, örneğin /parent/child/

UNIX'teki torun?:

Tümünü oluşturmak için -p seçeneğiyle mkdir komutunu kullanabilirsiniz.

tek seferde ana dizinler

$ mkdir -p yazılımı/java/app/config

$ pwd /ev/john

$ cd yazılımı/java/app/config/

[# ~/software/java/app/config]$

Devamını okuyun: http://www.java67.com/2012/09/10-linux-and-unix-interviewquestions-

answer-wipro-tcs-capegemini.html#ixzz5Iksbn3uU

================ SQL SORGULARI ===============

SQL? YAPI SORGU DİLİ

SQL AÇIKLAMALARI:

DML: VERİ MANİPÜLASYON DİLİ(SDET JOB)

SEÇME

SOKMAK

GÜNCELLEME

SİLMEK

BİRLEŞTİRMEK

DDL: VERİ TANIMLAMA DİLİ ( GELİŞTİRİCİ )

OLUŞTURMAK

DEĞİŞTİR

DÜŞÜRMEK

ADINI DEĞİŞTİRMEK

KIRMIZI

YORUM

DCL: VERİ DENETLEYİCİ DİLİ

HİBE

GERİ ÇEKMEK

TRANSİT KONTROLÜ

İŞLEMEK

GERİ DÖNME

KAYDETME NOKTASI

-- bir tabloda çift satır nasıl yazdırılır

first\_name, manager\_id'yi seçin

çalışanlardan

first\_name, manager\_id ile gruplandır

sayısı (\*) > 1 olan;

-- listeden belirli bir yinelenen birey nasıl yazdırılır

yönetici kimliği 145

seçme \*

çalışanlardan

burada first\_name = 'Peter'

ve manager\_id = 145;

------------ İç/sol/sağ/dışarı KATIL -----------

-- İç birleştirme Ans: Bir SQL Birleştirme kullanılır

-- iki veya daha fazla tablodaki verileri birleştirmek için,

-- aralarındaki ortak bir alana dayalı. İçin

-- örneğin, aşağıdaki iki tabloyu göz önünde bulundurun.

e.first\_name, d.manager\_id, d.location\_id'yi seçin

çalışanlardan e departmanlara katıl d

e.department\_id üzerinde = d.department\_id;

e.first\_name, d.manager\_id, d.location\_id'yi seçin

çalışanlardan e ayrılma departmanlarından ayrıldı d

e.department\_id üzerinde = d.department\_id;

e.first\_name, d.manager\_id, d.location\_id'yi seçin

çalışanlardan e sağ bölümlere katıl d

e.department\_id üzerinde = d.department\_id;

e.first\_name, d.manager\_id, d.location\_id'yi seçin

çalışanlardan e tam katılım departmanlarından d

e.department\_id üzerinde = d.department\_id;

------------ NATURAL JOIN ----------- niteleyici

-- Doğal birleştirme

-- aynı ada sahip sütunlarda birleşir

-- isim değişirse sorgu çalışmayı durdurur

-- yeni sütunlar eklenirse sorgu çalışmayı durdurabilir

-- veya yanlış verileri göster

-- e'nin yanında niteleyici olamaz.

ad\_adı seç,

Soyadı,

departman\_kimliği,

Bölüm Adı

çalışanlardan doğal katılım departmanları;

------------ KARTESYEN KATILIMI -----------

-- Genelde istediğin gibi değil

-- genellikle birleştirme koşulu kaçırıldığı için

-- size beklenenden daha fazla kayıt verir

-- çapraz birleştirme anahtar sözcüğünü kullanın

ilk\_adı seç

Bölüm Adı

çalışanlardan çapraz katılım departmanları;

ilk\_adı seç

Bölüm Adı

çalışanlardan, departmanlardan;

------------ KENDİNE KATIL -----------

-- bir tabloya kendi başına katıl

-- bir tablodaki bir kayıt başka bir kayıtla ilişkili olduğunda kullanılır

aynı tabloya kaydet

-- çalışan\_kimliği // yönetici\_kimliği

e.employee\_id öğesini seçin,

e.first\_name,

e.soy\_adı,

m.ad\_adı,

m.soy\_adı,

m.çalışan\_kimliği

çalışanlardan e

sol katılmak çalışanlar m

e.manager\_id üzerinde = m.employee\_id;

------------ ÇOKLU TABLOLAR BİRLEŞTİRME -----------

-- Çoklu Tablolar

-- veritabanının büyük avantajı

-- sadece başka bir birleştirme anahtar sözcüğü ekliyoruz

seçme

e.çalışan\_kimliği,

e.first\_name,

j.başlangıç\_tarihi,

l.iş\_başlığı

çalışanlardan e

job\_history j'ye katıl

e.employee\_id üzerinde = j.employee\_id

işlere katıl l

j.job\_id = l.job\_id üzerinde;

================================================= ==========

------------ instr işlevi -----------

-- diyelim ki o belirli pozisyonun bulunduğu sayıyı gösterir

EABEL için bana B harfinin numarasını yazdır

ad\_adı seç,

Soyadı,

e-posta,

instr(e-posta,'B')

çalışanlardan;

-- sayı olarak adın uzunluğu.

uzunluk (ad\_adı), ad\_adı seçin

çalışanlardan;

------------ İşlev içinde yuvalama işlevi -----------

-- bir dizinin son 4'ü olan 4568'den başlayarak yazdırmanıza izin verir.

telefon numarası

ilk\_adı seç

Soyadı,

e-posta,

telefon numarası,

instr(telefon\_numarası,4568)

çalışanlardan;

------------ Tarih fonksiyonları -----------

==== bugünün Tarihi nasıl yazdırılır ====

bunun için herhangi bir tabloya ihtiyacınız yok sql zaten onunla birlikte geliyor

sysdate'i seç

ikiliden;

-- Steven King'in adını, soyadını ve işe alınma tarihini yazdırın ve

5 yıl ekle göster

işe alma tarihinden itibaren ve takma ad olarak inceleme\_tarihi olarak adlandırın.

ad\_adı seç,

Soyadı,

kiralama\_tarihi,

ADD\_MONTHS(hire\_date,60) inceleme\_tarihi olarak

çalışanlardan

burada first\_name = 'Steven'

ve last\_name = 'Kral';

nasıl yazdırılır bir çalışan şirkette ne kadar süredir çalışıyor?

---- IN DAYS ay arasındaki fonksiyon kullanılırken ilk parametre

sysdate olacak

sistem tarihi ile kiralama saati arasındaki tarihi oluşturun.

ad\_adı seç,

Soyadı,

kiralama\_tarihi,

Month\_between(sysdate, kiralama\_tarihi) olarak

time\_with\_company

çalışanlardan;

--- yıllarda 12'ye böleriz ve daha sonra 1 ondalık sayıya yuvarlarız

ad\_adı seç,

Soyadı,

kiralama\_tarihi,

yuvarlak (months\_between(sysdate, kiralama\_tarihi)/12, 1) olarak

Year\_with\_company

çalışanlardan;

------------ YUVARLAK sayı fonksiyonları -----------

20.59'dan 21'e kadar sayıları yuvarlar

min\_maaş seçin,

YUVARLAK (min\_maaş)

işlerden;

---------- SQL RÖPORTAJ SORULARI ---------

-- çalışanın en yüksek ikinci maaşını bulun

max(maaş) seç

çalışanlardan

maaşın olmadığı yerde

(çalışanlardan maksimum (maaş) seçin);

- Her departmandan Maksimum Maaş bul

departman\_kimliğini seçin, maks(maaş)

çalışanlardan

gruplandırmak

departman\_kimliği;

-- geçerli tarihi yazdır

sysdate'i seç

ikiliden;

-- farklı çalışanın adını yazdırın

-- işe alım tarihi 17-HAZİRAN-03 ve 21-MAYIS-07 arasında olanlar

farklı bir ad seçin

çalışanlardan

'17-JUN-03' ve '21-MAY-07' arasındaki kiralama tarihi;

-- maaşı 10000'e eşit veya daha fazla olan çalışanı bulun

seçme \*

çalışanlardan

maaş >= 10000 olduğunda;

-- adı M ile başlayan çalışanların adını bulun

ilk\_adı seç

çalışanlardan

burada ad 'M%' gibi;

-- çalışanlar için yıllık maaş yazdır

ad, soyad, maaş\*12'yi seçin

çalışanlardan;

-- çalışanlardan Lex'i yazdır

seçme \*

çalışanlardan

burada first\_name = 'Lex';

-- 4000 ila 8000 arası kazanan çalışanlar için maaş yazdırın

seçme \*

çalışanlardan

4000 ile 8000 arasında maaş;

-- departman 60 ve 80'de çalışan çalışanları listeleyin

seçme \*

çalışanlardan

departman\_kimliği burada (60,80);

- çalışana göre azalan sırayla çalışan kimliğini ve soyadını alın

İD

çalışan\_kimliği, soyadı seçin

çalışanlardan

çalışan\_kimliği açıklamasına göre sipariş;

-- çalışanların soyadını ve soyadlarının uzunluğunu yazdırın

-- aynı soyadına sahip kaç çalışanın olduğunu da sayın

soyadını seç, say(\*) ,UZUNLUK(soy\_ad)

çalışanlardan

soyadına göre gruplandır;

- maksimum maaş ile çalışan kaydını seçin

seçme \*

çalışanlardan

maaş = (çalışanlardan maksimum (maaş) seçin);

-- çalışan tablosundaki en yüksek maaşı seçin

max(maaş) seç

çalışanlardan;

-- kimliğe göre çalışan aralığını seçin

seçme \*

çalışanlardan

nerede çalışan\_kimliği 100 ile 113 arasında;

--geri dönen çalışan adı, en yüksek maaş ve bölüm adı

e.first\_name'yi seçin,

e.maaş,

d.departman\_adı

çalışanlardan e departmanlara katıl d

açık (e.department\_id = d.department\_id)

nerede maaş (çalışanlardan maks(maaş) seçin);

-- her biri için en yüksek maaş çalışan adı bölüm adını döndür

Bölüm

e.first\_name'yi seçin,

e.maaş,

d.departman\_adı

çalışanlardan e departmanlara katıl d

açık (e.department\_id = d.department\_id)

maaş nerede

(departman kimliğine göre çalışanlar grubundan maksimum(maaş) seçin);

-- bir çalışanın başkan mı yoksa satış görevlisi mi olduğu bir satır seçin

-- temsilci ve çalışan 15000'den fazla kazanıyorsa

soyadı, iş\_kimliği, maaş seç

çalışanlardan

nerede (job\_id = 'SA\_REP'

veya job\_id = 'AD\_PRES')

ve maaş > 10000;

-- bir çalışan başkansa ve şundan fazlasını kazanıyorsa satırı seçin

15k

-- veya çalışan bir satış temsilcisiyse

soyadı, iş\_kimliği, maaş seç

çalışanlardan

nerede job\_id = 'SA\_REP'

veya job\_id = 'AD\_PRES'

ve maaş > 15000;

-- FI\_ACCOUNT içinden en yüksek maaşı alan çalışanları seçin

soyadı, iş\_kimliği, maaş seç

çalışanlardan

nerede job\_id = 'FI\_ACCOUNT'

ve maaş =

(maksimum(maaş) seçin

çalışanlardan

burada job\_id = 'FI\_ACCOUNT');

--ülke\_adının bulunduğu ülkeler tablosunda kaç kayıt olduğunu bulun

içlerinde "in"

sayıyı seç(\*)

ülkelerden

nerede ülke\_adı '%in%' gibi;

--doğal birleştirme

seçme \*

çalışanlardan, işlerden

burada çalışanlar.job\_id = iş.job\_id;

-- sorgu: ilk adı iş unvanını ve maksimum maaşı döndürür

--doğal birleştirme

işleri seçin.job\_title ,

işler.max\_maaş,

çalışanlar.first\_name

çalışanlardan, işlerden

burada çalışanlar.job\_id = iş.job\_id;

-- kendi kendine Katıl

a.first\_Name, b.first\_name seçin

çalışanlardan a, çalışanlardan b

burada a.first\_name = b.first\_name;

-- join ve on kullanarak açık iç birleştirme ..

a.first\_Name, b.first\_name seçin

çalışanlardan bir

çalışanlara katıl b

a.first\_name = b.first\_name üzerinde;

-- sağ birleşime katılır

c.first\_name,c.department\_id, j.min\_salary,j.job\_title seçin

çalışanlardan c

sağ birleştirme işleri j

c.job\_id üzerinde = j.job\_id

c.first\_name göre sipariş;

-- departman kimliğinde kaç çalışan olduğunu ve kimliğin 100 etiket olduğunu yazdırın

-- çalışanlar olarak

çalışan olarak count(department\_id) seçin

çalışanlardan

departman\_id=100;

SQL GELİŞTİRİCİ SORGUSU

-- SORGU: çalışanların içindeki her şeyi yazdırın

SEÇME \*

ÇALIŞANLARDAN;

-- Sorgu: çalışanların adını ve soyadını yazdır

ad, soyadı seç

çalışanlardan;

-- Sorgu: Ülkelerden ülke adını ve bölge kimliğini yazdırın

ülke\_adı, bölge\_kimliği seçin

ülkelerden;

-- Sorgu: Arjantin'i ülkeler tablosundan yazdırın.

ülke\_adı seç

ülkelerden

nerede country\_name = 'Arjantin';

--Sorgu: Çalışanlardan ad ve soyadı yazdırın;

ad, soyadı seç

çalışanlardan;

--Sorgu: Komisyon pct'sinin boş olmadığı tüm sütunları yazdırın

seçme \*

çalışanlardan

burada komisyon\_pct boş değil;

--Sorgu: Çalışanlardan birlikte ilk ad soyadını yazdırın ve

-- sütunu tam ad olarak yeniden adlandırın ve tablo adının altta olduğundan emin olun

dava

--Örnek ad: Ellen Abel -> EllenAbel

--Örnek Sütun Başlığı: FIRST\_NAME --> tam ad

ad seçin || last\_Name "tam ad" olarak

çalışanlardan;

--Sorgu: çalışanlar tablosunun bileşenlerini yazdırın, nelerden oluşur

çalışanları tanımlayın;

--Sorgu: Veritabanı Şemasını yazdır

tablo\_adı seç

user\_tables'dan;

--Sorgu: Adı yazdırın maaşı ve yıllık maaşı yazdırın

çalışanlar

-- ve anual\_salary olarak yeniden adlandırın

anual\_salary olarak ad, maaş, maaş\*12'yi seçin

çalışanlardan;

--Sorgu: Çalışanlardan yıllık +300 maaş yazdırın

çalışanlardan maaş\*12+300 seçin;

--Sorgu: adı yazdırın ve " mercedes rock'larını" birleştirin

ad seçin || q'[' mercedes kayaları]'

çalışanlardan;

--Sorgu: Çalışan kimliğinin daha az olduğu çalışanlardan her şeyi seçin

-- 120'den fazla

seçme \*

çalışanlardan

nerede çalışan\_kimliği < 120;

--Sorgu: büyük A adının ilk harfini içeren adı yazdır

ilk\_adı seç

çalışanlardan

burada ad 'A%' gibi;

--Sorgu: Çalışanlardan benzersiz iş kimliği ve ad yazdırın

farklı job\_id seçin

çalışanlardan;

--İç birleşim

--Query: Bir çalışan kimliğinin departman adını verin

departman\_adı, çalışan\_kimliği, ad\_adı, soyadını seçin

çalışanlardan e ayrılma departmanlarından ayrıldı d

e.department\_id = d.department\_id üzerinde

burada d.department\_id null değildir;

departman\_adı, çalışan\_kimliği, ad\_adı, soyadını seçin

çalışanlardan e ayrılma departmanlarından ayrıldı d

e.department\_id üzerinde = d.department\_id;

--Sorgu: departman tablosunun ve çalışan tablosunun olup olmadığını nasıl öğrenebilirim

--tam olarak aynı

(çalışan\_kimliğini seçin

çalışanlardan)

eksi

(departman\_kimliğini seçin

çalışanlardan);

(departman\_kimliğini seçin

çalışanlardan)

eksi

(çalışan\_kimliğini seçin

çalışanlardan);

========= Yumuşak Beceriler =========

1. Bize kendinizden ve çerçeveniz/mevcut projenizden bahsedin.

Şu anda kıdemli bir otomasyon Mühendisiyim ve şu anda

otomasyon yapmak, ne zaman ihtiyaç duyulursa

Ben de manuel test yapıyorum. ile çalışma deneyimim var

bankacılık ve sağlık hizmetleri web uygulamaları. Sahibim

benim gibi java gibi çeşitli araçları kullanma konusunda derin deneyim

programlama dili, selenyum, maven, testNG, Junit. Jenkins için

sürekli entegrasyon, paralel için yönetim aws için Jira

komutlarım için yürütme ve unix.

Ben bir inanca bağlıyım, 3 temel değere sahibim, tutku,

fırsat ve dürüstlük.

Hedef odaklıyım, detay odaklıyım ve her zaman yapılacak işlere odaklanıyorum.

tamamlanacak.

2. Zaman dar olduğunda, sadece iyi bir kod yazar mısın yoksa

gözden geçirmek için her zaman fazladan zaman ayırırsın

yazdığınız ve daha iyi hale getirmek için cilaladığınız kodu?

Her zaman kaliteli ürün sunarım, bunun nedeni yıllardır

ne kadar önemli olduğunu öğrendim

organize olmak ve aynı zamanda görevinize odaklanmak. yapmıyorum

tv izleyerek zaman kaybetmek gibi

Ya kodlama yapıyorum ya da yeni teknoloji öğreniyorum. sahip olduğum zaman

katılmak için boş zaman

buluşma toplantıları. Bilgiyi paylaşmayı ve aynı zamanda öğrenmeyi severim

diğerlerinden.

3. Nihai ve nihayet arasındaki fark?

final bir anahtar kelimedir, final sınıfa kısıtlamalar koymak için kullanılır

yöntemler ve değişkenler

bir sınıf final olarak işaretlenmişse, bu sınıf miras alınamaz

başka bir sınıf tarafından.

nihayet bir try catch ifadesinde kullanılır, bu bir bloktur, nihayet

birlikte istisna işleme için kullanılır

dene ve yakala blokları ile.

4. Projenizde aşırı yüklemeyi nasıl kullanıyorsunuz?

Yöntemlerimi oluştururken aşırı yükleme kullandım

mevcut projemde bir örnek, sahip olduğum listbox sınıfımız var

Bir oluşturulan

public class Liste Kutusu {

Liste seçin;

public Listbox(Liste seçin) {

this.list = liste;

}

public void select(int i) {

this.list.selectByIndex(i);

}

public void select(Dize metni) {

this.list.selectByVisibleText(metin);

}

public void deSelect(int i) {

this.list.deselectByIndex(i);

}

public void deSelect(Dize metni) {

this.list.deselectByVisibleText(metin);

}

5. 18 milyon satırlık veriyi bir yerden başka bir yere nasıl taşırsınız?

bir diğer?

veritabanı türüne bağlıdır. kadar karmaşık olabilir

spagetti yemek

6. Fonksiyonel test cihazı ile manuel test cihazı arasındaki fark nedir?

(Sanırım ürün sahibi benim gibi farklı isimler olduğunu düşündü.

zaman zaman bahseder)

projemdeki işlevsel testler, duman testimizdir. onun

çevremizin önemli yönlerini kapsayan.

manuel test, normalde benim durumumu anlamak için yaptığım şeydir.

işlevsellik. bir şey olup olmadığını kontrol etmek ve görmek için

otomatikleştirilemez veya bir şeyin otomasyona ihtiyacı yoksa.

7. Sizinle geliştirici arasında bir

bug, son söz kimde? (PO dedim)

Geliştiricimin hatamı reddettiği bir örneğimiz vardı. en

nedeni şu ki

gerekliliği anlamam onun için geçerli değildi

anlayış. üstesinden gelmek için ne yaptık

bu argüman, açıklığa kavuşturulması için BA'mızla konuşuldu.

gereklilik. belki ihtiyacı anlamıştır

benim anladığımdan farklı. Yani o günden itibaren

planlamak için yaptığımız her toplantı

temiz olmaya ve çok soru sormaya özen gösteriyoruz. sonunda biz

Ürünü teslim edenler olacaktır.

8. İş arkadaşlarınız arasındaki çatışmayı nasıl çözersiniz?

Çatışma her zaman vardır mesela bizim meslektaşımız gihan vardı

Kimin öfkesi her zaman yoldaydı.

Bir şey istediği gibi gitmezse çok çabuk patlardı.

Ekip olarak kendisine işaret ettik.

bir çözüm bulmak için mevcut araçlar. çok fazla araç var

stackoverflow gibi avantajlardan yararlanmak için çevrimiçi

guru99, bazı yardım sağlayanlar arasında. Böylece nasıl yapılacağını öğrendi

tepkileriyle ilgilen artık.

9. Stresle nasıl başa çıkıyorsunuz?

Bildiğiniz gibi özellikle IT üzerinde her zaman stres var. öldük

çizgiler ve test için sınırlı zamanımız var.

takım arkadaşlarıma patronum karıma sorabilirsin ve hep merak ederler

nasıl bu kadar sakin kalabilirim. nasıl bu kadar rahat olabilirim

nedeni şu. işe geldiğimde stres çantamı bırakıyorum

kapıda. sabaha taze başlıyorum

programıma ve teslim tarihlerime bakıyorum. hedeflerime öncelik veriyorum

yapılması gerekenler esas alınır.

10. Sprint çok yakında bitiyor ve serbest bırakmanız gerekiyor, ancak

küçük bir sorun buldunuz, ne yapardınız?

(Ekiple veya PO ile konuşun dedim) Ve dediler ki: Ama ürün sahibi

kapatmanı istiyor, çok geç,

tahliyeyi yapmalıyız.. (O zaman PO ne olursa olsun dedim

diyor)

Bir ekip üyesi olarak bunun önemini değerlendirmek zorundayız.

serbest bırakmak. Bir ürünü iyi teslim etmeyi tercih ederim

yarım bir üründen daha yapılır. bunun için geç kalacağım

sonuçta geceler biz kararlı bir ekibiz

ve söz verdiğimiz şeyi yerine getirmeyi seviyoruz.

11. Bize liderlik rolünü üstlendiğiniz ve bir

iş yerinde bir şeyler yapmak için grup.

Teşekkürler Şu anda aşağıdakileri içeren bir proje üzerinde çalışıyoruz.

veritabanımızı test ediyoruz. bizim PO

sprintin ortasında bize geldi ve bir

farklı proje. o projeye ihtiyacı vardı

kısa sürede bitecek, başka bir ekibimiz de vardı.

bize yardım ediyor çünkü bu projenin bir zaman çizelgesi vardı

ve yaptığım şey, ekibimizi topladığımız yaklaşım hakkında konuştuğum bir şeydi.

bu görevi tamamlamak için alacaktır. dayalı olarak düzenledik

işlevsellik ve ne gerekli öncelik. ayardan sorumluydum

Scrum ustamız BA'mızla konuşarak toplantılara katıl

yardıma ihtiyacı olan takım arkadaşları için bilet oluşturma

açıklamalar. aynı zamanda akran kod incelemesi yapıyordum.

Bahsettiğim gibi, bu kadar küçük bir zaman diliminde bu kadar çok şey yapabiliriz.

hatalar. Bu yüzden kodları gözden geçirmek için geç kaldım

işimizin iyi yapıldığından emin olmak için. aynı zamanda aramalar aldım

bana bir karım olduğunu hatırlatan bir kadından. öyleydi

mevcut ve zorlu. ama anladık. ahh ayrıca eve dönüş yolunda

Karım için biraz çiçek alacağımdan emin olabilirim.

12. Aynı/yüksek önceliğe sahip 2 kullanıcı hikayeniz var, nasıl olurdu?

ikisini de zamanında bitirdiğinizden emin misiniz?

kendimi organize et ve bir seferde 1 görev çalış. insanları gördüm

her iki testte de sonunda acele ediyorlar

kodlarını bitirmek için. çünkü akılları her kullanıcıya odaklanmaz

hikaye ama 2. o zaman neden hayatını zorlaştırıyorsun

Ölü hatlarımı tamamlayabileceğimi biliyorum, bu yüzden o anda 1 yapıyorum.

yapmadan önce derinden anlıyorum

13. Ekibinizi istediğiniz yeni bir araç/fikir hakkında nasıl ikna edersiniz?

herkesin karşı olduğu takıma getirmek için mi?

Selenium Grid'i tanıttım. Sauce Labs kullanıyorlardı

ve özellikle için zorluydu

testçiler. kendimiz yapmak zorunda olduğumuz için sevmediler

çok fikir. Onlara bir Demo gösterdikten sonra ve ayrıca

Kurulumu ve bunu kendi başımıza yapmanın faydalarını açıklamak

sevdiler. Çünkü kendi ayarlarımızı yapardık

raporlar ve ayrıca farklı şekillerde başlatabiliriz

tarayıcılar ve işletim sistemi.

14. Demo yaptınız mı?

Farklı Demo türleri gerçekleştirdim. çünkü bizde

Bana verilenlerden yararlandığım kaynaklar

bir demo benim paralel yürütmemdi ve diğeri küçük bir

projemde ekibime İşlevselliğimizi sundum.

eğitimimizin bir parçası olarak daha iyi olmamıza yardımcı olacak bir Demo düzenliyoruz

profesyonelce. ekibime katıldığımda tanıştırdım

çok fazla fikir var ama zaman kısıtlaması nedeniyle sadece

bazılarını başarmak. kadar geniş bir hayal gücüm var

proje geliyor ve ayrıca alabileceğimiz farklı yaklaşımlar.

15. Yalnızca test mi yapıyorsunuz yoksa geliştiricilere ve/veya BA'ya yardımcı mı oluyorsunuz?

işleriyle mi?

Akranlara yardım etme konusunda güvenilirimdir. şimdiki halimde

farklı türlerde katıldığımız proje

toplantılar. BA'm bir keresinde benden bazı kullanıcı hikayelerini incelememi istedi

onunla. şimdi geri bildirimimi sağlayabildiğim için

test tarafında, test senaryoları yazmasına ve öğrenmesine yardımcı oldum.

işlevsellik.

16. Mevcut projenizde her şeyin olabileceğine inanıyor musunuz?

otomatik? (Hayır dedim o yüzden manuel test cihazımız var)

hayır her şey otomatikleştirilemez çünkü hala manuel olarak yapmamız gerekiyor

otomatikleştirmeden önce test edin ve burası

otomatikleştirilebilen veya otomatikleştirilemeyen bir özellik buluyoruz. bir bulduğumda

hikaye veya otomatikleştirilemeyen bir işlevsellik

Ben tatmin olana kadar manuel olarak test ettiğimden ve otomatikleştirdiğimden emin oluyorum

otomasyona ne gerek var. tabiki karşı çıkamam

gereksinimler, belgeler otomatikleştir diyorsa, otomatikleştireceğim.

Ama bu yüzden planladığımız toplantılarımızı yapıyoruz.

ve bu yüzden BA/PO toplantılarına katılmayı seviyorum.

otomasyona neyin ihtiyaç duyduğuna dair düşüncelerime katkıda bulunun ve

ıvır zıvır.

17. done tanımının ne olduğunu bize açıklar mısınız?

Şu anda projem için kodumu bitirdiğimde emin oluyorum

onun .......bunun üzerinde çalışması gerekiyor....

18. Matematik becerileriniz ne kadar iyi?

Bu harika. Paslanmış ama uzun zaman önce bir bankada çalışıyordum.

19. Kaç tane ortamınız var?

şu anki işimde sermayede 5 tane var, dev çevrem var,

QA ortamı, regresyon ortamı, ön üretim -

ve üretim.

20. Belirli bir görevi tamamladığınızda, başkalarını mı gözden geçirirsiniz yoksa

sadece gönder? (doğru cevap tüm ekiple gözden geçirmektir)

ekip olarak görevimi bitirdiğimde kodlarımızı gözden geçiriyoruz.

inceleyerek gönderiyoruz.

1. Bana kendinden bahset?

Şu anda sermayesi olan kıdemli bir otomasyon Mühendisiyim. ben

kariyerime manuel test cihazı olarak başladım

dikkatimi çeken bir şey büyüme tutkusuydu. Yani ben

1 yıl sonra otomasyon yapmaya başladı

Evolent, BCBS, büyüme konusunda çok fazla destek sağladı ve

yeni araçlar öğrenmek. Şu anda projemde

BDD çerçevesi ile çalışıyorum. java'yı ana dilim olarak kullanmak,

özellik dosyalarım için selenyum ve salatalık.

maven, jenkins, test için testNG ve ayrıca Test ile çalıştım

NG. Proje yönetimimiz için kullandığımız Jira, bizim için

kod depoları. API testim için yeniden güvenceli kullanıyorum

manuel test aracı olarak kütüphane ve postacı ve

tarayıcı konsolu. veritabanı ve arka uç için JDBC kullanıyorum ve

manuel sorgular sql geliştirici kullanıyorum.

Şu anda bir java, linux ve Scrum Master sertifikam var. ben

kendime meydan okumayı severim ve hedeflere ulaşmayı severim

Ekibimi motive etmeyi ve aynı zamanda değer sağlamayı seviyorum.

2. Projeniz nedir?

sermaye 1'deki projem, kullanıcının manipüle edebilmesi içindir.

ayarlar tercihleri. Şu anda otomatikleştirdiğim şey

kişisel hesaplar, harici hesaplar ve

dil. bir kullanıcı olarak hesabı yapılandırabilirler ayrıca harici ekleyebilirler

hesaplar

veri girildiğinde bir API çağrısı oluşturur. emin ol

özellik dosyamda negatif, pozitif senaryonun yanı sıra api'yi otomatikleştir

ve veritabanı.

3. Gelecekte geliştirici olmak istiyor musunuz?

Bir geliştirici olmak, kendinizi değişikliklere zorluyor. Severim

sadece geliştirici olmak için değil, aynı zamanda

ayrıca BA/SMaster olmak. elimden geldiğince öğrenmek istiyorum.

Çünkü farklı rolleri anlamak farklı

Görüşler, yaptığım işte daha iyi olmama yardımcı oluyor. Yani bir değişiklik gelirse o zaman ben

açığım.

3. Otomasyonda neyi sevmiyorsunuz? (SQL dedim. Tamam dedi.

SQL ile başlayın ..)

API testi yapmıyorum. Yanlış gidebilecek birçok şey var.

normalde çünkü biz de uğraşıyoruz

örneğin kredi kartı, API, Kredi API'si gibi diğer API ile

emin olmak

Veriler kullanıldığında düzgün bir şekilde aktarılır. bu yüzden test ediyorum

yeniden püskürtülmeli ve aynı zamanda

zamanında bitirmek zorundayız.

4. Aynı verilere sahip olmaları gereken iki tablo var. Nasıl yapabilirsiniz

Dene?

onun bir API olduğunu söyleyelim, ne yaptığımı, üzerinde depolanan verileri test ediyorum.

girilen verilere karşı veritabanı.

her şeyden önce, anahtar değerlerimin uyuşup uyuşmadığını test ederim, test etmeliyim

içerideki veriler json yanıtımı söyleyelim.

veritabanı ile ilgiliyse, eksi ile test ederim. emin olurum

A ve B ve B ve A'yı kontrol etmek için

5. Aynı sayıda satıra sahip olmaları gereken iki tablo var.

Nasıl test edebilirsiniz?

SQL satır sayısını kullanıyorum.

6. Bir masa var. Adı, soyadı, maaşı, depID'si var. ben

her bölümün ortalama maaşını bilmek istiyorum

7. SQL'de JOINS nedir?

birleşimler birden çok tablodan veri sorguladığımda kullanırım, kullanışlıdır

eşleşen kayıtları bulurken ve bunun için iç birleştirme kullanıyorum

farklı birleşim türleri vardır: iç birleşim, sol birleşim gibi,

doğru, tam katılım. kendi kendine katılır.

8. Birlik, Birlik Tümü

birlik ve birlik arasındaki fark, birliği kullandığımız zamandır

kopyaları ve boş değerleri kaldırır, ayrıca sıralar

9. Java'da 'son' nedir? Ne zaman kullanıyorsun?

son olarak, normalde kısıtlamak için kullanılan bir yöntem veya değişkende kullanırız.

Bir sınıf

10. Java'da OOP kavramları?

kapsülleme

Soyutlama

polimorfizm

miras

11. Ters Dize yöntemi

Dize adı = "alex"

string temp = new StringBuilder(""+name).reverse().toString();

System.out.println(temp);

Dizi adı = "alex";

for(int i = name.lenght()-1; i >=0; i--){

ters = ters+isim.charAt(i);

}

System.out.println(rerse);

12. Salatalığı çerçevenizde nasıl kullanıyorsunuz?

Salatalık, hikayenin ve kabul kriterlerinin

uygulandı

herhangi bir kullanıcının anlayabileceği bir dilde.

kullandığımız dil ahududu ve her kullanıcı hikayesi

======= RÖPORTAJ SORULARI ======

Sizin için önemli olan çalışan nedir?

insanlar = benim için önemli olan takım arkadaşlarım. yanında olacağım

Alışveriş merkezi

gün

istikrar

büyüme

ağ oluşturma

İşveren sizin için önemli olan nedir?

dikenli problemler: Hangi problemlerle karşılaştığınızı söyleyebilir misiniz?

çözülmemiş zorluklar: Bu zorluğu çözmek için ne yapabilirim.

Bu pozisyon neden uygun hale geldi? biri mi ayrıldı

(AKA: BENİM için yeterince iyi misin?)

**Mülakat/Konular Çalışma Kılavuzu**

- Andy L.(parti 7)

Mülakat Hazırlığı

**Bana kendinden bahset**

1. Öncelikle bana zaman ve fırsat verdiğiniz için teşekkür ederim arkadaşlar.

şirketinizin.

2. Şimdiye kadar 6 yıldır BT sektöründeyim ve şu anda ekibimde bir S **DET'im .**

3. Farklı endüstri türlerinde çalıştıktan sonra ağ oluşturma konusunda alan bilgisi edindim,

bankacılık ve sağlık.

4. Manuel test cihazı olarak başladım ve sonunda otomasyon testinde uzmanlaştım

5. Ön uç testi ve arka uç testi konusunda maruz kaldım ve deneyimim var

6. Mevcut projemde kullandığım araçlar Selenium WebDriver, Java, TestNg, , salatalık

ön uç için.

a. Arka uç için SQL, jdbc ve oracle veritabanı

b. Projemde API testi üzerinde de çalıştım ve Restful , postman ve

güvence altına alınmış kitaplık

c. Ayrıca sürüm kontrolü için github ve sürekli entegrasyon/dağıtım için Jenkins kullandım.

ve çerçeve organizasyonu için maven ve proje yönetimi ve hata takibi için jira

d. amazon aws bulut makine sunucu yönetimi

7. Kariyerimin başlarında şelale metodolojisine maruz kaldım ve şu anda bir

Agile scrum ekibi ve toplantılara katılmak ve eserleri analiz etmek

8. Ayrıca veri odaklı ve davranışsal olan kendi çerçevemi oluşturdum ve tasarladım.

POM'a dayalıdır ve ayrıca mevcut çerçevelere de uyum sağlayabilirim

9. Motivasyonu yüksek, düşük gelirli bir yoksulluk içinde büyüyen bir birey olarak

Çok çalışmanın meyvesini verdiği çok genç bir yaş ve başarıya aç olmalısınız. pozitifim

kişi ve ben şirketlerin hedeflerine ulaşmasına ve onlarla birlikte büyümesine yardımcı olmayı seviyoruz. ben takım oyuncusuyum

haklı olduğum ekibim çok güvenilir olduğumu ve iyi iletişim kurduğumu söyler.

Sprint sırasında hedeflerimize ulaşmamızı sağlayacak beceriler.

**Mülakata hazırlanmak için sorular**

Otomasyonda karşılaştığınız zorluklar nelerdir?

Günlük aktiviteler?

Bana projenden bahseder misin?

Kuvvetli yönleriniz nelerdir?

Zayıf yönlerin neler?

Bana üzerinde çalıştığınız bir test vakasını tarif edin?

Bana üzerinde çalıştığınız bir test senaryosunu açıklayın

Bana bir liderlik rolü üstlendiğiniz ve bir grubu işinizde bir şeyler yapması için yönlendirdiğiniz bir zaman söyleyin.

Bana kupa böceğini tarif eder misin?

Scrum takım kompozisyonunuz nedir?

1

Sprintin ne kadar sürüyor?

Haftada kaç test vakasını otomatikleştiriyorsunuz?

Haftada kaç kusur buldunuz?

**KG Testi**

**Test stratejisi nedir?**

● Yazılım projesi için bir test planı, aşağıdakileri tanımlayan bir belge olarak tanımlanabilir:

Bir yazılım test etme çabasının kapsamı, amacı, yaklaşımı ve vurgusu

**Deneme planı nedir?**

- Test/test süreciyle ilgili belgeler

- Oluşmaktadır

- Giriiş

- Test stratejisi

- Test hedefleri

- Test varsayımları

- Veri yaklaşımı

- Kapsam ve test seviyeleri

- Yürütme stratejisi

- Testin nasıl gerçekleştirileceğini ve tanımlama sürecini açıklar

ve kusurları rapor etmek ve düzeltmeleri düzeltmek ve uygulamak

- Test yönetim süreci

- Testin lojistiğini ve ortaya çıkan tüm olayları ele alır

yürütme sırasında

- Test uygulaması

- Test ortamı

- Geliştiriciler, test kullanıcıları ve iş ekibi tarafından yürütülmeden önce gözden geçirilmelidir

**Test vakası nelerden oluşur?**

- Test adımlarını gerçekleştirmeden önce uyulması gereken ön koşul veya ön koşul

- Test adımları

- Test sonuçları- Başarılı/Başarısız

- Otomasyon durumu

- Yürütme Tarihi

**Test Senaryosu Nedir?**

- Test edilen uygulamanın uçtan uca işlevselliğinin beklendiği gibi çalıştığından emin olun

- Test cihazının, kontrol etmek ve gerçekleştirmek için ayağını son kullanıcının ayakkabısına koyması gerekir.

test altındaki uygulamayı nasıl kullandıklarına ilişkin eylem

- TS, kendisiyle ilişkili birçok test senaryosuna sahip olabilir, TS'yi çalıştırmadan önce yapmamız gereken

senaryo için test senaryolarını düşünün

- Test Senaryosu: Giriş sayfasını doğrulayın

2

- Test Durumu 1: Geçerli bir kullanıcı adı ve şifre girin

- Test Durumu 2: Parolanızı sıfırlayın

- Test Durumu 3: Geçersiz kimlik bilgilerini girin

- Her test durumunda, yürütme için ayrıntılı adımlar ve koşul vardır

**Test senaryosu ve Test komut dosyası arasındaki fark?**

**Test durumu :**

● Test durumu, girdi değerlerini, beklenen çıktıyı belirten bir belgedir.

ve testi yürütmek için ön koşullar.

● Bu terminoloji çoğunlukla Manuel Test için kullanılır

● Senaryonun nasıl test edileceğine ilişkin alt düzey ayrıntıların bir düzenidir.

**Test Komut Dosyası:**

● Yazılım testindeki bir test komut dosyası, aşağıdakiler üzerinde gerçekleştirilecek bir dizi talimattır:

Sistemin beklendiği gibi çalıştığını test etmek için test edilen sistem.

● Bu terminoloji çoğunlukla Otomasyon Testi için kullanılır

**Kalite Güvencesi/Kalite Kontrol Nedir?**

- QA, çıktının kalitesini doğrulamak için kullanılır

**Çerçevenizi sıfırdan nasıl oluşturabilirsiniz?**

- Test Çerçevesi - test senaryoları oluşturmak ve tasarlamak için kullanılan yönergeler ve kurallar

**1. Ortamı Ayarlayın; JDK, MAVEN, ECLIPSE IDE'yi kurun**

**2. Maven projesi oluşturun**

**a. ArtifactID - projenizin adı**

**b. GroupID - projenizi tüm projelerde benzersiz bir şekilde tanımlar**

**3. Maven Deposundan Bağımlılıklar Ekleyin**

**a. Selenyum Java**

**b. salatalıklı sandviç**

**c. <özellikler> içinde JRE Sistem kitaplığı 1.8**

**d. TestNG**

**e. WebDriverManager\_BoniGarcia**

**f. salatalık Java**

**g. Salatalık TestiNG**

**h. Apache POI.XML**

**i. APACHE POI**

**j. JDBC**

3

**k. Emin olabilirsiniz**

**ben. Gson**

**m. Log4j**

**4. Çerçeve Yapısı (paketleri) oluşturun**

**a. Sayfalar**

**i. Web öğeleri ve yöntemler**

**b. Fasulye**

**i. Özel sınıflar**

**c. koşucu**

**i. Cuckesrunner - kodlar oluşturur ve hedeflenen html raporunu depolar**

**ii. Sigara içmek**

**iii. gerileme**

**d. Adım Tanımları**

**i. Gerçek kodlar ve kanca sınıfı**

**e. testler**

**i. Veri odaklı testler**

**f. JDBC**

**g. API**

**h. Araçlar**

**i. Yapılandırma Okuyucu**

**ii. Sürücü sınıfı (Singleton)**

**iii. tarayıcıUtils**

**iv. DBUtils**

**v. APIUtils**

**i. Configuration.properties**

**j. Testng\_runner.xml**

**i. Koşucu sınıfından biri**

**Paketleri com.app.utilities ile çağırın**

**5. Kaynaklarda özellik dosyası (.feature) içeren özellik klasörü oluşturun**

**- Bir kornişon dilinde yazılmış dosya yürütülebilir belirtimi**

**6. Senaryonuzu yazmayı bitirdikten sonra, cukesRunner'ınızı dryRun=false ile çalıştırın, bu size**

**StepDefinition sınıfında saklayacağınız uygulama kodu**

**7. Kodunuzu nasıl çalıştırırsınız?**

**- Koşucu sınıfını kullanın - kodları çalıştırın ve salatalık raporu ve html raporları oluşturun**

**8. Çerçeveyi github'a gönderin**

**- Yeni repo oluştur**

**- Git URL'sini kopyalayın ve Eclipse'e gidin**

**- Git deposunu yapılandırın ve url'yi ekleyin**

**- Projeye sağ tıklayın, ekibi tıklayın, taahhüt et'e tıklayın ve git evrelemesinde olacaksınız**

**- ŞİMDİ KODUN GITHUB'DA**

4

**9. Sırada JENKINS ENTEGRASYONU**

Jenkins

- Açık kaynak otomasyon sunucusu

- Yazılım geliştirme sürecinin insan dışı kısmını otomatikleştirmeye yardımcı olur

- Sürekli entegrasyona izin verir

- Dev'in kodu düzenli aralıklarla paylaşılan bir depoya entegre etmesini gerektiren geliştirme uygulaması

- Bağlantı noktası 8081 yerel ana bilgisayardır

**Projenizi jenkins'te çalıştırmak için**

- jenkins hesabına giriş yapın

- Proje oluştur - serbest stil

- Eklentileri yükleyin -salatalık raporu ve git

- Kaynak kodu yönetimi altında git ve geçmiş git url'yi seçin

- Tetikleyici oluştur - periyodik olarak derlemeyi seçin

- Üst düzey maven'i çağır

- Maven versiyonu; MAVEN\_HOME

- Hedefler ; **temiz doğrulama -Drunner=smoke\_runnerxml**

- Oluşturma sonrası eylemler altında

- Salatalık raporlarını seçin

- Düzenlenebilir e-posta bildirimini seçin

- Düzenlenebilir e-posta bildirimi

- Derleme günlüğü ekleyin; derleme günlüğünü seç

- Gelişmiş ayarlara tıklayın

- Arıza-Herhangi biri

- Gelişmiş'i tıklayın

- Alıcı listesi - raporu alacak olan e-posta adresi. Birden fazla ise virgül ekleyin

- Tetik ekle'yi tıklayın - her zaman başarısızlık gibi

- Derleme günlüğü ekleyin; derleme günlüğü ekle'yi seçin

- Kaydetmek

- Son adım

- Şimdi oluştur'u tıklayın ve test çalışacaktır ve salatalık raporunuzu verir

**Yazılım Testi Nedir?**

1. Kullanarak yazılım hatalarını bulmak amacıyla bir programı veya uygulamayı yürütme süreci.

fonksiyonel ve otomasyon araçları

2. Bir yazılım programını/uygulamayı doğrulama/doğrulama süreci

3. Testçiler, geçmek için değil, kırmak için test etmelidirler.

**Yazılım Geliştirme Yaşam Döngüsü Nedir?**

**1. İhtiyaç Toplama**

a. Proje yaklaşımını ve fizibilite çalışmasını planlamak için kullanılan bilgilerin toplanması

b. Riskleri belirleme

5

c. KG gereksinimlerinin planlanması

**2. Gereksinimleri Tanımlama**

a. Ürün gereksinimlerini tanımlama/belgeleme aşaması

b. SRS (Yazılım Gereksinimi Belirtimi)

**3. Tasarım**

a. Uygulama nasıl oluşturulacak

**4. Geliştirme (kodlama)**

a. Gereksinimlere göre, geliştiriciler uygulamayı yazacak

i. Tasarım aşamasında seçilen araç/tekniklerin kullanılması

**5. Test**

**6. Dağıtım**

a. Serbest bırakma ürünü

**7. Bakım**

a. Ürünün stabil olduğundan emin olmak

b. Hatalarla ilgili müşteri raporuna bakma ve düzeltme

**STLC nedir?**

- Sistematik ve planlı bir şekilde yürütülen test sürecidir.

- STLC sürecinde kaliteyi artırmak için farklı faaliyetler yürütülmektedir.

ürün.

**Kaç ortamınız var?**

- Gelişim

- Birim testi

- Test ortamından daha az kararlı

- Test ortamı

- Manuel test burada gerçekleşir

**- Üretim ortamını tam olarak çoğaltır**

- Değişiklikler aralıklarla dağıtılır

- Otomatik duman testleri burada yapılır

- Otomasyon testleri burada çalıştırılır

- Otomatik Entegrasyon testleri burada çalışır

- Üretim Öncesi Ortam

- UAT ortamı

- Demo burada gerçekleşir

- yük/performans testi burada gerçekleşir

- Değişiklikler büyük aralıklarla dağıtılır

- Burada otomatikleştirilmiş büyük regresyon testleri (yayınlanmadan önce)

- Bu teste test planında karar verilir

- Çok kararlı

- Üretim ortamı

**Kod üretim ortamına nasıl dağıtılır?**

- Yerelden

1. çekme ve basma özelliğini kullanarak kodu Git'e kontrol edin

2. Birim testlerini çalıştırın

3. Değişiklikleri sunucuya dağıtın

6

a. Geliştirici kodu her kontrol ettiğinde jenkins tarafından otomatik olarak yapılır. Ondan sonra

dağıtım değişikliklerini Dev ortamına geçirir

- Dev ortamından

- Jenkins tarafından yapılan Test ortamındaki değişiklikleri dağıtın

- Programlanabilir veya manuel olarak tetiklenebilir

- Testten

- Jenkins tarafından yapılan değişiklikleri dağıtın

- Ön üretimden

**Sigara testi**

- Test ortamına karşı çalışır

- Değişikliklerin Test ortamına dağıtıldığı her zaman çalıştır

- Geliştirme ortamında çalıştırılabilir

**Regresyon Testi**

- UAT ortamına karşı çalışır

- Yeni değişikliklerin herhangi bir kusurla sonuçlanıp sonuçlanmadığını öğrenmek için

- Büyük hata düzeltmelerinden ve her sürümden sonra çalışır

**RTM nedir?**

- Gereksinim İzlenebilirlik Matrisi

a. Test senaryolarını gereksinimlere bağlama

**Regresyon Hiyerarşisi Nedir?**

**1.Programlama dili**

**2.Üst katman**

**3. test çerçevesi**

**4. yapı aracı**

**5.frontend-seleneium web sürücüsü**

**6.backend-jdbc ve yeniden güvence altına alınmış**

**7.jenkins -ci/cd**

**8.veri odaklı - apache poi**

**modüler**

**Yeniden kullanılabilir**

**bağımsız**

**Bana Çerçeveni Tanımla**

ÇERÇEVEMİ ve görüşmeciye nasıl açıklanacağı - Andy Lam ve biraz Alex

// **Veriye Dayalı ve Davranışa Dayalı** - Hibrit çerçeveye dayalı

// **Maven** - yalnızca bağımlılık yönetimi için değil, aynı zamanda komut istemi olarak oluşturma aracı

pom xml dosyasını kullanan bir araç, ayrıca dumanımı çalıştıran belirli bir **xml dosyasına da sahibim,**

// regresyon ve işlevsellik testleri

// Programlama dili olarak **Java - çalışan ön uç, arka uç (api) ve Java kullandığım veritabanı**

Verileri depolamak ve karşılaştırmak için **toplama çerçevesi**

// hassas/yeniden kullanılabilir verileri depolayan bir **özellikler dosyam da var - url, şifre, tarayıcı**

7

// ve verileri Java'ya biçimlendirdikten sonra, akışı kontrol etmek ve verileri onaylamak için **Testng test aracını kullanıyorum.**

kusurları bulmak için

Ayrıca , yalnızca oluşturmak ve kullanmak için **tekli desen** kullanan yardımcı program paketinde **Driver sınıfını oluşturdum.**

bir evrensel web sürücüsü

**ÖN UÇ** ; **selenyum web sürücüsü** ve çerçevemde **Sayfa nesne modelini** şu şekilde kullanıyorum

tasarım desenim; - sayfa nesneleri oluşturun; web öğelerini tanımlayın ve

//ve bir webelement değişkeni olarak saklayın,POM = ELEMENT/YÖNTEM BAZLI YENİDEN KULLANILABİLİRLİK

SAYFA OBJESİ

// @FindBy kullanarak web öğelerimi başlatmak için **Sayfa Fabrikası tasarım desenini de kullanıyorum -**

daha kolay/uygun

//Yarar; tarayıcı kullanımı - burada hayatınızı kolaylaştıran statik yeniden kullanılabilir kod, kodlar yapmak için

tarayıcıyı otomatikleştirmek daha kolay;

**ARKA UÇ (API)**

// Seri hale **getirme** işlemi için çerçevenizde **RESTASSURED kullanarak** ve

gerçekleşecek **serileştirme**

// bu şekilde **json yanıtını** bir Java toplama veri yapısında saklarsınız (/ yüksek seviye üretirim

pojos ve harita nesneleri) ve verileri beklenen değerle onaylayın (ayrıca Java verilerinde depolanır)

yapı)

// ayrıca bir api yardımcı program sınıfım var - yeniden kullanılabilir kodlar - bir satırın bir pojo oluşturduğu yöntem

// ÇERÇEVEMİ ÇAĞRILAMADAN ÖNCE manuel test için **postacı kullanıyorum**

**VERİTABANI testi**

sql sorguları üretmek için manuel i **sql geliştiricisi**

//OTOMASYON İÇİN ; Java'yı entegre etmek için **JDBC kitaplığını kullanarak bir CONNECTION alıyorum** .

oracle veritabanı

//sonra SQL sorgularını kullanarak **STATEMENTS oluşturma** ve ardından verileri bir **RESULTSET'e kaydetme**

nesne.

// Java veri yapılarını içerideki verileri depolamak ve karşılaştırmak için kullanıyorum

//ve **DATA DRIVEN ve CUCUMBER BDD** çerçevesini kullandığım için tüm bu testler

**özellik dosyaları** içinde saklanır

// **FEATURE FILE'dan** kodlar oluşturmaya ve bunları uygulamaya yardımcı olan **RUNNER sınıflarım var**

b adlı bir dosyaya

// ayrıca tüm testlerimden önce ve sonra çalışan kodlarımı uygulayan **HOOK sınıfına sahip - bu**

senaryo kullandığımda tetiklenen **TAKECREENSHOT arayüzümü** çağırdığım yer

arayüz (senaryo başarısız olduğunda)

//başarısız olan adımdayken bir resim çek

**kornişon dilinin** beklenen değerine dayalı kodlarımı burada sakladım

8

**DDT**

**senaryo ana hatlarıyla** çalışacağım ,

bu, boru hattını kullanarak **örnekler oluşturduğum** ve verileri depoladığım yer

// eğer büyük miktarda test verisi varsa, genellikle harici bir dosyada (excel) bulunur, bu yüzden **Apache POI'yi** kullanırım

**DDT EXCEL OTOMASYONUNU ÇAĞIRIN** ve excel dosyasından okuyun ve verileri saklayın

//java veri yapısına

// temelde yüksek riskli kodlarımı günlüğe kaydetmek için **log4j2 adında bir günlük aracım da var**

// ve son olarak raporlarım için, çerçevemde salatalıkta **Rerun.txt kodunu kullanıyorum**

salatalık sandviç kitaplığı tarafından oluşturulan "rerun:target/rerun.txt"

//bu benim başarısız salatalık özellik dosyalarımı saklayacak

// o zaman , başarısız senaryoların bulunduğu yere sahip **failScenario koşucu sınıfım da var.**

(tekrar çalıştır.txt)

//başarısız bir Senaryo xml dosyası oluşturuyorum

//bu yüzden ne zaman özellik dosyalarında başarısız olsam mvn komutunu kullanırım; **mvn -Drunner=başarısız Senaryolar.**

başarısız testlerimi çalıştırmak için **xml dosyası**

//raporlama - salatalık raporları adı verilen hedef klasörde bulunan html raporunu kullandım

-"html:hedef/salatalık raporu"

**Paralel test** - koşucular ve **maven** oluşturmak için **salatalık jvm-paralel eklentisi kullanıldı**

testleri çalıştırmak için **güvenli eklenti**

**Sürekli entegrasyon** için (jenkins)

//devops yapılandırmayla ilgilenir

// github yolu var

//ama araç benim bir mvn komutum çağrıldı - mvn doğrulama -drunner=smoketest.xml

test cihazı tarafından sağlanan - xml dosyası

//raporlar için, her derlemede testin grafiksel bilgilerini veren bir salatalık raporu ve

ekran görüntüsü

İnşadan sonra salatalık raporu hakkında konuşun

**1. Sayfa Nesne Modeli Nedir?**

a. Kod fazlalığını azaltır ve kodu düzenler

b. Öğeleri tanımlamaya ve onu bir sayfa nesnesi değişkeni olarak saklamaya yardımcı olur

c. Saklandığı yere bağlayabilirsiniz

d. PageFactory tasarım deseni eklendi

2. Otomasyon aracım olarak Selenium WebDriver

a. Önce manuel olarak test edin:

i. Başlangıç aşaması

1. Fonksiyonel test

ii. arka uç

1. Veritabanı - SQLDeveloper IDE

9

2. API - Postacı

b. Selenium'u şunlarla entegre edin:

i. Uzman

1. Test paketi

2. Yardımcı paket

a. kullanıcı arayüzü

b. DB

c. API

3. Yapılandırma dosyası

a. Özellikleri

4. Sürücü sınıfı

a. Tek tonlu tasarım deseni

i. Özel bir kurucuya sahip olun

ii. salatalık

1. BDD sürecinde işbirliğini kolaylaştırın

2. Öykünün ve kabul kriterlerinin kolay bir şekilde anlatılmasını sağlar.

dil.

iii. Git - kaynak kontrolü

iv. Jenkins

v. Java

1. Koleksiyonlar Çerçevesi

2. Apache İÇN

3. JDBC

4. İçiniz rahat olsun

3. Kod yazmak için Java

4. Davranış Odaklı Gelişim

a. için doğru standartları karşıladığından emin olmak için müşteri ile birlikte gelişmek

müşteri

5. Veriye Dayalı Geliştirme

a. Farklı veri kümelerine karşı aynı test senaryosunu yürütme

b. Test akışı verilere göre değişmemelidir

6. Salatalık raporlaması

a. Hedef klasör

b. Jenkins

**Şelale Metodolojisi**

Şelale nedir?

Şelalenin Dezavantajları?

1. İletişim sorun

a. hiyerarşi

i. Kötü iletişim kanalı akışı

2. Bir sonraki aşamaya geçtiğinizde, bir daha yukarı çıkamazsınız

Şelalenin Faydaları

10

1. Daha fazla öngörülebilirlik

**Atik**

Çevik Çerçeve

**- Rol**

- Ürün sahibi

- Saldırı ustası

- Takım

**- Törenler**

- Sprint Planlama

- Sprint İncelemesi

- Sprint Retrosu

- Günlük Scrum

- Bakım seansı

**- Eserler**

- Ürün birikimi

- Sprint iş yığını

- Tükenmişlik Tablosu

- gösteren grafik

**Çevik Nedir?**

- Kısa yinelemelerde yazılım geliştirmeye dayalı bir metodoloji

- Scrum

- Takım, bir sonraki sprint için iş miktarını planlıyor

- Kanban

- Sprint planlaması yok, hikayeler olduğu gibi alınır ama yine de

diğer her Şey

**Agile'ın bileşenleri nelerdir?**

**- Ürün İş Listesi**

- Gereksinimleri + teknik görevleri listeler

- Ürün sahibinin istediği biziz. Gereksinimler

- Yapılması gereken kullanıcı hikayeleri

- Tamamlanmamış Görevler

**- Sürat koşusu**

- Zaman aralığı

- 2-4 hafta

**- Sprint İş Listesi**

- Bu sprint için seçtiğimiz hikayelerin listesi

- Sprint boyunca üzerinde çalışılacak hikayeler

- Sprint sonunda yapılmalıdır

**- Sprint Planlama**

- Product Backlog'a gidiyoruz ve hikayeleri görüyoruz

- Hız ve kapasiteye göre hikaye seçimi

**- Hız**

11

- Bir sprintte teslim edilen hikaye puanı/demo sayısı

- Örneğin: ekip 30 hikaye puanı planlamışsa (İş değeri)

bir sprintte ve daha sonra planlandığı gibi teslim edebilecek kadar kullanıcı hikayesi

takımın hızı 30

**- Kapasite**

- Bir sprint için mevcut toplam saat sayısı Takımın kapasitesidir.

- Tatil ve pto saatlerini hesaplar

**- Günlük Scrum**

- Her gün takıma geçen gün ne yaptığımızı, ne yapacağımızı bildiriyoruz.

yap, engelleyici var mı?

- 15 dakikadan fazla değil

**- Sprint Demo/İnceleme**

- Sadece bir demo

- Müşteriye sprint'i ne yaptığımızı gösterin

- Ürün sahibi geri bildirimde bulunabilir

**- Sprint Retrospektifi**

- Takım son sürat koşusunu analiz edecek

- Sorunları + nedenlerini tartışın

- Karar ekip tartışmasına dayalı olacaktır

- Patron yok

**- Sevk edilebilir ürün/artış**

- Ürünün parçası yapılır ve ek işlevsellik formu almaya devam eder.

her sprint

- Artış, geliştirme ekibinin Bitti Tanımına uygun olmalıdır

- Artış, PO tarafından kabul edilebilir olmalıdır-

**- Sprint Bakım seansı**

- Sprint planlamasından önce olur

- Takım, Scrum master ve PO, iş öğelerinin alakalı olduğundan emin olmak için bir araya gelir

ve yararlı

- Kullanıcı hikayelerinin PO'larına sorular sorun

- Kabul kriterlerini yeniden tanımlayın

- Yeni hikayeler yazmak

- Destanları kullanıcı hikayelerine bölme

- Doğru tahminde bulunmak için hikayeyi anlayın

- Az/fazla tahmini önlemek için

**Done'un tanımı nedir?**

“ Bitti Tanımı ”nı karşılaması gerekir.

- Kabul kriterleri yerine getirildi

- Ürün sahibi, kullanıcı hikayelerini kabul eder

**Doğrulama ve Doğrulama nedir?**

- Doğrulama geliştirme sırasında gerçekleşir

- Doğru şeyi inşa edip etmediğinizi kontrol ederken

- Doğrulama

12

- Yaptıktan sonra onaylıyorsunuz, geliştirdikten sonra yapması gerekeni yapıyor

BT.

- Test Ediciler /Geliştirici - doğrulama

- Test kullanıcıları - doğrulama

**Ready'nin tanımı nedir?**

- Kabul Kriterleri temizlendi/incelendi

- Puan/saat verilir

**BurnDown Grafiği nedir?**

- İşin tamamlanma hızının ve ne kadar iş yapıldığının grafiksel gösterimi

yapılması gereken kalıntılar

**Kullanıcı Hikayesi Nedir?**

- “ Bir müşteri olarak.. İstiyorum ki … .Böylece … ..

**Epik nedir?**

- Bir *destan* , büyük bir çalışma grubunu yakalar. Esasen kırılabilecek büyük bir kullanıcı hikayesidir .

birkaç küçük hikayeye bölünür. Bir destanı tamamlamak için birkaç sprint gerekebilir.

**Kabul kriterleri nedir?**

- Kullanıcının bakış açısından tanımlanan gereksinimlerin beyanları

bir hikayenin ne zaman " bittiğini " ve beklendiği gibi çalıştığını belirleyin

- 3 parça örnekleri

- Giriş - geçerli e-posta adresi

- Süreç - işaretleme mesajlaşma

- Sonuç - pazarlama mesajı tasarımı, pazarlama tarafından sağlanan özelliklerle eşleşir

**Sprint iş akışı nedir?**

- Bir hikayenin yapılmasından yapılmasına ve yaşam döngülerine nasıl geçtiği - bir şey olduğunda ne olur?

bloke vb.

Ne tür Test vakaları

Farklı senaryoları ele alıyorsun

a. Pozitif

b. Olumsuz

c. Sınır Değer Analizi

**Bir haftada genellikle kaç Test vakası tamamlıyorsunuz?**

- 10 küçük test vakası

- 7-8 orta

- 2-3 büyük

**Otomatikleştirmek için attığınız adımlar nelerdir?**

1. İşlevi öğrenin

a. Okuma gereksinimleri

b. BA ile bilgi transferi oturumu

c. Takım arkadaşlarına sor

2. Manuel olarak test edin

a. Her adımı doğru anladığınızdan emin olmak

b. Beklenen sonuçları anlayın

13

3. Otomatikleştirin

a. POM sayfaları oluşturun

i. Kullanacağınız gerekli öğeleri/yöntemleri ekleyin ve ekleyin

Sayfa Fabrikası tasarım deseni

ii. Singleton desenli bir sürücü sınıfı oluşturun

b. TestNG Assertions'ı kullanarak testleri doğrulayın

**JBirimi**

- Ek açıklamalar

- @Dersten önce

- Sınıftaki test yöntemlerinden herhangi birinden önce bir kez çalıştırın

- @Dersten sonra

- Sınıftaki tüm testler yapıldıktan sonra bir kez çalıştırın

- @Önceki

- @Test'ten önce çalışır

- @Sonrasında

- @Test'ten sonra çalışır

- @Ölçek

- Çalıştırmak için test yöntemi

- @Yoksayır

- Test yürütme sırasında bazı ifadeleri yoksaymak istiyorsanız kullanılır

**TestNG**

- Ek açıklamalar

- Öncelik

- Hangi numaradan başladığınız veya ardışık olup olmadığı önemli değil

- Ör: @Test(öncelik=0)

- DependsOnMethods= ” test yöntemi adı ”

- Birden fazla test adı ekleyebilir

- İlki başarısız olursa, 2. test hiç çalışmayacaktır.

- İlk yöntem başarısız olursa, raporunuz 2. testin başarısız olacağını gösterecektir.

atlandı

**- 2 yöntem birbirine bağlıysa, neden birleştirmiyorsunuz?**

**tek sınav mı?**

- Her test yönteminde Bc, farklı test ediyoruz

işlevsellik

- Mümkün olan en küçük parçayı tek bir testte test etmek istiyoruz

- Fonksiyonel test

- Etkinleştirilmiş

- @Test(öncelik=0, etkin=yanlış)

- Bu bir testi atlar

- TestNG tarafından alınacak ve raporda yer alacak

- @BeforeClass vs @BeforeMethod

- BM - her testiniz olduğunda çalışacaktır

14

- BC - @BeforeMethod'dan önce bile her şeyden önce çalışır

- Gruplar

- Ör: @Test(öncelik=0, gruplar= ” duman ” )

- Grup eklerseniz, bazen @Önce ve @Sonra çalışmaz, bu nedenle

AlwaysRun=true öncesi ve sonrası

- xml dosyasına eklenebilir

- Parametre açıklaması

- @İsteğe bağlı ek açıklama

- xml'yi çağırarak belirli bir test yapmanızı sağlar

- Testng'de paralel yürütme

- xml dosyasına yazın;

- paralel= ” testler ” iş parçacığı sayısı= ” 4 ”

- Thread-count, aynı şekilde kaç tarayıcı açmak istediğinizdir.

zaman

- Toplu çalışma

- xml dosyasında her şeyi çalıştırmak için .\* ekleyebilirsiniz

- Ör: <paket adı= ” .\* ” ></paket>

- Testng'in kendi raporları var

- xml çalıştırdığınızda size test-output klasöründeki raporu verir.

- Test raporunu html olarak içerir

**TestNG ve JUnit**

15

**GitHub**

GitHub nedir?

- Versiyon kontrol sistemi

- Belgelerin yeni/eski sürümünün kaydını tutar

- Dosya setini yönetir/depolar

**Depo nedir?**

- Dosyaların kaydedildiği klasör

- Tek, dosya koleksiyonları veya tek projeler içerebilir

**Uzak Depo nedir?**

- Sunucuda barındırın (GITHUB)

16

- Değişikliklerimiz yerelden uzak depoya gidiyor

**Yerel Depo nedir?**

- Genellikle bilgisayarınızda

- Çalışma Dizini, dizin ve HEAD'den oluşan değişikliklerimiz burada yapılır.

**Git komutları nelerdir?**

- Ekle - hazırlama alanına ekleyin

- Taahhüt - çalışma dizininden ve yerel depodan ekleyin

- Push - uzak depoya ekle

- Çek - değişiklikleri uzaktan çalışma dizinine al

- url ile klonla - url'yi dizine klonlar

- Git sürümü - size git sürümünü verin

- hangi şubede olduğunuzu , izlenmeyen değiştirilmiş dosyaları gösterir

- Menşei - uzaktan kumandanın adı

- Usta - şubenin adı

- Git ekle .

- Sahne alanına ekleme

- Özyinelemeli ekleme

- Her şeyi ekler

- Git commit -m “ mesajı tüm dosyalar için geçerli olacak ”

- Git taahhüt -am

- Git push Origin nameOfBranch

- Git yoksay

- Not defteri.gitignore

- Not defterinde hazırlama alanına eklemek istemediğiniz dosyaları ekleyin

- SİPARİŞ VERMEK İÇİN .GITIGNORE DOSYASINI REPO YAPMAK ZORUNDASINIZ.

GIT'TE YOK ETMEK İSTEDİĞİNİZ DOSYALAR

- Bazı dosyalar önemli değildir ve git'e itilmemelidir

- Bir dosyayı kaldırma

- GIT EKLE KALDIR POM

- BUNU KABUL ET

- VE İt

Kendi şubesini oluşturmak

Git **ödeme -b** nameOfBranch master

Git'i terminalde nasıl kullanırım?

- Yeni bir repo oluşturmak için;

- echo "# SqlMentor" >> README.md

git init

git README.md ekle

git commit -m "ilk taahhüt"

git uzaktan kaynak ekle https://github.com/Andylam224/SqlMentor.git

git push -u kökenli usta

- Mevcut bir depoyu itin

17

- git uzaktan kaynak ekle https://github.com/Andylam224/SqlMentor.git

git push -u kökenli usta

- Varsayılan düzenleyici

- Ardından kaçışa basın

- ":x!" Shift tuşuna basın

- Kaydet ve çık

- ":q!"

- Kaydet ve çık yok

- Çekme isteği nedir?

- Git birleştirme fetch\_head --allow-ilgisiz-geçmişler

- Hızlı ileri sarma olmayan bir sorunu çekme sorunu çözüldü

**Jira**

**Jira nedir?**

- Proje yönetimi aracı ve kusurların izlenmesine yardımcı olur

- Planlama ve zaman yönetimine izin verir

- Teslim tarihlerini/atamaları takip eder

Yalnızca biriktirme listesinde ve aktif sprintlerde test edici

**Aktif Sprint Tahtası**

- **İş Akışı** : Yapılacaklar>Devam Ediyor(burada da engellenebilir)>İnceleme(Teknik müşteri incelemesi)

bitti'ye geçmeden önceki kod>Bitti

**- Engellendi**

- Hikayeye gidin ve seçeneklere tıklayın ve daha fazla seçeneğe tıklayın> engellendi

- Neden engellendiği hakkında bir yorum yazın

- Scrum master en kısa sürede uğraşmak zorunda kalacak

- Blokta bir günden fazla hiçbir şey olmamalıdır

- Düzeltildikten sonra, engelleyici çözüldü olarak değiştirebilirsiniz

**- Jira terimleri nelerdir?**

- Sorun

- Yapmanız ve düzeltmeniz gereken biz

- Sorun Türleri

- Hikaye

- Görev

- Böcek

- Epik

**- Epik ve biletlerin farkı nedir**

- Epic BA tarafından yazılmıştır

- Biletler testçiler tarafından oluşturulur

- Açıklama kutusu

18

- Hata bildirme örneği

- Kutuya yazarsın

- Hata ne hakkında

- Hangi işlevsellik bozuluyor

- Hatayı yeniden oluşturma adımları nelerdir?

- Gerekli verilerle

- Hata raporunu ve ekran görüntüsünü ekleyin

- Beklenen sonuçlar

- Fiili sonuç

**JIRA'dan Kullanıcı Hikayelerini nasıl otomatikleştirirsiniz?**

1. Açıklamaya bakın - Çevik hikaye

2. Özellik dosyası oluşturun ve dosyayı jira story.feature olarak kaydedin

a. Kabul kriterlerinde yer alan senaryoyu ekleyin

3. cukesRunner'ı dryRun=true ile çalıştırın

4. Yöntemleri uygulayın

**5.** JIRA'DAKİ TEST DURUMLARINI OTOMATİKLEMEDEN ÖNCE HER ZAMAN MANUEL TEST EDİN

İLK

**Jira Eklentileri**

**- Zefir**

**- Röntgen**

**SPA nedir?**

- Tek Sayfa Başvurusu

- Sayfa bir kez yüklenir ve artık yüklenmez

- Sayfanın bazı bölümleri yükleniyor ancak tüm web sayfası yeniden yüklenmiyor/değişmiyor

- Örnek AJAX

- İstemci sayfaya bir şey eklediğinde/kaydettiğinde, sayfa AJAX kullanır.

sunucuyla iletişim kurun, ardından sunucu JSON'u sayfaya geri gönderin ve

öğeler güncellendi

**HTML**

**CSS**

- sınıf= .C

- AT

- id=#id değeri

- E#id değeri

- isim= [isim=N]

- Bağlantı = değer

- İçerir= E[= ' değer ' ]

- İle başlar

- ile biter

**Apple'ın önceki tüm kardeşleri nasıl edinilir**

19

Xpath: "//ul/li[içerir(metin(),'Apple Mobiles')]/preceding-sibling::li"

**Bu "Samsung Cep Telefonları" verecek**

**Apple'ın aşağıdaki tüm kardeşleri nasıl edinilir**

Xpath: "//ul/li[içerir(metin(),'Apple Mobiles')]/following-sibling::li"

**Bu, önceki tüm kardeşlere (Nokia Mobiles, HTC Mobiles, Sony Mobiles, Micromax) verecektir.**

**cep telefonları)**

**CSS'yi nasıl kullanıyorsunuz?**

- Bana projende kullandığın bazı yöntemleri söyle

**Bir web sayfasındaki bir metni nasıl bulursunuz?**

**/** /tagname[içerir(metin(), ' metin ' )] - belirli testler içerir

//tagname[.= ' text ' ] - tam metni içeriyor - bazen çalışmıyor

**Selenyum**

**- Selenyumda birkaç istisna belirtin**

**- Web Tabloları/ızgaraları nasıl ele alınır?**

- **Tablo** - tablo için kullanılan etiket - veriler ızgara biçiminde düzenlenir

- sütun adı için **th - etiketi**

- Örnek - <tr>

<th>FirstName</th> - en üst satırdaki sütun adları

<th>Soyadı</th>

<th>Yaş</th>

</tr>

- **tr -** etiketi bir satırı belirtmek için kullanılır, tüm sütun için geçerlidir

- **td-** etiketi, bir satırdaki bir sütunu belirtmek için

- Örnek - <tr>

<td>Danny</td> - ilk satırdaki gerçek veriler

<td>Smith</td>

<td>29</td>

</tr>

- Bazı masalarda **tbody var**

- Tablonun verilerini belirtmek için kullanılır, genellikle sütun içermez

isimler ( **th** )

**- staleElementReferenceException ile nasıl başa çıkılır? (mülakat sorusu)**

- İlk önce öğeyi, sayfayı veya öğeyi bulduktan sonra bu istisnayı alırsınız.

eleman yenilendi ve selenyum orijinal elemanla çalışamıyor

artık değil

- genel boolean retryingFindClick( By by) { boolean sonuç = false ; int

denemeler = 0 ; while (deneme < 2 ) { deneyin {

20

driver.findElement(by).click(); sonuç = doğru ; ara ; }

yakalamak ( StaleElementException e) { } attampts++; } sonuç döndür ; }

**- FindElement vs FindElements**

- FE tek eleman döndürür

- FES, web öğelerinin bir listesini döndürür

- Birden fazla eleman bulunursa;

- Fe ilk elemanı döndürür

- FES, tüm öğelerin listesini döndürür

- Eleman bulunamazsa;

- Fe, NOSUCHELEMENT istisnasını atar

- FES boş bir dizi listesi döndürür

**- Driver.close vs Driver.quit?**

- Driver.close - yalnızca geçerli sekmeyi kapatır, **Sürücü nesnesini boş yapmaz**

- Driver.quit - tarayıcıyı tüm sekmelerle kapatır, **nesneyi NULL yapar**

**- Dinamik kimliği nasıl ele alıyorsunuz?**

- Kimliğin statik kısmını bulun ve bir konumlandırıcı yazın (xpath veya css)

- Ardından Startswith, içerir, EndsWith'i kullanın

- //E[birlikte biter(@A, ' t ' )]

- //E[ile başlar(@A, ' t ' )]

- //E[içerir(@A, ' t ' )]

**- Açılır listeler nasıl ele alınır?**

- **Select** etiketini bulup ondan bir nesne oluşturuyorum

- Eski; Webelement element=driver.findElement(by.id( “ falan ” );

- ör;Listeyi seç=yeni Seç(öğe)

- 3 yöntem kullanabilirim

- SelectByVisibleText() - görüntülenen teste göre seçeneği belirleyin. Alır

dize parametresi

- SelectByIndex() - sayıma göre seçeneği seçer. İnt param alır

- SelectByValue() - değerin değerine göre seçeneği seçer

öznitelik. Dize parametresini alır

- getFirstSelectedOption() - seçilen seçeneği döndürür

- getAllSelectedOption() - seçilen tüm seçenekleri döndürür

(birden fazla seçilmiş seçenek mümkündür)

- getOptions()

- Web öğeleri listesindeki tüm seçenekleri döndürür

**- Ya seçme etiketi yoksa?**

- Açılır menü için etiketi ayrı olarak seçmelisiniz.

web öğesi.

- Ardından manuel olarak tıklama yöntemini kullanın

**- Rastgele ortaya çıkan dinamik bir açılır pencere varsa ne olur?**

- Uyarı ile dene/yakala kullanın

21

**- Bir web sayfası nasıl büyütülür?**

- driver.manage().window().maximize();

**- Selenium'daki anahtar sınıf nedir?**

- Klavyeden tuşlara basma seçeneği sunar

- Anahtar.ENTER

- SendKeys() yöntemine GEÇİLMELİDİR

- Eski; .sendkeys( “ şarj cihazı ” + tuşlar.ENTER)

**- Thread.sleep() nedir?**

- Yetişmek için selenyumu yavaşlatır

- İstisna atar, bu yüzden işlemeli veya atmalıdır

**- Selenium'da aşırı yüklenmiş Yöntemleri nasıl kullandınız?**

- İki değerin eşit olup olmadığını söylerken kullanırım

TestNG'den Assert.assertEquals(gerçek, Beklenen)

- String, Objects, int, boolean değerleri parametrelerini girebilirsiniz.

**- Selenium'da çerçeveler nasıl işlenir?**

- Çerçeveler - bir html sayfasını diğerine gömmek için kullanılır

- Adımlar

1. iframe'i bulun

2. Başka bir iframe'e geçin

a. driver.switchTo().frame();

i. .frame() - string, Integer, webElement, name veya id alır

doğrudan parametre olarak

ii. driver.switchTo().frame(webElement);

iii. driver.switchTo().frame();

1. Şimdi bir tane bulmak istiyorsanız 2. karedesiniz.

2. çerçevenin dışındaki öğe ( sizin

şu anda açık) - **NosuchElementException'ı atar**

2. Önceki kareye geri dönmeniz gerekiyor

a. driver.switchTo().parentFrame()

i. Bir seviye yukarı gider

b. driver.switchTo().defaultcontent()

i. En tepeye gider

3. Sayımı kullanarak geçiş yapabilir

a. sürücü.switchTo(0)

**i. olmayan her şeyi sayar**

**varsayılan çerçeve**

iv. Bu yöntemler, size bağlı olarak farklı sonuçlar verebilir.

hangi tarayıcıyı kullanıyorsun

**Neden NoSuchElementException alıyoruz?**

- Konumlandırıcının doğru olup olmadığını kontrol edin

- Zamanlamanın doğru olup olmadığını kontrol edin

- Öğenin bir iframe içinde gizlenip gizlenmediğini kontrol edin

22

**Birden çok çerçeve nasıl işlenir?**

- 4 kare varsa, belirli bir noktaya ulaşmak için her birinden art arda geçmeniz gerekir.

çerçeve. 1. kareden 3. kareye geçemiyorum.

**Başsız tarayıcı nasıl işlenir**

- Örnek htmlunitdriver - selenyumdan

- WebDriver=yeni htmlunitdriver()

- Çok kararlı değil

- Phamtonjstarayıcı

- Daha kararlı

- tarayıcı=yeni phantomjsbrowser()

**Birden çok pencere/sekme nasıl işlenir?**

- Selenyum tek pencerede kalır

- Bir pencere açarsanız ve ardından 5 sekme açılırsa, selenyum ilk sekmeye odaklanır.

pencere

- Yeni bir penceredeyseniz ve Selenium'a varsayılan olarak bir öğe yazdırmasını söylüyorsanız

kullanıcının odağı yeni pencerede olsa bile çalışmaya devam edecek

**- Yeni pencereye geçilmelidir**

- 1. windowHandle() kullanın

- Driver.getWindowHandle()

- Selenium her tarayıcı açtığında bir dizin verecek

Sayfanın kimliği - olarak adlandırılan Tutamaçlar

- Geçerli sayfanın tanıtıcısını/kimliğini döndürür (dize olarak)

- driver.switchTo().window(dize tutamacı)

- driver.getWindowHandles() - birden çok pencere için

**dizi pencere tutamacı** döndürür

**- Başlıkları kullanarak geçiş yapın**

- for(string tanıtıcı: driver.getWindowHandles()){

- driver.switchTo().Window(tutamak)

- if(driver.getTitle().equals(targetTitle)

- Kırmak;

-}

**Örtük bekleme ve Açık Bekleme arasındaki fark?**

**- örtülü Bekle**

- driver.manage().timeouts().implicitlyWait(3,TimeUnit..saniye)

- Ne zaman bir şey arıyorsan, bulamazsan bekle

3 saniye boyunca ve bu 3 saniye boyunca bulmaya devam edin

- Öğe bulunamazsa **NoSuchElementException atar**

zaman aralığı boyunca

**findElement** yöntemini kullandığımızda devreye girer

- Sadece bir kez ayarlanır (kancalar sınıfında)

**- Açık Bekleme**

- **WebDriverWait'i kullanın**

23

- Belirli eylemleri/belirli işlemleri beklemenizi sağlayan sınıf

web öğesi/yer bulucu

- Örnek - " id " kimliğine sahip öğe görünür hale gelene kadar bekleyin

- WebDriverWait'in iki şeye ihtiyacı var

- Öğe veya konum belirleyici

- Durum (ne bekliyorsunuz)

- Ör: WebDriverWait bekle=yeni WebDriverWait(sürücü,5)

- Zaman parametresi her zaman saniye cinsindendir

- wait.until( **BeklenenKoşullar.VisibilityofElemen**

**tLocated(By.cssSelector( “ #avlanan ” );**

- wait.until() - bekleme eylemlerini başlatır

- Bu yöntem öğeyi bulacak ve

görünür, o zaman elemana kadar bekleyecek

görünür

- Bulamazsa, **timeoutException'ı atar**

- Ayrıca boole referansını wait.until içinde saklayabilir

beklenen koşul.invisibilityofElementLocated

- BeklenenKoşul.elementTobeClickable()

- Her zaman beklendiği gibi çalışmıyor

- Zamanın sadece %90'ı

**- FluentWait nedir?**

- Açık beklemenin alt türü ancak **koşulları geçersiz kılabilirsiniz**

- Bekle<WebDriver>bekle=yeni

FluentWait<Web sürücüsü>(sürücü).withTimout(5,timeUnit.saniye).po

llingEvery(100,zaman birimi.

milisaniye).ignoring(NoSuchElementException.class);

**3 bekleme ne zaman kullanılır?**

**Örtülü Bekleme:** Örtülü bekleme sırasında Web sürücüsü onu hemen bulamazsa, çünkü

kullanılabilirliği nedeniyle, DOM'yi yoklamaya (yaklaşık 250 milisaniye) devam edecektir.

öğe. Öğe belirtilen Süre içinde mevcut değilse bir

NoSuchElementException yükseltilecek. Varsayılan ayar sıfırdır. Bir zaman belirlediğimizde,

Web Sürücüsü, WebDriver nesne örneğinin süresini bekler.

**Açık Bekleme:** Belirli bir öğenin birden fazla

yüklemek için dakika. Bu durumda, Örtük beklemeye büyük bir zaman ayırmayı kesinlikle sevmiyorsunuz,

sanki bunu yapıyormuşsunuz gibi tarayıcınız her eleman için aynı süreyi bekleyecektir.

Bu durumdan kaçınmak için gerekli öğeye ayrı bir zaman koyabilirsiniz.

sadece. Bunu takip ederek tarayıcınızın örtük bekleme süresi her öğe için kısa olacaktır.

ve belirli bir eleman için büyük olurdu.

**Akıcı Bekleyin:** Diyelim ki sadece 1 saniyede görünen bir öğeniz var.

ve bir süre görünmesi dakikalar alır. Bu durumda, akıcı beklemeyi kullanmak daha iyidir, çünkü

24

bu, öğeyi bulana veya son zamanlayıcı çalışana kadar tekrar tekrar bulmaya çalışacaktır.

dışarı. Örnek AJAX veya JQuery'dir

**Öğenin mevcut/görünür/etkin olup olmadığı ve mevcut metin nasıl kontrol edilir?**

1. Mevcut Öğeyi kontrol etmek için:

2. if (driver.findElements( By .xpath( "value" )).size() != 0 ){

System .out.println( "Element Mevcut" ); } başka {

System .out.println( "Element Yok" ); }

3. Veya

4. if (driver.findElement( By .xpath( "değer" ))!= null ){

System .out.println( "Element Mevcut" ); } başka {

System .out.println( "Element Yok" ); }

5. Görünür'ü kontrol etmek için:

6. if ( driver.findElement( .cssSelector ( "a > font" )).isDisplayed()){

System .out.println( "Eleman Görünür" ); } başka {

System .out.println( "Element Görünmez" ); }

7. Etkinleştir'i kontrol etmek için:

8. if ( driver.findElement( .cssSelector ( "a > font" )).isEnabled()){

System .out.println( "Eleman Etkinleştirildi" ); } başka {

System .out.println( "Element Devre Dışı" ); }

9. Mevcut metni kontrol etmek için

10. if (driver.getPageSource().contains( "Kontrol edilecek metin" ) )){

System .out.println( "Metin var" ); } else { System .out.println( "Metin

yok" ); }

**JavaScriptExecutor kullanıyor musunuz?**

Bu, kendi JavaScript'imi yazmama yardımcı oluyor

JavaScriptExecutor jsExecutor=(JavaScriptExecutor)sürücü;

- Bu sınıfı kullanarak tarayıcıya JS komutları gönderebiliriz.

-exeexeScript(); komutu yerine getirir

- Parametrenin içinde JS kodunu koyduğunuz yer

jsExecutor.executeScript(“alert('UYARI: Bu gereksiz bir mesajdır');”

- Bu kod bir JS açılır penceresi açar

Ayrıca is .executeScript(“js code”,element); 2 parametresini de koyabilirsiniz.

- Kaydırma için kullanılır (selenyum kaydırma ile iyi değildir, bir meydan okuma olduğunu söyleyebilirsiniz

Şartlar ve koşullar sayfasında çalışırken, okumanız gereken yer

Devam'a tıklamadan önce sayfa. Selenyum ve eylemler sınıfını kullanmayı denediğimde

işe yaramadı, bu yüzden javaexecutor kullandım ve bir öğeye tıkladım;

25

**Aşağı kaydırmak** için :

WebDriver sürücüsü = yeni FirefoxDriver (); JavascriptExecutor jse =

( JavascriptExecutor ) sürücüsü; jse.executeScript( "window.scrollBy(0,250)" , "" );

VEYA, aşağıdaki gibi yapabilirsiniz:

jse.executeScript( "scroll(0, 250);" );

**Yukarı kaydırmak** için :

jse.executeScript( "window.scrollBy(0,-250)" , "" ); VEYA, jse.executeScript( "kaydırma(0,

-250);" );

JavaScript, selenyumdan çok daha fazla kontrole sahiptir.

**- Eylemler sınıfı nasıl kullanılır?**

- Fareyi kontrol et

- Gelişmiş kullanıcı etkileşimleri için yöntemler sağlayan sınıf

- Gezinme

- Çift tıklama

- Sağ tık

- Taslak

- Sürükle ve bırak

- karıştır/eşleştir operatörleri

- Eylemler eylemi=yeni Eylemler(sürücü)

- Eylem yöntemleri

- Tıklayın()

- tutmak()

- inşa etmek()

- rol yapmak()

- tuşa basma()

- moveToElement(eleman)

- dragAndDrop(kaynak,hedef).perform()

- sendKeys() - genellikle kullandığımızdan farklı

- Sendkeys işlemini farklı elementler üzerinde yapmanızı sağlar

- webelement'ten gelen normal gönderme anahtarları bir

giriş metni olmayan bir şey için istisna.

- Uzun yol

action.moveToElement(source),clickAndHold().moveToElement(t

arget).release().perform();

**- perform() kullanılmadığı sürece eylemler çalışmaz**

26

**- Zincirleme metotları kullanıyorsanız, perform()'dan önce build()'i kullanmalısınız.**

**- Dosya indirme ve yükleme**

- İndirmek

- Selenium'un kendisi dosya indirmelerini doğrulayamaz, indirme bağlantısını tıklayabilir ancak

tarayıcının dışına çıkamaz ve indirilen dosyayı açamaz

- Bunun için başka araçlar kullanılmalıdır - Robot ve autoIT

- Yüklemek

- Selenium yüklemeyi gerçekleştirir, ancak bunu gerçekle karşılaştırıldığında farklı şekilde yapar

kullanıcı

- Adımlar

1. Yükleme penceresini tetikleyen öğeyi bulun

2. Yüklemek istediğiniz dosyanın yolunu bulun

a. Bir Dizeye Depolayın

i. Örn: dize

dosya= ” C:\\Kullanıcılar\\Andy\\Masaüstü\\klasör1\\dosya.anahtar ” ;

ii. Ardından driver.findElement(yükle

buton).sendkeys(dosya);

**Uzman**

**Maven nedir?**

- Koşucu sınıfımı çağıran POM xml dosyası olarak adlandırılan bir derleme aracı ve komut istemi aracı

ve bağımlılıklarımı yönetiyor

- Komut İstemi - mvn arketipi;oluştur

- Proje oluşturur

- Bir # seçin - enter tuşuna basın

- Bir # seçin - enter tuşuna basın

- **Grup Kimliği** ; Com.nameOfProject (genellikle ters çevrilmiş bir alan adı,

com.example.foo

- **ArtifactID** ; test-maven-projesi

- Sürüm - girin

- Paket - girin

-Y; girmek

**Maven eseri nedir?**

Bir yapı, bir Maven deposuna dağıtılan bir dosyadır, genellikle bir JAR'dır.

Bir Maven yapısı, derlenmiş bir JAR ve bir "kaynak" gibi bir veya daha fazla yapı üretir.

KAVANOZ.

Her yapıtın bir **grup kimliği** (genellikle com.example.foo gibi ters çevrilmiş bir alan adı), bir

**yapı kimliği** (yalnızca bir ad) ve bir **sürüm** dizesi. Üçü birlikte benzersiz bir şekilde tanımlar

eser.

27

Örnek: <groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenyum-java</artifactId>

<sürüm>3.11.0</sürüm>

Bir projenin bağımlılıkları yapıtlar olarak belirtilir.

**Bana maven yaşam döngüsünü açıkla;**

**- Komutlar yalnızca belirli pom xml dosyasının bulunduğu dizinde çalışabilir**

**bulunan**

1. 3 yerleşik yapı yaşam döngüsü

a. Varsayılan

i. Proje dağıtımınızı yönetir

b. Temiz

i. Proje temizliğini gerçekleştirir

c. Alan

i. Projenin site belgelerinin oluşturulmasını yönetir

2. Bir yapı yaşam döngüsü aşamalardan oluşur

a. Doğrula

i. Projenin doğru olduğunu ve gerekli tüm bilgilerin mevcut olduğunu doğrulayın

b. derlemek

i. Projenin kaynak kodunu derleyin

ii. Hedef klasör oluşturuldu

1. Raporlar burada saklanacak

c. Ölçek

i. Uygun bir birim test çerçevesi kullanarak derlenmiş kaynak kodunu test edin.

ii. Kodun paketlenmesini veya dağıtılmasını gerektirmemelidir

iii. Mvn -D(DeğişkenAdı)=testadı

1. Parametreye göre özel testler çalıştırın

d. paket

i. Derlenmiş kodu alın ve JAR gibi dağıtılmış bir biçimde paketleyin

e. Doğrulayın

i. Kalite kriterlerini sağlamak için entegrasyon testlerinin sonuçlarını kontrol eder

karşılandı

f. Düzenlemek

i. Bağımlılık olarak kullanmak için paketi yerel depoya yükleyin

g. Dağıtmak

i. Yapı ortamında yapılır, son paketi uzaktan kumandaya kopyalar

diğer geliştiriciler ve projelerle paylaşmak için depo

28

**Maven projesini Eclipse projesine nasıl dönüştürürsünüz?**

- Mvn tutulması

**Java projeleri nasıl yapılır?**

1. Klasörler/paketler oluşturun

2. Kitaplıklar/bağımlılıklar ekleyin

3. Sınıf dosyaları oluşturun

4. Derleme

5. Testleri çalıştırın

6. Dağıt

**Bağımlılıklarınızı/kütüphanelerinizi nerede buluyorsunuz?**

- Mvnrepository.com

- Maven çalışmıyorsa projeyi güncelleyin

- pom dosyanızın içinde bağımlılıklar olduğunda ve güncellemeyi kullandığınızda, maven

JAR dosyalarını internetten çekip projenize ekleyecektir.

**m2 klasörü nedir?**

- Jar dosyalarınızın/depolarınızın bilgisayarınızda kaydedildiği yer

**POM xml dosyası nedir?**

- Tüm projeyi yöneten bir dosya

- Bir maven komutu çalıştırdığınızda, her şey pom.xml üzerinden yapılmalıdır.

**Araçların sürümleri**

RestAssured Library çıkış tarihi: 2015

**Log4j**

- Herhangi bir uygulama tarafından kullanılır

- Örnek: LOG4J2

- Apachel'den

- Aktiviteyi kaydeder

- Dev loglara bakacak, saate bakacak, ip adresine gidecek ve ne olup bittiğine bakacak.

bir hata vardı

Kaydediciler, uygulamaların çok önemli bir parçasıdır ve gerçekleşen her adımı/olayları tutar.

zaman damgası

Normalde günlükler programlı olarak .log dosyasına yazılır

Herhangi bir çerçeveye veya uygulamaya eklemek için hazır araçlar/kütüphaneler vardır.

Java'da en ünlü günlük kitaplığı/çerçevesi apache'den LOG4J'dir.

Günlüklerin amacı:

- Uygulamayla ilgili yaşayabileceğiniz sorunları ayıklamamıza yardımcı olun.

29

Bazen uygulamada bir hata bulunduğunda, geliştiriciler önce günlükleri kontrol eder. Görmek için

kullanıcının hangi adımları attığı ve uygulamanın beklendiği gibi davranmadığı.

Günlükler, sorunun kaynağını bulmanıza yardımcı OLABİLİR (test açısından değil, uygulama açısından)

Java'da LOG4J

**Test otomasyonunda günlüklerin rolü nedir?**

Test çalıştırmalarımızın durumunu görmek için konsol veya html raporuna bakıyoruz. Bir şey başarısız olursa, biz

oradan bul.

Çerçevemize giriş uygularsak, otomasyona bakmanın başka bir yolu olacaktır.

yürütme adımları ve testimiz başarısız olduğunda sorunu bulmamıza yardımcı olacaktır.

**salatalık**

**Salatalık BDD çerçevesinin bileşenleri nelerdir?**

1. Özellik dosyaları

2. Cukes Runner

3. Adım tanımı

TEKNİK OLMAYAN PPL'NİN ANLAMASI İÇİN

BAĞIMLILIK - BDD BAĞIMLIDIR

POM.XML'DEKİ MVN DEPOSU FILEEEEEEE

CUCUMBER.IO'DAN SALATALIK BDD

tdd tekniklerini birleştirin

davranış odaklı

- Akışı ifade edin - müşteri davranışı

- Öğelere odaklanmayın

**özellik dosyası nedir?**

- Belirli bir özelliği veya işlevi test eden senaryolardan oluşur

- Özellik ana hikayedir, senaryolar ise hikayenin test durumlarıdır (özellik)

**Cukes Runner nedir (Runner sınıfı)**

- Testleri sıkı bir şekilde çalıştıran, adım tanımı için kodlar üreten bir sınıf

- @sigara testi

- Cukesrunner

- CUCKESRUNNER'DA NEREDE YERİNİ GÖSTEREN BİR ÖZELLİK KONUMUM VAR

ÖZELLİĞİM BULUNDU

**Adım tanımlama sınıfı nedir?**

- Gherkin dili ile başlayan adımlardan oluşan bir sınıf

**- Adım tanımının cukes Runner veya child ile aynı pakette olduğundan emin olun.**

**paket (ebeveyn veya kardeş değil)**

**-**

**Kornişon nedir?**

30

- Özellik dosyaları tarafından kullanılan dil

- Özellik, Senaryo, Verilen, Sonra, Ne Zaman, Ve, Ama, Arka Plan, Senaryo Anahattı,

Örnekler

**@CucumberOptions ne yapar?**

- Salatalık testlerinin çalışmasını özelleştirmek için kullanılan etiket

- @CucumberOptions içine şunları ekleyebilirsiniz:

- kuru çalışma

- Eklenti

- “ Güzel ”

- Konsola daha fazla bilgi ekler

- Size etiket, senaryo, yöntem bilgisi verir.

- "html:hedef/salatalık raporu"

- Hedef/salatalık raporu klasöründe bulunan html raporu oluşturur

- “ json:hedef/salatalık.json

- Etiketler

- Etiketler özellik yolunda bulunmalıdır

- Birden fazla etiket ekleyebilir...tags= “ @Dog,@Cat ”

- Özellikler - özellik dosyalarının bulunduğu konum

- Tutkal - adım tanımlama adımlarının nerede aranacağı. **kanca sınıfı da yapıştırıcının bir parçasıdır.**

**Raporlarınızı salatalıkta nasıl görebilirsiniz?**

- Hedef klasöre git

- Sistem gezgini ile aç

- Hedef>salatalık raporu>dizine gidin - yaptığınız testleri gösterir

**JUnit ile Salatalık nasıl çalıştırılır?**

- Salatalık ekleyin -junit bağımlılığı

- cukesRunner sınıfının üstüne @RunWith(Cucumber.class) ekleme

**Salatalık Testng ile nasıl çalıştırılır?**

- Salatalık -testng bağımlılığı ekleyin

- CukesRunner'ı AbstractTestNGCucumberTests'e genişletin

**Salatalıktaki kancalar nelerdir?**

- @Before kullanan sınıf - her salatalık senaryosundan önce çalışır

- @After - her senaryodan sonra çalışır

- Senaryo başarılı veya başarısız olursa olsun her zaman çalışır

**- Class, stepdefinition ile aynı pakette olmalıdır**

- Inhook sınıfının ekran görüntülerini uyguladım

**- DryRun=true ise HookClass çalışmaz**

- Senaryoyu önce/sonra yöntemimde parametre olarak kullanıyorum

**Salatalıkta ekran görüntüsü nasıl alınır?**

- Aftermethod'umda bir kod kullanıyorum:

- TakeScreenShot arayüzünü kullanıyorum

- Ekran görüntüsünü bayt veya dosya olarak saklayabilirsiniz

@Sonrasında

public void yırtma(Senaryo senaryosu) {

31

if(scenario.isFailed()) {

// ekran görüntüsü alma

final byte[] ekran görüntüsü = ((Screenshot Alır)

Driver.getDriver()).getScreenshotAs(OutputType.BYTES);

//ekran görüntüsünü rapora ekleme

senaryo.embed(ekran görüntüsü, "image/png");

}

**Koşucu sınıfınızı etiketsiz çalıştırırsanız ne olur?**

- Tüm özellik dosyaları yukarıdan aşağıya doğru çalışacak, ancak yalnızca içinde bulunan özellik dosyaları

@CucumberOptions “ özellikler= ”

**DDT ile Salatalık nasıl çalıştırılır?**

- Salatalık tabloları kullanıyorum:

- |Ana Sayfa|

- |E-postalar|

- |Belgeler|

- |Projeler|

(DataTable arg1) ile bir yöntem elde edersiniz

- DataTable parametresinde bunu şu şekilde değiştirebilirsiniz:

Liste<YourType>,List<List<E>>,List<Harita<K,v>>,ve Harita<K,v>

- Liste için sırayla yazdırır

- Harita için sipariş yok

**Geçebileceğim değişken türlerini nasıl sınırlayabilirim?**

- Kornişon parantez içine ekleyebilirsiniz (İşbirliği|Satış|İşaretleme,vb)

- Ör: @When("^ **İşbirliği|Satış|Pazarlama|Etkinlikler|Destek|Tümü) üzerine geldim**

menü$")

- public void i\_hover\_over\_the\_Collaboration\_menu(Dize menüsü) {

- geçiş(menü) {

- durum "Satış":

- BrowserUtils.hover(dashboard.sales);

- kırmak;

- durum "Pazarlama":

- BrowserUtils.hover(dashboard.marketing);

- kırmak;

- durum "Destek":

- BrowserUtils.hover(dashboard.support);

- kırmak;

- "İşbirliği" vakası:

- BrowserUtils.hover(dashboard.collaboration);

- kırmak;

- durum "Etkinlikler":

- BrowserUtils.hover(dashboard.activities);

32

- kırmak;

- durum "Tümü":

- BrowserUtils.hover(dashboard.all);

- kırmak;

-}

**Ya iki parametreli bir senaryonuz varsa (sınırlayıcı parametre, tablo**

**parametre)?**

Örnek :

Senaryo: İşbirliği menü seçeneklerini doğrulayın

suiteCRM'de oturum açtığıma göre

İşbirliği menüsünün üzerine geldiğimde

Ardından, İşbirliği için aşağıdaki menü seçenekleri görünür olmalıdır:

|Ana Sayfa|

|E-postalar|

|Belgeler|

|Projeler|

Bu senaryoda bir masam var ve işbirliğini sadece işbirliği ve

diğer menü kategorileri

Çözüm:

@Then("^aşağıdaki menü seçenekleri şunlar için görünür olmalıdır:

(İşbirliği|Satış|Pazarlama|Etkinlikler|Destek|Tümü):$")

public void follow\_menu\_options\_should\_be\_visisble\_for\_Collaboration **(Dize**

**menü,Liste<String> seçenekleri** ) {

- **Dize menüsü** 5 menü seçeneğini temsil eder

((İşbirliği|Satış|Pazarlama|Etkinlikler|Destek|Al)

- **List<String>options** , tabloları temsil eder;

|Ana Sayfa|

- |E-postalar|

- |Belgeler|

- |Projeler|

**DDT için salatalık senaryosunu nasıl kullanırım?**

- Mevcut projemde Örneklerle Senaryo Anahattı kullanıyorum

- Senaryo özellik dosyamda, veriye dayalı olarak bir değişken kullandığımda kullanırım

“ <değişken> ”

- Sonra Örneklerde:

- |değişken| - sütun adı

- |veri 1| - 1. sıra

- |veri 2|- 2. satır

- |veri 3| - 3. sıra

33

- Bu nasıl çalışır, her satır testi çalıştıracak ve senaryo kaç tane olduğuna bağlı olarak çalışacaktır.

Örnekte sahip olduğunuz satır

**Senaryo Anahattı nedir?**

- Veriye dayalı testler için kullanılır

- Aynı salatalık adımlarına sahip olun, ancak senaryodan sonra verileri kullanarak tablo olarak veri sağlıyoruz.

anahtar kelime örnekleri

**Arka Plan Nedir?**

- Salatalığın kendine has bir yöntemi vardır.

- Kancalı olan java içindir

- Özellik dosyası içinde bir senaryodan ÖNCE çalışan bir adım

- Tüm senaryolardan önce sadece üstüne koyabilir

- Arka planlara boru hatları yerleştirilemiyor

- Yalnızca senaryo taslağında

**Haritalar salatalıkta nasıl kullanılır?**

**- Senaryo Dışı Bir Anahat Kullanmak**

- Senaryo: Bir harita kullanarak kişi oluşturun

- suiteCRM'de oturum açtığıma göre

- Yeni bir kişi oluşturduğumda:

- | ad\_adı | John |

- | soyadı | Smith |

- | cep telefonu | 801-888-8889 |

- O zaman "John Smith" için iletişim bilgilerini görmeliyim

Sol taraf anahtardır ve sağ değerdir - yalnızca 2 sütun

-

- **Bir Senaryo Anahattı Kullanma**

- Senaryo Anahattı: Bir harita kullanarak kişi oluşturun

- suiteCRM'de oturum açtığıma göre

- Yeni bir kişi oluşturduğumda:

- | ad\_adı | <first\_name> |

- | soyadı | <lname> |

- | cep telefonu | <cep\_phone> |

- |office\_phone|<office\_phone>|

- O zaman "<first\_name> <lname>" için iletişim bilgilerini görmeliyim

-

- Örnekler:

- |first\_name|isim|cep\_telefonu|ofis\_telefonu|

- |Michael|Jackson|1234567890|2345678891|

- |Bonnie|Garcia|4569871234|4567890987|

-

- def adımında yazıyorum;

-

34

- @When("^Yeni bir kişi oluşturuyorum:$")

- public void i\_create\_a\_new\_contact(Map<String,String>contact) {

- // kişi oluştur iletişim kutusunu aç

- Her iki senaryo için de çalışır

**POJO salatalıkta nasıl kullanılır?**

- contactBean sınıfı oluşturun

- Tüm değişkenleri ekle

- Alıcı/ayarlayıcıları ekleyin

- Fasulye özellik dosyası oluşturun

- contactBean sınıfındaki değişkenleri içeren ilk satırı içeren bir tablo oluşturun

- Tablonun altına değerler ekleyin

- Parametreli uygulama yöntemi (List<ContactBean>contacts)

Senaryo: Kişi oluştur

suiteCRM'de oturum açtığıma göre

Yeni bir kişiyi kaydettiğimde:

**| ad | soyadı | bölüm | ofisTelefon | cepTelefon | e-posta |**

**| Steve | kapılar | BT | 3456758888 | 1234329999 | SteveGates123@gmail.com |**

O zaman "Steve Gates"in iletişim bilgilerini görmeliyim.

**Java**

**finalize() nedir?**

Bir nesnenin finalize() yöntemi, çöp toplayıcı tarafından nesneyi kaldırmadan önce çağrılır.

bellekten nesne (bu nesneye referans olmadığında). nesneyi yazabilirsin

finalize yönteminde temizleme kodu.

korumalı void finalize() Fırlatılabilir {} atar

● her sınıf, finalize() yöntemini Java.lang.Object öğesinden devralır

● **yöntem çöp toplayıcı tarafından artık belirlenmediği zaman çağrılır.**

**nesneye referanslar var**

● Object finalize yöntemi hiçbir eylem gerçekleştirmez ancak herhangi bir işlem tarafından geçersiz kılınabilir.

sınıf

● normalde Java dışı kaynakları temizlemek, yani bir dosyayı kapatmak için geçersiz kılınmalıdır

● finalize() işlevini geçersiz kılarsa , try-catch-finally kullanmak iyi bir programlama uygulamasıdır

ifadesi ve her zaman super.finalize() öğesini çağırmak için . Bu bir güvenlik önlemi

nesneler tarafından kullanılan bir kaynağı yanlışlıkla kapatmayı kaçırmadığınızdan emin olun

çağrı sınıfı

● korumalı void finalize() Fırlatılabilir { try { close (); // kapalı açık

dosyalar } nihayet { süper .finalize(); } }

● çöp toplama sırasında finalize() tarafından oluşturulan herhangi bir istisna, işlemi durdurur

sonuçlandırma, ancak aksi halde yoksayılır

● finalize() hiçbir zaman herhangi bir nesnede birden fazla çalıştırılmaz

Örneğin;

35

sınıf OpenAFile {

FileInputStream aFile = null;

OpenAFile(Dize dosya adı) {

denemek {

aFile = new FileInputStream(dosyaadı);

} yakalama (java.io.FileNotFoundException e) {

System.err.println("Dosya açılamadı" + dosya adı);

}

}

}

İyi davranmak için OpenAFile sınıfı, dosya sonlandırıldığında dosyayı kapatmalıdır. işte

OpenAFile sınıfı için sonlandırma yöntemi:

korumalı void finalize () Fırlatılabilir { atar

if (aFile != boş) {

aFile.close();

aDosya = boş;

}

}

**Sorun şu ki, finalize() yönteminize koyduğunuz herhangi bir kod,**

**çalışması garantilidir, bu nedenle içine önemli bir kod koymayın**

system.gc() nedir?

- JVM'ye, belleği boşaltmak için Çöp toplayıcıyı çalıştırma isteği

- Her zaman çalışmıyor

**Java Koleksiyon Çerçevesi**

Toplamak

- İki çeşit Koleksiyon (Karıştırmamaya dikkat edin)

- java.util.Collection - Set ve List'ten arabirim genişletilir (uygulanmaz)

- java.util.Collections - koleksiyonlarla kullanım için statik yardımcı yöntemler içeren bir sınıf

- Ekleme, kaldırma, içerme, boyut ve yineleyici vb.

- **Set** (Benzersiz şeyler) - TEKRARLAMALARA İZİN VERMEMEKTEDİR

- Kümeyi Uygulayan Sınıflar

**- HashSet**

- Herhangi bir kopya istemediğiniz ve umursamadığınız zaman kullanın

yinelediğinizde sipariş verin

- Sırasız

- Sınıflandırılmamış

**- BağlantılıHashSet**

- HashSet'in sipariş edilen sürümü

- Yineleme sırasını önemsediğinizde HashSet üzerinden kullanın

**- SortedSet**

36

**- Ağaç Kümesi**

- Öğeler, aşağıdakilere göre artan sırada olacaktır:

elementlerin doğal düzeni

- Yapıcıyı kendi uygulamanızı uygulamak için de özelleştirebilir

doğal düzenin kuralları

- **Liste** (şeylerin listesi) - dizine önem verir

- Liste uygulayan sınıflar

**- Bağlantılı liste**

- Endeks konumuna göre sıralanmıştır

- Elemanlar birbirine çift bağlıdır

- Yığın ve kuyruk uygulamak için iyi bir seçim

- Arraylist'ten daha yavaş yinelenir ancak hızlı ekleme ve silme

**- Vektör**

- ArrayList ile aynı AMA vektör yöntemleri senkronize edilir (iş parçacığı için güvenli)

**- Dizi Listesi**

- Hızlı yineleme

- Hızlı rastgele erişim

- sıralı (dizine göre)

- Sıralanmamış (ancak sıralamak için Collections.sort()'u çağırabilir)

**Harita** (benzersiz kimliğe sahip şeyler)

**- Önemli:** Harita ile ilgili sınıfların ve arabirimlerin hiçbiri Koleksiyon formunu genişletmez

- Map'in uygulama sınıfları Collection değil, koleksiyonlar olarak düşünülür .

- Haritayı uygulayan sınıflar

**- Hashtable**

- HashMap ile aynı AMA HashTable yöntemleri senkronize edilir

(UNUTMAYIN. YALNIZCA YÖNTEMLER SENKRONİZE EDİLMİŞTİR, DEĞİL

SINIFLAR VEYA DEĞİŞKENLER)

- Hashtable, NULL bir şeye sahip olmanıza izin vermez (HİÇBİR BOŞ YOK)

**- BağlantılıHashMap**

- Ekleme siparişini (veya isteğe bağlı olarak erişim sırasını) korur

- Öğe eklemek/kaldırmak için Hashmap'ten daha yavaş ama HIZLI

İTERASYON

**- HashMap**

- Sınıflandırılmamış

- Sırasız

- Bir koleksiyonda bir boş ANAHTAR ve birden çok boş değere izin verir

**- SortedHarita**

- Ağaç Haritası

Set,List ve Map'in uygulama sınıfları HİÇBİR ZAMAN hem sıralı hem de sırasız olamaz.

diğer tüm kombinasyonlar olsun.

HashMap

37

- Anahtar seti()

- Map.keySet() - bir dizi Anahtar döndürür

- Map.keySet().size - tuşların sayısını döndürür

**Araçların Sürüm ve Yayın tarihleri**

salatalık

- Çıkış tarihi = 2014

**Java**

**- IS-A ve HAS-A ilişkisi arasındaki fark nedir?**

- IS-A mirasa dayalıdır

- Bu şey o şeyin bir türü

- HAS-A ilişkileri kullanıma dayalıdır

- Ör: A sınıfı HAS -AB, eğer A Sınıfındaki kodun B sınıfının bir örneğine referansı varsa

- Halk Atı{

- Özel Halter myHalter;

- Genel boşluk atlama(){

- Sysout " atlıyorum "

- Attan gelen atlama yöntemini kullanmak için bir Halter örnek değişkeni çağırıyorsunuz

sınıf - bunun yaptığı şey, Horse HAS-A Halter'in olduğu anlamına gelir.

- Horse sınıfının bir Halter'i vardır, çünkü Horse, Halter türünde bir örnek değişkeni bildirir.

Wehn kodu, Horse nesnesinin Halter örnek değişkeninde tie() öğesini çağırır

-}

Soyut sınıfın kurucuları varken arayüzde bir tane yok

**Yerinde bir dize nasıl tersine çevrilir?**

public static void main( String args[])

{ Dize numarası = "1234" ;

System .out.println( "orijinal Dize:" + sayı);

= inPlaceReverse(sayı);

System .out.println( "ters çevrilmiş Dize: " + ters çevrilmiş); }

genel statik Dize inPlaceReverse ( son Dize girişi)

{ son StringBuilder oluşturucu = yeni StringBuilder (giriş); int uzunluk =

oluşturucu.uzunluk();

for ( int i = 0 ; i < uzunluk / 2 ; i++) { son karakter akımı = builder.charAt(i);

final int otherEnd = uzunluk - i - 1 ;

38

builder.setCharAt(i, builder.charAt(otherEnd));

// builder.setCharAt(otherEnd, geçerli);

} builder.toString() döndür ; }

**SQL**

İlişkisel Veritabanı terimleri

SQL select deyimleri için yetenekler

**- Projeksiyon**

Bir sorgu tarafından döndürülen tablodaki **sütunları** seçin

**- Seçim**

Bir sorgu tarafından döndürülen bir tablodaki **satırları** seçer

**- Katılmak**

Bağlantıyı belirterek **farklı tablolarda** saklanan verileri bir araya getirir

onların arasında

**Tuple -** tek sıra

**Tablo** - ilişki

**Sütun** - nitelik

**Alan** - kesişme noktasında veya bir satır ve bir sütunda bulunur

- İçinde değer olmayan bir alana boş değer denir.

**SQL** - Yapılandırılmış sorgu dili

- Çeşitli görevler için ifadeler sağlayın

- Veri sorgulama

- Tabloya satır ekleme, güncelleme, silme

- Nesneleri oluşturma, değiştirme, değiştirme ve düşürme

- Veritabanına ve nesnelerine erişimi kontrol etme

- Veritabanı tutarlılığı ve bütünlüğü

**DML** - Veri işleme

- Seçme

- Sokmak

- Güncelleme

- Silmek

- Birleştirmek

**DDL** - Veri Tanımlama dili

- Oluşturmak

- Değiştir

- Düşürmek

- Yeniden isimlendirmek

- kes

- Yorum

39

**DCL** -

- Hibe etmek

- Geri çekmek

**İşlem Kontrolü (DML deyimleri tarafından yapılan değişiklikleri yönetir)**

- İşlemek

- Geri alma

- Kaydetme noktası

SQL Geliştirici

- Tablo>Satırlar/Kayıt/Sütunlar

- Geliştirme ortamı (sorguları kullanarak veritabanını manuel olarak test etme)

- 2.1 -2009

- 3.0 - 2011

- Sürüm 4.0 - 2013(en son)

- Migration sürümüne sahip(1.2)

- kullanıcılara üçüncü taraf DB'deki verilere göz atmak ve geçiş yapmak için tek bir nokta sağlar

bu DB'den Oracle'a

- Windows, Linux ve Mac OS x'i destekler

**SQL yan tümcesi**

- Seçme

- itibaren

**SQL Deyimleri Yazma**

- Anahtar kelimeler büyük, sütunlar ve tablo adları küçük harfle yazılır

- İfadeler büyük/küçük harfe duyarlı değildir

- Cümleler genellikle ayrı satırlara yerleştirilir

- Anahtar kelimeler kısaltılamaz veya satırlara bölünemez

SQL deyimlerini yürütme

- Komut Dosyasını Çalıştır (f5)

**Aritmetik İfadeler**

**- Operatörleri herhangi bir tümcede kullanırsınız (From tümcesi hariç)**

- Tarih ve Zaman Damgası ile - yalnızca toplama ve çıkarma kullanabilir

- Ekle (+)

- Çıkart (-)

- Çarpmak (\*)

- Bölmek (/)

- Bir SQL ifadesinin herhangi bir yan tümcesinde aritmetik operatörleri kullanabilirsiniz (FROM

madde)

- Tarih ve zaman damgası ile yalnızca toplama/çıkarma işlemlerini kullanabilirsiniz

**Öncelik kuralları:**

1. Çarpma ve bölme, toplama ve çıkarmadan önce gerçekleşir.

2. Operatörler soldan sağa gider

3. Parantezler kuralları geçersiz kılar

**kısıtlamalar**

**- boş olamaz**

40

**- Birincil anahtar**

**- Geçersiz değil**

**Boş değer**

- Kullanılamayan, atanamayan, bilinmeyen veya uygulanamayan bir değer

- Null, sıfır veya boşlukla aynı şey değildir

**- Sıfır bir sayıdır**

**- Boş alan bir karakterdir**

- herhangi bir veri türünün sütunları boş değerler içerebilir, **ancak bazı kısıtlamalar (boş değil ve**

**birincil anahtar)** , sütunda boş değerlerin kullanılmasını engeller.

- Bir aritmetik ifadedeki herhangi bir sütun değeri null ise, sonuç null olur. eğer denersen

bir sayıyı 0'a bölersen hata alırsın. Bir sayıyı null ile bölmeye çalışırsanız sonuç null olur.

**Sütun Takma Adı**

- Bir sütun başlığını yeniden adlandırır

- Takma ad başlığı tamamı büyük harfle görünür ( küçük harf veya özel için “” )

- Sütun adı ve takma ad arasında anahtar sözcük olarak (yalnızca **As** anahtar sözcüğü için seçim yan tümcesinde)

- Boşluk veya özel karakter kullanıyorsanız veya büyük/küçük harfe duyarlıysa **çift**

**alıntı işaretleri**

- Rakamlar için, rumuz ilk sıradaysa çift tırnak işaretine ihtiyacınız vardır.

- Aksi takdirde çift alıntı yapmanıza gerek yoktur.

**birleştirme operatörü**

- Sütunları veya karakter dizilerini diğer sütunlara bağlar

- Kullanma || (çift değil tek tırnak ile çalışır)

- Boşluk eklemek için tek tırnak kullanın

- Karakter ifadesi olan bir sonuç sütunu oluşturur

- Tarih ifadelerini diğer ifadeler veya sütunlarla da birleştirebilir

- Birleştirme operatörü ile boş değerler

- Bir boş değeri bir karakter dizisiyle birleştirirseniz, sonuç bir karakterdir

sicim

- LAST\_NAME || LAST\_NAME değerlerinde NULL sonuç

**Değişmez Karakter Dizeleri.**

**- Karakter, tarih ve numara**

- Karakter değişmezi ve tarih değerleriyle tek tırnak kullanın

- Soyadını SEÇ || **' bir '** ||job\_id OLARAK çalışanlardan "Çalışan Ayrıntıları";

- Alıntı yapmanıza gerek olmayan numara

**Alternatif alıntı(q) operatörü**

- Kendi tırnak işareti sınırlayıcınızı belirtin.

Herhangi bir sınırlayıcı seçin. •

- Bunları {},[],(),<> kullanabilir

Okunabilirliği ve kullanılabilirliği artırın.

departman\_adı SEÇ || ' Bölüm' || **q'['nin Yönetici Kimliği: ]'** || manager\_id AS

departmanlardan "Departman ve Yönetici";

- Birleştirmede tek alıntıyı kullanmak için

41

**SQL Oracle tablo şeması nasıl bulunur**

tablo\_adı seç

user\_tables'dan;

**Belirgin**

- DISTINCT niteleyicisinden sonra birden çok sütun belirtebilirsiniz. DISTINCT niteleyicisi

seçilen tüm sütunları etkiler ve sonuç,

sütunlar.

**Select anahtar sözcüğünden hemen sonra** belirgin olmalıdır

- Boş yazdıracak mı?

- Evet, ilk olacak

**Betimlemek**

- Bir tablonun yapısını göster

- Ör: çalışanları TANIMLAYIN

- Null, bu sütunun değerlerinin bilinmeyebileceğini belirtir. NOT NULL şunu gösterir

bir sütun veri içermelidir. Tür, bir sütunun veri türünü görüntüler.

- SAYI(p,s) En fazla p basamağı olan, s basamaklı sayı değeri

ondalık noktanın sağı

- Eski numara (4,2)

- 2, ondalık virgülden sonra 2 hane olduğu anlamına gelir (ör: 3.01)

- VARCHAR2(s) Maksimum boyut s değişken uzunluklu karakter değeri

- TARİH MÖ 1 Ocak 4712 ile MS 31 Aralık arasındaki tarih ve saat değeri

9999.

- CHAR(lar) s boyutunun sabit uzunluktaki karakter değeri

**nerede cümlesi**

- Koşul (doğru/yanlış/bilinmeyen değer)

**- WHERE yan tümcesi FROM yan tümcesini takip eder.**

- WHERE yan tümcesi sütunlardaki, değişmez değerlerdeki, aritmetik ifadelerdeki veya

fonksiyonlar.

- Üç unsurdan oluşur: • Sütun adı • Karşılaştırma koşulu • Sütun adı,

sabit veya değerler listesi

- WHERE yan tümcesinde sütun diğer adını kullanamazsınız.

**Karakter Dizisi ve Tarihler**

- Karakter dizileri ve tarih değerleri tek tırnak içine alınır. •

- Karakter değerleri büyük/küçük harf duyarlıdır ve tarih değerleri biçime duyarlıdır. •

- Varsayılan tarih görüntüleme formatı DD-MON-RR'dir.

- Soyadı, iş\_kimliği, departman\_kimliğini SEÇİN NEREDE son\_adı = çalışanlardan

'BALİNA';

- Kiralama\_tarihi = '17-ŞUBAT-96' NEREDE çalışanlardan soyadı SEÇİN;

**Karşılaştırma operatörü**

- Bu operatörler, bir ifadeyi başka bir değerle karşılaştıran veya

ifade.

- where cümlesinde kullanılır

= - eşittir

42

> - daha büyük

>= - daha büyük veya eşit

< - daha az

<= - daha az veya eşit

<> -eşit değil (!= ve ^= aynı şey)

… .ve … Arasında

- SEÇİN soyadı, maaş ARASINDAKİ YERLERDE çalışanlardan maaş

2500(alt limit) VE 3500(üst limit) ;

**- ÖNCE ALT LİMİT BELİRLENMELİDİR**

**içinde** ( **üyelik koşulu** )

- VEYA koşulları kümesi

- Listedeki değerleri test etmek için IN operatörünü kullanın:

- SEÇİN çalışan\_kimliği, soyadı, maaş, yönetici\_kimliği çalışanlardan

WHERE manager\_id IN (100, 101, 201);

- Listede karakterler veya tarihler kullanılıyorsa, bunlar tek bir işaretle kapatılmalıdır.

alıntı işaretleri ('').

- Sorgu: Adları Andy ve Alex olan çalışanlardan ilk ad seçin

- ilk\_adı seçin

- çalışanlardan

- burada ilk\_ad ('Andy','Alex')

- ad adına göre sırala;

**- Alex masada değilse ne olur?**

**- Yalnızca Andy'yi yazdırır, başka bir şey yazmaz**

**Beğenmek**

- Geçerli arama dizisi değerlerinin joker karakterli aramalarını gerçekleştiren operatör

- Arama koşulları, değişmez karakterler veya sayılar içerebilir:

- %, sıfır veya birçok karakter anlamına gelir

- \_ bir karakteri belirtir

**'S%' GİBİ** first\_name YERİNDE çalışanlardan ilk\_ad SEÇİN

- Joker karakterleri birleştirme

- İki joker karakteri (%,\_) için değişmez karakterlerle birleştirebilirsiniz.

desen eşleştirme

- Soyadı seçin

- Çalışanlardan

- Last\_name **' \_o% ' gibi nerede ;**

**- Kaçış tanımlayıcısı**

- Gerçek % ve \_ sembollerini aramak için Kaçış tanımlayıcısını kullanabilirsiniz.

- çalışan\_kimliği, soyadı, iş\_kimliği'ni seçin

- çalışanlardan

- nerede job\_id **'%SA\\_%' Escape '** \' gibi;

43

**Boş**

**Is Null** operatörü ile operatör ve boş değerler için testler

- Soyadı, manager\_id'yi seçin

- Çalışanlardan

- manager\_id **null olduğunda;**

**- = ile test edemezsiniz, bc bir boş değer herhangi bir değere eşit veya eşit olamaz**

**Mantıksal operatörler**

**Ve**

- Her iki bileşen koşulu da doğruysa true döndürür

**Veya**

- Bileşen koşullarından herhangi biri doğruysa True döndürür

**Değil**

- Koşul yanlışsa true döndürür

- çalışan\_kimliği, soyadı, iş\_kimliği, maaşı seçin

- çalışanlardan

- job\_id **(** 'IT\_PROG','FI\_ACCOUNT');

**BETWEEN** gibi diğer SQL operatörleriyle de kullanılabilir ,

**GİBİ VE NULL**

- job\_id'nin **olmadığı yerde** ( ' AC\_ACCOUNT ' , ' AD\_VP ' )

- Maaşın 10000 ile 15000 **arasında olmadığı durumlarda**

- Last\_name ' &A% ' **gibi değil**

- komisyon\_pct'nin **boş olmadığı yer**

**Öncelik sırası**

44

ACCINNO -hatırlamanın kolay yolu

1. Aritmetik

2. Birleştirme

3. Karşılaştırma

4. (Not )null, like, in

5. Arasında değil

6. Eşit değil

7. Mantıklı değil

8. Ve

9. Veya

**Veya, Ve** sipariş önceliği

Sorgu: Bir çalışan başkansa ve 15.000 ABD Dolarından fazla kazanıyorsa veya

çalışan satış temsilcisidir ”

soyadı, iş\_kimliği, maaş seç

çalışanlardan

45

nerede job\_id='SA\_REP'

**veya** job\_id='AD\_PRES'

**ve** maaş>15000;

Sorgu: Bir çalışanın başkan veya satış temsilcisi olması durumunda satırı seçin ve

çalışan 15.000 dolardan fazla kazanıyor

soyadı, iş\_kimliği, maaş seç

çalışanlardan

nerede **(** job\_id='SA\_REP'

**veya** job\_id='AD\_PRES' **)**

**ve** maaş>15000;

**Maddeye Göre Sırala**

**-** Satırları sıralama

- ASC (varsayılan olarak) - artan (küçükten büyüğe)

- DESC - azalan (büyükten küçüğe)

**- Order By select ifadesinde SON HÜKÜM OLMALIDIR**

**-** Order By deyimi kullanılmazsa **, SIRALAMA SİPARİŞİ TANIMLANMAMIŞTIR.**

- Ayrıca, satırların döndürülüp döndürülmeyeceğini belirtmek için **NULLS FIRST** veya **NULLS LAST kullanabilirsiniz.**

sipariş sırasında **ilk veya son görünmelidir**

First\_name seçin

çalışanlardan

ada **göre sırala desc nulls last** ;

Sıralama (artan)

- Tarih değerleri ilk önce en erken değerle görüntülenir (01-JAN-92 görüntülenir

01-OCAK-95'ten önce

- **Boş değerler , Artan** diziler için en son ve **ilk olarak** görüntülenir .

**Azalan** diziler

- Sırayla sütun takma adını kullanabilir

**Sütunun sayısal konumunu kullanarak sıralama** :

soyadı, iş\_kimliği, departman\_kimliği, kiralama\_tarihi seçin

çalışanlardan

**3** ile sipariş ;

- 3 kiralama\_tarihi

**Birden çok sütuna göre sıralama:**

Ör: Soyadı, departman\_kimliği, maaşı seçin

çalışanlardan

**departman\_kimliğine göre sırala, maaş açıklaması;**

- Department\_id'yi artan şekilde sıralar

- Maaş azalan sıralar

46

**İkame değişkeni (daha sonra bahsedeceğiz)**

**Fonksiyonlar**

- Fonksiyonlar bazen argüman alır ve her zaman bir değer döndürür

**Tek satırlı SQL işlevleri**

- **Yalnızca tek sıralarda çalışın**

- Satır başına bir sonuç döndür

- Her satır için bir sonuç döndürür

- Tek sıralı fonksiyon türleri

- Karakter

- Sayı

- Tarih işlevi

- Tüm tarih işlevleri, DATA veri türünden bir değer döndürür.

Bir sayı döndüren MONTHS\_BETWEEN işlevi

- Dönüştürmek

- Genel

- SELECT, WHERE AND ORDER BY yan tümcesinde kullanılabilir;İç içe yerleştirilebilir

- Sütun veya ifade olabilen argümanları kabul eder

**Karakter işlevleri**

**- Büyük/küçük harf dönüştürme işlevleri**

- DAHA DÜŞÜK

- Alfa karakter değerlerini küçük harfe dönüştürür

Örnek: alt seç('HAA')

ÇALIŞANLARDAN;

- ÜST

- Alfa karakter değerlerini büyük harfe dönüştürür

Örnek: üst ('haaaa') seçin

çalışanlardan;

- INITCAP

- Her kelimenin ilk harfi için büyük harf; diğer tüm harfler küçük

Örnek: initcap('haa ha') seçin

çalışanlardan;

**- Karakter manipülasyon fonksiyonları**

- BİRLEŞTİR(m,n)

- Birinci karakter değerini ikinci karakter değerine birleştirir; ile aynı

birleştirme operatörü(||)

- m negatif ise sayma karakterin sonundan başlar

değer

- n atlanırsa, dizenin sonuna kadar olan tüm karakterler döndürülür

47

- SUBSTR(string,m,n)

- Başlangıç karakteri konumundan belirli karakterleri döndürür, m,n

karakter uzunluğu

- UZUNLUK

- İfadedeki karakter sayısını döndürür

- INSTR(m,n)

- Adlandırılmış bir dizenin sayısal konumunu döndürür

- İsteğe bağlı olarak aramaya başlamak için bir m konumu sağlayabilirsiniz ve

dizenin oluşumu n

- M ve n varsayılan olarak 1'dir

- LPAD|RPAD

- Bir ile n karakter uzunluğunda sola dolgulu bir ifade döndürür

karakter ifadesi

- n karakter uzunluğunda sağ dolgulu bir ifade döndürür.

karakter ifadesi

- KIRMIZI

- Bir karakter dizesinden baştaki veya sondaki karakterleri (veya her ikisini) kırpın

- trim\_character veya trim\_source bir karakter değişmeziyse, kullanılmalıdır

tek tırnak işareti

- YER DEĞİŞTİRMEK

- Bir karakter dizisi için bir metin ifadesi arar ve bulunursa onu değiştirir

belirtilen bir değiştirme dizesi ile.

48

49

**Sayı Fonksiyonları**

- Yuvarlak(sütun|ifade, n)

- Değeri belirli bir ondalık sayıya yuvarlar

- n atlanırsa, ondalık basamak yok (n negatifse, ondalık basamağın solundaki sayılar

yuvarlatılmış)

- Tur (45.926.2) = 45.93

- Trunk

- Değeri belirli bir ondalık basamağa keser

- n atlanırsa, n varsayılan olarak sıfır olur

- Kesik (45.926.2) = 45.92

- Mod

- Bölmenin kalanını döndürür

- Mod(1600, 300) =100

**Tarihlerle Çalışmak**

- Varsayılan tarih görüntüleme formatı **DD-MON-RR'dir**

- Sistem tarihi işlevi

- Tarih ve saati döndürür

**sysdate'i** seçin

ikiliden;

**Tarihli Aritmetik**

, ( **sysdate-hire\_date)/7'yi** hafta olarak seçin

çalışanlardan

departman\_id=90;

**- Tarih Manipülasyon İşlevleri ( tüm tarih işlevleri, HARİÇ** DATE veri türünü döndürür

SAYISAL BİR DEĞER OLAN **MONTHS\_BETWEEN )**

- AY\_BETWEEN(tarih1,tarih2)

50

- tarih1 ile tarih2 arasındaki ay sayısını bulun

- tarih1 tarih2'den sonraysa sonuç pozitiftir

- tarih1 tarih2'den önceyse sonuç negatiftir

- iki tarih arasındaki ay sayısı

- ADD\_MONTHS(tarih;n)

- Bugüne kadar n takvim ayı ekler

- n'nin değeri bir tam sayı olmalıdır ve negatif olabilir

- Tarihe takvim ayları ekleyin

- NEXT\_DAY(tarih, ' char ' )

- Takip eden haftanın belirtilen tarihin sonraki günü

- LAST\_DAY(tarih)

- Ayın son günü

- YUVARLA(tarih[, ' fmt ' ])

- fmt biçim modu tarafından belirtilen birime yuvarlanmış tarihi döndürür

- fmt atlanırsa, tarih en yakın güne yuvarlanır

- TRUNC (tarih[, ' fmt ' ])

- Günün saat bölümünün, belirtilen birime kesildiği tarihi verir.

biçim modeli fmt tarafından belirtilen

- fmt atlanırsa, tarih en yakın güne kısaltılır

**Çok sıralı işlevler (grup işlevleri)**

- Satır kümesi başına bir sonuç döndür

- Her satır kümesi için bir sonuç döndürür

Grup işlevleri

- Grup başına bir sonuç vermek için satır kümeleri üzerinde çalışın

Grup İşlevlerinin Türü (boş değerleri yok sayar)

-ORT

- SAYMAK

- saymak(\*)

- Satır sayısını döndürür

- Yinelenen satırları içerir

- Boş değerler içerir

- say(ifade)

**Null olmayan değerlere** sahip satır sayısını döndürür

- MAKS

- MIN

- TOPLA

- DEĞİŞİKLİK

- STDDEV

51

**Grup Fonksiyonları ve Null Değerler**

**Veri Grupları Oluşturma**

- Maddeye göre gruplandır

**Birleşim Türleri**

Doğal birleşimler

- Doğal birleştirme maddesi

- Aynı ada sahip iki tablodaki tüm sütunlara göre.

- Tüm eşleşen değerlerde eşit değerlere sahip iki tablodan satırları seçer.

sütunlar

- İsimler değişirse sorgu çalışmayı durdurur

- Yeni sütunlar eklenirse sorgu çalışmayı durdurabilir

- Sütunlarda farklı veri türü varsa, bir hata döndürülür.

- Doğal birleştirme kullanılması kesinlikle TAVSİYE EDİLMEMEKTEDİR

- Cümleyi kullanma

- On cümlesi

**İç birleşim**

Dış birleşimler:

- Sol dış katılma

**Soldaki** tablodan eşleşen veriler ve eşleşmeyen veriler (bu şu anlama gelir:

**sağ** taraftaki değerler boş olacak)

- Sağ dış birleştirme

**Sağ tablodan** eşleşen veriler ve eşleşmeyen veriler (bu,

**sol** taraftaki değerler boş olacak)

- Tam dış birleştirme

**Her iki** tablodan eşleşen veriler ve eşleşmeyen veriler

Çapraz birleşimler

- Tabloları birleştirir ve her tablonun toplam satırlarını çoğaltır

- Örnek- A tablosunun 20 satırı ve B tablosunun 5 satırı vardır

- Çapraz birleştirme size 100 satır verecektir

**Operatörleri ayarla**

**- Birlik**

**- Birlik tüm**

**- Eksi**

**- kesişme**

- Takip edilecek kurallar;

- 2 bağımsız sorguya ihtiyaç var

- Select deyiminde **aynı sayıda sütun**

- Aynı sırada aynı veri türü

52

- İlk isim ise, maaş

- İkinci ad, ad, maaş olmalıdır

- İkinci maaş ise, ad\_adı = **veri türü farklı olduğu için çalışmaz**

**Birlik**

- Birden çok sorgudan veri alır

**- Yinelenenleri kaldırır ve sonucu sıralar**

**- Boş değerleri kaldırır**

(departman\_kimliği, departman\_adı seçin

bölümlerinden)

**Birlik**

(çalışan\_kimliği, soyadını seçin

çalışanlardan);

- 2 bağımsız sorgunuz daha olabilir

- Sadece birlik koyun

**Birlik tüm**

- 2 bağımsız sorgudan birleştirilmiş satırları döndürür

- Kopyaları tutar ve sonuç sıralanmaz

**Eksi**

2. sorguda **OLMAYAN** 1. sorgudaki kayıtları döndürür

- İLK SORGU KARAR VERİR

- 1. sorgunun sonuçlarını alacak ve 2. sorgu ile karşılaştıracak ve yalnızca

2. sorguda görünmüyor.

**- İki sorgu arasındaki farkı bulmanıza yardımcı olacaktır - Eksi Operatörü**

- İlk sorgu - **Tahta Kaşık** , Elma, Muz, **Salatalık**

- İkinci sorgu - Elma, Muz, Kivi, Kabak

- Sonuç = Tahta Kaşık, Salatalık

Sorgu: Tüm çalışanlar ve departmanlar için emp kimliklerini, dep kimliklerini, dep adlarını görüntüleyin

**Eksi**

Yalnızca tüm çalışanlar ve departmanlar için emp kimliklerini, dep kimliklerini, dep adlarını görüntüleyin

çalışanı olan departmanlar

(e.employee\_id, e.department\_id, d.department\_name seçin

çalışanlardan e tam dış birleştirme departmanlarından d

e.department\_id=d.department\_id üzerinde)

**Eksi**

(e.employee\_id, e.department\_id, d.department\_name seçin

çalışanlardan e departmanlara katıl d

e.department\_id=d.department\_id üzerinde);

53

**DataBase1(MYSQL)'den DataBase2(Oracle)'a Veri Taşıma**

- Her ikisinde de aynı veritabanı şeması

- MYSQL'de TableA

- ORACLE'da TableB

- İki tablonun aynı sütunlara sahip olduğundan nasıl emin olabilirsiniz?

TableA'dan \* seçin

**EKSİ**

TableB'den \* seçin;

**- Aynı verilerse, sonuçlarda hiçbir şey görünmemelidir**

**- Bir fark varsa, bu kayıtlar gösterilecektir.**

**kesişme**

Her iki sorguda da yalnızca **ortak satırları tutar**

- Kopyalar varsa ondan kurtulur

- 1. sorgu: A,B,C

- 2. sorgu:A,C,D

- Sonuç: A,C

Çalışan\_kimliği, soyadını seçin

çalışanlardan

nerede çalışan\_kimliği (100.104.105)

**kesişmek**

Çalışan\_kimliği, soyadını seçin

çalışanlardan

burada çalışan\_kimliği (100,104,106);

DML (Test kullanıcıları genellikle bunu yapar)

- Seçme

- Sokmak

- Güncelleme

- Silmek

DDL

- Oluşturmak

- Değiştir

- Düşürmek

- Yeniden isimlendirmek

- kes

- Yorum

DCL

- Hibe etmek

- Geri çekmek

İşlem (bazen güncellememiz ve silmemiz gerektiğinde kullanılır)

- İşlemek

54

- Geri alma

- Kaydetme noktası

SQL ile deneyiminiz var mı?

- Evet, ilişkisel veritabanlarıyla çalıştım ve DDL ile çok rahatım ve

DML komutları

Oluştur - tablo oluşturur

**tablo oluştur** Stud

(

Student\_id numarası(5)birincil anahtar,

last\_name varchar2(30) boş değil,

ders\_kimliği numarası(4) boş

);

O zaman Stud'u **tarif et**

- Size Stud tablosunun ayrıntılarını gösterir

Yapın **: tablo\_adı seçin**

**user\_tables'dan;**

- Size mevcut tüm tabloları gösterir

**Sokmak**

**Stud'a** ekle

değerler(200,'Jones',101);

Kayıtları güncellemek için update komutu kullanılır.

( **GÜNCELLEMEYİ KULLANIRKEN DİKKATLİ OLUN. ŞARTLI BİR AÇIKLAMAYA İHTİYACINIZ OLMALIDIR.**

**TABLODAKİ HER ŞEYİ GÜNCELLEME**

**Veritabanı tetikleyicileri** denen bir şey var , bir kayıtta değişiklik yaparsanız,

diğer tablolarda da değişiklik yapın

saplamayı **güncelle**

Course\_id=102 olarak ayarla

burada last\_name='Jones';

**Geri alma** - eklemeden önce geri döner

- Ctrl-z gibi

**Savepoint** - bir noktayı kaydeder

**Taahhüt** : kullanıldığında, taahhüt ifadesinden önce geri alamazsınız

55

geri alma;

**Stud'a** ekle

değerler(200,'Jones',101);

**taahhüt** ;

--Kaydetme noktası

**savepoint before\_delete;**

**Sil** - veritabanı tablosundaki verileri silin

saplamadan **sil**

burada öğrenci\_id=200;

**savepoint'e geri alma Before\_delete;**

İşin püf noktası, önce **kaydetme noktası ,** ardından kaydetme **noktası adına geri dönmektir .**

**\*\*\*DDL'yi geri alabilirsiniz, ancak DML'yi geri alamazsınız**

Düşür vs Kes

- Her ikisi de Veri Tanımlama dili komutlarıdır

- **Drop** , verileri ve tabloyu da kaldırır

- **Kes** , tüm verileri kaldırır ancak boş tabloyu tutar

- Hiçbiri geri alınamaz

**kes** saplama

- Tablodaki tüm verilerden kurtulur ancak tabloyu tutar

**Düşürme** masası Saplama

- Verileri ve tabloyu kaldırır

**Değiştir** vs **güncelleme**

- **Değiştir** veritabanı yapısını değiştirir

- Tablo adını değiştirme

- Veri tipleri

- **Güncelleme** , bir veri satırını günceller

- Verilerle anlaşmalar

56

**EN DÜŞÜK MAAŞ NASIL BULUNUR.**

**İlişkili alt sorgu kullanmamız gerekiyor.**

SEÇ \* çalışanlardan e1

NEREDE n= (ÇALIŞANLARDAN SAYI SEÇİN(FARKLI(maaş)) e2

NEREDE e1.maaş <= e2.maaş);

**API**

**API** nedir ?

- Uygulama programlama Arayüzü

- Bir uygulamadan (istemci tarafı) (DB) istek alan ve

başka bir uygulamaya çevir(sunucu tarafı)(Veritabanı)

**Web Hizmetleri Nedir?**

- API internetten geçiyor

- İnternet üzerinden veya yerel olarak çalıştırılabilir

**Projenizde hangi Web Servislerini kullanıyorsunuz?**

- Temsili Aktarım Durumu olan Restful kullanıyorum ve XML ile iletişim kuruyor

ve JSON ama şu anki projem JSON kullanıyor

**JSON nedir?**

- JavaScript Nesne Gösterimidir

- Temelde hafif bir XML sürümü

- Anahtarda: Değer biçimi

- Anahtar her zaman çift tırnak içindedir ve eğer dizi çift tırnak içindeyse ve sayılar hayır ise değer

tırnak

- Tamamen http protokolüne dayalıdır - bu nedenle tarayıcıdaki bağlantıya ulaşır ve sonuçları görür

**Nasıl ve nereye istek gönderiyorsunuz?**

- Rest kullandığım için uç noktaları var. Geliştiricilerim genel URL'ler ve istekler oluşturuyor

o url'ye gönderilir

**Web hizmetleri dışındaki herhangi bir API kullanıyor musunuz?**

- Tarayıcı için Selenium API, veritabanı için JDBC ve API için RestAssured kullanıyorum

**Projenizde API'yi nasıl test edersiniz?**

- Mevcut projemde sadece şirketlerimizin api'sini değil, diğer harici api'leri de test ediyoruz.

Örneğin, yetkili son kullanıcının bilgilerini kolayca bizim için transfer etmek için linkedIN api kullanıyoruz .

veri tabanı. Bir test kullanıcısı olarak bir API isteği gönderir ve durum kodunu, yanıt gövdesini doğrularız.

ve api url'sinin uç noktalarını kontrol etmek beklendiği gibi çalışıyor

- Örneğin projemde API'nin Pozitif/Negatif testini de yapıyorum.

57

- Olumlu - geçerli istekler, başlıklar, parametreler ve json gövdesi gönderiyorum ve

yanıtın 200/201 olduğunu doğrulayın

- Olumsuz- geçersiz istekler, başlıklar, parametreler ve gövde gönderiyorum,

durumun 200 olmamasını bekliyorum

**API'niz için API dokümantasyon web siteniz var mı?**

- Evet, api belgelerimiz için swagger kullanıyoruz ve burada açıklama ve

API uç noktalarının yönergeleri

**Tüm API uç noktaları, tüm Http protokollerini kullanabilir mi?**

- Bu url'nin GET, POST, PUT veya DELETE ile çalışıp çalışmadığına API geliştiricim karar verir.

istekler

**API'nizi manuel olarak nasıl test edersiniz?**

- Postacı kullanıyorum - bu, REST API URL'sini test eden bir REST API istemci aracıdır

**Rest API'deki İstek türleri nelerdir?**

- Al, Gönder, Koy ve Sil istekleri var

- Okuma verilerini alın

- Gönderi veri oluşturur

- Güncelleme verilerini koy

- Sil verileri siler

**REST API'deki başlıklar nelerdir?**

- Kabul Et.(Content Type.JSON) türünü kullanıyorum - ne aldığımı kontrol ediyor

JSON veya XML biçimi

- Ve ContentType.(Contenttype.Json) - gönderdiğim şeyin JSON'da olması gerektiğini kontrol ediyor

biçim

**RestAssured Kitaplığı nedir?**

- BDD biçiminde olan ve kullanarak java kodunu entegre etmeye yardımcı olan web hizmeti olmayan bir API

json'dan veri çıkarmak ve bir java'ya dönüştürmek için seri hale getirme ve seri hale getirme

Verileri beklenen veriye depolamak, doğrulamak ve doğrulamak için nesne.

**Projenizde Enum'u nasıl kullanıyorsunuz?**

- Yanıt türümün JSON biçiminde olduğundan emin olmak için içerik Türü kullanıyorum

**Yanıt verilerinin boyutunu doğrulamak için hangi yöntemleri kullanıyorsunuz?**

- Hamcrest'ten Matchers kullanıyorum

- hasitemler()

- eşittir()

**JsonPath nedir?**

Yanıt gövdesini doğrulamanın başka bir yolu

-JsonPath j=yanıt.jsonpath;

**100 limitli parametreye ve yol parametresine ihtiyacım olan bir yöntemi nasıl yazarım?**

**çalışan kimliği = 100?**

- Yazardım ;

- .and().params("sınır",100)

- .and().pathParams("çalışan\_kimliği", 110)

58

**Response arayüzünü nasıl kullanırdım?**

**Raporlama**

Mvn Verify, başarısız olsa bile testleri çalıştırır (arızayı yok sayar)

- Tüm testin bitmesini bekler

- Yapı yapılandırmamızda buna sahip olduğumuz için başarısızlığı yok sayar

<testFailureIgnore>doğru</testFailureIgnore>

-

- Doğrulama, testten sonra gelen bir Maven yaşam döngüsüdür

Bir şey başarısız olursa Mvn testi testi çalıştırmayı durduracak

Aldığımız orijinal html raporu o kadar iyi değil, istatistiksel verilere ihtiyacımız var

Eski; "html:hedef/salatalık raporu"

- Kaç testin başarılı/başarısız olduğu gibi yüzdeler

fazlası için Salatalık Sandviçini kullanacağız **(bu, pom xml'de bir bağımlılık dosyasıdır)**

istatistiksel veri raporlama

cukesrunner'da ekleyin; "json:hedef/salatalık.json"

- Bir JSON dosyasından bir html raporu

- Bu raporun nasıl çalıştığı, raporu oluşturmak için Json dosyasının kullanılmasıdır.

- Sürüm 3.15 (videodan)

- Pom'a yeni bir yapı xml ekleyin (zaten pom dosyanızda, TestProject adındaki dosyada)

- Yalnızca bu json raporlamasını alacaksınız (grafikler ve istatistikler içeren salatalık raporu.

Bu rapor yalnızca SİZİN görmeniz için yerel olacaktır, Jenkins için değil) YALNIZCA yürütürseniz

MVN Doğrulama

- AMA HER ZAMAN BİR **JSON DOSYASI ALACAKSINIZ(salatalık raporundan farklı)**

TEST YAPTIKTAN SONRA, BİLE DOĞRULANDI

- Bu JSON dosyası, Jenkins için çok önemlidir - salatalık raporu için

Eklenti

- TestProject yapısı:

- <id>yürütme</id>

- <phase> **doğrulama** </phase> - bu nedenle html (json) raporu yalnızca

doğrulama kullanarak

- <hedefler>

- <goal>oluştur</goal>

</hedefler>

- Rapor ayrıca size bir json dosyası verecektir.

- Doğrulamayı kullanarak testleri çalıştırmak için, pom dosyasına sağ tıklayın ve maven derlemesine tıklayın …

- Ayrıca parametreler de ekleyebilirsiniz (koşucu değişkeni ve xml dosyası olan değer gibi)

- Hedefleri yazın: doğrulayın

Bunu komut satırında çalıştırmak için

- pom dosyasının konumuna gidin ve **mvn doğrulama yazın**

- Sözdizimi mvn<lifecycle/hedef> şeklindedir

59

mvn doğrulama kullanarak Yürütme Sırası

1. pom dosyasına karşı çalıştırın

2. Pom dosyası xml dosyasını çalıştırır

3. Xml, cukesrunner dosyasını çalıştırır

4. Cukesrunner salatalık özellik dosyasını/testini çalıştırır

json salatalık raporu ekran görüntüsü gösteriyor mu?

**Jenkins**

- Sürekli Entegrasyon ve Dağıtım aracı

- Jenkins'in 3 bileşeni

1. Kod değişikliği

a. Devs, uygulama kodunda değişiklik yapar

2. Testi

a. CI aracı, değişiklikleri otomatik olarak alır ve uygulamayı test eder

3. Dağıt

a. CI aracı, uygulamayı değişikliklerle birlikte dağıtır

**Jenkins'in işi nedir**

- Her şey bir iş yaratarak yapılır

- Jenkins'in programına göre gerçekleştirdiği bir görev

- Birkaç adımdan yapılmıştır

- Ne zaman çalışacağını belirleyen bir tetikleyiciye sahip olabilir

- Çalıştırmanın sonuçlarını otomatik olarak raporlar

**Sürekli Entegrasyon nedir?**

- Yazılımın kodu her değiştiğinde, otomatik olarak kurulur ve test edilir

**Sürekli dağıtım nedir?**

- Kod değişiklikleri otomatik olarak oluşturulur, test edilir, dağıtılır ve bir sürüm için hazırlanır.

üretme

**Jenkins'i koruyor musun?**

- Özel Ekip, çevre ekibi, mimari ekip tarafından yapılır.

- Ama onlara testlerim ve yapılandırma bilgilerim hakkında bilgi ve ayrıca e-postaları veriyorum.

bildirimler gönderin.

- Git yolu

-Mvn kodu; hedefler - derleyin veya doğrulayın -drunner=xml, vb.

- Belirli testler için zaman çizelgesi

Regresyon testinizi jenkins üzerinde çalıştırırsanız, otomatik olarak çalışırsa nasıl çalıştırırsınız?

**Regresyon testinde neler oluyor?**

- Ekibe, test planına, kapsama ve iş değerine bağlıdır

**Jenkins'te kaç iş var?**

- Otomatik testler için kişisel olarak 2-3 iş kurdum

- duman için 1

60

- Duman günde 2,3 kez çalışıyor, tüm

ortamlar çalışıyor ve çalışıyor

- Tam regresyon (manuel ve otomatik testlerin çalıştırılması)?

- Her üretim çıkışından önce (3 sprintten sonra)

- Sadece çok kararlı test durumları tam regresyonda saklanır

- İşlevselliği güncelleme

- Küçük gerileme

- Sprint sonunda koşar

- Belirli modüller ve işlevlerle ilgili testler

- Hangi modülün çalıştırılacağını belirtmek için etiketleri kullanıyorum

**Kod ortamınıza nasıl dağıtılır?**

- Geliştiriciler kodu yazar, test eder, ardından dev'den test ortamına jenkins'te dağıtılır

- Ya olmazsa?

- Geliştiricinizle konuşun ve dağıtmasını isteyin

**Etiketlere göre nasıl arama yaparsınız?**

**Eclipse'de** arama aracını açmak için ctrl-h'yi kullanabilir ve etiket adını buraya koyabilirsiniz ve

ne tür bir dosya, feature.file gibi

**selenyum ızgara**

- Otomatik testlerinizi farklı tarayıcılarda (ve farklı tarayıcılarda) çalıştırma olanağı verir.

sürümleri) ve platformlar (temelde İşletim sistemleri ve sürümleri. Pencere, Linux,

Mac) VISGRID

- Bu araç, çok sayıda Testiniz varsa (500'den fazla) kullanışlıdır.

- WebDriver driver=new Chromedriver() yapmak yerine

- **WebDriver sürücüsü yapın=yeni RemoteWebDriver(url,yetenekler)**

- Yapıcılarda 2 parametre içerir

**- Selenyum Izgarası nasıl çalışır?**

- Grid, Hub ve düğümlerden oluşan bir kurulumdur.

**- Hub** , tüm düğümlerin bağlandığı merkezi bir makinedir.

- **IP adresi ve port numarası var** , Ardından Hub'ı Düğümlere bağlarsınız

- Hub olarak adlandırılan bir ana makineye ve birden çok düğüme (bu makineler

aslında testlerinizi çalıştırın)

- Yürütme sırası;

- Kodunuz>uzak sürücü>Selenium Hub>Selenium düğümleri (belki

birden fazla)

- Selenium Hub'ınızdan testlerinizi çalıştırmasını istediniz

- Ardından Selenium hub, hub'a bağlı bir düğüm bulacaktır.

ve testinizi oradan çalıştırın

- İstediğiniz kadar düğüme sahip olabilirsiniz, ancak yalnızca bir hub'a sahip olabilirsiniz.

**Selenyum ızgaranızı nasıl kurarsınız?**

- Hub farklı bir sunucu makinesidir ve her düğüm ayrı bir sunucu makinesidir

- Merkezimiz ve düğümlerimiz **Amazon AWS Ec2 makinelerinde kurulacak (ideal olarak)**

61

- Hub'ınızdan testler yapmasını isteyebilirsiniz ve ardından hub bir düğüm bulup testinizi çalıştıracaktır.

Buradan

- Aynı makine veya sunucuda hub ve birden fazla düğüme de sahip olabiliriz.

- Her düğüm belirli bir konfigürasyonla HUB'a kaydolur ve HUB bunun farkındadır.

düğümde mevcut tarayıcı

- Belirli bir tarayıcı için HUB'a bir istek geldiğinde (İstenen yeteneklerle birlikte)

nesne), HUB, istenen tarayıcı için bir eşleşme bulunursa, çağrıyı \*buna\* yönlendirir.

belirli GRID Düğümü ve ardından çift yönlü olarak bir oturum kurulur ve yürütme

başlar

**Çapraz tarayıcı/platform nerede çalışıyor?**

- Jenkins'in kurulu olduğu Amazon AWS makinesinde çalışmaktadır. Ama normalde,

şirket jenkins, geliştiriciler ekibi, devops, dağıtım ekibi ve QA tarafından kullanılır

otomasyon ekibi

- Orada tavsiye edilmiyor

- İdeal olarak Jenkins>Github>Maven>Runner class>Selenium Hub'dır ve aşağıdakilerden birinde çalıştırın

farklı sunucuda yapılandırılmış düğümler

- Kanca sınıfınızda, webDriver'ınızı yerel bir sürücüye yönlendirmek yerine,

onu **RemoteDriver() olarak değiştirin;** ve sahip olduğu bulut makinesine işaret etmesini sağlayın

Merkez

**- Çoklu tarayıcı testindeki zorluklar nelerdir?**

- Bir şey tıklamıyor

- Görünmez

- Bazı öğeler bir tarayıcıda diğer tarayıcıdan farklı görünüyor

**- Hangi tarayıcıları test ediyorsunuz?**

- Krom

-FireFox

- IE/Kenar

- Safari

- Opera

**- Çoklu tarayıcı testini nasıl otomatikleştirirsiniz?**

1. Çerçevemdeki özellikler dosyamda tarayıcıyı başka bir şeyle değiştirin

- Testlerimi farklı tarayıcıda çalıştırmak istediğimde

- Eski; “ Browser=chrome ” dan “ Internet Explorer ” a

**- Bu yöntem, testleriniz 500 testten azsa işe yarar**

2. Çerçevemde Selenium Grid uyguladım ve testleri farklı bulutlarda çalıştırabilirim

farklı tarayıcılar kullanan makineler

**Başarısız testler ile ne yapmalı?**

- Otomasyon yürütme raporuna bakın

- Başarısızlığın nedenini öğrenin

- Adımları manuel olarak yapmaya çalışın, manuel geçiyorsa otomasyon sorunu

- Yani düzeltirsin, yeniden çalıştırırsın ve geçip geçmediğini görürsün

62

- Uygulama sorunu varsa

- Bir kusur oluştur

- Kusur düzeltilirken, **Ad-hoc testi kullanarak manuel olarak test ediyorum**

- Kusur bir **gösterici değilse,** o zaman başka testler yapar ve otomatikleştirirsiniz .

- Öyle ise beklemeniz gerekir, başka test yapamazsınız.

- Yeniden çalıştırırken, yalnızca yeniden çalıştırmak istediğiniz testi test etmek için @ReRun etiketini kullanıyorum

**Ad-Hoc testi nedir?**

- Uygun planlama ve dokümantasyon olmadan gerçekleştirilmiştir

- Bu yöntem kullanılarak bulunan kusurların tekrarlanması zordur çünkü hizalanmış test durumları yoktur

bu senaryolar için

- Resmi test uygulamasından sonra gerçekleştirildi

**SauceLabs - bulut Izgara hizmeti. Birden çok tarayıcıyla birden çok Platforma erişim**

- Bir sürü çalıştırabilmeniz için bulut makinesi sağlar

- Paralel testi iyi yapıyor

**SauceLabs ile nasıl test yapılır?**

Genellikle “ WebDriver driver=new FireFoxDriver();

Şimdi yapacağımız;

- DesiredCapabilities caps=DesiredCapabilities.firefox();

caps.setCapability ( " platform " , " Windows 7 " );

caps.setCapability( “ sürüm ” , “ 38,0 ” );

WebDriver sürücüsü=yeni RemoteWebDriver(yeni

URL( " http://YOUR\_USERNAME:YOUR\_ACCESS\_@ondemand.saucelabds.com:80/wd/hub ” ,ca

not)

**Testlerinizi soslabs/herhangi bir hazır selenyum ızgarasında çalıştırma adımları**

1. İstenen Capabilities nesnesini oluşturun ve istediğiniz işletim sistemi türü, tarayıcı ile belirtin.

çalıştırmak için testler. Selenyum Izgara ile birlikte kullanılır.

a. DesiredCapabilities caps=DesiredCapabilities.firefox();

caps.setCapability ( " platform " , " Windows 7 " );

caps.setCapability( “ sürüm ” , “ 38,0 ” );

**HUB url'si** ile RemoteWebDriver oluşturun :

WebDriver sürücüsü=yeni RemoteWebDriver(yeni URL( **URLOFHub), büyük harfler** );

3. soapLabsDemo sınıfı oluşturun

- @BeforeTest'te

- DesiredCapabilities(Selenium'dan gelir)

caps=DesiredCapabilities.firefox();(hangi tarayıcıyı seçer)

caps.setCapability ( " platform " , " Windows 7 " );

caps.setCapability( “ sürüm ” , “ 38,0 ” );

Sicim

URL= ” http://YOUR\_USERNAME:YOUR\_ACCESS\_@ondemand.saucelabds.com:80/wd/hub ”

- Bu Selenium Hub adresi

**URI= birleşik kaynak tanımlayıcısı**

63

**URL=birleşik kaynak bulucu**

**SauceLabs kullanarak nasıl rapor verirsiniz?**

- JIRA sunucuma bağlanıyorum

**AWS'de Selenium Grid'i nasıl kurarsınız?**

1. 2 bulut sunucusu (Ec2)

a. 1 HUB olacak

b. 1 düğüm olacak

2. 1. makinede **Selenium StandAloneServer'ı indirin**

a. Remote Selenium WebDriver'ı çalıştırmak için gereklidir.

i. Çok fazla yapılandırma

ii. Komut satırını kullan

1. HUB'ı kurun; java -jar selenyum-sunucu-bağımsız-3.5.3.jar - rol

merkez

a. http://localhost:444/grid/console

2. Düğüm1'i kurun

a. Java -jar selenyum-sunucu-bağımsız-3.5.3.jar -rol düğümü -

hub http://locahost:4444/grid/register

3. Düğüm2

a. Java -jar selenyum-sunucu-bağımsız-3.5.3.jar -rol düğümü

-hub http://localhost:4444/gird/register -port 7777

**3. Visgrid'i de kullanabilirsiniz**

a. 1. makinede indir

i. jar dosyasını aç

1. Maksimum oturumu 10'a ayarlayın

2. Merkezi başlat

3. Port 4444 (hub'ınızın olduğu yer. Değiştirebilirsiniz ama hatırlayın)

ii. Düğüm oluştur'u tıklayın

1. Düğüm için bir tarayıcı seçin

2. Örnek sayısını yazın

3. Ekle'ye tıklayın

a. Tarayıcıyı ec2 makinesinde açın

b. Tür: localhost:4444 (bu SeleniumGrid hub'dır) (Şimdi

sadece HUB'ımızı çalıştırdı)

c. konsola tıklayın

d. Sayfayı yenile

e. Şimdi düğümleri göster (hepsi aynı makinede)

f. Başka bir düğüm ekle (şimdi 2 düğüm gösterecek)

iii. Şimdi başka bir makineye gidin ve oradan düğümler oluşturun ve onu

HUB

iv. 2. makineye geçmeden önce

1. aws konsoluna gidin

2. Hub'ınızı tutan örneğe gidin

a. Güvenlik gruplarına tıklayın = sihirbazı başlat

64

b. Gelene git

i. Hub'ın bağlantı noktası numarasını (4444) ekleyin ve

Onu kurtar

v. 2. makineye gidin

1. 1. makine için kullandığınız visgrid'in aynı jar dosyasını indirin

2. jdk'yi indirin (dosyayı açmak için)

3. Kavanozu açın> hub'ı başlat> düğüm oluştur> HUB'ı Geçersiz Kıl'a tıklayın (biz

ilkinde bir tane daha olduğu için bu göbeğe ihtiyacımız var

makine)>makine 1 ip adresi ve bağlantı noktası numarası ekle: 4444>ekle

**Testlerinizi paralel olarak birden çok iş parçacığında nasıl çalıştırıyorsunuz?**

- İş parçacığı, bir işlem veya uygulama çalıştırma örneği gibidir

- 4 yol

1. Farklı etiketlerle birden fazla cukesrunner oluşturabiliriz

a. Eski. cukerunner'da “ @Test ” etiketi var

b. Smokerunner'da @smoke var

c. Regresyon koşucusu @Regression'a sahiptir

2. Testng xml oluşturun ve bu koşucu sınıfını tek bir testin altına ekleyin

a. Tüm 3 koşucuyu bir xml'ye ekleyin

b. Sonra verbose=2(xml dosyasının üstüne) yanına “ paralel = ” sınıfları ekleyin.

iş parçacığı sayısı = ” 10 ” >

3. Ardından, sürücümüzün hub'a giden uzak bir Web Sürücüsü açtığından emin olun.

a. Kodu sürücü sınıfına ekleyin

4. Testng xml'yi kendi başına veya maven kullanarak çalıştırın.

**Linux Komutları (büyük/küçük harfe duyarlı)**

- **yeniden başlat**

- sistemi yeniden başlatır

- **adam**

- Size komutun talimatını verir

- Ör: “ **adam yeniden başlat** ”

**- mikdir**

- Dizin (klasör) oluşturur

**- cd**

- Dizini değiştir

**- ls**

- Dizin içeriğini listele

**- pwd**

- Geçerli çalışma dizininin adını yazdır

- Size tam konumu verir

- Ör: /home/Andy/Desktop

**- LL**

65

- Uzun liste formatı

**- ls -la**

- Dosyaları ve gizli dosyayı yazdırır

**- açık**

- Temiz ekran

**- cd..**

- Üst dosyaya gider (kök dosyaya değil)

**- cd /**

- Üst kök dosyaya gider

- **cd ~**

- Kullanıcı dosyasının ana sayfasına gider

**- grep**

- Bir desenle eşleşen bir çizgi yazdırır

**- df - h**

- Disk alanı kullanımını yazdırır

**- tepe**

- Linux görevlerini görüntüler (görev yöneticisi gibi)

**- Bir hesap nasıl oluşturulur**

- Kullanıcı

**- kullanıcı Andy'yi ekle**

- Grup

**- grupCbertek'i ekleyin**

- Gruba kullanıcı ekleme

**- useradd -G Cybertek Andy**

**- Andy'nin kimliği**

- bu kişinin ayrıntılarını yazdırır (Andy'nin cybertek'e sahip olduğunu gösterir)

Ağı yapılandırma/değiştirme

**- vi /etc/sysconfig/network**

**- crontab**

- Dosyanızın çalışması için bir zamanlayıcı ayarlar (jenkins gibi bir program oluşturun)

- İzinleri ayarlama

**- chmod**

- Dosya modu bitlerini değiştir

- Sipariş sahibi, grubu, diğerleri

- Dosya klasör ise, d önde

- Klasör değilse d yoktur

- chmod 777

- r- oku

- w-yaz

- x-yürüt

- rwxrwxrwx (777)

- Sahip, grup ve diğerlerine erişim sağlar

66

- Çok tehlikeli

- Kullanmayın

- Chmod 644

- -rw-r--r--

- Sahibine (okuma ve yazma), gruba (salt okunur), diğerlerine (salt okunur) erişim izni verin

- Varsayılan erişim

- Standart

- grep kullanarak bir dosya nasıl bulunur (eğer dosyanın adını biliyorsanız)

- grep ' başarısız test dosyasının adı ' /home/Andy/Test1/TestScenario (konum) >

/home/Alex/OtomasyonDosyası

- Şimdi dosya bu konumun içinde; /home/Alex/OtomasyonDosyası

' name ' olan herhangi bir dosyayı bulun

**- grep ' başarısızlık senaryosu ' \***

**TekTon**

genel sınıf InstanPage { özel statik InstanPage örneği=null;

özel WebDriver sürücüsü; private InstanPage(){ } genel Web Sürücüsü

openBrowser(){ sürücü=yeni FirefoxDriver(); }

driver.manage().window().maximize(); dönüş sürücüsü; } genel statik

InstanPage getInstance(){ if(instance==null){ örnek = yeni

InstanPage(); } dönüş örneği; } }

67

Yumuşak Beceri Mülakat soruları

Genel Giriş / Davranış Soruları:

**1. Neden yazılım kalite analizini seçtiniz?**

**meslek/kariyer?**

Her zaman bir kullanıcının ürünlerine baktım .

perspektif. Kullanıcıya bir

sorunsuz bir deneyim, kaliteli bir ürün değil.

Herhangi bir ürünün kullanım kolaylığı ve kullanım kolaylığı,

yazılım uygulamaları da dahil olmak üzere her zaman ilham kaynağı olmuştur.

ben.

Bu kişilik özelliğinden dolayı, bakmaktan zevk alıyorum

herhangi bir üründeki boşluk için. bir şeyleri kırmayı seviyorum

analitik olarak aşağı inin ve kutunun dışında düşünerek

üzerinde düşünülmemiş boşluklar var.

Bu mükemmel ürünler üretme aşkı beni bir seçim yaptı

mesleğim/kariyerim olarak kalite analizi.

**2. Tipik bir iş gününüzü tanımlayın**

İş yerindeki günüm tipik olarak bir

O gün çalışmam gerekenlerin “ yapılacaklar ” listesi. A

liste aşağıdakilerden herhangi birini içerebilir, ancak bunlarla sınırlı değildir.

Takip etmek:

● E-postamı kontrol edin (herhangi bir

katılmam gereken toplantı)

● Günlük scrum toplantısına veya herhangi bir

bilgi transferi toplantısı.

● Görev üzerinde çalışmaya devam edin, gereksinimleri okuyun

hangi tür testlerin daha iyi anlaşılması için

inşa etmem gereken kasalar

● Senaryolarda otomasyon komut dosyaları yazma

otomasyona ihtiyaç duyulan yerlerde

● Testin yürütülmesi ve koordinasyonun sağlanması

sahip olduğum kusurları kontrol etmek için geliştiriciler

bulundu

● Geliştirme ekibiyle toplantı yapmak için

günlük kusur günlüğünü ve planını iletin

kusurları düzeltme ve yeniden test etme etrafında.

**3. Kendinizi bundan 5 yıl sonra nerede görüyorsunuz?**

biraz el almak istiyorum

bünyesinde iş yapma deneyimi

Bilgi Teknolojileri endüstrisi. kadar öğrenmek isterim

daha teknik rekabetçi olmak mümkün

kişi. İdeal olarak, gerçek olmak isterim

Test alanında yazılım geliştiricisi.

(Not: Bazı yöneticiler zaten

stres. kiralamak konusunda rahat değiller.

kendinden daha zeki insan. Yani

onları nasıl yapacağını bilmelisin

senin olduğunu hissetmelerine izin vererek rahat

teknik olarak iyi, ama sıfır iş parçacığı olacaksın

onların pozisyonuna. )

**4. Bir takım oyuncusu musunuz yoksa yalnız bir kurt musunuz?**

Her iki durumda da bulundum. Süre

Önyüklemek için ekip çalışmasının gerçekten gerekli olduğunu hissediyorum

coşku ve üretkenlik, ben de

rahat kendi başıma çalışıyorum. Benden

deneyim, beyin fırtınası, gitme gibi şeyler

kullanıcı hikayeleri ve senaryoları üzerinden ve takma

kullanıcı hikayelerinde veya kabul kriterlerinde boşluklar

çok takım çalışmasına ihtiyaç var. Ancak, var

dokümantasyon, hazırlama gibi şeyler

sunumlar, en iyi veri analizi

başlangıçta tek başına yapılır.

**5. Küçük takımlarda mı yoksa büyük takımlarda mı çalışmayı seversiniz?**

kadar büyük takımlarda çalışma şansım oldu.

12'den fazla testçi olarak ve ayrıca küçük

4 veya 5 kişilik ekipler Her durumun kendine özgü

artıları ve eksileri, rahat hissediyorum ve çalışıyorum

her iki şekilde de iyi.

Örneğin, büyük bir ekiple

daha fazla kaynak ve daha iyi bir bölüm

iş gücü. Ancak, koordinasyon

zorlu ve kaotik

Benzer şekilde, küçük takımlar çevik olabilir ve

çevik stil geliştirme için uygundur.

Ancak kapsamı sınırlı olmalıdır.

ekip boyutu nedeniyle ve genellikle personel

ince gergin ve mus uzun saatler çalışır.

**6. Baskı altında mı çalıştınız?**

Baskı altında çalışmak garip değilim .

İyi baskı - çok fazla sahip olmak gibi

üzerinde çalışılacak ödevler / görevler veya ve

yaklaşan son tarih kalmama yardım et

motive ve üretken. Tabii ki, orada

çok fazla baskının yol açabileceği zamanlar

strese; Ancak işime öncelik verebilirim.

ve beni engelleyen son teslim tarihlerini karşılamak

iş sırasında sık sık stresli hissetmekten.

Örneğin, geliştirici yapamadığında

bir otomasyon olarak kodu zamanında dağıtın

test cihazı hiçbir şey yapmamalı ve beklememeliyiz. İçinde

bu durumda kabul kriterlerini okudum/

gereksinim, senaryoları analiz etme ve oluşturma

test durumu veya sözde kodumu yaz. Yani

bir kez bittiğinde, işimin %70'i de tamamlanmış olur,

bu nedenle, teslim tarihime yetişebilir veya bitirebilirim

işim son teslim tarihine göre.

Stresimi bu şekilde hallediyorum ya da kaçınıyorum

basit önceliklendirme ile gereksiz stres,

odaklanın ve işi bitirin.

**7. Bu pozisyona neden başvurdunuz? ya da neden gerekir**

**seni işe alıyoruz?**

Boş pozisyonunuzun işime tam olarak uyduğunu hissediyorum.

deneyim, beceri ve nitelikler. Bu nedenle, ben

önemli katkılarda bulunabilecektir.

şirketinizin yanı sıra potansiyelimi yerine getirin.

çalıştığım bir şirkette çalışmak istiyorum

gerçek bir fark yaratabileceğimi hissediyorum. biraz yaptıktan sonra

şirketiniz hakkında araştırma keşfettim

mükemmel bir üne sahip olmasıdır. ben de

Şirketinizin misyonundan büyük ölçüde etkilendim

ifade, değerler ve kültür. bunu şiddetle hissediyorum

senin çalışmak istediğim bir kuruluşsun

ve ile ilişkilendirilebilir.

Şirketinizin bana yardımcı olacağına inanıyorum

kariyerimi istediğim yönde geliştirmek

Git.

Mükemmel bir aday olacağımı düşünüyorum ve

Bu pozisyon için uyumlu uyum.

**8. Neden şimdi bir değişiklik arıyorsunuz?**

Şu anki işverenimde X için çalıştım

yıl sayısı ve bu harika bir yolculuk oldu.

Hala işimi seviyorum ve bu işte çok iyiyim.

Ancak, açısından bir tavana çarptım

burada gelişme ve ben bir

yeni zorluklarla yüzleşme, başarma fırsatı

belirli temel kariyer özlemleri ve içinde büyümek

sanayi. İş tanımını okuduğumda,

gibi göründüğü için hemen ilgimi çekti

becerilerim için mükemmel bir eşleşme ve harika bir

kuruluşunuza değer katma fırsatı.

**9. QA olarak güçlü ve zayıf yönleriniz nelerdir?**

Güçlü:

● QA'ların tüm yazılıma girişi vardır

gelişme süreci. Doğrulama yazılımı

rolün sadece bir yönü - ama çoğu son

kullanıcılar herhangi bir şeyle ilgilenmiyor

uygulamalar onlar için yapabilir. Sahibim

hakkında çok net bir anlayış elde etme yeteneği

ile birlikte iş gereksinimleri

teste başlamadan çok önce iş mantığı

strateji.

● Hızlı öğrenme

● İletişim becerileri

● Baskı altında çalışın veya stresle başa çıkın

zayıf yönler

Geçerken en büyük zayıflığım

kendi işime çok eleştirelim. yapmak için hep düşündüm

mükemmel ve hatasız işler üretmek, her birine gitmek zorunda

ve her ayrıntı, bu benim işime yararken

performans, ancak aşırılıklara gitmek mümkündür. Sahibim

ayrıca kontrol ederek kolayca zaman kaybedebileceğimi ve

aynı şeyleri tekrar kontrol etmek. mesela ben bulduğumda

kusur, bunu doğrulamak için 2-3 kez çoğaltmak harika

bir hata ya da değil. Ama yaptığım şey, hatayı 6-7 kez yeniden oluşturmak

farklı test verileri veya ortam kullanarak. ama sonra ben

bunun sadece zaman kaybı olduğunu fark etti. Yani şimdi ben her zaman

kendime ve kaliteme güvenmek için bilinçli bir çaba sarf etmek

diğer görevlere daha fazla odaklanın.

10 **. Benim için herhangi bir sorunuz var mı?**

Evet, birkaç sorum var, söyler misin?

ekip yapısı ve kiminle çalışacağım hakkında?

Ekibinizin şu anda karşılaştığı zorluklar nelerdir?

**Temel KG soruları**

**1.Yazılım Geliştirme Yaşam Döngüsü Nedir?**

Sistem (veya yazılım) geliştirme yaşam döngüsü

(SDLC), projede kullanılan kavramsal bir modeldir.

bir süreçte yer alan aşamaları tanımlayan yönetim

bir başlangıçtan bilgi sistemi geliştirme projesi

bakımı yoluyla fizibilite çalışması

tamamlanmış uygulama

Aşağıdaki farklı aşamaları içerir:

1.Gereksinim

2. Tasarım aşaması

3.Kodlama

4. Test

5.Sürüm (Üretim)

6. Bakım

**2. Yazılım Testi Yaşam Döngüsü Nedir?**

Yazılım testinin kendi yaşam döngüsü vardır. Başlar

gereksinimleri inceleyerek ve analiz ederek.STLC

olması gereken belirli adımları olan test sürecini ifade eder.

sağlamak için belirli bir sırayla yürütülür.

kalite hedeflerine ulaşılmıştır.

Aşağıdaki farklı aşamaları içeriyordu:

1.Gereksinim Analizi

2. Test planı

3.Test geliştirme (test senaryosu yazma, inceleme

test durumu)

4.Test yürütme

5.Test kapatma.

**3. İhtiyaç belgesi nedir?**

Başka bir deyişle, beklenen tüm işlevler

**“Gereksinimler”** açısından belgelenmiştir

**Gereksinim belgesi** denir . Bu

**Sistem** anlamına gelen SRS belgesi olarak da adlandırılır.

**Gereksinim Spesifikasyon Belgesi.**

**4. Modül Nedir?**

Bir “ Modül ” , belirli bir özelliğe sahip bir yazılım bileşenidir.

görev. İçine girebilen bir “ bağ ” olabilir.

bileşen detayı.

**5. İnşa Nedir?**

Yazılımın farklı modüllerinin her biri

Hazırlanırlar, Configuration ile tek bir klasöre konurlar.

Yönetim Ekibi (CMT) ve buna yapı denir. İçinde

başka bir deyişle, geliştiriciler kodlarını paylaşıma koydu

konum (klasör) ve tüm bu kodlar (modüller) birleştirilir

birlikte çalışan eksiksiz bir uygulamadır.

**6. Yapı Dağıtımı ile ne kastedilmektedir?**

CMT tarafından bu şekilde hazırlanan Derleme farklı bir adrese gönderildiğinde

Test Ortamları, Yapı Dağıtımı olarak adlandırılır.

**7. Gereksinimler karşılanıyorsa uygulamayı nasıl test edersiniz?**

**müsait değil?**

Başvuru için ihtiyaç belgesi mevcut değilse,

hakkında yapılan varsayımlara dayalı olarak bir test planı yazılabilir.

uygulama. Yapılan varsayımlar iyi olmalı

test planında belgelenmiştir. (Kitapçı cevap).

A-1 Gereklilik mevcut değilse,

mümkün olduğunca fazla bilgi toplamak en iyisidir.

rakiplerden son kullanıcılar, istemci ve benzeri uygulamalar.

Araştırmamıza dayanarak uygulamayı hala test edebiliriz. İçin

örneğin bir e-ticaret uygulamasını test etmemiz gerekirse

toplamak için amazon.com'u analiz edebiliriz

temel gereksinimleri ve buna göre testlerimizi gerçekleştirin.

A-1.2 Herhangi bir resmi belgemiz olmadığında (örn.

FSD,BRD.SRS) referans olarak kullanılabilir, yardım alabiliriz

uygulamanın önceki sürümlerinden, kusur açıklaması

ve yorumlar, tel kafesler vb.

iyi etki alanına sahip bazı üyeleri takıma dahil edin

bilgi. Ayrıca doğrudan geliştiricilerle konuşabiliriz ve

Uygulamayı anlamamıza yardımcı olan iş analistleri

davranış. için referans dokümanlar oluşturabiliriz.

yeni ekip üyelerinin olmalarına yardımcı olacak test ekibi

hızlı üretken.

A-2.3 Mevcut şirketimde herhangi bir

dahil olmayan üretim hatası bileti

Kabul Kriterleri (şart). Bu durumda

geliştirici masası, temel neden senaryoları hakkında tartışır ve

bir not, analiz edin, bir test senaryosu oluşturun, bunları yürütün

sorunun çözüldüğünden emin olun ve ardından testimi tamamlayın.

**8. Meslektaş değerlendirmesi nedir?**

Meslektaş incelemesi, herhangi bir hata veya kusur bulma sürecidir.

çeşitli belgeler, ekip üyeleri tarafından yürütülür. en

Akran değerlendirmesinin amacı, kusuru mümkün olan en kısa sürede bulmaktır.

sonraki adıma dağıtılmadan önce mümkündür.

**9. Test senaryosunu kim onaylar?**

Test senaryolarının onaylayıcısı bir kuruluştan diğerine değişir.

sonraki. Bazı kuruluşlarda, ekip lideri/KG lideri onaylayabilir

Test senaryoları, bir başkası ekip üyeleri tarafından onaylanırken

akran değerlendirmesi yoluyla

**10.Yeterli test vakası olduğunu nasıl anlarsınız?**

**bir sistemi veya modülü test etmek için mi oluşturuldu?**

Bu, RTM'ye sahip olmamızın bir nedenidir (Gereksinim

İzlenebilirlik Matrisi) kaç tane gereksinimin olduğunu söyleyebiliriz.

test vakaları tarafından kapsandı ve hala kaç tanesi kaldı

RTM. Başka bir deyişle, haritalayan ve çizen bir belgedir.

test senaryoları ile kullanıcı gereksinimlerini izler. Asıl amaç

Gereksinim İzlenebilirlik Matrisinin tüm testlerinin

durumlarda hiçbir işlevin kaçırılmaması için

Yazılım testi yapıyor.

**11. Test planı nedir? test planlarını kim yazar**

Test etmek için ayrıntılı yaklaşımı açıklayan bir belge

yazılım ve nihai iş akışının ne olacağı. BT

test edilecek özelliklerden, test edilemeyecek özelliklerden oluşur

test edilmiş, yaklaşma, giriş kriterleri, çıkış kriterleri, test

çevre, eğitim ihtiyaçları, kaynaklar, roller ve

sorumluluklar, riskler ve acil durum planı.

Test Planlamasında yer alan görevler nelerdir?

1. Gereksinimleri anlayın ve analiz edin 􀀀

2. Risk analizi

􀀀 3. Test Stratejisinin Uygulanması

􀀀 4. Test Tahminleri

􀀀 5. Takım oluşumu

􀀀 6. Test Planı belgeleri 􀀀

7. Konfigürasyon Yönetimi planlaması 􀀀

8. İzlenebilirlik Matrisi

􀀀 9. Test Ortamı kurulumunu tanımlayın

Test Lideri Test Planını hazırlar

**12. Test stratejisini kim oluşturur? ana nelerdir**

**içine dahil edeceğiniz içerikler?**

Test yöneticisi, test stratejisini oluşturur. Bu bir

şablonu olan şirket düzeyinde belge

genellikle PMO veya eşdeğeri tarafından tavsiye edilir

paylaşılan hizmetler grubu Daha sonra proje için özelleştirilmiş

için benzersiz hususlara dayalı olarak test yöneticisi tarafından

özel uygulama. Yapacağım tipik şeyler

bir test stratejisi belgesine şunları dahil edin:

● Test türleri

● Test etmeden önce tamamlamamız gereken adımlar

● sayısı gibi ayrıntıları içeren test yaklaşımı

kullanıcılar, test senaryolarının oluşturulması, yürütme yaklaşımı,

mülkiyet vb.

● KG zaman çizelgeleri

● Aktörler ve günlük adımlarla test süreci.

● İlgili formatlarda hazırlanacak belgeler.

**13.Test stratejisi ve test planı aynı şeyi mi tanımlıyor?**

**amaç?**

Evet, test stratejisinin ve test planının nihai amacı,

aynı, yani yürütmek için bir kılavuz veya el kitabı olarak çalışmak

yazılım test süreci, ancak yine de ikisi de farklı

**14. Test senaryosu nedir ve test senaryolarını kim oluşturur?**

nasıl yapılacağını adım adım açıklayan belgelerdir.

test gerçekleştirin. şunları içerir: test senaryosu kimliği, test senaryosu adı

, ön koşul, adım açıklaması, test verileri, beklenen sonuç ve

gerçek sonuç ve başarılı/başarısız. Test kullanıcıları test senaryoları oluşturacaktır.

**15. Test senaryosu, test senaryosu arasındaki farklar nelerdir?**

**ve test komut dosyası?**

Test senaryosu, kullanıcının çoklu

uygulamaları kullanırken karşılaşılabilecek eylemler. Bu

neyin test edileceği hakkında. Test vakalarının yüksek seviyesidir.

Yazılım testinde test komut dosyası bir dizi talimattır

Bunu test etmek için test edilen sistemde gerçekleştirilecek

sistem beklendiği gibi çalışır. Bu terminoloji çoğunlukla

otomasyon testi için kullanılır

Test durumu, girişi belirten belgelerdir.

değerler, beklenen çıktı ve yürütme için ön koşullar

test. Bu terminoloji çoğunlukla manuel test için kullanılır. BT

nasıl test edileceği ile ilgilidir. Bu, düşük seviyeli ayrıntıların bir düzenidir.

senaryo nasıl test edilir

**16. Doğrulama ve doğrulama nedir?**

Doğrulama, biz olup olmadığımızdan emin olmak için bir süreçtir.

ürün hakkını oluşturmak, yani gereksinimleri doğrulamak için

ürünü geliştirip geliştirmediğimizi doğrulamak için

buna göre ya da değil. Burada yer alan faaliyetler Denetimler,

İncelemeler, İncelemeler.

Doğrulama, doğru olanı inşa edip etmememiz gereken bir süreçtir.

ürün yani elimizdeki ürünü doğrulamak için

geliştirilen doğru ya da değil. Bununla ilgili faaliyetler,

Yazılım uygulamasının test edilmesi.

17. **Doğrulama ile doğrulama arasındaki fark nedir?**

**yazılım testinin doğrulama yaklaşımı?**

Geliştirme boyunca doğrulama yapılır

geliştirilmekte olan yazılımda aşama, oysa

validasyon üretilen nihai ürün üzerinden gerçekleştirilir

geliştirme sürecinden sonra

belirtilen gereksinim ve şartname.

18. **Yazılım Testinde Giriş Çıkış Kriterleri Nelerdir?**

Giriş kriterleri – Bir işlem yapıldığında çalışması gereken bir süreçtir.

sistem başlar. Aşağıdaki eserler içerir.

SRS (Yazılım Gereksinimi Belirtimi)

FRS (İşlevsel Gereksinim Belirtimi)

Test durumu

Test planı

Test stratejisi

Çıkış Kriterleri – Testin ne zaman tamamlanması gerektiğini bildirir

ve ürün ne zaman piyasaya sürülmeye hazır olmalıdır. O içerir

aşağıdaki eserler.

Test Özeti Raporu

Metrikler

Hata Analizi raporu

**19. Uzun vadeli bir projede, birçok kez gereksinimler**

**değiştirmek. Test planları da değişiklikle birlikte değişiyor mu?**

**Gereksinimler?**

Çoğu zaman, evet. Gereksinimler değişirse, tasarım

belgeler ve özellikler (söz konusu modül için

gereksinimleri uygulayan) da değişecektir.

Bu nedenle, yalnızca o kısım için test planı ve test senaryoları

ayrıca güncellenmesi gerekir. Bunun nedeni “ Kaynak

Tahsis ” , test planının bir bölümüdür. İhtiyacımız olacak

yeni test senaryoları yazmak, test senaryolarını gözden geçirmek ve yürütmek.

Bu nedenle, kaynak tahsisi yapılmalıdır.

buna göre. Sonuç olarak, test planı değişecekti.

**20. Uygulamada gözlemlediğiniz bazı yaygın sorunlar nelerdir?**

**testi zorlaştırabilecek projeler? Ayrıca sağlamak**

**Bu zorlukların üstesinden gelmenin yolları.**

Yazılım geliştirmede bazı yaygın sorunlar

Testi zorlaştırabilecek süreçler şunlardır:

Yanlış veya eksik gereksinimler:

Gereksinimlerin olmasının birçok nedeni vardır.

yanlış veya eksik. Bu doğrudan etkiler

geliştirilmesi ve ardından test edilmesi. Bazen

gereksinimler, geri bağlanacak ayrıntı düzeyinde olmayabilir.

granül test durumu. Bu gibi durumlarda, test ekibinin

bir test vakasının sonucunun ne olduğuna dair en iyi tahminde bulunun

olmalı ve bu onu belirsiz ve sorunlu hale getirebilir

ürün UAT'ye gittiğinde. Bunu çözmek için bir test cihazı

ayrıca BA gibi davranmalıdır. Ne zaman bulursan

net bir test senaryosunun tanımlanamadığı gereksinim veya

yeterli ayrıntı mevcut değil, konuşmalı ve

BA'dan eksik bilgileri almasını isteyin. Bu irade

zaman tüketecek, ancak ürünün size

teslim etmek, kullanıcıların gerçekten istediği şeydir.

Gerçekçi olmayan program:

Gerçekçi olmayan program ve test için ayrılan kısa bir zaman.

Bunu halletmenin yolları var. Bunun bir yolu, testin

vakalar açıkça önceliklendirilir, böylece yüksek ve

İlk önce orta vakalar. Alternatif olarak, daha azını da yapabiliriz

temel özellikleri kapsayan belgeler ve daha fazla geçici test

sağlanan süre içinde sistemin.

21 **. Bildirdiğiniz ayıp giderilemezse ne yaparsınız?**

**sonra bile geliştirme ekibi tarafından çoğaltılabilir.**

**kusura tekrar tekrar daha fazla ayrıntı mı ekliyorsunuz?**

Kusurun ekran görüntülerinin zaten sağlandığını varsayıyorum

bu durumda. Tek alternatif bir video oluşturmak olacaktır.

ve kusurlu olarak yükleyin.

Ancak, hatanın bir hatadan kaynaklandığı zamanlar vardır.

yeniden yaratılamayan tamamen rastgele bir durum. böyle bir

durumu ayrı olarak not edeceğim ve olup olmadığını görmek için test etmeye devam edeceğim.

tekrar olur. Değilse, düşük öncelikli, düşük öncelikli olarak dahil edilebilir.

olasılık hatası.

**22.Geliştirici ve geliştirici arasındaki bir anlaşmazlığı nasıl çözeceksiniz?**

**test cihazının kusurun düzeltilmesini istediği test cihazı,**

**geliştirici bunu bir kusur olarak görmüyor mu?**

Test kullanıcıları ve geliştiriciler, gereksinime bir

farklı bakış açısı. Yani benim yaklaşımım şuna bakmak

müşterinin bakış açısıyla gereksinimi ve

karar.

Yakın tarihli bir projeden bir örnek verebilirim. Bu proje

bir kullanıcının doldurması gereken birçok çevrimiçi form içerir. Bu formlar

yüzlerce alana sahip ve bir kullanıcının bir veya daha fazlasını seçmesi gerekiyor

açılır listeden yüzlerce seçenek arasından değerler. Yine de,

Bir açılır listenin şu anda kullanıcı dostu olmadığını test ederken fark ettim.

tümü, bir diziden birden çok değer seçecekseniz

yüzlerce değer. Çoklu seçim kutusu çok daha iyi bir seçenektir.

Geliştirici, onu inşa ettikleri için bir hata olduğunu düşünmedi

tam olarak gereksinim özelliklerine göre. Ancak,

bir son kullanıcı için çalışmadım. Bunu onlara açıkladığımda,

onda bir değer gördüler ve sonunda bir kusur olarak kaydedildi.

**23. Lütfen yüksek öncelikli-düşük önem derecesine sahip birer örnek veriniz,**

**düşük öncelikli-yüksek önem derecesi ve düşük öncelikli-düşük önem derecesi**

**kusur.**

Düşük öncelik-düşük önem derecesi:

Herhangi bir yazım hatası, yazı tipi boyutu veya rengi.

Düşük öncelikli-yüksek önem derecesi:

Şirket logosu rengi beklendiği gibi değil. Veya alt tarafından yazılmış

dava.

Yüksek öncelikli-yüksek önem derecesi:

Uygulama oturum açma işlevi çalışmıyor.

**24. Öncelik ve önem arasındaki farkı ayırt edin.**

Öncelik, işletmenin düzeltme veya kaldırma ihtiyacını tanımlar

tanımlanmış kusur, şiddet ise tanımlamak için kullanılır

bir kusurun bir sistemin işleyişi üzerindeki etkisi

**25. Hangi parametre kusuru düzeltmenin önemini anlatır?**

**bir müşterinin bakış açısından.?**

Hata önceliği, kullanıcıyı değerlendirecek parametredir.

kullanıcı / müşterinin kusur düzeltmesinin önemi

perspektif. Öncelik ne kadar yüksekse, o kadar önemlidir

kusuru düzeltin.

**26. Hata şiddeti ile hata arasındaki farkı açıklayın**

**öncelik?**

Önem Derecesi : Hata/Kusur şiddeti, aşağıdakilerin etkisi olarak tanımlanabilir.

uygulamadaki hata. Kritik, Majör veya Minör olabilir. İçinde

basit kelimeler, sisteme ne kadar etkisi olacak

belirli bir kusur nedeniyle

Öncelik : . Ana odak, kusurun ne kadar sürede düzeltilmesi gerektiğidir. BT

bir kusurun çözülmesi gereken sırayı verir. geliştiriciler

göre bir sonraki kusuru hangi kusuru gidermeleri gerektiğine karar verin.

öncelik. Yüksek, Orta veya Düşük olabilir.

Geliştirme ekibi önce yüksek öncelikli kusurları alır

daha yüksek şiddette. Genel olarak, önem derecesi şu şekilde atanır:

Test Kullanıcısı / Test Lideri ve öncelik, Geliştirici/Ekip tarafından atanır

Kurşun/Proje Lideri.

**27. Çok az olduğunda yaklaşım ne olmalıdır?**

**test için verilen süre ve testi tamamlamanız gerekiyor**

**bu kısa zaman dilimi içinde?**

Yüksekliğe karar vermek için BA ve geliştiricilere danışın

öncelikli test durumları Önce yüksek öncelikli test senaryolarını yürütün.

Ad hoc testler her zaman daha kısa sürede iyi sonuçlar verir. bu

Buradaki anahtar, uygulamanın farklı alanlarını

ad hoc test yapmak için farklı ekip üyeleri.

**28. Hata triyajı nedir?**

Hata triyajı, hatanın ne kadar önemli olduğunun belirlenmesidir.

sistemde tanımlanan diğer hatalarla ilgili olarak. Adımlar

ilgili olacaktır:

Hatayı analiz edin

Adıma uygun hata önceliği

Hatayı uygun hata sahibine atayın

Gerekirse hata şiddetini ayarlayın

Gerekirse kusurları ertelenmiş duruma taşıyın.

**29. Gizli kusur nedir?**

Gizli kusur, arızaya neden olmayan mevcut bir kusurdur.

çünkü belirli bir dizi koşul asla oluşmadı

eşzamanlı

**30** \_ **Bir hata oluştuğunda test edenin sorumluluğu nedir?**

**test sırasında yakalanmadı, ancak test sırasında müşteri tarafından bulundu**

**UAT?**

Bir müşteri herhangi bir hatayı geliştirme ekibine bildirdiğinde ,

istemciyi bir sorunla baş başa bırakan sistemin en büyük dezavantajı

teslim edilen sistemin olumsuz izlenimi. Böyle bir durumda,

test eden kişi, istemci tarafından bildirilen hatayı kendi sisteminde yeniden oluşturmalıdır.

Yerel sistemlerde yeniden üretilmiyor, sadece yerel sistemlerde üretiliyorsa,

istemcinin sistemi, tutarsız bir hata olarak etiketlenmelidir.

Hatayı tutarsız olarak işaretleyebilir ve geçici olarak kapatabiliriz.

böcek.

**31. Fonksiyonel test türleri nelerdir?**

Birleştirme testi

Duman testi

akıl sağlığı testi

Entegrasyon testi

Sistem testi

Gerileme testi

UAT (kullanıcı kabul testi)

**32.Farklı Test Seviyeleri nelerdir?**

4 test seviyesi vardır 􀀀 a.

● Birim/Bileşen

● Entegrasyon testi 􀀀 c

● Sistem Testi 􀀀 d

● Kabul Testi

**33.Yeniden Test Etme ile Yeniden Test Etme arasındaki fark nedir?**

**Gerileme testi?**

Hata düzeltmelerini doğrulamak için Yeniden Test yapılır ve

Arızanın giderilip giderilmediğini kontrol etmek için regresyon testi yapılır

çalışan mevcut işlevselliği bozdu

değişiklikleri yapmadan önce iyi.

**34.İşlevsel olmayan testler neden aynı derecede önemlidir?**

**fonksiyonel test?**

İşlevsel testler, sistemin işlevlerini test eder

ve yazılımdan önce belirtilen özellikler

gelişme süreci. Yalnızca amaçlananı doğrular

yazılımın belirtilen koşullara karşı işleyişi

gereksinim ve spesifikasyon ancak performansı

sistemin beklenmedik bir şekilde çalışması

gerçek dünyadaki koşullar ve koşullar

Kullanıcıların sonunda ve müşteriyi karşılamak için çevre

memnuniyet, işlevsel olmayan testlerle yapılır

teknik. Böylece, işlevsel olmayan test,

Yazılımın işlevsel olmayan özellikleri.

**35.Hangisi daha iyi bir test metodolojisidir: kara kutu**

**test mi yoksa beyaz kutu testi mi?**

Hem kara kutu hem de beyaz kutu test yaklaşımı,

kendi avantajları ve dezavantajları. Siyah kutu

test yaklaşımı, testçilerin harici olarak test etmelerini sağlar

Sistem, belirtilen gereksinim temelinde ve

şartname ve kapsamı sağlamaz

sistemin iç yapısını test ederken,

beyaz kutu test metodolojisi doğrular ve onaylar

yazılım kalitesini, dahili testleriyle

yapısı ve çalışması.

**36.Kara kutu ve beyaz kutu ise, neden gri kutu?**

**test yapmak?**

Gri kutu testi, üçüncü bir test türüdür ve bir hibrittir.

kara kutu ve beyaz kutu test yaklaşımının şekli,

harici olarak test etme kapsamı sağlayan

türetilen test planlarını ve test senaryolarını kullanan sistem

iç yapı bilgisi ve anlayışı

sistemin. 􀀀

**37. Statik ve dinamik testler arasındaki fark**

**yazılım.**

Statik ve dinamik arasındaki temel fark

test yaklaşımı, birincisinin içermemesidir

sistemi test etmek için kodun yürütülmesi, oysa

ikinci yaklaşım, doğrulamak için kod yürütülmesini gerektirir

ve sistem kalitesini doğrulayın.

**38. Kullanılan çeşitli metodolojileri veya teknikleri listeleyin**

**statik test altında.**

1. Muayene

2. İzlenecek yollar

3. Teknik incelemeler

4. Resmi olmayan incelemeler

5. Meslektaş değerlendirmeleri

**39.Smoke ve Sanity testi, yazılımı test etmek için kullanılır**

**inşa eder. benzerler mi??**

Bununla birlikte, hem duman hem de akıl sağlığı testi için kullanılır

test yazılımı oluşturulur ancak test etmek için duman testi kullanılır

akıl sağlığı kararsız olan ilk yapı

testler, nispeten kararlı yapılar üzerinde yürütülür.

regresyon yoluyla birçok kez geçirmişti

test yapmak.

**40. Keşif testi neden tercih edilir ve kullanılır?**

**çevik metodoloji?**

Çevik metodoloji hızlı yürütme gerektirdiğinden

küçük yinelemeli döngüler yoluyla süreçlerin,

bu nedenle hızlı ve keşif amaçlı testler gerektirir

hangi dokümantasyon çalışmasına bağlı değildir

ve kademeli olarak test cihazı tarafından gerçekleştirilir

yazılım anlayışı, çevik için en uygun

çevre.

**41. Pozitif ve negatif test nedir?**

Pozitif test, amaçlanan ve amaçlananı test etme faaliyetidir.

ile beslenirken sistemin doğru çalışması

geçerli ve uygun girdi verileri ise negatif

test, sistemin davranışını ve yanıtını değerlendirir

geçersiz giriş verilerinin varlığında.

**42.Sistem testi kabulden nasıl farklıdır?**

**test yapmak?**

Sistem testi, test etmek için bakış açısıyla yapılır.

belirtilen gereksinimlere karşı sistem ve

şartname, kabul testi ise

sistemin ihtiyaçları karşılamaya hazır olması ve

Bir kullanıcının beklentileri.

**43. Veritabanı testinin önemi nedir?**

Veritabanı, bir yazılımın kalıtsal bir bileşenidir

uygulamanın arka uç sistemi olarak çalıştığı için

uygulama ve farklı veri türlerini saklar ve

birden fazla kaynaktan bilgi. Böylece, çok önemli

bütünlük, geçerlilik sağlamak için veritabanını test edin,

saklanan verilerin doğruluğu ve güvenliği.

**44.Test Tasarım aşamasında yer alan görevler nelerdir?**

i. Test senaryoları oluşturma

􀀀 ii. Test senaryosu belgeleri

􀀀 iii. Test verisi toplama

**45.Gereksinim İzlenebilirlik Matrisi nedir?**

arasındaki ilişkiyi gösteren belge

Gereksinimler ve Test Durumları.

**46.İzlenebilirlik Matrisini kim hazırlar ve günceller?**

Test Lideri veya Ekip Lideri, İzlenebilirlik Matrisi oluşturur

ve Test Ediciler, tüm süreç boyunca İzlenebilirlik Matrisini günceller.

STLC.

47. **Yazılım Kalite Güvencesi (SQA) Nedir?**

Yazılım kalite güvencesi bir şemsiye terimdir,

için çeşitli planlanmış süreç ve faaliyetlerden oluşan

Tüm yazılımın standardını izlemek ve kontrol etmek

kalite niteliğini sağlamak için geliştirme süreci

nihai yazılım ürününde.

48. **Yazılım Kalite Kontrolü (SQC) Nedir?**

Yazılım kalitesine benzer bir amaçla

güvencesi, yazılım kalite kontrolü,

yazılım geliştirme süreci yerine

kalite yönünü elde etmek ve sürdürmek

yazılım ürünü.

49. **Yazılım Testi Nedir?**

Yazılım testi , aşağıdakilerin bir alt kategorisi olarak görülebilir:

kaldırmak için kullanılan yazılım kalite kontrolü

yazılımda bulunan kusurlar ve kusurlar ve

daha sonra ürünü iyileştirir ve geliştirir

kalite.

50. **Yazılım kalite güvencesi (sqa),**

**yazılım kalite kontrolü (sqc) ve yazılım testi**

**benzer terimler?**

Hayır, ancak hepsinin nihai amacı aynıdır, yani

ve yazılım kalitesini korumak.

51. **O zaman QA, QC ve QC arasındaki fark nedir ?**

**Test yapmak?**

SQA, hem SQC'yi hem de SQC'yi kapsayan daha geniş bir terimdir.

içinde test eder ve yazılım geliştirmesini sağlar

süreç kalitesi ve standardı ve ardından

nihai ürün ayrıca, kullanılan test

yazılım kusurlarını tanımlama ve tespit etme, aşağıdakilerin bir alt kümesidir.

SQC. 􀀀

**52.Giriş kriterleri ve çıkış kriterleri neden belirlenir ve**

**tanımlı?**

Giriş ve çıkış kriterleri tanımlanmış ve belirtilmiştir.

belirli bir test sürecini başlatmak ve sonlandırmak veya

sırasıyla aktivite, belirli koşullar, faktörler

ve gereksinimler karşılanıyor veya karşılanıyor. 􀀀

53. **Yazılım testi yapıtı nedir?**

Yazılım testi yapıtı veya test yapıtı,

boyunca oluşturulan belgeler veya somut ürünler

test amacıyla test süreci veya

ekip arasındaki yazışmalar ve

müşteri. 􀀀

**54.Yazılımda neden hata/kusur var?**

Çeşitli nedenlerden dolayı yazılımda bir hata veya kusur oluşur.

yanlış anlama veya yanlış anlama gibi sebepler ve koşullar

gereksinimler, zaman kısıtlaması, deneyim eksikliği,

hatalı üçüncü taraf araçlar, dinamik veya son değişiklikler,

vb.

**55. Yazılımda bulunan hatalar veya kusurlar nasıl sınıflandırılır?**

**yazılım?**

Bir hata veya kusur önceliğe göre kategorize edilebilir

ve ciddiyet temeli, burada öncelik

iş perspektifinden kusuru düzeltmek veya ortadan kaldırmak,

ciddiyet, çözme ihtiyacını belirtirken veya

Yazılım gereksinimindeki kusuru ortadan kaldırmak ve

kalite bakış açısı.

56. **Hata yaşam döngüsü nedir?**

Hata veya Kusur yaşam döngüsü tüm yolculuğu tanımlar

veya çeşitli aşamalardan geçen bir kusurun ömrü veya

evreleri, tanımlandığı andan itibaren ve

kapatma.

**57. Hata, kusur ve başarısızlık arasındaki fark.**

Yazılım mühendisliğinde hata, hatayı tanımlar.

programcılar tarafından yapılır. Kusur yansıtır

üretim sahasında hataların tanıtımı ve sonuçları

nedeniyle beklenen çıktıdan sonuçlarda sapma

programlama hataları Arıza , sistemin

varlığı nedeniyle işlevleri yerine getirememe

kusur. yani kullanıcı tarafından keşfedilen kusur.

**58. Test, aşağıdakilere göre nasıl farklıdır?**

**hata ayıklama?**

Tespit etmek amacıyla testler yapılır ve

kusurları test ekibi tarafından bulmak, oysa

hata ayıklama, geliştiriciler tarafından düzeltmek veya düzeltmek için yapılır

kusurlar

**59. Gerçekleştirmek için farklı yaklaşımlar nelerdir?**

**yazılım testi?**

Genel olarak, gerçekleştirmek için iki yaklaşım vardır

yazılım testi bkz. Manuel test ve Otomasyon.

Manuel test, üzerinde test senaryolarının yürütülmesini içerir.

yazılım test cihazı tarafından manuel olarak

otomasyon süreci otomasyon kullanımını içerir

test görevini otomatikleştirmek için çerçeve ve araçlar

komut dosyası yürütme. 􀀀

**60. Otomasyonun manuele göre avantajı nedir?**

**test yaklaşımı ve tersi?**

Manuel test yaklaşımına kıyasla ,

otomasyon , gerekli çabayı ve zamanı azaltır.

büyük miktarda test komut dosyasını tekrar tekrar yürütme

ve daha uzun bir süre boyunca sürekli olarak

doğruluk ve hassasiyet.

Ancak, %100 otomasyon testi

imkansız. Manuel test hala gereklidir. bu

manuel testin avantajı, Manuel testin

her türlü uygulama üzerinde yapılır, uygulama

otomatikleştirilmeden önce manuel olarak test edilir, tercih edilir

gereksinimlerin değiştiği projelerde

sık sık ve GUI'nin değiştiği ürünler

sürekli olarak, test cihazının geçici test yapmasına izin verir,

Keşif testi.

**61. Geçici testler ile keşif amaçlı testler arasındaki fark**

**test yapmak?**

Hem geçici testler hem de keşif testleri,

olmadan sistemi test etmenin resmi olmayan yolları

uygun planlama ve strateji. Ancak, geçici olarak

test, bir test cihazı yazılım konusunda bilgilidir ve

özellikleri ve böylece testi gerçekleştirir

keşfederken, öğrenir ve

sırasında yazılım hakkında daha fazla bilgi edinin.

test eder ve böylece sistemi kademeli olarak test eder.

boyunca yazılım anlayışı ve öğrenme

test süreci. 􀀀

**62. Ne zaman, ne ve neden otomatikleştirmeli?**

Yürütme sırasında otomasyon tercih edilir

için testlerin tekrar tekrar yapılması gerekir.

daha uzun süre ve belirtilen süre içinde

son tarihler. Ayrıca, otomasyona ilişkin bir ROI analizi

maliyet-fayda modelini analiz etmek istenmektedir.

otomasyon. Tercihen fonksiyonel, regresyon ve

fonksiyonel testler otomatikleştirilebilir. Ayrıca, testler

doğruluk ve kesinlik gerektiren ve

otomasyon için zaman alıcı olarak kabul edilebilir,

veri odaklı testler de dahil.

**63. Otomasyonda karşılaşılan zorluklar nelerdir?**

Uygulamada karşılaşılan ortak zorluklardan bazıları

otomasyon

1. Başlangıç maliyeti çok yüksektir.

bakım maliyetleri. Böylece, uygun gerektirir

otomasyondaki yatırım getirisini değerlendirmek için analiz.

2. Artan karmaşıklıklar.

3. Sınırlı süre.

4. Uygun özelliklere sahip, yetenekli bir test uzmanı gerektirir.

programlama bilgisi.

5. Otomasyon eğitim maliyeti ve süresi.

6. Doğru ve uygun araçların seçimi ve

çerçeveler.

7. Daha az esnek.

8. Test planlarını ve vakaları güncel tutmak ve

bakımlı

**64. Regresyon yapmak için hangisi daha iyi bir yaklaşım**

**test: manuel mi yoksa otomasyon mu?**

Otomasyon daha iyi avantaj sağlayacaktır

regresyon gerçekleştirmek için kılavuzla karşılaştırma

test yapmak. 􀀀

**65. Regresyon testinin hangi bölümü olmalı?**

**otomatik?**

**Veya**

**66. Otomatik testi ne zaman tercih edersiniz?**

**manuel test?**

Test senaryoları yüksek öncelikli test senaryolarıysa

işlevsellik kritik işlevsellik ise

test durumları duman testinin bir parçasıysa

Test senaryosu çok uzunsa ve yürütülmesi çok zorsa

manuel olarak

Önceliğe dayalı regresyon testi durumları.

Test senaryolarını mümkün olduğunca otomatikleştirmeliyiz.

**67.Ne tür testler sadece manuel olarak yapılır veya yapılmaz**

**otomatik**

1. Dinamik olarak değişen işlevsellik

2. Daha fazla İnsan etkileşimi gerektiren test durumları

3. Bir Resmin içeriğini yakalayamıyoruz

4. Web Uygulamasının Görünümü ve Hissi.

5. tek seferlik test.

**68. Bir yazılım uygulamasının %100 olup olmadığı**

**test edildi mi?**

Hayır, yazılım testi durumlarının ilkelerinden biri olarak

kapsamlı test mümkün değildir.

**69. Teste ne zaman başlamalı ve ne zaman durdurulmalı?**

Temel olarak, yazılım derlemesinin kullanılabilirliği üzerine, test

süreç başlar. Ancak, test erken başlatılabilir

geliştirme süreci ile birlikte en kısa sürede

gereksinimler toplanır ve kullanılabilir. Dahası,

test, yazılımın gereksinimlerine bağlıdır

şelale modelinde olduğu gibi geliştirme modeli, test

test aşamasında yapılırken, çevik testte

çoklu ve kısa yinelemede gerçekleştirilir

Çevrim. 􀀀 Test, olduğu gibi sonsuz bir süreçtir.

bir yazılımı %100 hatasız hale getirmek imkansızdır. Fakat

durdurmak için belirtilen belirli koşullar var

gibi testler :

● Son tarihler

● Test takımlarının ve komut dosyalarının eksiksiz yürütülmesi.

● Bir test için belirtilen çıkış kriterlerini karşılama.

● Yüksek öncelikli ve önem derecesine sahip hatalar belirlenir ve

çözüldü.

● İşlevlerin eksiksiz test edilmesi ve

özellikleri.

**70. İlgili risklerin farklı biçimlerini özetleyin**

**yazılım testi.**

Yazılım testinde yer alan farklı risk türleri şunlardır:

bütçe riski, teknik risk, operasyonel risk, planlanmış

risk ve pazarlama riski

**71. Bir test vakasını ne oluşturur?**

Bir test senaryosu birkaç bileşenden oluşur. Bazı

bunlar test paketi kimliği, test senaryosu kimliği, açıklama,

ön koşul, test prosedürü, test verileri, beklenen

sonuçlar, test ortamı.

72. **Test senaryolarına neden ve nasıl öncelik verilir?**

Yürütme için test vakalarının bolluğu nedeniyle

Verilen test süresi içinde,

test senaryolarına öncelik verin. Test önceliklendirmesi şunları içerir:

test vakalarının sayısında azalma ve & seçimi

sadece bazılarına dayananlara öncelik vermek

özel kriterler.

**73. Bir testçinin rolleri ve sorumlulukları nelerdir?**

**veya bir QA mühendisi?**

Bir QA mühendisinin birden fazla rolü vardır ve şunlarla sınırlıdır:

kaliteyi tanımlamak gibi çeşitli sorumluluklar

parametreleri, test stratejisini tanımlama, testi yürütme,

ekibe liderlik etmek, kusurları veya test sonuçlarını raporlamak.

**74. Test kapsamının ve kod kapsamının olup olmadığı**

**benzer terimler?**

Hayır, kod kapsamı, kodun yüzdesini tutar

yazılım yürütme sırasında kapsanırken, test

kapsanacak test senaryolarıyla ilgili kapsam endişeleri

özel işlevsellik ve gereksinim.

**75.Farklı türlerini listeleyin**

**yazılımda kullanılan belgeler/belgeler**

**test yapmak.**

a. Test planı

b. Test senaryosu

c. Test durumları

d. İzlenebilirlik Matrisi

e. Test Günlüğü ve Raporu

**76.Üretimde bir uygulama varsa ve bir modül**

**kodu değiştirildi, sadece bunu tekrar test etmek gerekli mi?**

**modül veya diğer tüm modüller test edilmeli mi**

**ilave olarak?**

Regresyon testi yapmak ve

hepsini kontrol et

diğer modüller de. En azından sistem testi

olmak

gerçekleştirildi.

77. **Negatif test durumu nedir?**

Negatif test senaryoları, bir ortamda test etme fikrine dayalı olarak oluşturulur.

yıkıcı bir şekilde. Örneğin, ne olacağını test etmek

uygulamaya uygun olmayan girdiler giriliyor

**78. Rastgele testi açıklayın.**

Rastgele test, uygulamanın nasıl işlediğini kontrol etmeyi içerir

rastgele oluşturulan girdi verileri. Veri türleri tipik olarak

yoksayılır ve rastgele bir harf, sayı ve diğer diziler

karakterler veri alanına girilir

**79.Duman testini tanımlayın.**

A1-Duman testi, olmayan bir yazılım testi şeklidir.

kapsamlıdır ve yalnızca en önemli bileşenlerini kontrol eder.

yazılım ancak daha ayrıntılı olarak kontrol etmez. (kitapçı yolu)

A2- Şu anki projemde duman testi yapıyoruz.

uygulama, diğer büyük testleri gerçekleştirmek için yeterince kararlı

faaliyetler.

● Duman testi, uygulamanın tüm alanlarını test etmek için kullanılır.

çok derine inmek.

● Duman testi her zaman otomatik bir test veya yazılı bir dizi

testler. Her zaman senaryo yazılır.

● Duman testi, uygulamanın her bölümünü içerecek şekilde tasarlanmıştır

kapsamlı veya ayrıntılı olmayan bir şekilde.

● Duman testi her zaman en önemli işlevlerin

bir programın çalışıyor, ancak daha ince ayrıntılarla uğraşmıyor.

**80. Akıl sağlığı testinde hangi adımlar yer alır?**

Akıl sağlığı testi, duman testine çok benzer. bu ilk

emin olmak için yapılan bir bileşenin veya uygulamanın test edilmesi

en temel düzeyde çalıştığını ve kararlı olduğunu

daha ayrıntılı testlere devam etmek için yeterli.

● Akıl sağlığı testi, bir veya birkaç alana odaklanan dar bir testtir

işlevsellik, ancak tam olarak veya derinlemesine değil.

● Akıl sağlığı testi genellikle yazılı değildir.

● Akıl sağlığı testi, küçük bir değişiklikten sonra küçük bir

uygulamanın bir kısmı hala çalışıyor.

● Akıl sağlığı testi, kanıtlamak için yapılan üstünkörü bir testtir.

uygulamanın spesifikasyonlara göre çalıştığından emin olun.

Bu test seviyesi, regresyon testinin bir alt kümesidir.

**8. Sistem testi ile sistem testi arasındaki fark nedir?**

**entegrasyon testi?**

Sistem testi için tüm sistem bir bütün olarak kontrol edilir,

entegrasyon testi için ise, arasındaki etkileşim

bireysel modüller test edilir.

**Atik**

**1. Çevik nedir?**

**Çevik yazılım geliştirme, bir grup yazılım geliştirmeyi ifade eder.**

**Gereksinimlerin ve**

**çözümler, kendi kendini organize eden çapraz işlevler arasındaki işbirliği yoluyla gelişir.**

**takımlar**

**2. Projenizde ne tür bir Çevik metodoloji kullandınız?**

**önceki proje?**

Scrum, Agile'ın bir alt kümesidir. için hafif bir süreç çerçevesidir.

çevik geliştirme ve en yaygın kullanılanı.

● Bir “ süreç çerçevesi ” , uygulanması gereken belirli bir uygulamalar dizisidir.

Bir sürecin çerçeve ile tutarlı olması için izlenir.

(Örneğin, Scrum süreç çerçevesi aşağıdakilerin kullanılmasını gerektirir:

Sprints adı verilen geliştirme döngüleri, XP çerçevesi çift gerektirir

programlama vb.)

● “ Hafif ” , sürecin ek yükünün olabildiğince küçük tutulduğu anlamına gelir.

mümkün olduğu kadar, mevcut üretken zaman miktarını en üst düzeye çıkarmak için

faydalı işler yapmak.

**3. Bana hücumu tarif edebilir misin?**

Scrum, çok yaygın olarak kullanılan Çevik çerçevedir.

yazılım geliştirme. Srum'da birkaç rol vardır:

**Ürün sahibi** : Ürün sahibi

ürün adı verilen projeye oluşturulan istek listesi

biriktirme. Ürün sahibi genellikle ürüne öncelik verir

biriktirme öğesi ve sprint biriktirme listesi ile birlikte gelir.

**Scrum master** :

takım doğru yöne. Takımın daha fazlası olması için koçluk yapmak

çevik ve daha üretken. Herhangi bir engelleyicimiz varsa, o veya

Engelleyicimizi kaldırmak için kişiye gidiyor.

**Takım** : normalde scrum geliştirme takımı 4-9

testçiler ve geliştiriciler dahil olmak üzere ekip üyeleri.

**Olaylar:**

**Sprint planlama toplantısı** : Sprint planlama bir

olan bir ScrumMaster'ı içeren işbirlikçi çaba

toplantıyı kolaylaştıran, konuyu netleştiren bir Ürün Sahibi

Ürün biriktirme listesi kalemlerinin ayrıntıları ve ilgili

kabul kriterleri ve tanımlayan Tüm Çevik Ekip

sprintlerini karşılamak için gerekli çalışma ve çaba

bağlılık.

**Günlük stand up** : her gün günlük stand up yapacağız

Aşağıdaki soruları cevaplamak için toplantı:

● Dün ne yaptın?

● Bugün ne yapacaksın?

● Engelleyici var mı?

**Sprint İnceleme Toplantısı** . Scrum'da, her sprint

potansiyel olarak sevk edilebilir bir ürünü teslim etmek için gerekli

artış. ... Yani her sprintin sonunda bir sprint

gözden geçirme toplantısı yapılır. Bu toplantı sırasında,

Scrum ekibi neler başardıklarını gösterir

sprint sırasında. Tipik olarak bu biçimini alır

yeni özelliklerin bir demosu.

**Sprint retrospektif toplantısı** : Sprint'in amacı

Sprint Retrospektifi, uygulamaları geliştirmek,

Bir sonraki Sprint için ekip çalışması ve çevre

önceki Sprint'in nasıl gittiğine bağlı olarak.

toplantı aşağıdaki konuları tartışır:

● Ne iyi gitti?

● Ne iyi gitmedi?

● Nelerin iyileştirilmesi gerekiyor?

**Ürün biriktirme listesi** - Yapmak istediğimiz tüm uygulama

geliştirmek

**Sprint biriktirme listesi** - yapacağımız öğelerin listesi

belirli bir sprintte gelişmek

**4. Scrum takımını nasıl tanımlarsınız?**

Benim için ekip, bir grup insanı paylaşan bir grup insandır.

aynı amaç, aynı yöne hareket eden, birbirine güvenen

diğer ve kim etkili bir şekilde iletişim kuracak ve

harika bir ürün oluşturmak için birbirleriyle işbirliği yapın.

**5. Scrum'da karşılaştığınız zorluk nedir?**

Scrum, çapraz fonksiyonel takımı vurguladığından (

geliştirici test edebilmeli ve test edenler

geliştirebilir). Geliştirme ekibinin bir parçası olmak zor

geleneksel bir QA test cihazı olarak. Çünkü genellikle KG'ler

kod yazmayı bilmek. Bu yüzden kendimi tutmalıyım

çok rekabetçi insan. ne zaman vaktim olsa ben

daha fazla kodlama öğrenmek.

**6. Sprint'iniz ne kadar sürüyor?**

Şu anki projemde 4 hafta / 2 hafta. Ne kadardır

sprintin burada mı? (geri sorabilirsin, yapman gerektiğini unutma

ATM gibi davranmayın. Genelde insanları unuturlar Sadece

soruları yanıtlıyor. Sohbet etmeye çalışın.)

**7. Kullanıcı hikayesi nedir?**

Yazılım geliştirme ve ürün yönetiminde, bir

kullanıcı hikayesi, resmi olmayan, doğal bir dil açıklamasıdır.

bir yazılım sisteminin bir veya daha fazla özelliği. Bir kullanıcı hikayesi

yakalamak için Çevik yazılım geliştirmede kullanılan bir araç

bir son kullanıcıdan bir yazılım özelliğinin açıklaması

perspektif. Bir kullanıcı hikayesi, kullanıcının türünü, ne olduğunu açıklar.

istiyorlar ve neden. Bir kullanıcı hikayesi, bir

bir gereksinimin basitleştirilmiş açıklaması.

**8. Destan nedir?**

Bir Destan, sahip olduğu büyük bir iş parçası olarak tanımlanabilir.

tek bir ortak amaç. Bir özellik olabilir, müşteri

istek veya iş gereksinimi. ... Bu ayrıntılar

Kullanıcı Hikayelerinde tanımlanmıştır. Bir epik genellikle daha fazlasını alır

tamamlamak için bir sprint.

**9.Kabul Kriterleri Nedir?**

Kabul kriterleri ne yapılması gerektiğini tanımlar.

bir kullanıcı hikayesini tamamlayın. sınırlarını belirlerler

hikaye ve ne zaman çalıştığını onaylamak için kullanılır

amaçlanan. Örneğin, kullanıcı hikayesi için < Müşteri olarak ben

Satın alabilmek için bir ürünü arayabilmek istiyorum.>

Yukarıdaki kullanıcı için aşağıdaki kabul kriterlerine sahip olabiliriz

Öykü:

1. Müşteri, ürüne göre arama yapabilmelidir

isim.

2. Müşteri, ürün kimliğine göre arama yapabilmelidir

3. Kullanıcı, arama sonucunu şu ölçütlere göre sıralayabilmelidir:

(sıcak satış, fiyat, derecelendirme, trend)

Tüm koşullar karşılanırsa, hikayeyi biliyoruz.

başarıyla geliştirildi. Ayrıca test senaryolarını temel alarak yazıyoruz

Kabul şartları.

**10. Otopark nedir?**

Agile'da bunun anlamı şudur:

Toplantıda, gerçekte olmayan bir sorununuz olduğunda

diğer prople ile ilgili tartışmaya devam etmemeliyiz

toplantıdaki o öğeyi çünkü diğerlerini boşa harcıyoruz

**park yeri** maddesi yapalım > demek

bu konuyla ilgilenen herkes şu andan sonra konuşabilir

toplantı.

**11. Fare deliği nedir?**

Çevikte çok fazla iletişim olduğundan

takım, takım birçok şeyi tartışmak zorunda. Ama bazen

tartışma bir konu için çok uzun sürecek ve

gerçekten üretken. < **fare deliği >** diyeceğiz, demek ki biz

bu konuyu fazla uzatmamalı ve hareket etmemelidir.

ileri.

**12.Avantajları ve dezavantajları nelerdir?**

**Çevik Model?**

**Çevik Modelin Avantajları:** 􀀀 1.

diğer modellere kıyasla çok yüksek bir proje. 􀀀 2. Can

Gereksinimlerdeki değişiklikleri herhangi bir zamanda kabul edin. 􀀀 3.

Çalışan yazılımlar sıklıkla teslim edilir. 􀀀 4. Bu

değil, değişime yanıt vermeyi vurgular.

kapsamlı planlama ve dokümantasyon. 􀀀 5. Bu

Ürün Geliştirme için önerilir.

**Çevik Modelin Dezavantajları:** 􀀀 1. Pahalı Model olarak

daha fazla sayıda kaynağa ihtiyaç vardır. 􀀀 2. Kompleks

Yönetmek. 􀀀 3. Gerekli konulara vurgu yapılmamaktadır.

tasarlamak ve belgelemek. 􀀀 4. Proje kolayca

herhangi bir iletişim boşluğu varsa yoldan çekilin.

**13. Scrum Modeli Nedir?**

Scrum, yinelemeli ve artımlı bir çevik yazılımdır.

yazılım yönetimi için geliştirme metodolojisi

gelişim. 􀀀 Bu modelde, Sistem küçük parçalara bölünmüştür.

Sprintler olarak bilinen parçalar. Her sprintin süresi

bir hafta ile üç hafta arasında değişmektedir. Sonunda

sprint, ekip üyeleri ve paydaşlar değerlendirmek için bir araya gelir

Projenin ilerleyişi ve daha fazla planın belirlenmesi

eylem. Bu değerlendirme, mevcut durumun izini sürmeye yardımcı olur.

iş hattını belirtin ve yeniden çalışın ve projeyi tamamlayın

zamanında ve sadece spekülasyon yapmak veya daha fazlasını tahmin etmekle kalmayın

sonuç.

**14. Çevik Model neden diğer SDLC'den bu kadar popüler?**

**modeller?**

Çevik Model, esnekliği nedeniyle popülerdir.

gereksinimlerdeki değişiklikleri benimsemek ve aynı zamanda

yazılımı mümkün olan en kısa sürede teslim etmek. Örneğin,

sprint incelemesinden / demosundan sonra, müşteri beğenmezse

geri bildirimlerini alıp geliştirebileceğimiz bir şey

ürün. **Gereksinim değişikliği tamam.**

Yinelemeli bir geliştirme süreci olduğundan,

geliştirme ekibi işlevsellik parçası geliştirebilir,

geri bildirim alın ve bir sonraki yinelemeyi iyileştirin. yani ürün

sürekli iyileştirilecektir.

İsraf, scrum yardımıyla çevik bir şekilde ortadan kaldırılır

usta. Örneğin, engellenirsem beklemek zorunda değilim

ve zamanımı boşa harcama. Ekip üyeleri iletişim kurduğundan

birbirimizle verimli bir şekilde daha üretken olabiliriz

tekrarlanan çabanın önlenmesi.

**========= SELENYUM =========**

**Otomasyon Çerçevesinde OOPS**

• ÖZET

• ARAYÜZ

• MİRAS

• POLİMORFİZM

• YÖNTEM AŞIRI YÜKLEME

• YÖNTEM GEÇERSİZ KILMA

• KAPSÜLLEME

**SOYUTLAMA**

Sayfa Nesne Modeli tasarım deseninde yazıyoruz

konum belirleyiciler (id, ad, xpath vb. gibi) bir

Sayfa Sınıfı. Bu konumlandırıcıları testlerde kullanıyoruz

ancak bu konumlandırıcıları testlerde göremiyoruz.

Kelimenin tam anlamıyla yer belirleyicileri testlerden gizleriz.

Soyutlama, bilgiyi gizleme yöntemidir.

iç detayların uygulanması ve gösterilmesi

kullanıcılara işlevsellik.

Soyutlama hakkında daha fazla bilgi edinin

**ARAYÜZ**

Selenium'da hepimizin bildiği temel ifade

Web Sürücüsü

sürücü = yeni FirefoxDriver();

WebDriver'ın kendisi bir Arayüzdür. peki bu ne

demek ki WebDriver sürücüsü = yeni

FirefoxSürücü(); Firefox'u başlatıyoruz

Selenium WebDriver kullanan tarayıcı. Ayrıca

bir referans değişkeni oluşturduğumuz anlamına gelir

(Sürücü) arabiriminin (WebDriver) ve

bir Nesne oluşturma. Burada WebDriver bir

Daha önce belirtildiği gibi arayüz ve

FirefoxDriver bir sınıftır.

Java'daki bir arayüz, bir sınıfa benzer ancak

hem arayüz hem de sınıf iki farklı

kavramlar. Bir arayüzün yöntemleri olabilir ve

değişkenler sadece sınıf gibi ama yöntemler

arayüzde bildirilenler varsayılan olarak soyuttur.

Burada Arayüz hakkında daha fazla bilgi edinin .

**MİRAS**

Çerçevede bir Temel Sınıf oluşturuyoruz

WebDriver arayüzünü başlatır, WebDriver bekler,

Temel Sınıfta özellik dosyaları, Excel'ler vb.

Temel Sınıfı aşağıdaki gibi diğer sınıflarda genişletiriz:

Testler ve Yardımcı Sınıf olarak. Bir sınıfı genişletme

diğer sınıfa Kalıtım olarak bilinir.

Burada Miras hakkında daha fazla bilgi edinin .

**polimorfizm**

Aşırı yükleme ve geçersiz kılma kombinasyonu

Polimorfizm olarak bilinir. Polimorfizm bize izin verir

bir görevi birden çok şekilde gerçekleştirin.

Polimorfizm, bir nesnenin üstlenme yeteneğidir.

birçok form. En yaygın kullanımı

OOP'deki polimorfizm, bir üst sınıf

başvuru, bir alt sınıf nesnesine başvurmak için kullanılır.

- DİNAMİK POLİMORİİZM -

GEÇERSİZ KIRMA

- STATİK POLİMORİİZM - AŞIRI YÜKLEME

Burada Polimorfizm hakkında daha fazla bilgi edinin .

**YÖNTEM AŞIRI YÜKLEME**

Selenium'da örtük bekleme kullanıyoruz. örtük bekleme

aşırı yüklenmeye bir örnektir. Örtük beklemede

gibi farklı zaman damgaları kullanıyoruz

SANİYE, DAKİKA, SAAT vb.

Aynı ile birden çok yönteme sahip bir sınıf

ad ancak farklı parametrelere Yöntem denir

Aşırı yükleme

Burada Aşırı Yükleme hakkında daha fazla bilgi edinin .

**YÖNTEM GEÇERSİZ KILMA**

Zaten olan bir yöntemi kullanıyoruz

değiştirilerek başka bir sınıfta uygulandı.

parametreler.

Alt sınıfta bir yöntem bildirmek

ana sınıfta zaten mevcut denir

Yöntem Geçersiz Kılma. Örnekler almak ve

farklı sürücülerin yöntemlerinde gezinme

Selenyum.

Burada örneklerle Geçersiz Kılma hakkında daha fazla bilgi edinin

**KAPSÜLLEME**

Bir çerçevedeki tüm sınıflar bir örnektir

Kapsülleme. POM sınıflarında beyan ederiz

@FindBy kullanan veri üyeleri ve

veri üyelerinin başlatılması yapılacak

Bunları yöntemlerde kullanmak için Yapıcı kullanma.

Kapsülleme, bir bağlama kodu mekanizmasıdır

ve verileri tek bir ünitede bir araya getirir.

Burada Kapsülleme hakkında daha fazla bilgi edinin

başka konuları tartışmak istiyorum

Otomasyon Çerçevesinde kullanıyoruz.

**WEB ELEMANI:**

Web öğesi, tanımlamak için kullanılan bir arayüzdür.

Bir web sayfasındaki öğeler.

**WEB SÜRÜCÜSÜ:**

WebDriver, başlatmak için kullanılan bir arayüzdür.

Firefox, Chrome gibi farklı tarayıcılar,

Internet Explorer, Safari vb.

**TARAFINDAN BUL:**

FindBy, Sayfa Nesnesinde kullanılan bir açıklamadır

Elemanları tanımlamak için model tasarım deseni.

**ELEMAN BUL:**

Öğe Bul, POM'da tanımlamak için bir yöntemdir

Bir web sayfasındaki öğeler.

Java Öğrenin – Selenium için özelleştirilmiş bir gönderi

Otomasyon Test Cihazları

**- Tarayıcı nasıl kullanılır (krom)**

**Selenium'daki bildirimler?**

**- @Listener'ın kullanımı nedir?**

**TestNG'de açıklama?**

TestNG dinleyicileri raporları yapılandırmak için kullanılır

ve günlüğe kaydetme. En yaygın kullanılanlardan biri

dinleyiciler *ITestListener* arayüzüdür. BT

*onTestStart* gibi yöntemlere sahiptir ,

*onTestSuccess* , *onTestFailure* , *onTestSkipped* e

tc.

Pratik Örnek

**- Bir grup test senaryosu nasıl çalıştırılır**

**TestNG'yi mi kullanıyorsunuz?**

TestNG, gelişmiş performans göstermenizi sağlar

test yöntemlerinin gruplandırılması. Sadece beyan edemezsin

bu yöntemler gruplara aittir, ancak

diğer grupları içeren grupları belirtin. O zamanlar

TestNG çağrılabilir ve bir

belirli grup grupları (veya normal ifadeler)

başka bir set hariç.

Gruplar, testng.xml dosyanızda belirtilir ve

<test> veya <suite> etiketinin altında bulunur . Gruplar

<suite> etiketinde belirtilen tüm <test> etiketlerine uygulanır

altında.

Tüm Gönderiyi Görüntüle

**- Parametreli test nedir?**

**Test NG?**

*Parametreli testler* , geliştiricilerin çalışmasına izin verir

kullanarak aynı testi tekrar tekrar

farklı değerler.

Bu parametreleri ayarlamanın iki yolu vardır:

• *testng.xml ile -* Pratik Örnek

• *Veri Sağlayıcılarla –*

Pratik Örnek

**- Test senaryosu önceliği nasıl ayarlanır?**

**Test NG?**

*Öncelik* özelliğini kullanıyoruz

@Test *ek* açıklamaları. Öncelik olmaması durumunda

ayarlayın, ardından test komut dosyaları alfabetik olarak yürütülür

emir.

**- TestNG.xml nasıl oluşturulur ve çalıştırılır?**

TestNG çerçevesinde, yapmamız gereken

ve işlemek için T **estNG XML dosyası oluşturun**

çoklu test sınıfları Testimizi yapılandırıyoruz

çalıştır, test bağımlılığını ayarla, dahil et veya hariç tut

herhangi bir test, yöntem, sınıf veya paket ve set

XML dosyasında öncelik vb.

Komple Gönderi İçin

**- TestNG Assert nedir ve listesi**

**desteklenen bazı ortak İddialar**

**TestNG tarafından?**

TestNG Asserts, durumu doğrulamamıza yardımcı olur

test çalıştırmasının ortasındaki testin. Temelli

TestNG İddialarında, bir

başarılı test, yalnızca testi tamamlarsa

herhangi bir istisna atmadan çalıştırın.

tarafından desteklenen bazı ortak iddialar

TestNG vardır

• assertEqual(Dize gerçek,Dize

beklenen)

• assertEqual(Dize gerçek,Dize

beklenen, Dize mesajı)

• assertEquals(boolean gerçek, boolean

beklenen)

• assertTrue(koşul)

• assertTrue(koşul, mesaj)

• assertFalse(koşul)

• assertFalse(koşul, mesaj)

**- Mevcut ek açıklamalar nelerdir?**

**Test NG?**

@ÖnceTest

@AfterTest

@Dersten önce

@Dersten sonra

@ÖnceYöntem

@SonraYöntem

@BeforeSuite

@AfterSuite

@BeforeGroups

@AfterGroups

@Ölçek

**- Tarayıcı Çerezleri nasıl silinir?**

**Selenyum Web Sürücüsü?**

**- Veritabanı testi nasıl yapılır**

**Selenyum?**

JDBC, bize izin veren SQL düzeyinde bir API'dir.

SQL deyimlerini yürütün. bir

Java Programlama arasındaki bağlantı

Dil ve veritabanı.

JDBC Sürücüsünü kullanarak şunları yapabiliriz ..

i. Veritabanı bağlantısı kurun

ii. Veritabanına SQL Sorguları Gönderin

iii. Sonuçları işleyin

**- Sürekli Entegrasyon Nedir? CI**

Sürekli Entegrasyon bir geliştirme uygulamasıdır

doğruluğundan emin olmayı amaçlayan bir

yazılım. Her taahhütten sonra, bir dizi test çalıştırılır

sağlamak için yazılımı otomatik olarak test edin ve

yazılımın herhangi bir şey olmadan çalışıp çalışmadığı

kırılır. Herhangi bir test başarısız olursa, hemen alacağız

geribildirim "derleme bozuk" veya " Derleme başarısız" diyor

Basit bir deyişle, sürekli entegrasyon bir süreçtir

bir yazılımın doğruluğunu doğrulamak.

Bunları kullanarak test takımının yürütülmesini planlayabiliriz.

CI Araçları.

**İstenen yetenekler nedir?**

Selenium'da SSL'yi işlemek için istenen yetenekleri kullanıyoruz

chrome tarayıcıda sertifikalar

DesiredCapabilities örneğini oluşturmamız gerekiyor

**- Öğe Kullanarak Nasıl Vurgulanır**

**Selenyum Web Sürücüsü?**

JavaScript Executor arayüzünü kullanarak,

belirtilen öğeyi vurgulayabilir

Pratik Örnek

**- Sürükle ve Bırak Nasıl Gerçekleştirilir**

**Selenium WebDriver'da Eylem?**

Selenium, işlemek için “Eylemler” sınıfı sağlar

sürükle ve bırak

Pratik Örnek

**- Çift Tıklama İşlemi Nasıl Gerçekleştirilir**

**Selenium WebDriver'da mı?**

Çift tıklama eylemi yapmak için Actions sınıfını kullanıyoruz

selenyum içinde.

Pratik Örnek

**- Sağ Tıklama İşlemi Nasıl Gerçekleştirilir**

**(Bağlam Tıklaması) Selenium WebDriver'da?**

Selenium WebDriver'da Actions sınıfını şu şekilde kullanıyoruz:

Sağ Tıklama (İçerik Tıklama) eylemini yapın.

Pratik Örnek

**- Web Sayfasını Aşağı veya Yukarı Kaydırma**

**Selenyum Web Sürücüsünü Kullanıyor musunuz?**

JavaScript **scrollBy** () yöntemi,

belirtilen piksel sayısına göre belge.

Pratik Örnek

**- Tarayıcı Penceresi Nasıl Yeniden Boyutlandırılır**

**Selenyum Web Sürücüsünü Kullanıyor musunuz?**

Tarayıcı penceresini belirli bir şekilde yeniden boyutlandırmak için

boyutları, yeniden boyutlandırmak için 'Boyut' sınıfını kullanıyoruz

tarayıcı penceresi.

**- Çerçeveler arasında nasıl geçiş yapılır**

**Selenyum?**

Aşağıdaki kodu kullanarak geçiş yapabiliriz

çerçeveler arasında.

**- Selenium'da bir dosya nasıl yüklenir**

**Web Sürücüsü?**

Anahtarları gönder yöntemini kullanarak

**- Kurtarma'yı nasıl kullanabilirsiniz?**

**Selenium WebDriver'daki Senaryo?**

Selenium içinde “ *Try Catch Block”* kullanarak

Web Sürücüsü Java testleri.

**- Sayfanın avantajları nelerdir?**

**Nesne Modeli Çerçevesi?**

**Kodun yeniden kullanılabilirliği** – Kod elde edebiliriz

kodu bir kez yazıp kullanarak yeniden kullanılabilirlik

farklı testlerde.

**Kod sürdürülebilirliği** – Temiz bir

test kodu ve sayfaya özel arasındaki ayrım

konumlandırıcılar ve düzen gibi kodlar

kodu korumak çok kolay hale gelir. BT

test bakımını geliştirir ve kodu azaltır

çoğaltma.

**Nesne Deposu** – Her sayfa

Java sınıfı olarak tanımlanır. içindeki tüm alanlar

sayfa bir arayüzde şu şekilde tanımlanacaktır:

üyeler. Sınıf daha sonra uygulayacak

arayüz.

**Okunabilirlik** – Okunabilirliği artırır

test kodu ve sayfa arasındaki temiz ayrım

özel kod

**- Sayfa arasındaki fark nedir?**

**Nesne Modeli (POM) ve Sayfa Fabrikası?**

**Sayfa Nesnesi** , bir web'i temsil eden bir sınıftır.

sayfa ve işlevleri ve üyeleri tutun.

**Sayfa Fabrikası** , web'i başlatmanın bir yoludur

içinde etkileşim kurmak istediğiniz öğeler

bir örneğini oluşturduğunuzda sayfa nesnesi.

**- Sayfa Fabrikası nedir?**

Bildiğiniz gibi 'Sayfa Nesne Modeli'

bir testte bir uygulamayı temsil etmek

çerçeve. Uygulamadaki her 'sayfa' için,

'sayfaya' referans vermek için bir Sayfa Nesnesi oluşturuyoruz

diğer yandan 'Sayfa Fabrikası' bir yoldur

'Sayfa Nesne Modeli'ni uygulamak.

**- Sayfa Nesne Modeli Nedir?**

**Selenyum? POM**

Sayfa nesne modeli (POM) şu durumlarda kullanılabilir:

her türlü çerçeve

Bir sayfa nesnesi, nesne yönelimli bir sınıftır.

sayfanızın bir arayüzü olarak hizmet eder.

Test Edilen Uygulama (AUT). O zaman testler

bu sayfa nesne sınıfının yöntemlerini kullanın

Kullanıcı ile etkileşime girmeleri gerektiğinde

Bu sayfanın arayüzü (UI). Fayda şu ki

sayfa için kullanıcı arayüzü değişirse, testler

kendilerinin değişmesi gerekmez, sadece

sayfa nesnesindeki kodun değişmesi gerekiyor.

**- Nesne Deposunu nasıl oluşturursunuz?**

**senin projen?**

**-**

Bir kullanıcı bir testi kaydettiğinde, nesneler ve

özellikler varsayılan olarak bir Nesnede yakalanır

Depo. QTP, bu Nesne Deposunu şu amaçlarla kullanır:

komut dosyalarını oynatın. Selenyum'a gelince,

varsayılan Nesne Deposu kavramı yoktur.

Bu, Nesne olmadığı anlamına gelmez

Selenyum'da Depo. Olsa da

varsayılan bir tane yok, hala kendimizinkini yaratabiliriz. İçinde

Selenyum, nesneleri konumlandırıcı olarak adlandırırız (örneğin

Kimlik, Ad, Sınıf Adı, Etiket Adı, Bağlantı Metni,

Kısmi Bağlantı Metni, XPath ve CSS). Nesne

depo, nesnelerin bir koleksiyonudur. Biri

Nesne Deposu oluşturmanın yolları, hepsini yerleştirmektir.

konum belirleyiciler ayrı bir dosyada (yani, özellikler

dosya). Ancak en iyi yol, Sayfa Nesnesini kullanmaktır.

modeli. Sayfada Nesne Modeli Tasarımı

Desen, her web sayfası bir

sınıf. Belirli bir konu ile ilgili tüm nesneler

bir web uygulamasının sayfası bir sınıfta saklanır.

**- Yapamadığımız bazı senaryoları listeleyin**

**Selenium WebDriver kullanarak otomatikleştirilsin mi?**

1. Bitmap karşılaştırması kullanılarak yapılamaz

Selenyum Web Sürücüsü(fotoğraflar = Bitmap)

2. Captcha'yı otomatikleştirmek şu şekilde mümkün değildir:

Selenyum Web Sürücüsü

3. Selenium kullanarak barkod okuyamıyoruz

Web Sürücüsü

**- Selenium'da Ajax aramaları nasıl yapılır?**

**Web Sürücüsü?**

AJAX, istemciden HTTP istekleri gönderir.

sunucu ve ardından sunucunun yanıtını işleyin

tüm sayfayı yeniden yüklemeden.

Gönder düğmesine tıkladığınızda,

gerekli bilgiler web'de görünebilir

tarayıcıyı yenilemeden sayfa.

Bazen bir saniyede yüklenebilir ve

bazen daha uzun sürebilir. bizde yok

yükleme süresi üzerinde kontrol. En iyi yaklaşım

selenyumda bu tür durumları ele almak

dinamik beklemeleri kullan

1. **titleIs()** – Beklenen koşul

belirli bir başlığa sahip bir sayfa.

1 bekle. Kadar(ExpendedConditions.titleIs(“Günün Fırsatı”));

2. **elementToBeClickable()** – Beklenen

koşul, bir öğenin tıklanabilir olmasını bekler

yani üzerinde mevcut/görüntülenen/görünür olmalıdır

ekranın yanı sıra etkin.

1 wait.until(ExpectedConditions.elementToBeClickable(By.xpath("xpath")));

3. **alertIsPresent()** – Beklenen koşul

bir uyarı kutusunun görünmesini bekler.

1 wait.until(ExpectedConditions.alertIsPresent()) !=null);

4. **textToBePresentInElement()** –

beklenen koşul, sahip bir eleman için bekler

belirli bir dize deseni.

1 wait.until(ExpectedConditions.textToBePresentInElement(By.id(“title'”), “bulunacak metin”));

**- Captcha'yı otomatikleştirmek mümkün mü**

**Selenyum kullanarak?**

Hayır, captcha'yı otomatikleştirmek mümkün değildir ve

barkod okuyucu.

**- Test verilerini nasıl okursunuz?**

**üstün mü?**

Excel dosyalarını işlemek için Apache POI kullanıyoruz

Selenyum Web Sürücüsü. Hepimizin bildiği gibi Selenyum

yalnızca Web tarayıcı otomasyonunu destekler. Biz

gibi üçüncü taraf API'nin yardımını almanız gerekiyor

Excel'i işlemek (okumak ve yazmak) için Apache POI

Selenium WebDriver kullanan dosyalar.

APACHE POI'DEKİ JAVA ARAYÜZLERİ VE SINIFLARININ LİSTESİ

XLS VE XLSX DOSYASINI OKUMAK İÇİN

**- JavaScriptExecutor nedir ve içinde**

**JavaScriptExecutor hangi durumlarda yardımcı olacaktır**

**Selenyum otomasyonunda?**

Genel olarak bir örnek vereyim, biz

tıklama () yöntemini kullanarak bir öğeye tıklayın

Selenyum.

Örneğin:

1 **-** sürücü. findElement(By.id("Id Değeri")).click();

Bazen web kontrolleri iyi tepki vermiyor

selenyum komutlarına karşı ve karşılaşabiliriz

yukarıdaki ifadeyle ilgili sorunlar (tıklayın ()). İle

Böyle bir durumun üstesinden geliriz,

JavaScriptExecutor arayüzünü kullanın.

Bunu kullanarak halledebileceğimiz bazı senaryolar

Arayüzler:

1. Selenium WebDriver'da Metin Yazmak İçin

Keys() yöntemini gönder

2. Selenium WebDriver'da bir Düğmeye tıklayarak

JavaScript

3. Onay Kutusunu işlemek için

4. Selenyumda Uyarı Pop penceresi oluşturmak için

5. Javascript kullanarak tarayıcı penceresini yenilemek için

6. Tüm web sayfasının iç metnini

Selenyum

7. Web sayfamızın başlığını almak için

8. Alan adını almak için

9. Bir web sayfasının URL'sini almak için

10. Bir uygulamada Kaydırma yapmak için

Selenyum

11. Yalnızca ekranda görünen bir Alt Menüye tıklamak için

Menüde farenin üzerine gelin

12. Javascript kullanarak farklı sayfaya gitmek için

**- Bir JavaScript değişkeni nasıl okunur**

**Selenyum Web Sürücüsü?**

JavascriptExecutor kullanarak

**- Birden fazla web nasıl bulunur**

**Listedeki eleman?**

**- İçindeki gizli öğeler nasıl işlenir?**

**Selenyum Web Sürücüsü?**

JavaScript Executor kullanarak gizli öğeleri işleyebiliriz.

**- Windows tabanlı nasıl işleyebiliriz?**

**açılır?**

Selenyum, Windows tabanlı desteklemiyor

uygulamalar. Bir otomasyon test aracıdır

yalnızca web uygulaması testini destekler.

Windows tabanlı açılır pencereleri

Selenyum gibi bazı üçüncü taraf araçları kullanarak

AutoIT, Robot sınıfı olarak şimdiye kadar

bu araçları kullandım ama sabırsızlıkla bekliyorum

gerekirse öğrenme.

**- Web tabanlı açılır pencereyi nasıl halledebiliriz?**

Uyarı açılır pencerelerini işlemek için geçiş yapmamız gerekiyor

uyarı penceresine gidin ve Selenium'u arayın

WebDriver Alert API yöntemleri.

***Uyarı*** Arayüzü kullanıyoruz .

***Uyarı*** Arayüzü, aşağıdakiler için bazı yöntemler sağlar :

pop-up'ları idare et

Bir paketi içe aktarmamız gerekiyor

***org.openqa.selenium.Uyarı***

Selenium'daki uyarıları işlemek için.

driver.switchTo().alert();

**Açık uyarıyı ele almak için:**

1 Uyarı uyarısı = driver.switchTo().alert();

**Tamam düğmesine tıklamak için:**

1 uyarı.kabul();

**İptal düğmesine tıklamak için** .

1 uyarı.kapat()

**Uyarıda bulunan metni almak için** .

1 uyarı.getText();

**Uyarı kutusuna metni girmek için**

1 alert.sendkeys(String stringToSend);

**Kimlik bilgilerini ileterek Kimlik Doğrulamak için**

1 alert.authenticateUsing(Kimlik bilgileri)

Pratik Örnek.

**- Bir web üzerinde fareyle üzerine gelme nasıl yapılır**

**WebDriver kullanan öğe?**

Actions sınıfını kullanarak

**- Ekran görüntüsü nasıl alınır**

**Selenyum Web Sürücüsü?**

Selenyum bir arayüz sağlar

olan *TakesScreenshot* olarak adlandırılır.

için kullanılabilecek *getScreenShotAs* yöntemi

test edilen uygulamanın ekran görüntüsünü alın.

**Açılır menüde bir değer nasıl seçilir?**

*Select* sınıfını kullanarak

**Seçme Yöntemlerinin Türleri:**

i. **selectByVisibleText** Yöntemi

ii. **selectByIndex** Yöntemi

iii. **SelectByValue** Yöntemi

**DeSelect Yöntemlerinin Türleri:**

i. deselectByVisibleText Yöntemi

ii. deselectByIndex Yöntemi

iii. deselectByValue Yöntemi

iv. deselectAll Yöntemi

**- Bir elemanın olup olmadığı nasıl bulunur**

**web sayfasında görüntüleniyor mu?**

Bir öğenin olup olmadığını doğrulamak için Selenyum Web Sürücüsünde

düğmeler, açılan kutular gibi görüntülenir,

onay kutuları, radyo düğmeleri, kullandığımız etiketler…

**1. Görüntüleniyor()**

1 boolean elePresent = driver.findElement(By.xpath("xpath")).isDisplayed();

**2.Selected()**

1 boolean eleSelected= driver.findElement(By.xpath("xpath")).isSelected();

**3. Etkinleştirildi()**

1

boolean eleEnabled= driver.findElement(By.xpath("xpath")).isEnabled();

**- Fark ne**

**driver.findElement() ile**

**driver.findElements() komutları?**

driver.findElement() ve arasındaki fark

driver.findElements() komutları-

• findElement(), aşağıdakilere dayalı olarak tek bir WebElement (önce bulunur) döndürür:

bulucu parametre olarak geçti. findElements() bir liste döndürürken

WebElements, tümü iletilen konumlandırıcı değerini karşılayan.

• findElement()- sözdizimi

WebElement textbox=driver.findElement(By.id(“textBoxLocator”));

findElements()- sözdizimi

Liste <WebElement> öğeleri = element.findElements(By.id(“değer”));

• İkisi arasındaki diğer bir fark ise şudur: eğer eleman bulunamazsa

findElement (), NoSuchElementException'ı atar, oysa

findElements(), 0 öğeden oluşan bir liste döndürür.

**- driver.close() ile driver.close() arasındaki fark nedir?**

**driver.Quit() yöntemleri?**

Bu iki yöntemin (driver.close ve driver.quit) amacı,

hemen hemen aynı. Her ikisi de bir tarayıcıyı kapatmamıza izin veriyor ama yine de orada

bir farktır.

*driver.close():* Geçerli WebDriver örneğini kapatmak için

*driver.quit():* Açılan tüm WebDriver örneklerini kapatmak için

**- Arasındaki fark nedir**

**driver.getWindowHandle() ve**

**Selenyum'da driver.getWindowHandles()**

**Web Sürücüsü?**

*driver.getWindowHandle()* – Geçerli olanın tanıtıcısını döndürür

sayfa (benzersiz bir tanımlayıcı)

*driver.getWindowHandles()* – Bir dizi tutamaç döndürür.

tüm sayfalar mevcuttur.

**- Kullanarak bir tarayıcıyı yenilemenin yolları nelerdir?**

**Selenyum Web Sürücüsü?**

Selenium'da bir sayfayı yenilemenin birden çok yolu vardır.

• Belirtildiği gibi *driver.navigate().refresh()* komutunu kullanma

45. soruda

• Geçerli URL'de driver.get(“URL”) kullanma veya

*driver.getCurrentUrl()* kullanarak

• Geçerli URL'de driver.navigate().to(“URL”) kullanma

veya *driver.navigate().to(driver.getCurrentUrl());*

• Web sayfasındaki herhangi bir metin kutusunda *sendKeys(Keys.F5)* kullanma

**- Test Stratejisi ve Test Planı arasındaki fark**

**Test Planı Test Stratejisi**

• Yazılım için bir test planı

proje olarak tanımlanabilir

tanımlayan bir belge

kapsam, amaç,

yaklaşım ve vurgu

bir yazılım testinde

çaba göstermek

• Test stratejisi bir dizi

açıklayan yönergeler

test tasarımı ve

nasıl test edileceğini belirler

yapılması gereken

• Test planının bileşenleri

dahil- Test planı kimliği,

test edilecek özellikler,

test teknikleri, test

görevler, özellikler başarılı veya başarısız

kriterler, test çıktıları,

sorumluluklar ve

zamanlama, vb.

• Testin Bileşenleri

strateji içerirhedefler

ve kapsam,

dokümantasyon biçimleri,

test süreçleri, ekip

Raporlama yapısı,

müşteri iletişimi

strateji vb.

• Test planı tarafından yürütülür.

bir test yöneticisi veya lider

nasıl yapılacağını anlatan

test, ne zaman test, kim

test edecek ve ne test edecek

• Bir test stratejisi yürütülür

proje tarafından dışarı

müdür. ne diyor

teknik türü

takip et ve hangisi

test edilecek modül

• Test planı şunları anlatır:

şartname

• Test stratejisi anlatıyor

genel hakkında

yaklaşımlar

• Test planı değişebilir • Test stratejisi değiştirilemez

değişti

• Test planlaması yapılır.

olası sorunları belirlemek

ve bağımlılıklar

tanımlamak için

riskler.

• Uzun vadeli bir plandır.

action.Sen soyutlayabilirsin

olmayan bilgi

projeye özel ve koymak

test yaklaşımı içine

• Bir test planı mevcuttur

bireysel olarak

• Daha küçük projede, test edin

strateji genellikle bulunur

bir testin bir bölümü olarak

plan

• Projede tanımlıdır.

seviye

• Organizasyonda belirlenir

seviye ve tarafından kullanılabilir

birden fazla proje

**Selenium'da tarayıcı penceresini nasıl büyütebiliriz?**

Selenyumdaki tarayıcı penceresini büyütmek için büyütme *kullanıyoruz*

*()* yöntemi. Bu yöntem, geçerli pencereyi büyütüyorsa,

zaten maksimize edilmemiş

**driver.manage().window().maximize();**

**- Selenium'da geçerli sayfa URL'si nasıl alınır?**

Geçerli sayfa URL'sini getirmek için *getCurrentURL() kullanıyoruz*

**- Bir tarayıcıda ileri geri gezinebilir miyim?**

**Selenyum Web Sürücüsü?**

Gezinme arabirimini, bir

tarayıcı. Geriye, ileriye ve geriye gitme yöntemleri vardır.

bir sayfayı yenileyin.

**driver.navigate().forward();** - sonraki web'e gitmek için

tarayıcının geçmişine atıfta bulunan sayfa

**driver.navigate().back();** - öncekine döner

tarayıcının geçmişine atıfta bulunan web sayfası

**driver.navigate().refresh();** - mevcut web'i yenilemek için

sayfa böylece tüm web öğelerini yeniden yükler

**driver.navigate().to(“url”);** – yeni bir web tarayıcısı başlatmak için

pencere ve belirtilen URL'ye gidin

**- Fark ne**

**driver.get *()* ve *driver.navigate.to(“url”) arasında* mı?**

*driver.get():* Bir URL'yi açmak için, tamamına kadar bekleyecektir.

sayfa yükleniyor

*driver.navigate.to():* Bir URL'ye gitmek için

tüm sayfa yüklenene kadar bekleyin

**- *driver.get()* yönteminin alternatifi nedir?**

**Selenium WebDriver kullanarak bir URL açılsın mı?**

*driver.get(“url”)* yöntemine alternatif yöntem

driver.navigate.to *(“url”)*

**- Bir test yürütmesini 5 saniye boyunca nasıl duraklatabilirsiniz?**

**belirli nokta?**

**Java.lang.Thread.sleep** *(uzun milisaniye)* yöntemini kullanarak

yürütmeyi belirli bir süre için duraklatabiliriz. duraklatmak için 5

saniye, parametreyi 5000 (5 saniye) olarak geçmemiz gerekiyor

**- Selenium'da metin kutusundaki ENTER tuşuna nasıl basılır**

**Web Sürücüsü?**

Selenium WebDriver kullanarak ENTER tuşuna basmak için,

sabit ENTER ile Selenium Enum Keys'i kullanın.

**- Selenium WebDriver kullanarak bir form nasıl gönderilir?**

Bir form göndermek için öğede “gönder” yöntemini kullanıyoruz

**- Selenium kullanarak bir köprüye nasıl tıklanır**

**Web Sürücüsü?**

Köprüye tıklamak için Selenium'da click() yöntemini kullanıyoruz

**- Selenium kullanarak bir öznitelik değeri nasıl elde edilir**

**Web Sürücüsü?**

getAttribute(değer) kullanarak;

Parametre olarak iletilen özelliğin değerini döndürür.

HTML:

1 <input name="nameSelenium" value="valueSelenium">Yazılım Test Malzemesi</input>

Selenyum Kodu:

1

2

3

String nitelikValue = driver.findElement(By.name("nameSelenium")).getAttribute("değer");

System.out.println("Kullanılabilir öznitelik değeri:"+attributeValue);

Çıktı: değerSelenyum

**- Bir web öğesinin metni nasıl alınır?**

getText() yöntemini kullanarak

**- Metin kutusundaki metni kullanarak nasıl temizlenir**

**Selenyum Web Sürücüsü?**

clear() yöntemini kullanarak

**- Arama yapmadan metin kutusuna metin nasıl girilir**

**sendKeys()?**

**- Selenium kullanan metin kutusuna metin nasıl girilir**

**Web Sürücüsü?**

sendKeys() yöntemini kullanarak

**- Selenium WebDriver'da Akıcı Bekleme Nedir?**

Deneyimlerime göre FluentWait maksimum miktarı tanımlayabilir

ile belirli bir durum ve sıklık için bekleme süresi

atmadan önce durumu kontrol etmek için

" *ElementNotVisibleException* " istisnası.

Sözdizimi

Örnek

**- Selenium WebDriver'da WebDriver Wait Nedir?**

WebDriverWait ile belirli bir öğeye uygulanır

tanımlanmış *beklenen durum* ve *zaman* . Bu bekleme sadece uygulanır

belirtilen öğeye. Bu bekleyiş aynı zamanda bir

bir öğe bulunamadığında istisna.

Bunlar, Açık Beklemede kullanılabilecek bazı Koşullardır.

1. uyarıIsPresent()

2. elementSelectionStateToBe()

3. elementToBeClickable()

4. elementToBeSelected()

5. frameToBeAvaliableAndSwitchToIt()

6. görünmezlikOfTheElementLocated()

7. görünmezlikOfElementWithText()

8. entityOfAllElementsLocatedBy()

9. entityOfElementLocated()

10. textToBePresentInElement()

11. textToBePresentInElementLocated()

12. textToBePresentInElementValue()

13. başlıkIs()

14. başlıkİçerir()

15. görünürlükOf()

16. GörünürlükOfAllElements()

17. görünürlükOfAllElementsLocatedBy()

18. GörünürlükOfElementLocated()

pratik örnek

**- Selenium WebDriver'da Örtülü Bekleme Nedir?**

Örtük beklemeler, WebDriver'a belirli bir süre beklemesini söyler.

bir istisna oluşturmadan önce geçen süre. Bir kez ayarladık

zaman, WebDriver öğeye dayalı olarak öğeyi bekleyecektir.

bir istisna atmadan önce belirlediğimiz zaman. Varsayılan ayar

0 (sıfır). yapmak için biraz bekleme süresi ayarlamamız gerekiyor

WebDriver gerekli süreyi beklemek için.

pratik örnek

**- Selenium'da bulunan bekleme türleri nelerdir?**

**Web Sürücüsü?**

Selenium'da Implicit gibi üç tür bekleme görebiliriz.

Bekler, Açıkça Bekler ve Akıcı Bekler.

• Örtülü Beklemeler – Ayrıntılı gönderiyi görüntülemek için tıklayın

• Açık Beklemeler – Ayrıntılı gönderiyi görüntülemek için tıklayın

• Akıcı Bekleyişler – Ayrıntılı gönderiyi görüntülemek için tıklayın

**- Karşılaştığınız farklı istisnalar nelerdir?**

**Selenium WebDriver'da?**

Mevcut projemde karşılaştığım istisnalardan bazıları:

1. ElementNotVisibleException

2. StaleElementReferenceException

**Öğe Görünmez İstisna:**

Bu istisna, bir yer bulmaya çalıştığınızda atılacaktır.

şu anda görünmeyen web sayfasındaki belirli öğe

DOM'da mevcut olmasına rağmen. Ayrıca bazen, eğer

xpath ile bir öğeyi bulmaya çalışıyorlar

iki veya daha fazla elemanla ilişkilendirir.

**Eski Öğe Referans İstisnası:**

bir öğe referans istisnası atıldı

durumlarda, birincisi ikincisinden daha yaygındır.

Eski öğe referansının iki nedeni:

1. Öğe tamamen silindi.

2. Öğe artık DOM'a bağlı değildir.

Bu eski öğe referans istisnasıyla karşı karşıya kalırız.

Etkileşimde olduğumuz element yok edilir ve sonra yeniden yaratılır.

Yeniden. Bu olduğunda, öğenin referansı

DOM bayat olur. bu nedenle alamıyoruz

elemana atıfta bulunur.

Genellikle karşılaştığımız diğer bazı istisnalar şunlardır:

• WebDriverException

• IllegalStateException

• Zaman aşımıİstisnası

• NoAlertPresentException

• NoSuchWindowException

• NoSuchElementException

***Web sürücüsü sürücüsü = yeni* kod satırını açıklayın**

***FirefoxSürücü();* ?**

1 Web sürücüsü sürücüsü = yeni FirefoxDriver();

' *WebDriver* ' bir arayüzdür ve biz bir nesne yaratıyoruz.

FirefoxDriver sınıfının bir nesnesini başlatan WebDriver yazın.

**- Mevcut doğrulama noktaları nelerdir?**

**Selenyum?**

Selenium WebDriver'da yerleşik özellik yoktur.

doğrulama noktaları. Tamamen kodlama stilimize bağlı.

Doğrulama noktalarından bazıları

• Sayfa başlığını kontrol etmek için

• Belirli bir metni kontrol etmek için

• Belirli bir öğeyi kontrol etmek için (metin kutusu, düğme, bırakma

aşağı vb.)

**- Selenium'da Yumuşak İddia ve Sert İddia nedir?**

**Soft Assert: Soft Assert,** *@Test* Soft sırasında hataları toplar

Assert, bir assert başarısız olduğunda bir istisna oluşturmaz ve

assert ifadesinden sonraki adımla devam eder.

**Hard Assert** : Hard Assert atar

*AssertException* bir assert ifadesi olduğunda hemen

başarısız olur ve test paketi sonraki *@Test ile devam eder*

**- Assert ve Verify arasındaki fark nedir?**

**Selenyum'da mı?**

**Assert:** Basit bir deyişle, assert koşulu doğruysa, o zaman

program kontrolü bir sonraki test adımını yürütecektir, ancak

koşul yanlış, yürütme duracak ve bir sonraki test adımı

idam edilmeyecektir.

**Doğrulayın:** Basit bir deyişle, testte herhangi bir duraklama olmayacak

doğrulama koşulu doğru veya yanlış olsa bile yürütme.

**- Mutlak Yol ile Mutlak Yol arasındaki fark nedir?**

**Göreceli yol?**

Mutlak XPath kök düğümden başlar ve biter

istenen alt öğenin düğümü ile. üst ile başlar

HTML düğümü ve giriş düğümü ile biter. bir ile başlar

aşağıda gösterildiği gibi tek eğik çizgi (/).

1 /html/body/div[3]/div[1]/form/table/tbody/tr[1]/td/input

Göreli XPath, HTML arasındaki herhangi bir düğümden başlar

geçerli öğenin düğümüne sayfa (son düğümü)

eleman). Gösterildiği gibi çift eğik çizgi(//) ile başlar

aşağıda.

1 //giriş[@id='eposta']

**- “/” ve “//” arasındaki fark nedir?**

**Tek Eğik Çizgi “/” –** XPath oluşturmak için tek eğik çizgi kullanılır.

mutlak yol, yani XPath başlamak için oluşturulur

belge düğümünden/başlangıç düğümünden seçim.

**Double Slash “//” –** XPath oluşturmak için çift eğik çizgi kullanılır

göreli yolla yani XPath başlamak için oluşturulur

belgenin herhangi bir yerinden seçim.

**- XPath nedir?**

Öğeleri bulmak için XPath kullanılır. XPath kullanarak,

bir XML belgesindeki öğeler ve nitelikler arasında gezinme

metin kutusu, düğme, onay kutusu gibi web öğelerini bulmak için

Bir web sayfasındaki resim vb.

**- Selenium'da bulunan Konumlandırıcılar nelerdir?**

Selenium WebDriver'da 8 farklı tür vardır.

konum belirleyiciler:

1. Kimlik – Pratik örnek

2. ClassName – Pratik örnek

3. İsim – Pratik örnek

4. Etiket Adı – Pratik örnek

5. LinkText – Pratik örnek

6. PartialLinkText – Pratik örnek

7. XPath – Pratik örnek

8. CSS Seçici – Pratik örnek

Konumlandırıcılar hakkındaki ayrıntılı gönderiyi görmek için buraya tıklayın .

**- Selenium Grid'i ne zaman kullanıyorsunuz?**

Selenium Grid, aynı veya farklı testi yürütmek için kullanılabilir

birden fazla platformda ve tarayıcıda aynı anda komut dosyaları

dağıtılmış test yürütme elde etmek için

**- Selenium Grid'in avantajları nelerdir?**

Test senaryolarının paralel olarak çalıştırılmasına izin vererek testten tasarruf sağlar

uygulama vakti.

Çoklu tarayıcı testine izin verir

Test senaryolarını çoklu platformda yürütmemizi sağlar

**- Selenium Grid'de hub nedir?**

Hub, testi kontrol eden bir sunucu veya merkezi bir noktadır.

farklı makinelerde yürütme.

**- Selenium Grid'deki bir düğüm nedir?**

Düğüm, göbeğe bağlı olan makinedir. olabilir

Selenium Grid'de birden çok düğüm olun.

**====== MAVEN RÖPORTAJ SORULARI ====**

**- Maven tarafından gerçekleştirilen görevler nelerdir?**

Yukarıda bahsedilen tüm problemler Maven tarafından basitleştirilmiştir ve

aşağıdaki gibi çeşitli görevleri yerine getirir:

• Maven kullanılarak kolayca proje oluşturulabilir.

• Maven tarafından tek tip bir inşa süreci sağlanır.

• Proje ile ilgili tüm bilgiler Maven tarafından sağlanmaktadır.

Günlük belgesi, çapraz başvurulan kaynaklar, posta listesi gibi bilgiler,

bağımlılık listesi, birim test raporları vb.

• Maven'in yeni özelliklerinin taşınması kolaydır.

• Apache Maven aşağıdaki özniteliklerin yönetilmesine yardımcı olur-

• Yapılar

• Belgeler

• Raporlama

• SCM'ler

• Sürümler

• Dağıtım

**- Kullanmakta olduğunuz mvn sürümünü nereden biliyorsunuz?**

Aşağıdaki komutu yazın -

mvn - sürüm

**- Maven Build Yaşam Döngüsünün aşamaları nelerdir?**

Aşağıdaki aşamalar -

• **validate −** projenin doğru olduğunu ve gerekli tüm bilgilerin mevcut olduğunu onaylayın

mevcut.

• **derleme -** projenin kaynak kodunu derleyin.

• **test −** derlenmiş kaynak kodunu uygun bir birim testi kullanarak test edin

çerçeve. Bu testler, kodun paketlenmesini veya

konuşlandırılmış

• **paket −** derlenmiş kodu alın ve dağıtılabilir olarak paketleyin

JAR gibi bir format.

• **entegrasyon testi –** gerekirse paketi işleyin ve bir

entegrasyon testlerinin çalıştırılabileceği ortam.

• **doğrulama -** paketin geçerli olduğunu ve kaliteyi karşıladığını doğrulamak için tüm kontrolleri çalıştırın

kriterler.

• **kurun −** paketi yerel depoya kurun;

yerel olarak diğer projelerde bağımlılık.

• **dağıtma −** bir entegrasyon veya yayın ortamında yapılır, son durumu kopyalar

diğer geliştiricilerle paylaşmak için uzak depoya paketleyin ve

projeler.

**- Bu komut *mvn clean bağımlılık:kopya bağımlılık paketi ne işe yarar* ?**

Bu komut projeyi temizleyecek, bağımlılıkları kopyalayacak ve

*pakete* kadar tüm aşamaları yürütmek ).

**- Yerel deponuz için varsayılan konum nedir?**

~/m2./depo.

**- JAR dosyasını yerel depoya yükleme komutu nedir** ?

mvn kurulumu

**- Projenizi çevrimdışı nasıl oluşturabilirsiniz?**

Komutu kullanın -

mvn o paketi .

**========== API RÖPORTAJI ============**

**- API nedir?**

Basit bir deyişle, API duruyor

**Bir** uygulama **programlama** arayüzü için . **\_** API, bir

iki yazılım uygulaması arasındaki arayüz ve

birbirleriyle iletişim kurmak için iki yazılım uygulaması

**- API'de kullanılan yaygın protokollerden bazılarını adlandırın**

**Test yapmak?**

API Testinde kullanılan protokollerden bazıları şunlardır:

• HTTP

• DİNLENME

**- Dinlendirici API için HTTP yöntemleri nelerdir?**

**HTTP Yöntemi CRUD**

Oku

PUT Güncelle/Değiştir

YAMA Kısmi Güncelleme/Değiştirme

SİL Sil

**- SABUN ve REST arasındaki fark?**

**SABUN:**

1. SOAP, iki bilgisayarın

XML belgesini paylaşarak iletişim kurun

2. SOAP yalnızca XML biçimini destekler

3. SOAP önbelleğe almayı desteklemiyor

4. SOAP, REST'ten daha yavaştır

5. SOAP, yakından bağlantılı özel masaüstü uygulaması gibidir.

sunucuya

6. SOAP, HTTP üzerinde çalışır ancak mesajı zarflar

**DİNLENME:**

1. REST, ağ tabanlı bir hizmet mimarisi ve tasarımıdır.

yazılım mimarisi

2. REST, farklı veri formatlarını destekler

3. REST önbelleğe almayı destekler

4. REST, SOAP'tan daha hızlıdır

5. REST istemcisi tıpkı bir tarayıcı gibidir ve standart kullanır

yöntemler Bir uygulamanın içine sığması gerekir

6. REST, meta bilgileri tutmak için HTTP başlıklarını kullanır

**- Üzerinde yapılan yaygın testler nelerdir?**

**API'ler?**

API'ler üzerinde gerçekleştirdiğimiz yaygın testlerden bazıları aşağıdaki gibidir.

1. Dönüş değerinin giriş koşuluna dayalı olup olmadığını doğrulayın.

API'lerin yanıtı, isteğe bağlı olarak doğrulanmalıdır.

2. Sistemin sonucu doğrulayıp doğrulamadığını doğrulayın

API herhangi bir veri yapısını güncellerken

3. API'nin başka bir olayı tetikleyip tetiklemediğini veya

başka bir API iste

4. Dönüş değeri olmadığında API'nin davranışını doğrulayın

**- API Testinin avantajları nelerdir?**

• API Testi, GUI ile karşılaştırıldığında zaman açısından etkilidir

Test yapmak. API test otomasyonu, daha az kod gerektirir, böylece

daha hızlı ve daha iyi test kapsamı sağlar.

• API Testi, test maliyetini düşürmemize yardımcı olur. API ile

Test, GUI Testinden önce küçük hatalar bulabiliriz.

Bu küçük hatalar GUI sırasında daha da büyüyecek

Test yapmak. API Testinde bu hataları bulmak

Şirket için maliyet etkin.

• API Testi dilden bağımsızdır.

• API Testi, Temel İşlevselliği test etmede oldukça faydalıdır.

API'leri bir kullanıcı arayüzü olmadan test edebiliriz. GUI'de

Test ediliyor, uygulama hazır olana kadar beklememiz gerekiyor

Temel işlevleri test etmek için.

• API Testi, riskleri azaltmamıza yardımcı olur.

**- API Testinde tam olarak neyin doğrulanması gerekiyor?**

Temel olarak, API Testinde, API'ye şununla bir istek göndeririz:

bilinen verileri ve yanıtı analiz ediyoruz.

1. Veri doğruluğu

2. HTTP durum kodları

3. Tepki süresi

3. API'nin herhangi bir hata döndürmesi durumunda hata kodları

4. Yetkilendirme kontrolleri

5. Performans testi, güvenlik gibi işlevsel olmayan testler

test yapmak

**- API için en çok kullanılan bazı şablonları listeleyin**

**belgeler?**

API dokümantasyon şablonlarından bazıları aşağıdaki gibidir.

• Havalı

• FlatDoc

• RestDoc

• API planı

• Arduvaz

• Miredot

• Web hizmeti API Spesifikasyonu.

**- API testinde karşılaşılan temel zorluklar nelerdir?**

API testi yaparken karşılaştığımız zorluklardan bazıları şunlardır:

• Uygun parametrelerin ve bunların kombinasyonlarının seçilmesi

• Parametreleri uygun şekilde kategorize etmek

• Uygun arama sıralaması gereklidir, çünkü bu,

testlerde yetersiz kapsama

• Çıktının doğrulanması ve doğrulanması

• GUI'nin olmaması nedeniyle girdi sağlamak oldukça zordur

değerler

**- Ne zaman karşılaştığımız hata türleri nelerdir?**

**API testi yapmak?**

API testi yapılırken gözlemlenen sorunlar şunlardır:

• Stres, performans ve güvenlik sorunları

• Yinelenen veya eksik işlevsellik

• Güvenilirlik sorunları

• Uygunsuz mesajlaşma

• Uyumsuz hata işleme mekanizması

• Çok iş parçacıklı sorunlar

• Uygun olmayan hatalar

**- En sık kullanılan bazı HTTP yöntemlerini adlandırın?**

HTTP yöntemlerinden bazıları şunlardır:

**GET:** Bir sunucudan veri almanızı sağlar.

**POST:** Mevcut bir dosyaya veri eklemenizi veya

sunucudaki kaynak

**PUT:** Bir sunucudaki mevcut bir dosyayı veya kaynağı değiştirmenizi sağlar.

**DELETE:** Bir sunucudan veri silmenizi sağlar.

**YAMA:** Bir kaynağa kısmi değişiklikler uygulamak için kullanılır

**SEÇENEKLER:** İletişim seçeneklerini tanımlamak için kullanılır.

hedef kaynak için

**HEAD:** Bir GET'inkine benzer bir yanıt ister.

istek, ancak yanıt gövdesi olmadan

**- Oluşturmak için PUT yerine GET isteğini kullanabilir misiniz?**

**kaynak?**

Hayır, GET isteği yalnızca salt okunur haklara izin verir. Size olanak tanır

bir sunucudan veri almak, ancak bir kaynak oluşturmak için değil. KOY

veya bir kaynak oluşturmak için POST yöntemleri kullanılmalıdır.

**- PUT ve POST arasındaki fark nedir**

**yöntemler?**

PUT ve POST yöntemleri bazen

her biri ne zaman kullanılmalıdır. POST isteğini kullanarak amacımız

PUT ile sunucuda yeni bir nesne oluşturmak için

istek, amacımız bir nesneyi başka bir nesneyle değiştirmektir.

İstemci sayfayı kullanıcıya gönderdiğinde POST kullanılmalıdır.

sunucu ve ardından sunucu, istemcinin onu nereye koyduğunu bilmesini sağlar.

İstemci konumunu belirttiğinde PUT kullanılmalıdır.

sayfa

**- API test yaklaşımını açıklayın.**

• API'ler için uygun test senaryoları yazın ve aşağıdaki gibi test tekniklerini kullanın:

doğrulamak için sınır değer analizi, denklik sınıfı vb.

işlevsellik.

• İki veya daha fazla katma değerli parametre kombinasyonunun çağrılarını doğrulayın.

• API programının kapsamını ve temel işlevlerini tanımlayın.

• Doğru giriş parametrelerini tanımlayın.

• Test senaryosu yürütme ve sonuçların beklenen sonuçlarla karşılaştırılması.

• Dosyalarla bağlantı vb. koşullar altında API davranışını belirleme.

**- Bir API'de gözlemlenen farklı çıktı türlerini kısaca açıklayın.**

Ana düşünce, her türlü koşulda doğru sonuçların döndürülmesidir.

Temel olarak, bir API'nin gözlemlenen çıktısı veya sonuçları şu şekilde üç bölüme ayrılır:

şöyle:

• Sonuç durum değerlerinin ' Başarılı ' veya ' Başarısız ' olarak döndürülmesi .

• Veri veya herhangi bir özel bilgi olarak sonuç.

• Herhangi bir API işlevine yapılan çağrının, başka bir API işlevine yapılan çağrıyı başlatacağı bir olay

API işlevi.

**- API testini ve Birim Testini ayırt edin.**

API testi ile Birim testi arasındaki fark, aşağıdakilerden anlaşılabilir:

aşağıdaki tablo:

**BİRİM testi API Testi**

Birim testi genellikle

her işlevselliğin olduğu test cihazları

ayrı ayrı test edilmiştir.

API testi, test kullanıcıları tarafından gerçekleştirilir

uçtan uca test için

işlevsellik.

Sınırlı bir kapsama sahip oldukları için

test, bu nedenle temel işlevler

sadece test için düşünülmüştür.

Daha geniş bir kapsama sahip oldukları için

test, işlevsel olan tüm sorunlar

test için düşünüldü.

Beyaz kutu testinin bir şeklidir. Kara kutu testinin bir şeklidir.

Genellikle, birim testi testten önce yapılır.

kod yapıya dahildir.

API testi, derlemeden sonra gerçekleştirilir

test için hazırdır.

Kaynak kodu bu formda yer almaktadır

test etmek.

Kaynak kodu bu formda yer almıyor

test etmek.

**============ SQL RÖPORTAJI =======**

**GUI Testi ile GUI Testi arasındaki fark nedir?**

**Veritabanı Testi?**

• GUI Testi, AKA Kullanıcı Arayüzü Testi veya Ön Uçtur

test yapmak

Veritabanı Testi, AKA arka uç testi veya veri testidir.

• GUI Testi, test edilebilir tüm öğelerle ilgilenir.

Kullanıcıya Menüler, Formlar vb. etkileşimlere açıktır.

Veritabanı Testi, test edilebilir tüm öğelerle ilgilenir.

genellikle kullanıcıdan gizlenir.

• GUI Testini gerçekleştiren test cihazının

Yapılandırılmış Sorgu Dilini bilmek

Veritabanı Testini gerçekleştiren test kullanıcısının,

Yapılandırılmış Sorgu Dilini bilmek

• GUI Testi, metin kutularının geçersiz kılınmasını içerir, kontrol edin

kutular, düğmeler, açılır menüler, formlar vb., büyük ölçüde görünüm

ve genel uygulamanın hissi

Veritabanı Testi, verilerin bütünlüğünü doğrulamayı içerir.

arka tarafta bulunan verilerle ön uçtaki veriler

son. Şemayı, veritabanı tablolarını, sütunları doğrular,

dizinler, saklı yordamlar, tetikleyiciler, veri çoğaltma,

yetim kayıtları, önemsiz kayıtlar. Güncellemeyi içerir

bir veritabanına kaydeder ve aynısını ön tarafta doğrular

son.

**- Veritabanında Tablo Nedir?**

Tablo, kayıtları bir yerde depolamak için kullanılan bir veritabanı nesnesidir.

Verileri tutan sütunlar ve satırlar biçimindeki alan.

**- Veritabanında Alan Nedir?**

Veritabanı tablosundaki bir alan, bir veriyi depolamak için ayrılmış bir alandır.

bir tablo içinde belirli bir kayıt.

**- Tablodaki sütun nedir?**

Sütun, bir tablodaki tüm öğeleri içeren dikey bir varlıktır.

tablodaki belirli bir alanla ilişkili bilgiler.

**- RDBMS nedir?**

RDBMS standları

**İlişkisel** Veri **Tabanı Yönetim** Sistemi için . **\_** \_ RDBMS (şimdiki değeri)

tabanlı bir veritabanı yönetim sistemi (DBMS)

ilişkisel modelde. İlişkisel veritabanından alınan veriler

Yapılandırılmış Sorgu Dili (SQL) kullanılarak erişilebilir

**- Farklı SQL komutları türleri nelerdir?**

SQL komutları aşağıdaki türlere ayrılmıştır:

• DDL – Veri Tanımlama Dili

• DML – Veri Manipülasyon Dili

• DQL – Veri Sorgulama Dili

• DCL – Veri Kontrol Dili

• TCL – İşlem Kontrol Dili

**- SQL'deki farklı DDL komutları nelerdir?**

DDL komutları yapıyı tanımlamak veya değiştirmek için kullanılır

veritabanı.

• OLUŞTUR: Veritabanları ve veritabanı nesneleri oluşturmak için

• ALTER: Mevcut veritabanı nesnelerini değiştirmek için

• DROP: Veritabanlarını ve veritabanları nesnelerini bırakmak için

• TRUNCATE: Bir tablodaki tüm kayıtları kaldırmak, ancak onun

veritabanı yapısı

• RENAME: Veritabanı nesnelerini yeniden adlandırmak için

**- SQL'deki farklı DML komutları nelerdir?**

DML komutları, mevcut verileri yönetmek için kullanılır.

veritabanı.

• SEÇ: Bir veritabanından belirli verileri seçmek için

• EKLE: Bir tabloya yeni kayıtlar eklemek için

• GÜNCELLEME: Mevcut kayıtları güncellemek için

• SİL: Bir tablodan mevcut kayıtları silmek için

**- SQL'deki farklı TCL komutları nelerdir?**

Yapılan değişiklikleri yönetmek için TCL komutları kullanılır.

DML ifadeleri ile.

• COMMIT: Değişiklikleri veritabanına yazmak ve saklamak için

• GERİ DÖNDÜRME: Son işlemden bu yana veritabanını geri yüklemek için

**- Alt Sorgu nedir?**

Bir Alt Sorgu, başka bir sorgu içindeki bir SQL sorgusudur. Bu bir

dönüş değerleri olan bir Select ifadesinin alt kümesi

ana sorgunun koşullarını filtrelemede kullanılır.

**- Bir sorguda yinelenen kayıtlardan nasıl kaçınılır?**

SQL SELECT DISTINCT sorgusu yalnızca döndürmek için kullanılır.

benzersiz değerler Tüm yinelenen değerleri ortadan kaldırır.

Ayrıntılı Gönderiyi Görüntüle

**- Yeniden Adlandır ve Takma Ad arasındaki fark nedir?**

'Yeniden adlandır', bir tabloya verilen kalıcı bir addır veya

kolon

'Takma ad', bir tabloya veya sütuna verilen geçici bir addır.

**- Katılmak nedir?**

Katıl, ilgili sütunları veya satırları alan bir sorgudur.

çoklu tablolar.

**- Farklı birleşim türleri nelerdir?**

Birleştirme türleri aşağıdaki gibidir:

• İÇ BİRLEŞTİRME

• SOL BİRLEŞTİR

• DOĞRU BİRLEŞTİRME

• DIŞ BİRLEŞTİRME

Tüm Gönderiyi Görüntüle

**- İç ve dış arasındaki fark nedir?**

**katılmak?**

En azından bazı olduğunda bir iç birleşim satırları döndürür

iki (veya daha fazla) tablo arasında eşleşen veriler

karşılaştırıldı.

Bir dış birleştirme, her iki tablodan aşağıdakileri içeren satırları döndürür:

tabloların birinden veya her ikisinden eşleşmeyen kayıtlar.

**- SQL kısıtlamaları nelerdir?**

SQL kısıtlamaları, bazılarını zorlayan kurallar kümesidir.

verilerin eklenmesi, silinmesi veya güncellenmesi sırasında kısıtlama

veritabanları.

**- SQL'de mevcut olan kısıtlamalar nelerdir?**

SQL'deki bazı kısıtlamalar şunlardır: Birincil Anahtar,

Yabancı Anahtar, Benzersiz Anahtar, SQL Null Değil, Varsayılan, Kontrol

ve Dizin kısıtlaması.

**- Birincil Anahtar nedir?**

Bir *PRIMARY KEY* kısıtlaması, bir dizideki her kaydı benzersiz bir şekilde tanımlar.

veritabanı tablosu. Birincil anahtara katılan tüm sütunlar

kısıtlama NULL değerler içermemelidir.

**- Bir tablo birden fazla *BİRİNCİL ANAHTAR içerebilir* mi?**

Kısa cevap hayır, bir tablonun içermesine izin verilmez

birden fazla birincil anahtar ancak bir bileşik olmasına izin verir

iki veya daha fazla sütundan oluşan birincil anahtar.

**- Bileşik *BİRİNCİL ANAHTAR* nedir?**

Bileşik *BİRİNCİL ANAHTAR* , üzerinde oluşturulan bir birincil anahtardır.

içinde birden fazla sütun (birden çok alanın birleşimi)

bir masa.

**- *YABANCI ANAHTAR* nedir?**

*YABANCI ANAHTAR* , iki tabloyu birbirine bağlamak için kullanılan bir anahtardır .

Bir tablodaki *YABANCI ANAHTAR BİRİNCİL ANAHTAR ile bağlantılıdır.*

Başka bir tablonun *ANAHTARI .*

**- Bir tablo birden fazla *YABANCI ANAHTAR içerebilir* mi?**

Bir tabloda birçok *YABANCI ANAHTAR olabilir.*

***UNIQUE* arasındaki fark nedir?**

**ve *BİRİNCİL ANAHTAR* kısıtlamaları?**

Bir tabloda yalnızca bir *BİRİNCİL ANAHTAR* olmalıdır

oysa herhangi bir sayıda *EŞSİZ* Anahtar olabilir.

*BİRİNCİL ANAHTAR NULL değerlere* izin vermezken Benzersiz

tuşu NULL değerlere izin verir.

**- *NULL* değeri nedir?**

*NULL* değeri olan bir alan, değeri olmayan bir alandır .

NULL değeri, sıfır değerinden *veya*

boşluklar içerir. *NULL* değeri olan bir alan ,

kayıt oluşturulurken boş bırakılmıştır. Farz etmek,

bir tabloda isteğe bağlı bir alan var ve

isteğe bağlı bir değer eklemeden bir kayıt ekleyin

alan daha sonra alan *NULL* değeriyle kaydedilecektir.

**- NULL değeri ile Sıfır arasındaki fark nedir,**

**ve Boş alan?**

Daha önce de belirttiğim gibi, Null değer değeri olmayan alandır

sıfır değerinden ve boşluktan farklıdır.

*Boş değer* , değeri olmayan bir alandır.

*Sıfır* bir sayıdır

*Boş alan* , sağladığımız değerdir. ASCII değeri

boşluk CHAR(32).

***NULL* Değerleri Nasıl Test Edilir ?**

*NULL* değeri olan bir alan, değeri olmayan bir alandır .

değer. *NULL* değeri diğer NULL ile karşılaştırılamaz

değerler. Bu nedenle, *NULL* değerleri test etmek mümkün değildir.

=, < veya <> gibi karşılaştırma operatörleriyle. İçin

bu, *IS NULL* ve *IS kullanmamız gerekiyor*

*NULL* operatörler DEĞİL.

1 tablo\_adı NEREDE sütun\_adı SEÇİN sütun\_adı NULL;

1 tablo\_adı NEREDEN sütun\_adı SEÇİN sütun\_adı NULL DEĞİLDİR;

**- SQL *NOT NULL* kısıtlaması nedir?**

*NOT NULL* kısıtlaması, içindeki değerin

dosyalanmış bir NULL olamaz

**- Birlik ve Birlik arasındaki fark nedir**

**Tüm komut?**

***Birlik:*** Yinelenen kayıtları atlar ve yalnızca farklı olanı döndürür

iki veya daha fazla seçim ifadesinin sonuç kümesi.

***Tümünü Birleştir:*** İçindeki kopyalar dahil tüm satırları döndürür.

farklı seçim deyimlerinin sonuç kümesi.

Performans açısından Union All, Union All'dan daha hızlıdır, Union All'dan beri

kopyaları kaldırmaz. Birlik sorgusu yinelenen değerleri kontrol eder

bu yinelenen kayıtları kaldırmak için biraz zaman harcar.

**Birlik** sorgusu çalıştırırsanız .

1

2

3

Tablo1'DEN \* SEÇİN

BİRLİK

Tablo2'DEN \* SEÇİN

**Çıktı:** Toplam 19 kayıt

**Union all** sorgusunu çalıştırırsanız .

1

2

3

Tablo1'DEN \* SEÇİN

BİRLİK TÜMÜ

Tablo2'DEN \* SEÇİN

**Çıktı:** Toplam 20 kayıt

**- Sahip Olmak ve Nerede Olmak arasındaki fark nedir?**

**madde?**

Bir veritabanından veri almak için yan tümce kullanılır.

belirli kriterleri belirtirken, bir Sahiplik yan tümcesi kullanılır

belirli gereksinimleri karşılayan verileri almak için 'GROUP BY' ile birlikte

Toplama işlevleri tarafından belirtilen kriterler. nerede cümlesi

Toplama işlevleriyle kullanılamaz, ancak

cümlesi olabilir.

**- SQL'de toplama işlevleri nelerdir?**

SQL toplama işlevleri, hesaplanan tek bir değer döndürür

bir sütundaki değerlerden. Bazı agrega

SQL'deki fonksiyonlar aşağıdaki gibidir

• AVG() – Bu işlev ortalama değeri döndürür

• COUNT() – Bu fonksiyon satır sayısını döndürür

• MAX() – Bu işlev en büyük değeri döndürür

• MIN() – Bu fonksiyon en küçük değeri döndürür

• ROUND() – Bu işlev, sayısal bir alanı

belirtilen ondalık basamak sayısı

• SUM() – Bu fonksiyon toplamı döndürür

**- SQL'de string fonksiyonları nelerdir?**

SQL dize işlevleri öncelikle dize için kullanılır

manipülasyon. Yaygın olarak kullanılan SQL dizelerinden bazıları

fonksiyonlar

• UZUNLUK() – Bir metin alanındaki değerin uzunluğunu döndürür

• LOWER() – Karakter verilerini küçük harfe dönüştürür

• UPPER() – Karakter verilerini büyük harfe dönüştürür

• SUBSTRING() – Bir metin alanından karakterleri çıkarır

• LTRIM() – Dosyadaki tüm boşlukları kaldırmaktır.

dizenin başlangıcı

• RTRIM() – Sondaki tüm boşlukları kaldırmak içindir.

sicim

• CONCAT() – Concatenate işlevi birden çok

karakter dizileri birlikte

• REPLACE() – Bir dizgenin içeriğini güncellemek için.

**- Bir dosyaya yeni Çalışan ayrıntıları nasıl eklenir?**

**Aşağıdaki ayrıntıları içeren Employee\_Details tablosu**

**Çalışan\_Adı: John, Maaş: 5500, Yaş: 29?**

**- Bir tabloya 'Maaş' sütunu nasıl eklenir**

**Çalışan bilgileri?**

**- Tüm kayıtları seçmek için bir SQL Sorgusu yazın.**

**masa?**

**- Bir Veritabanından Tüm Tabloların Listesi Nasıl Alınır?**

**-**

**- Adlı tüm Oyuncuları kaldırmak için komutu yazın**

**Oyuncular masasından Sachin.**

**- Her isim bir çalışandan yalnızca bir kez nasıl alınır**

**masa?**

DISTINCT anahtar sözcüğünü kullanarak, her birini alabiliriz

isim sadece bir kez.

**- SQL çıktısında bir sütun nasıl yeniden adlandırılır**

**sorgu?**

**- Çalışan\_Adı bulmak için bir SQL Sorgusu yazın**

**maaşı 5000'e eşit veya daha fazla olan**

**aşağıdaki tablo Çalışan\_Ayrıntıları.**

**Bir tablodan benzersiz kayıtlar nasıl alınır?**

DISTINCT anahtar sözcüğünü kullanarak.

**=========== JAVA ==============**

**JAVA RÖPORTAJ SORULARI**

**Java'nın beş özelliğini listeler misiniz?**

Bazı özellikler şunları içerir: Nesneye Yönelik, Platformdan Bağımsız, Sağlam,

Yorumlanmış, Çok iş parçacıklı

**Java Mimari Neden Tarafsız?**

Derleyicisi, mimariden bağımsız bir nesne dosyası formatı oluşturur.

derlenmiş kodun varlığı ile birçok işlemcide yürütülebilir olması

Java çalışma zamanı sistemi.

**Java, Yüksek Performansı nasıl etkinleştirdi?**

Java, yüksek performansı etkinleştirmek için Just-In-Time derleyicisini kullanır. Tam zamanında

derleyici, Java bayt kodunu çeviren bir programdır.

gönderilebilecek talimatlara yorumlanması gereken talimatları içerir

doğrudan işlemciye

**Java neden dinamik olarak kabul edilir?**

Gelişen bir ortama uyum sağlamak için tasarlanmıştır. Java programları taşıyabilir

doğrulamak ve doğrulamak için kullanılabilecek kapsamlı çalışma zamanı bilgisi

çalışma zamanında nesnelere erişimleri çöz.

**Java Sanal Makinesi nedir ve bağlamında nasıl ele alınır?**

**Java'nın platformdan bağımsız özelliği?**

Java derlendiğinde, platforma özel makinede derlenmez,

yerine platformdan bağımsız bayt koduna. Bu bayt kodu dağıtılır

hangi platformda olursa olsun sanal Makine JVM JVM tarafından yorumlanır

çalıştırılıyor.

**İki Java IDE'sini listele?**

Netbeans, Eclipse, vb.

C,C++anahtar kelimelerinden farklı olarakC ,C++anahtar kelimelerinden farklı bazı Java anahtar sözcüklerini listeleyin ?

Bazı Java anahtar sözcükleri içe aktarma, süper, nihayet vb.

**Nesne ile ne demek istiyorsun?**

Nesne bir çalışma zamanı varlığıdır ve durumu alanlarda depolanır ve davranış

yöntemlerle gösterilir. Yöntemler, bir nesnenin iç durumu üzerinde çalışır ve

nesneden nesneye iletişim için birincil mekanizma.

**Sınıfı tanımla?**

Sınıf, tek tek nesnelerin oluşturulduğu mavi bir baskıdır. Bir sınıf olabilir

bir nesnenin davranışını tanımlayan alanlar ve yöntemler içerir.

**Bir sınıf ne tür değişkenlerden oluşabilir?**

Bir sınıf, Yerel değişken, örnek değişkenler ve sınıf değişkenlerinden oluşur.

**Yerel Değişken nedir?**

Yöntemler, yapıcılar veya bloklar içinde tanımlanan değişkenlere yerel denir.

değişkenler. Değişken, yöntem içinde bildirilecek ve başlatılacaktır ve

yöntem tamamlandığında yok edilecektir.

**Örnek Değişken nedir?**

Örnek değişkenler, bir sınıf içindeki ancak herhangi bir yöntemin dışındaki değişkenlerdir. Bunlar

değişkenler, sınıf yüklendiğinde başlatılır.

**Sınıf Değişkeni Nedir?**

Bunlar, herhangi bir yöntemin dışında, bir sınıfta ile bildirilen değişkenlerdir.

statik anahtar kelime.

**Singleton sınıfı nedir?**

Singleton sınıfı kontrol nesnesi oluşturma, sayıyı bir ile sınırlama ancak

durum değişirse daha fazla nesne oluşturma esnekliğine izin verir.

**Yapıcı ile ne demek istiyorsun?**

Oluşturucu, yeni bir nesne oluşturulduğunda çağrılır. Her sınıfın bir

yapıcı. Bir sınıf için açıkça bir kurucu yazmazsak, Java

derleyici, o sınıf için varsayılan bir kurucu oluşturur.

**Bir sınıf için bir Nesne oluşturmanın üç adımını listeleyin?**

Bir Object önce bildirilir, ardından somutlaştırılır ve ardından başlatılır.

**Java'da bayt veri türünün varsayılan değeri nedir?**

Bayt veri türünün varsayılan değeri 0'dır.

**Java'da kayan nokta ve çift veri türünün varsayılan değeri nedir?**

Float ve double veri türünün varsayılan değeri, C/C++ ile karşılaştırıldığında farklıdır.

Float için 0.0f ve double için 0.0d

**Bir bayt veri türü kullanıldığında?**

Bu veri türü, büyük dizilerde, özellikle tamsayılar yerine yerden tasarruf etmek için kullanılır.

bir bayt, bir int'den dört kat daha küçük olduğundan.

**Statik değişken nedir?**

Statik değişkenler olarak da bilinen sınıf değişkenleri, statik değişkenler ile bildirilir.

bir sınıftaki anahtar kelime, ancak bir yöntemin, yapıcının veya bloğun dışında.

**Erişim Değiştirici ile ne demek istiyorsunuz?**

Java, sınıflar, değişkenler için erişim düzeylerini ayarlamak için erişim değiştiricileri sağlar.

yöntemler ve yapıcılar. Bir üyenin paket veya varsayılan erişilebilirliği var

erişilebilirlik değiştiricisi belirtilmediğinde.

**Korumalı erişim değiştiricisi nedir?**

Korumalı olarak bildirilen değişkenler, yöntemler ve yapıcılar

üst sınıfa yalnızca diğer paketteki alt sınıflar veya herhangi bir alt sınıf tarafından erişilebilir.

korumalı üyelerin sınıfının paketi içindeki sınıf.

**Senkronize Erişimsiz Değiştirici ile ne demek istiyorsunuz?**

Java, Access dışındaki işlevleri sağlamak için bu değiştiricileri sağlar.

Değiştiriciler, bir yönteme erişilebileceğini belirtmek için kullanılan senkronize

bir seferde yalnızca bir iş parçacığı.

**Java Operatörü önceliğine göre, hangi operatörün kabul edildiği**

**en yüksek önceliğe sahip olmak?**

Postfix operatörleri, yani [] . en yüksek önceliğe sahiptir.

**Switch deyiminde kullanılan değişkenler hangi veri tipleri ile kullanılabilir?**

Bir switch deyiminde kullanılan değişkenler yalnızca string, enum, byte, short,

int veya karakter.

**parseInt yöntemi ne zaman kullanılabilir?**

Bu yöntem, belirli bir String'in ilkel veri türünü elde etmek için kullanılır.

**String sınıfı neden değişmez olarak kabul edilir?**

String sınıfı değişmezdir, bu nedenle bir kez oluşturulduktan sonra bir String nesnesi olamaz

değiştirilecek. String değişmez olduğundan, birçok kişi arasında güvenle paylaşılabilir.

çok iş parçacıklı programlama için çok önemli kabul edilen iş parçacıkları.

**StringBuffer neden değişken olarak adlandırılıyor?**

String sınıfı değişmez olarak kabul edilir, böylece bir kez oluşturulduktan sonra bir String

nesne değiştirilemez. Çok fazla değişiklik yapma zorunluluğu varsa

karakter dizileri için StringBuffer kullanılmalıdır.

**StringBuffer ve StringBuilder sınıfı arasındaki fark nedir?**

StringBuffer'dan daha hızlı olduğu için mümkün olduğunda StringBuilder kullanın.

Ancak, iş parçacığı güvenliği gerekliyse, StringBuffer nesnelerini kullanın.

**Normal ifadelerle desen eşleştirme için hangi paket kullanılır?**

java.util.regex paketi bu amaçla kullanılır.

**java.util.regex hangi sınıflardan oluşur?**

java.util.regex üç sınıftan oluşur - Pattern sınıfı, Matcher sınıfı ve

PatternSyntaxException sınıfı.

**sonuçlandırma yöntemi nedir?**

Bir nesnenin finalinden hemen önce çağrılacak bir yöntem tanımlamak mümkündür.

çöp toplayıcı tarafından imha. Bu yönteme sonlandırma adı verilir ve

bir nesnenin temiz bir şekilde sonlandırılmasını sağlamak için kullanılabilir.

**İstisna nedir?**

İstisna, bir programın yürütülmesi sırasında ortaya çıkan bir sorundur.

İstisnalar, iş parçacığının yöntemi boyunca konumlandırılmış işleyiciler tarafından yakalanır

çağırma yığını.

**Kontrol Edilen İstisnalar ile ne demek istiyorsunuz?**

Bu, tipik olarak bir kullanıcı hatası olan veya düzeltilemeyecek bir sorun olan bir istisnadır.

programcı tarafından öngörülmüştür. Örneğin, bir dosya açılacaksa, ancak dosya

bulunamaz, bir istisna oluşur. Bu istisnalar basitçe olamaz

derleme sırasında dikkate alınmaz.

**Çalışma Zamanı İstisnalarını Açıkla?**

tarafından muhtemelen önlenebilecek bir istisnadır.

programcı. Kontrol edilen istisnaların aksine, çalışma zamanı istisnaları

derleme sırasında göz ardı edilmiştir.

**İstisna sınıfı altındaki iki alt sınıf hangileridir?**

Exception sınıfının iki ana alt sınıfı vardır: IOException sınıfı ve

RuntimeException Sınıfı.

**Atar anahtar kelimesi ne zaman kullanılır?**

Bir yöntem kontrol edilen bir istisnayı işlemezse, yöntemin bunu bildirmesi gerekir.

throwskeyword'ü kullanma. throws anahtar sözcüğü, bir

yöntemin imzası.

**throw anahtar sözcüğü ne zaman kullanılır?**

Yeni başlatılmış bir istisna veya bir istisna olarak bir istisna oluşturulabilir.

atış anahtar sözcüğünü kullanarak az önce yakaladığınız.

**İstisna İşleme altında nihayet nasıl kullanılır?**

Nihayet anahtar sözcüğü, bir try bloğunu izleyen bir kod bloğu oluşturmak için kullanılır.

Bir istisna olup olmadığına bakılmaksızın, nihayet bir kod bloğu her zaman yürütülür.

meydana geldi.len

**Kendi istisnalarınızı yaratırken nelere dikkat etmelisiniz?**

**Java?**

Kendi istisnanızı yaratırken -

• Tüm istisnalar Throwable'ın çocuğu olmalıdır.

• Otomatik olarak uygulanan, denetlenen bir istisna yazmak istiyorsanız

Handle veya Declare Rule ile Exception sınıfını genişletmeniz gerekir.

• Bir çalışma zamanı istisnası yazmak istiyorsanız,

RuntimeException sınıfı.

**Kalıtımı tanımla?**

Bir nesnenin diğerinin özelliklerini kazandığı süreçtir. İle

kalıtım kullanımı bilgiler hiyerarşik bir düzende yönetilebilir hale getirilir.

**Süper anahtar kelime ne zaman kullanılır?**

Yöntem, üst sınıfın yöntemlerinden birini geçersiz kılarsa, geçersiz kılınan yöntem,

super anahtar sözcüğü kullanılarak çağrılabilir. Ayrıca başvurmak için kullanılabilir

gizli bir alana.

**Polimorfizm nedir?**

Polimorfizm, bir nesnenin birçok form alma yeteneğidir. en

OOP'de polimorfizmin yaygın kullanımı, bir üst sınıf referansı olduğunda ortaya çıkar.

bir alt sınıf nesnesine atıfta bulunmak için kullanılır.

**Soyutlama nedir?**

OOP'de bir sınıfı soyut yapma yeteneğini ifade eder. azaltmaya yardımcı olur

karmaşıklık ve ayrıca sistemin sürdürülebilirliğini artırır.

**Soyut sınıf nedir?**

Bu sınıflar somutlaştırılamaz ve ya kısmen uygulanır ya da

hiç uygulanmadı. Bu sınıf bir veya daha fazla soyut yöntem içeriyor

bunlar sadece bir gövdesi olmayan yöntem bildirimleridir.

**Soyut yöntemler ne zaman kullanılır?**

Bir sınıfın belirli bir yöntemi içermesini istiyorsanız, ancak asıl

alt sınıflar tarafından belirlenecek bu yöntemin uygulanması,

üst sınıftaki yöntemi soyut olarak bildirin.

**Kapsülleme nedir?**

Bir sınıftaki alanları private hale getirme ve erişim sağlama tekniğidir.

halka açık yöntemlerle alanlara. Bir alan özel olarak bildirilirse, olamaz

sınıfın dışındaki herhangi biri tarafından erişilir, böylece içindeki alanları gizler.

sınıf. Bu nedenle kapsülleme, veri gizleme olarak da adlandırılır.

**Kapsüllemenin birincil faydası nedir?**

Kapsüllemenin ana faydası, uygulanan uygulamamızı değiştirme yeteneğidir.

kodumuzu kullanan başkalarının kodunu kırmadan kodlayın. Bununla

Kapsülleme, kodumuza sürdürülebilirlik, esneklik ve genişletilebilirlik sağlar.

**Arayüz nedir?**

Arayüz, soyut yöntemlerin bir koleksiyonudur. Bir sınıf bir uygular

arabirim, böylece arabirimin soyut yöntemlerini devralır.

**Arayüzün bazı özelliklerini verir misiniz?**

Şunları içerir:

• Arayüz somutlaştırılamaz

• Bir arayüz herhangi bir kurucu içermez.

• Bir arabirimdeki tüm yöntemler soyuttur.

**Java'da Paketler Tanımlansın mı?**

Bir Paket, ilgili bir gruplama olarak tanımlanabilir.

türler sınıflar,arayüzler,numaralandırmavenotasyonlarsınıflar ,arayüzler,numaralandırmasan

erişim koruması ve ad alanı yönetimi sağlayan açıklamalar .

**Paketler neden kullanılır?**

Paketler Java'da adlandırma çakışmalarını önlemek, kontrol etmek için kullanılır.

erişim, sınıfların, arayüzlerin aranması/yerleştirilmesi ve kullanılması,

numaralandırmalar ve açıklamalar vb. daha kolay.

**Çok iş parçacıklı programdan kastınız nedir?**

Çok iş parçacıklı bir program, aynı anda çalışabilen iki veya daha fazla parça içerir.

Böyle bir programın her parçasına iş parçacığı denir ve her iş parçacığı bir

ayrı yürütme yolu.

**Thread'in oluşturulabileceği iki yol nelerdir?**

İş parçacığı şu şekilde oluşturulabilir: Runnable arabiriminin uygulanması,

İplik sınıfı.

**Applet nedir?**

Applet, bir Web tarayıcısında çalışan bir Java programıdır. Bir applet olabilir

Java API'sinin tamamına sahip olduğu için tamamen işlevsel Java uygulaması

imha etmek.

**Bir uygulama hangi sınıfı genişletir?**

Bir uygulama, java.applet.Applet sınıfını genişletir.

**Java'da çöp toplamayı açıklar mısınız?**

Belleği boşaltmak için çöp toplama kullanır. Bu nesneleri temizleyerek

artık herhangi bir program tarafından referans alınmıyor.

**Değişmez nesneyi tanımla?**

Değişmez bir nesne oluşturulduktan sonra değiştirilemez.

**Bunun kullanımını yapıcılarla açıklayın?**

Değişkenler veya yöntemlerle kullanılır ve aynı sınıfın yapıcısını çağırmak için kullanılır.

**Set Arayüzünü Açıkla?**

Yinelenen öğeler içeremeyen bir öğe topluluğudur. Set

arabirim yalnızca Koleksiyondan devralınan yöntemleri içerir ve

yinelenen öğelerin yasak olduğu kısıtlaması.

**TreeSet'i açıkla?**

Öğeleri sıralı bir düzende istediğimizde uygulanan bir Kümedir.

**Karşılaştırılabilir Arayüz Nedir?**

collections.sort kullanarak koleksiyonları ve nesne dizilerini sıralamak için kullanılır ve

java.utils. Comparable arabirimini uygulayan sınıfın nesneleri,

Sipariş verilecek.

**Fırlatma ve fırlatma arasındaki fark?**

O içerir:

• Throw, as atışların kullanıldığı durumlarda bir istisnayı tetiklemek için kullanılır.

istisna beyanı.

• Atmalar olmadan, Kontrol edilen istisna şu durumlarda işlenemez:

kontrol edilen istisna, atışlarla yayılabilir.

**Java Programı altında kullanılan aşağıdaki satırı açıklayın -**

genel statik geçersiz ana Stringargs[] Stringargs[]

Aşağıda ayrı ayrı açıklama gösterilmektedir -

• public − erişim belirtecidir.

• statik - belirli bir örneğini başlatmadan ana çağrının yapılmasına izin verir

bir sınıfın örneği.

• void − derleyiciye main tarafından hiçbir değer döndürülmediğini onaylar.

• ana − bu yöntem, bir Java programının başlangıcında çağrılır.

• String args[ ] − args parametresi, String sınıfının bir örnek dizisidir

**JRE, yani Java Runtime Environment tanımlansın mı?**

Java Runtime Environment, Java Virtual Machine'in bir uygulamasıdır.

hangi Java programlarını yürütür. için minimum gereksinimleri sağlar.

bir Java uygulaması yürütmek;

**JAR dosyası nedir?**

JAR dosyaları Java Arşiv dosyalarıdır ve birçok dosyayı tek bir dosyada toplar. O tutar

Bir kütüphanede Java sınıfları. JAR dosyaları ZIP dosya biçiminde oluşturulmuştur ve .jar'a sahiptir.

Dosya uzantısı.

**WAR dosyası nedir?**

Bu Web Arşiv Dosyasıdır ve XML, Java sınıfları ve JavaServer'ı depolamak için kullanılır.

sayfalar. JavaServer Pages, Java koleksiyonunu dağıtmak için kullanılır

Servlet'ler, Java sınıfları, XML dosyaları, statik Web sayfaları vb.

**JIT derleyicisini tanımla?**

Bilgisayar programlarının çalışma zamanı performansını aşağıdakilere dayalı olarak iyileştirir:

bayt kodu.

Nesne yönelimli programlama dili ile nesne yönelimli programlama dili arasındaki fark nedir?

**nesne tabanlı programlama dili?**

Nesne tabanlı programlama dilleri, aşağıdakiler dışında OOP'lerin tüm özelliklerini takip eder.

Miras. JavaScript, nesne tabanlı programlama dillerine bir örnektir.

**Varsayılan kurucunun amacı nedir?**

Java derleyicisi, yalnızca kurucu yoksa varsayılan bir kurucu oluşturur.

sınıfta.

**Bir kurucu nihai hale getirilebilir mi?**

Hayır, bu mümkün değil.

**Statik blok nedir?**

Statik veri üyesini başlatmak için kullanılır, Ana yöntemden önce yürütülür

sınıf yükleme sırasında.

**Kompozisyonu tanımla?**

Diğer sınıfın referansını başka bir sınıf içinde tutmak olarak bilinir.

kompozisyon.

**İşlev aşırı yüklemesi nedir?**

Bir sınıfın aynı isimde fakat farklı parametreleri olan birden fazla fonksiyonu varsa,

Yöntem Aşırı Yüklemesi olarak bilinir.

**işlevi geçersiz kılmak nedir?**

Bir alt sınıf, zaten var olan bir yöntemin belirli bir uygulamasını sağlıyorsa

üst sınıfı tarafından sağlanan, Yöntem Geçersiz Kılma olarak bilinir.

**Aşırı Yükleme ve Geçersiz Kılma arasındaki fark?**

Yöntem aşırı yüklemesi, programın okunabilirliğini artırır. Yöntem

geçersiz kılma, zaten olan yöntemin özel uygulamasını sağlar.

aşırı yükleme durumunda süper sınıf parametresi tarafından sağlanan farklı olmalıdır,

geçersiz kılma durumunda parametre aynı olmalıdır.

**Son sınıf nedir?**

Son sınıflar oluşturulur, böylece o sınıf tarafından uygulanan yöntemler kullanılamaz.

geçersiz kılındı. Miras edilemez.

**NullPointerException nedir?**

Bir null öğesinin örnek yöntemi çağrılırken bir NullPointerException atılır

nesne, boş bir nesnenin alanına erişme veya alanı değiştirme vb.

**Bir iş parçacığının bekleme durumuna girme yolları nelerdir?**

Bir iş parçacığı, uyku yöntemini çağırarak, bloke ederek bekleme durumuna girebilir.

IO'da, bir nesnenin kilidini başarısız bir şekilde almaya çalışarak veya çağırarak

bir nesnenin bekleme yöntemi. Ayrıca çağırarak bekleme durumuna girebilir.

, kullanımdan kaldırılmış askıya alma yöntemi.

Tek CPU'lu bir bilgisayarda çoklu iş parçacığı nasıl gerçekleşir?

İşletim sisteminin görev zamanlayıcısı, yürütme süresini birden çok

görevler. Yürütme görevleri arasında hızla geçiş yaparak, izlenim yaratır.

görevlerin sırayla yürütülmesini sağlar.

**Bir iş parçacığının çalıştırma yöntemini ne çağırır?**

Bir iş parçacığı başlatıldıktan sonra, Thread sınıfının başlangıç yöntemi aracılığıyla JVM

iş parçacığı başlangıçta yürütüldüğünde, iş parçacığının çalışma yöntemini çağırır.

**FileNotFoundException için ifadeleri hangi sırayla yakaladığının önemi var mı?**

**ve IOException yazılır?**

Evet öyle. FileNoFoundException, IOException'dan miras alınır.

İstisnanın alt sınıfları önce yakalanmalıdır.

**Üretmek ve uyumak arasındaki fark nedir?**

Bir görev, verim yöntemini çağırdığında hazır durumuna döner. ne zaman bir görev

uyku yöntemini çağırır, bekleme durumuna geri döner.

**Neden Vector sınıfı kullanılıyor?**

Vector sınıfı, büyütülebilir bir dizi uygulama yeteneği sağlar.

nesneler. Dizinin boyutunu bilmiyorsanız, vektör çok kullanışlıdır.

veya sadece bir ömrü boyunca boyutları değiştirebilecek birine ihtiyacınız var.

programı.

Unicode, ASCII, UTF-16 ve UTF-8'i temsil etmek için kaç bit kullanılır?

karakterler?

Unicode 16 bit gerektirir ve ASCII 7 bit gerektirir. Her ne kadar ASCII

karakter seti sadece 7 bit kullanır, genellikle 8 bit olarak gösterilir. UTF-8

8, 16 ve 18 bit desenleri kullanan karakterleri temsil eder. UTF-16, 16 bit kullanır ve

daha büyük bit desenleri.

**Wrapper sınıfları nelerdir?**

Bunlar, ilkel türlere nesne olarak erişilmesine izin veren sınıflardır.

Örnek: Tamsayı, Karakter, Çift, Boole vb.

**Pencere ve Çerçeve arasındaki fark nedir?**

Frame sınıfı, Windows'u genişleten bir ana uygulama penceresini tanımlar.

bir menü çubuğuna sahip olabilir.

**Statik ve statik olmayan değişkenler arasındaki fark nedir?**

Statik bir değişken, sınıfla değil, bir bütün olarak sınıfla ilişkilendirilir.

bir sınıfın belirli örnekleri. Statik olmayan değişkenler ile benzersiz değerler alır

her nesne örneği.

**Serileştirme ve seri durumdan çıkarma nedir?**

Serileştirme, bir nesnenin durumunu bir bayt akışına yazma işlemidir.

Seri durumdan çıkarma, bu nesneleri geri yükleme işlemidir.

**Kullanım durumları nelerdir?**

Bir programın analizinin bir parçasıdır ve bir programın

karşılaşabileceği ve programın bu durumda hangi davranışı sergilemesi gerektiği

durum.

**Bir Java programında alt sınıf kullanımını açıklar mısınız?**

Alt sınıf, tüm genel ve korumalı yöntemleri devralır ve

uygulama. Ayrıca, tüm varsayılan değiştirici yöntemlerini ve bunların

uygulama.

**Menü öğesine menü kısayolu nasıl eklenir?**

b1 adında bir buton örneği varsa, arayarak menü kısayolu ekleyebilirsiniz.

b1.setMnemonic ′ F ′ ′ F ′ , böylece kullanıcı düğmeyi tıklatmak için Alt+F'yi kullanabilir.

**Hem uygulama hem de uygulama olarak kullanılabilecek bir Java sınıfı yazabilir misiniz?**

**bir uygulama?**

Evet, uygulamaya bir ana yöntem eklemeniz yeterlidir.

**Swing ve AWT bileşenleri arasındaki fark nedir?**

AWT bileşenleri ağırdır, oysa Swing bileşenleri

hafif. Ağır bileşenler, yerel pencerelemeye bağlıdır

araç takımı. Örneğin, java.awt.Button, ağır bir bileşendir.

Unix platformu için Java platformunda çalışan gerçek bir Motif düğmesine eşlenir.

**Yapıcılar ve diğer yöntemler arasındaki fark nedir?**

Yapıcılar, sınıfla aynı ada sahip olmalıdır ve bir değer döndüremez.

Normal yöntemler birçok kez çağrılabilirken, yalnızca bir kez çağrılırlar.

**Kalıtım kullanmanın herhangi bir sınırlaması var mı?**

Evet, kalıtım süper sınıftan ve arayüzden her şeyi miras aldığından,

alt sınıfı fazla kümelemeye ve bazen hataya açık hale getirebilir.

bazı durumlarda dinamik geçersiz kılma veya dinamik aşırı yükleme.

**ArrayStoreException ne zaman atılır?**

Öğeleri farklı diziler arasında kopyalarken, kaynak veya hedef

argümanlar dizi değil veya türleri uyumlu değil, bir

ArrayStoreException atılacak.

Bir sınıfın birden fazla kurucusu varsa, bir kurucuyu diğerinden çağırabilir misiniz?

Evet, bu sözdizimini kullanın.

**Uyuma ve bekleme yöntemleri arasındaki fark nedir?**

uyku 2000 2000 kodu ; ipliği tam olarak iki saniye bir kenara koyar. kod

wait 2000 2000 , iki saniyeye kadar beklemeye neden olur. Bir iş parçacığı beklemeyi durdurabilir

notify veya notifyAll çağrısını alırsa daha erken. Bekleme yöntemi şurada tanımlanmıştır:

Object sınıfı ve uyku yöntemi, Thread sınıfında tanımlanır.

**ArithmeticException ne zaman atılır?**

ArithmeticException, tamsayı sıfıra bölündüğünde veya alındığında atılır.

bir sayının sıfırdan kalanı Asla kayan noktaya atılmaz

operasyonlar.

**Geçici değişken nedir?**

Geçici değişken, sırasında serileştirilmeyen bir değişkendir.

Serileştirme sırasında varsayılan değeriyle başlatılan ve seri hale getirme,

**senkronizasyon nedir?**

Senkronizasyon, birden çok iş parçacığının erişimini kontrol etme yeteneğidir.

paylaşılan kaynaklar java'daki senkronize anahtar kelime, hangi kilitlemeyi sağlar?

paylaşılan kaynağa karşılıklı özel erişim sağlar ve veri yarışını önler.

**Koleksiyonlar API'sı nedir?**

Collections API, işlemleri destekleyen bir dizi sınıf ve arabirimdir.

nesnelerin koleksiyonları hakkında.

**Çöp toplama, bir programın bitmeyeceğini garanti eder mi?**

**hafıza?**

Çöp toplama, bir programın süresinin bitmeyeceğini garanti etmez.

hafıza. Programların bellek kaynaklarını daha hızlı tüketmesi mümkündür.

toplanan çöplerdir. Programların nesneler oluşturması da mümkündür.

çöp toplama işlemine tabi değildir.

**Applet sınıfının hemen üst sınıfı mı?**

Panel, hemen üst sınıftır. Bir panel, içinde bir

uygulama, diğer paneller de dahil olmak üzere başka herhangi bir bileşeni ekleyebilir.

**Hangi Java operatörü doğru ilişkiseldir?**

= operatörü doğru ilişkiseldir.

**Bir break ifadesi ile bir devam arasındaki fark nedir**

**Beyan?**

Bir break ifadesi, bağlı olduğu ifadenin sonlandırılmasına neden olur.

switch,for,do,orwhile anahtarı,for,do,orwhile uygular . Devam ifadesi kullanılır

geçerli döngü yinelemesini sonlandırmak ve kontrolü döngü ifadesine döndürmek için.

**Bir değişken özel olarak bildirilirse, değişkene nereden erişilebilir?**

Özel bir değişkene yalnızca içinde bulunduğu sınıf içinde erişilebilir.

beyan.

**System sınıfının amacı nedir?**

System sınıfının amacı, sistem kaynaklarına erişim sağlamaktır.

**İlkel Java türleri listelensin mi?**

Sekiz ilkel tür şunlardır: byte, char, short, int, long, float, double ve

boole.

**AWT kapsamında kırpma ve yeniden boyama arasındaki ilişki nedir?**

Bir pencere AWT boyama iş parçacığı tarafından yeniden boyandığında, kırpmayı ayarlar.

pencerenin yeniden boyanması gereken alanı.

**Container sınıfının hemen üst sınıfı hangi sınıftır?**

Bileşen sınıfı, anında süper sınıftır.

**Java çalışma zamanı sistemi tarafından hangi sınıf istisnalar oluşturulur?**

Java çalışma zamanı sistemi, RuntimeException ve Error istisnaları oluşturur.

Bir nesnenin sonlandırma yöntemi hangi koşullar altında çöp tarafından çağrılır?

kolektör?

Çöp toplayıcı, bunu algıladığında bir nesnenin sonlandırma yöntemini çağırır.

nesne ulaşılamaz hale geldi.

**Ölü bir iş parçacığı nasıl yeniden başlatılabilir?**

Ölü bir iş parçacığı yeniden başlatılamaz.

**Hangi aritmetik işlemler bir atışla sonuçlanabilir?**

**Aritmetikİstisna?**

Tamsayı / ve %, bir ArithmeticException'ın atılmasına neden olabilir.

**Boole türünün değişkeni otomatik olarak şu şekilde başlatılır?**

Boole türünün varsayılan değeri false'tur.

**Try ifadeleri iç içe geçebilir mi?**

Evet

**ClassLoader nedir?**

Sınıf yükleyici, sınıfları yüklemekten sorumlu bir nesnedir. Sınıf

ClassLoader soyut bir sınıftır.

**Arayüz ve Soyut sınıf arasındaki fark nedir?**

Soyut bir sınıf, varsayılanı uygulayan örnek yöntemlere sahip olabilir.

davranış. Bir Arayüz yalnızca sabitleri ve örnek yöntemleri bildirebilir, ancak

varsayılan davranışı uygulayamaz ve tüm yöntemler dolaylı olarak soyuttur. Bir

interface tüm genel üyelere sahiptir ve uygulaması yoktur.

**Statik değiştiricinin imzasından kaldırılırsa ne olur?**

**ana yöntem?**

Program çalışma zamanında "NoSuchMethodError" hatası veriyor.

**Örnek olarak bildirilen bir nesne başvurusunun varsayılan değeri nedir?**

**değişken?**

Açıkça tanımlanmadığı sürece boş.

**Üst düzey bir sınıf özel veya korumalı olabilir mi?**

Hayır, üst düzey bir sınıf özel veya korumalı olamaz. Ya "kamu" olabilir

veya değiştirici yok.

**Neden sarmalayıcı sınıflarına ihtiyacımız var?**

Bunları, bir yöntemin beklediği yerde, yöntem parametreleri olarak iletebiliriz.

nesne. Ayrıca yardımcı yöntemler sağlar.

**Hata ve istisna arasındaki fark nedir?**

Hata, çalışma zamanında meydana gelen geri dönüşü olmayan bir durumdur. Gibi

OutOfMemory hatası. İstisnalar, kötü koşullar nedeniyle oluşan koşullardır.

girdi vb. Örneğin, belirtilen dosya yaparsa FileNotFoundException atılır.

yok.

**Her try bloğunu bir catch bloğunun takip etmesi gerekli midir?**

Her try bloğunu bir catch bloğunun takip etmesi gerekli değildir. BT

ardından bir yakalama bloğu veya bir nihayet bloğu gelmelidir.

**Bir iş parçacığı oluşturulduğunda ve başlatıldığında, başlangıç durumu nedir?**

Bir iş parçacığı, oluşturulduktan sonra başlangıç durumu olarak hazır durumdadır ve

başladı.

**Locale sınıfı nedir?**

Locale sınıfı, program çıktısını bir

belirli bir coğrafi, politik veya kültürel bölge.

**Senkronize yöntemler ve senkronize ifadeler nelerdir?**

Senkronize yöntemler, bir veriye erişimi kontrol etmek için kullanılan yöntemlerdir.

nesne. Senkronize edilmiş bir ifade yalnızca bir iş parçacığı tamamlandıktan sonra yürütülebilir.

senkronize edilmiş dosyada başvurulan nesne veya sınıf için kilidi aldı

Beyan.

**Çalışma zamanı polimorfizmi veya dinamik yöntem gönderimi nedir?**

Çalışma zamanı polimorfizmi veya dinamik yöntem gönderimi, bir

geçersiz kılınan bir yönteme yapılan çağrı, derleme zamanında değil, çalışma zamanında çözülür.

Bu süreçte, referans üzerinden geçersiz kılınan bir yöntem çağrılır.

bir üst sınıfın değişkeni.

**Dinamik Bağlama geç bağlama geç bağlama nedir ?**

Bağlama, bir prosedür çağrısının yürütülecek koda bağlanması anlamına gelir.

çağrıya yanıt. Dinamik bağlama, bir

verilen prosedür çağrısı, çalışma zamanında çağrının zamanına kadar bilinmemektedir.

**Yapıcı miras alınabilir mi?**

Hayır, kurucu miras alınamaz.

**ArrayList'in dizilere göre avantajları nelerdir?**

ArrayList dinamik olarak büyüyebilir ve daha güçlü ekleme ve

dizilerden ziyade arama mekanizmaları.

**LinkedList'te silme neden ArrayList'ten daha hızlı?**

Bağlantılı listede silme işlemi hızlıdır çünkü yalnızca bir sonraki işaretçiyi güncellemeyi içerir

silinen düğümden önceki düğümde ve önceki işaretçinin güncellenmesi

silinen düğümden sonraki düğüm.

**ArrayList ve LinkedList'i ne zaman kullanacağınıza nasıl karar veriyorsunuz?**

Öğeleri listenin ortasından sık sık ekleyip çıkarmanız gerekiyorsa

ve yalnızca liste öğelerine sırayla erişirseniz, LinkedList kullanılmalıdır.

Öğeleri eklemeden veya çıkarmadan rastgele erişimi desteklemeniz gerekiyorsa

sondan başka herhangi bir yerden sonra ArrayList kullanılmalıdır.

**Değer Koleksiyonu Görünümü nedir?**

Harita Arayüzü'nün değerler yöntemi tarafından döndürülen bir koleksiyondur.

haritada değerler olarak bulunan tüm nesneleri içerir.

**Özel bir kurucuyu nerede ve nasıl kullanabilirsiniz?**

Özel kurucu, diğer sınıfların aşağıdakileri başlatmasını istemiyorsanız kullanılır.

nesne ve alt sınıflamayı önlemek için.T

**Tip döküm nedir?**

Tip dökümü, bir tipteki bir değişkene başka bir tipmiş gibi davranmak anlamına gelir.

**İpliğin yaşam döngüsünü tanımlayın?**

Bir iş parçacığı, bir programdaki bir yürütmedir. Bir iş parçacığının yaşam döngüsü şunları içerir:

• Yenidoğan durumu

• Çalıştırılabilir durum

• Çalışma durumu

• Engellenen durum

• Ölü durum

**Kısa türün aralığı nedir?**

Kısa türün aralığı - 2 1 5 215 ila 2^15 - 1'dir.

**Java, Varsayılan Argümanlara izin veriyor mu?**

Hayır, Java Varsayılan Bağımsız Değişkenlere izin vermez.

**Java'da baştaki sıfır ile gösterilen sayı hangisidir?**

Sekizli Sayılar, Java'da baştaki sıfır ile gösterilir, örnek: 06

**Java'da 0x veya 0X başta olmak üzere hangi sayı gösterilir?**

**Java programında import ifadesi nerede kullanılır?**

Paketten sonra program dosyasının başında import ifadesine izin verilir

Beyan.

Thread sınıfı altında askıya alma yöntemini açıklayın>

İş parçacığının yürütülmesini duraklatmak veya geçici olarak durdurmak için kullanılır.

**Thread sınıfı altında isAlive yöntemini açıklayın?**

Bir iş parçacığının hala çalışıp çalışmadığını öğrenmek için kullanılır.

**CurrentThread nedir?**

Geçerli iş parçacığına bir referans elde etmek için kullanılan genel bir statik yöntemdir.

**Thread sınıfı yürütme altındaki ana iş parçacığı açıklansın mı?**

Ana iş parçacığı otomatik olarak oluşturulur ve hemen çalışmaya başlar

bir program başladığında. Diğer tüm alt konuların kaynağı olan bir iş parçacığıdır.

**Bir appletin yaşam döngüsü hangi adımları içerir?**

Yaşam döngüsü aşağıdaki adımları içerir -

• Başlatma

• Başlangıç

• Durma

• Yok etme

• Boyama

**Uygulamalar altında init yönteminin rolü neden?**

Applet'i başlatır ve çağrılacak ilk yöntemdir.

**System.exit ise 0 0 ; try bloğunun sonuna yazılırsa, sonunda**

**blok hala yürütülüyor mu?**

Hayır, bu durumda nihayet bloğu yürütülmez çünkü dediğinizde

sistem.çıkış 0 0 ; kontrol hemen programdan çıkar ve böylece

sonunda asla yürütmez.

**Private olarak tanımlanan bir sınıfa paketinin dışından erişilebilir mi?**

Hayır, paketinin dışından erişim mümkün değildir.

**Statik bir yönteme veya statik bir bloka uygulanan kısıtlamalar nelerdir?**

**kod?**

Statik bir yöntem, bir örnek oluşturmadan örnek değişkenlere başvurmamalıdır.

örneğe başvurmak için "bu" işlecini kullanamaz.

**Bir Arayüz başka bir Arayüzü genişletebilir mi?**

Evet, bir Arayüz başka bir Arayüz devralabilir, bu nedenle bir Arayüz

birden fazla Arayüzü genişletin.

**Aşırı yükleme kullanılarak hangi nesne yönelimli Kavram elde edilir ve**

**ağır basan?**

polimorfizm

ilk ifade olarak bağımsız değişken olmayan yapıcıyı çağırır.

**Bir bayta çift değer atılabilir mi?**

Evet, çift değer bir bayta dönüştürülebilir b = (bayt) çift d = 232.232

========= MANTIK GÖRÜŞME SORULARI

============

"Diyelim ki sekiz özdeş topunuz var.

onlar biraz daha ağırdır ve size bir

denge ölçeği. en az kaç tane

bulmak için ölçeği kullanmanız gerekir.

daha ağır top?

**Cevap:** Kolaylık olsun diye topları 1-8 olarak adlandıralım. Önce {1,2,3} tartılırız

solda ve {4,5,6} sağda. Ortaya çıkabilecek üç senaryo var

bundan.

Sol taraf daha ağırsa, 1, 2 veya 3'ten birinin daha ağır top olduğunu biliyoruz.

Solda {1} ve sağda {2} tartın. Bunu yaparak, 1 mi yoksa 2 mi olduğunu bileceğiz

daha ağırdır. Dengelerse, o zaman 3 daha ağırdır.

Sağ taraf daha ağırsa, 4, 5 veya 6'nın daha ağır top olduğunu biliyoruz.

Solda {4} ve sağda {5} tartın. Bunu yaparak 4 veya 5 olup olmadığını bileceğiz

daha ağır. Dengelerse, 6 daha ağırdır.

{1,2,3} ve {4,5,6} dengelenirse, o zaman 7 veya 8'in daha ağır olduğunu biliyoruz

bir. Hangisinin daha ağır olduğunu bulmak için ikisini de tartın.

Sonsuz bir su kaynağınız ve 5

quart ve 3 quart kovalar, nasıl olursun

tam olarak 4 litre ölçmek? ve nedir

ihtiyacınız olan en az adım sayısı?

1. 5p'yi doldurun

2. 3 çeyrek 5'i 3p'ye aktarın

3. Boş 3p yapın

4. 5p'nin kalan 2 çeyreğini 3p'ye aktarın

5. 5p'yi doldurun

6. 1 quart 5p'yi 3p'ye aktarın

İki direğin tepesinden 80 metre (m) uzunluğunda bir kablo sarkıyor.

ikisi de yerden 50 m yüksekliktedir. arasındaki mesafe nedir?

iki kutup, bir ondalık basamağa, eğer kablonun merkezi: 10 m

yer üstünde?

Cevap: Kablo 80 m ise yarısı 40 m'dir. Ama dikkat 40

50 m'lik bir direğin tepesinden m, yerden 10 m yüksekliktedir.

Bu nedenle kablo doğrudan aşağı doğru asılıdır! kablo vardır

kendi üzerine ikiye katlanmalı ve iki kutup

çakışık ve 0 m aralıklı!

**- *3 adet anahtarın bulunduğu bir odadasınız.***

*Başka bir odada 3 ampul ve hangi anahtarın karşılık geldiğini bilmiyorsunuz*

*hangi ampul. Ampullerle odaya sadece bir kez girebilirsiniz. Yapamazsın*

*herhangi bir harici ekipman kullanın (güç kaynakları, dirençler, vb.). Nasıl buldun*

*hangi ampul hangi anahtara karşılık gelir?*

Cevap: A anahtarını 1 dakika boyunca açın. Kapatmak.

B anahtarını açın.

Ampullerin olduğu odaya girin.

Sıcak ampul A

Yanan ampul B'dir

Koyu ampul C'dir

Aşağıdaki özelliklere sahip bir depoya sahip 50 bisikletiniz var.

100 kilometre gidebilme kapasitesi...

Bu 50 bisikleti kullanarak ne kadar ileri gidebilirsiniz?

**Cevap** : Önce tüm bisikletleri 50 kilometre alın. O zamanlar,

tüm benzini bisikletlerin yarısından alın ve dökün

diğer yarısına. Artık tam dolu 25 bisikletiniz var.

tanklar.

Ardından 50 kilometre daha gidin ve tekrarlayın. Bu

şekilde toplam 350 kilometre yol kat edebilirsiniz.

1000 şişe meyve suyunuz var...

Biri zehir içerir ve tadı acıdır. nasılsın

en az sayıda yudumda mı buldunuz?

CEVAP: 10 yudum.

500 şişeden bir damla alın ve bir yudum alın.

Tadı acıysa o 500 şişeden birindedir.

Sonra aynısını 250 şişe, ardından 125 şişe için yapın,

ve saire ve saire...

Zehir olan tam şişeyi bulana kadar.

***Üç kutu var, biri sadece elma içeriyor, biri***

*sadece portakal içerir ve biri hem elma hem de portakal içerir. bu*

*kutular yanlış etiketlenmiştir, öyle ki hiçbir etiket gerçek durumu tanımlamaz.*

*etiketlediği kutunun içeriği. Sadece bir kutuyu açıp içine bakmadan*

*kutu, bir parça meyve çıkarırsın. Meyveye bakarak, nasıl*

*tüm kutuları hemen doğru şekilde etiketle?*

Cevap: Onu çizmek daha kolay.

Tüm etiketler yanlış etiketlendiğinde yalnızca 2 olası kombinasyon vardır.

Tek yapmanız gereken "Elma + Portakal" ile işaretlenmiş meyvelerden birini seçmek.

Apple ise, "Apple + Orange" öğesini "Apple" olarak değiştirin

"Elma", "Turuncu" olarak değiştirildi

"Turuncu olan, "Apple + Orange" olarak değişir

Turuncu ise, "Elma + Turuncu"yu "Turuncu" olarak değiştirin

"Apple", "Apple + Orange" olarak değiştirildi

"Turuncu" olan "Elma" olarak değişir