ile Alberto

DAVRANIŞSAL SORULAR

1. Bana kendinizden bahseder misiniz?

Öncelikle bana bu fırsatı verdiğiniz için teşekkür ederim ve zaman ayırdığınız için gerçekten çok teşekkür ederim. Benim adım Alper Aslan ama çoğunlukla Albert olarak bilinir.

• 6 yıldan fazla bir süredir **BT** endüstrisindeyim ve şu anda **ekibimde** bir **SDET'im** .

• Farklı endüstri türlerinde çalıştıktan sonra;

o Eğitim ve Finans sektörlerinde **alan** bilgisi kazandım .

o Scrum ustası ve Oracle Java Programcısı sertifikası aldım.

o **SDLC hakkında** güçlü bir anlayışa sahibim ve hem **Waterfall** hem de **Agile** ortamına aşinayım

o Aşağıdakiler dahil çeşitli test türlerinde oldukça ustayım; fonksiyonel, regresyon ve duman testi.

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

o Otomasyon çerçevesini sıfırdan geliştiriyorum. Pek çok "birçok" test komut dosyasını başarıyla tasarladım ve geliştirdim kullanarak **Veri güdümlü** , **Davranış odaklı** ve **Hibrid Çerçeveler** .

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 Sürekli entegrasyon için **JENKINS** ve farklı tarayıcılarda çoklu testler için **SELENIUM GRID** , işletim sistemleri ve paralel makineler

o Proje yönetimi ve hata takibi için **ComindWork** ama önceki projelerimde de JIRA deneyimi yaşadım.

o ve arka uç testi için **SQL**

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

o **BDD'yi** destekleyen POM'a dayalı en son "test çerçevemi" geliştirdim .

o Ayrıca, çerçevem JDBC kullanarak Veritabanı testini de destekliyor. Bugünlerde JOOQ'u öğreniyorum ve deniyorum. Yazmak gibi gerçek SQL sorguları. Biliyor musun bilmiyorum ama bu harika.

• Yumuşak beceri söz konusu olduğunda, kendimi düşünüyorum;

o **Pozitif bir** kişi, çapraz işlevli bir ekip üyesi

o **Hızlı öğrenen** ve değişen koşullara uyarlanabilir ve detay odaklı.

o Steve Jobs'un dediği gibi, " *ne kadar akıllı olursanız olun, harika insanlardan oluşan bir ekibe ihtiyacınız var* " ve efendim / hanımefendi bana inanın ben bir harika bir takım oyuncusu. Ben bir insanım.

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

o Her zaman **son teslim tarihine uyduğumdan** emin oluyorum (bunlardan herhangi birini detaylandırmamı isterseniz, daha ileri gitmekten memnuniyet duyarım ayrıntılı olarak.?)

• Bu hemen hemen benimle ilgili. Teşekkür ederim.

3. Sayfa

2. Rolünüzü açıklayın?

• POM'daki "test otomasyon çerçevemizi" geliştirip sürdürmekten ve yeni test eklemekten sorumluyum otomatik regresyon paketimizdeki vakalar.

• Esas olarak Not Verme, Devam, Davranış, Uygulama, Ebeveyn gibi Akademik özelliklerin otomasyonundan sorumluyum ve Öğrenci Modülleri. Buna ek olarak Business, "Ödeme Talep Yönetimi, Ücret Takibi, İK Dosyalama Modülleri.

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

• Mevcut projemde, geliştiriciler yeni işlevsellik eklediklerinde Regresyon testini yürütmekten sorumluyum. sprint uygulaması veya sonu.

• Şirkete katıldığımda, Otomatikleştirmemiz gereken büyük bir regresyon takımımız vardı. Otomasyon için regresyon kapsamı % 10 civarında çok düşüktü. Çok az otomatik test durumumuz vardı. Bu nedenle, Koşu için çok zaman harcıyorduk Gerileme. Tüm test senaryolarını regresyon paketinden çalıştırmak için yeterli zamanları olmadığından, birçok yeni kusurlar, bu nedenle proje için gerçekten otomasyona ihtiyaç vardı. Katıldığımdan beri uygulama işlevselliğini analiz ediyorlardı. Son iki yıldır **Regresyon takımını** analiz ediyorum, aşağıdaki gibi öncelikli test **senaryoları** buldum: işlevsellik en çok kullanılır, hangi işlevsellik en kritiktir, hangi işlevlerin manuel olarak yapılması uzun zaman alır yürütmek, sık sık bozulan işlevler.

• Otomasyon çerçevemi sıfırdan geliştirdim.

• Test senaryolarını otomatik hale getiriyorum. Test senaryolarını otomatikleştirdikten sonra, bazılarını günlük duman testi için aynı ölçüde tekrarlanacak şekilde programlıyorum. ihyaç olduğu gibi. Duman testi daha sonra raporunu bana ve ekibime gönderecek.

• **Her uygulama sürümünden önce tüm regresyon takımını çalıştırırım** . Test sonucunu analiz ederim. Başarılı-başarısız raporu veriyorum. ben Başarısız olan herhangi bir şeyin yanlış olup olmadığını görmek için yürütmeyi izleyin. Kodum nedeniyle başarısız olursa, kodumu düzeltmem gerekir. (Belki bu süre içinde uygulama çalışmıyordu ve betiğimi yanlış zamanda çalıştırdım.) Gerçekten bir kusur varsa,

kusur ve düzeltilene kadar tekrar test edin.
• Çapraz işlevli bir ekip üyesi olarak, işlevsel test uzmanlarına yardım ediyorum, onlara temel otomasyon çerçevesi, Java ve
Selenyum onları ekibin bir parçası yapmak, Hepsi ekibin üretkenliğini artırmak için. En azından test senaryolarını yürütebilirler ve
sonuçları analiz edin.
• ÇAPRAZ FONKSİYONEL bir ekip üyesi olarak, fonksiyonel test ekibine de ihtiyaç duyulduğunda yardımcı olmaya çalışıyorum.
manuel test durumları. Ve yeniden üretebileceğim herhangi bir kusur varsa, kusuru JIRA'ya kaydederim.
• Ayrıca her büyük sürümde prodüksiyon desteği veriyorum. Normalde üretim desteği hafta sonundadır. Ben geleceğim ve
dağıtımdan sonra üretim ortamında 'salt okunur' test senaryoları yürütün. Dağıtımla ilgili herhangi bir sorun varsa ben
sorunu en kısa sürede çözmek için Geliştirme ve Kalite Güvencesi ile hemen iletişime geçmeniz gerekir.
• Sprint tımar toplantısında, test edilebilir bir şey olduğundan emin olmak için kullanıcı hikayelerine her zaman geri bildirimde bulunurum ve
ölçülebilir. Örneğin: Böyle bir değişiklikten sonra bir kullanıcı hikayesi vardı ve uygulamada bu tür bir değişiklikten sonra performans
gelişmeli. İş adamlarına performans iyileştirme ile ne demek istiyorsunuz? Nasıl ölçüyorsun
gelişme? Bundan sonra daha iyi kullanıcı hikayeleri buldular (Agile'da gereklilik)
• Bunun yanı sıra, size gerçekten zevk aldığım sorumluluklarımdan birinin kullanıcı hikayesi oluşturma oturumları olduğunu söyleyebilirim, çünkü bu çok
kullanıcının bakış açısından ilginç. Çünkü uygulamayı her zaman test eden biziz. Ben düşünüyorum
son kullanıcı perspektifi. Kendimi son kullanıcının perspektifine koyarak iyi yaptığımı düşünüyorum. Bu nedenle, denediğimizde
kullanıcı hikayesi oturumları, kabul kriterlerimizi çok daha iyi hale getiriyoruz. İş Analisti, kullanıcı hikayelerinin üzerinden geçiyor,
kabul kriterlerini gözden geçiririz, sorular sorarız geri bildirimde bulunuruz, kullanıcı hikayelerini iyileştirir, böylece ekibimizi daha çok yaparız
üretken. Çünkü daha iyi ve net kabul kriterlerimiz var. Bu bizi, gereksinimlerimizi, kodumuzu daha iyi yapar.
daha net ve belirsiz olmak yerine, kullanıcı hikayesi oluşturma oturumunun kendisi açısından bazı kusurlardan kaçınıyoruz
bir kullanıcı hikayesi, açık olmayan kod yapma, bir şeyi yanlış yapma.
• Ayrıca, Çevik Scrum Ekibinin bir parçası olarak, gereksinim incelemeleri için çeşitli adım adım açıklamalı toplantılara katılıyorum
ve BA'ya değerli geri bildirimler sağlamak.
• Bu, mevcut projemdeki otomasyon mühendisi rolümün hemen hemen% 80'i kadardır.

4. sayfa

3. Günlük aktivitenizi anlatır mısınız?

• İşteki günlük aktivitelerim, Çoğunlukla sabah erken saatlerde işe giderim ve emin olmak için Duman testi sonuç raporunu kontrol ederim.
bu ortam çalışır durumda ve uygulama gün boyunca kararlı veya değil.
• Bir şeyler ters giderse, ekibime bir e-posta göndereceğim, böylece herkes gelmeden en kısa sürede ilgilenebilsinler
maksimum verimliliğe ulaşmak için çalışmak.
• Daha sonra önemli görevler veya bildirimler varsa e-postamı kontrol ediyorum, ayrıca herhangi bir toplantı olup olmadığını da kontrol ediyorum
o gün için ve ayrıca o gün hangi öncelikte yapılması gerektiğini gözden geçirmek için Jira'yı kontrol edin.
• Daha sonra, dün ne yaptığımı, bugün ne yapacağımızı konuşmak için scrum ekibimle günlük standup toplantısına katılacağım.
ve yolumda herhangi bir engel var mı? Bu toplantı yaklaşık 15 dakika sürer.
• Bundan sonra, masama geri dönüyorum ve regresyon takımlarından test senaryolarını otomatikleştirmeye başlıyorum. Ayrıca, test senaryolarını otomatik hale getiriyorum
geçilirse manuel olarak yaptıktan sonra sprint birikiminden. Sprintimizin sonuna geldik ve otomatikleştirmemizi, testimizi bitiriyoruz
Sprint Demosunu yürütmek, raporlamak, hazırlamak ve yürütmek
• Ayrıca haftada bir kodu gözden geçirmek için Kod Gözden Geçirme toplantıları yapıyoruz. bu gerçekten yardımcı oldu
• Benim için en tatmin edici, genç test uzmanları veya geliştiricilerle onları otomasyon konusunda eğittiğim rehberlik oturumlarım ve
bazen teknik olmayan beceriler. bunu kahverengi öğle yemeği olarak yapıyoruz ve genel üretkenliğimizi artırmamıza gerçekten yardımcı oluyor,
şirket kaynaklarını ve parasını korumakla kalmaz, aynı zamanda bize iş arkadaşlarımızı tanımak için değerli bir fırsat verir.
bence benim için en önemli şey günün sonunda birlikte çalıştığım insanlar.

4. Bana projenizden bahseder misiniz?

• Şu anda ekibimin özellikle arama İŞLEVSELLİĞİNE odaklandığı bir uygulama üzerinde çalışıyorum .

ÇERÇEVE

• Çerçevemde, test kodumu Java programlama dili ve Selenium WebDriver kullanarak geliştiriyordum.
• pom.xml'ye sahip olduğum bağımlılıklarımı yönetmek ve merkezileştirmek için Maven'i kullandım
• Çerçevem, her sayfa öğesinin bir sınıfa gittiği POM temel alınarak yapılandırıldı ve
bakım ve kodumu düzenli ve temiz tutmak.
o Ayrıca, uygulanmış adımımı sakladığım ayrı sınıflarım var ...
o Sürücülerim, koşucularım ve faydalı yöntemlerim için başka ayrı klasörler oluşturdum.
• Çerçevem, Davranış Odaklı Geliştirme (BDD) ve senaryo taslağını destekler.
• Hata izleme aracı olarak Jira kullanıyorum.
• Ayrıca Jenkins kullanarak sürekli entegrasyon elde ediyor ve testlerimi planlıyorum. (Y THING yaparak X THING başardım,
ve Z ÖLÇÜLEBİLİR SONUÇ elde edildi)
• Küçük bir dokunuş: Bu proje sırasındaki başarı, bir otomasyon test cihazı olarak büyümeme gerçekten yardımcı oldu.
SORUN X'i keşfetti, EYLEM Y'yi yaptı ve ardından ölçülebilir sonuç Z oldu.

5. En Büyük Başarınız Nedir?

• Başarılarımdan biri, ekip içinde büyük ve güvenilir bir ilişki kurmaktır.
• Teknik soruyorsanız: Son projeme katıldığımda, uygulamanın çok az "kimliği" vardı, bu nedenle
POM projemde bir Web sayfası öğesi bulup geliştiricilerle ve diğer ekip üyeleriyle ve tüm

birlikte, uygulamaya kendi başıma "Kimlik" koyma erişimini elde ettiğim çözümü bulduk.

- Bu benim için harikaydı, kendime ve başkalarına zaman kazandırdı. Bu nedenle, öğeleri bulmak için zaman harcamak yerine zamanımı daha fazla otomasyon testi komut dosyaları oluşturmak ve bunları yürütmek.
- Yeni araçlar bulmayı ve kullanmayı gerçekten çok seviyorum. Ekibe katılmadan önce sadece PNG formatında ekran görüntüleri almışlardı. Ama ben Geliştiriciler için sorunu daha anlaşılır hale getiren notasyonlu GIF animasyonlu görüntüleri kullanmaya başladık.

5.Sayfa

6. Neden iş arıyorsunuz? (Neden pazardasınız?)

- Mevcut projem yakında biteceği için iş arıyorum. Menajerim bana yeni aramaya başlamamı söyledi fırsatlar.

VEYA Alternatif cevap:

- Şirketim beni korumaktan hoşlanırlar. 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 konusunda gerçekten heyecanlıyım ... Daha önce yaptığımız telefon görüşmelerinde şirketin kültürü hakkında daha fazla bilgi edinmiştik. Yaparım inanılmaz bu kültürün (şirketin) bir parçası olduğum için çok mutlu ol.

8. Bundan 5 yıl sonra kendinizi nerede görüyorsunuz?

- "(ŞİRKET ADI) 'daki bu pozisyon beni gerçekten heyecanlandırdı çünkü beş yıl içinde, test konusunda derin uzmanlık ve bunun burada yapma fırsatım olacağını biliyorum. Ben de gerçekten heyecanlıyım Önümüzdeki birkaç yıl içinde daha fazla yönetimsel sorumluluk üstlenmek ve hatta bazı projelerde potansiyel olarak başı çekmek. Ya sahibim bazı harika yöneticilerle çalışacak kadar şanslıydım ve bu yüzden kendimi harika bir yönetici olarak geliştirmek, Gerçekten heyecanlıyım. "
- Birlikte büyüdüğüm ve önümüzdeki beş yıl içinde yeni zorluklar üstlenmeye devam ettiğim bir şirkette bir rol arıyorum. Ve ötesinde, unvanları kapatmıyorum, ancak daha fazla yönetim sorumluluğu almak istediğimi biliyorum. Ve nihayetinde adım at liderlik pozisyonuna.

ŞİRKET BİLİNMİYORSA

- "Yaptığım işte en iyisi olmaya kararlıyım ve becerilerimi geliştirmek için fırsatlara sahip olacağım bir yerde çalışmak istiyorum, ilginç projeler üstlenmek ve gerçekten öğrenebileceğim insanlarla çalışmak. Dünyadaki en yenilikçi düşünürlerden bazıları endüstri burada çalışıyor ve bu, burada kariyer yapmayı sevmemin büyük bir nedeni. "

9. Zayıf yönünüz nedir?

- Sanırım zayıflığım, bana ne zaman bazı sorumluluklar verildiğinde ve bunun için bir son tarih olduğunda, bir gün çalışıyorum ve gece, bazen haftada 7 gün. Bu benim aile hayatım için kötü; gerçek şu ki, işim bitmedikçe uyuyamam ödevler.

EN BÜYÜK zayıflık mı?

- En büyük zayıflığım, işime çok adanmış ve tutkulu bir insan olmamdı.
- Kulağa iyi bir kalite gibi gelse de, ekip üyelerinin aynı tutkuyu paylaşıp paylaşmadığını bilirsiniz. senin gibi çok hüsrana uğrayabilir. Sanki, insanların gereksinimlerini yerine getirmediğini gördüm, her zaman insanları bekliyordum karşılaştığımız tüm sorunlara karşı ciddiye alıyoruz.
- Bu zayıflığı düzeltmek için pratik, gerçekçi adımlar atmam gerektiğini fark ettim.
- Ve bence tüm ekibin başarısını sağlamak için diyalog ve bir tür iletişim yöntemi açabilmenin önemli olduğunu düşünüyorum. Scrum ekibimizdeki üyeler harika bir kimyaya sahip.
- Kahve toplantısı, kaçış odası, barbekü organizasyonu gibi iş dışında birçok etkinlik düzenledim sevdiğim :) .
- Bu, ekip üyelerimin aileleriyle ve özellikle çevreleriyle yakından bağlantı 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 olmak için bazı liderlik kurslarına kaydoldum ve takım üyelerimi zihniyetime kavuşturmak için.

Sayfa 6

- Sonuç olarak, scrum ekibimizin kimyasını geliştirdim ve projelerimiz en yüksek Müşteri memnuniyeti.

• Zayıflığımla mücadele konusunda gerçekten iyi bir deneyim yaşadım ve bir
sorun.

10. Güçlü yönleriniz nelerdir?

• Detaylara çok önem veren biriyim. İşimi son teslim tarihine göre önceliklendirebilirim.
• Ben de işime çok bağlıyım.
• Aynı zamanda dürüst bir insanım ve QA sürecinde beceri ve uzmanlığa sahibim.
• En güçlü yanlarımdan biri, bir grafik çizme becerisine sahip olmam, yani
Örneğin; *Bir paragrafı, olayı, vakayı veya bir cümleyi gördüğümde veya duyduğumda, hatta bir cümle bile olsa, zihnimde kolaylıkla grafiğini çizebilirim gerçek hayatta uygulayın.*
• Sunumlarda gerçekten iyiyim. Bir konuyu veya herhangi bir konuyu sunduğumda, bunu çok net ve yapılacak noktaya değiniyorum. Bana izin ver
size bir örnek vereyim;
Önceki şirketimde bir gözden geçirme toplantımız vardı ve 3 ekip üyesi vardı, üst yönetim her ekibe sordu
görüşlerini sundu. Sorumluluklarımızı ayırıyoruz ve tek tek anlatmaya hazırız ama maalesef her ekip
üye toplantı konseptini yanlış anladı çünkü ihtiyaçlar net değil ve fark ettik ki üst yönetimin
Bizi bir sunum bekliyordu ve ekip üyelerinden biri projemizi tanıtıyor ve ilk ekip sunumuna başladıktan sonra
gözden geçirdik, herkes birbirine baktı ve ekibin biraz gergin olduğunu ve
- Slack üzerine bu inceleme için bir grubumuz vardı ve onlara bir planım olduğunu düşünmeyin diye yazdım.
- Sadece konuya odaklanıyorum ve bir şablon hazırlıyorum ve ekip üyelerini gönderiyorum ve onlar kendi parçalarını hazırladılar ve birleştirdim
hepsi, o zaman 20 dakika içinde hazırız ve sonra hepsini kolayca sunabilirim.
Bu yüzden, gücüm ekibi düzenlememize ve sorunumuzu çözmemize yardımcı oluyor ve tüm ekip üyeleri bundan mutluydu çünkü biz
çok başarılı bir şekilde bitirdim ve sunuyorum ... Biz takımdık ve inanıyorum ki bir ekiple tutkuyla çalışırsanız,
elde edilebilir ve beklenen sonuca zamanında ulaşabiliriz, bu da müşteri memnuniyeti demektir, bu da demektir ki
şirket başarısı.
• Teknik En büyük güç:
o Java'da en güçlüyüm çünkü onu SEVİYORUM. nedenini açıklayın: artılar, faydalar vb.
• Nedeninin mantığını açıklayın: Kapanışlar, geri arama, vaatler benim gücümdür çünkü kapsamlı olmama uygundur
analitik becerilerim var, beynim zaten bu şekilde çalışmaya ayarlandı, bu yüzden JS dili bana çok doğal ve kolay geldi.

11. Son projeniz sırasında karşılaştığınız bir sorunu açıklayın.

• Mevcut projemde karşılaştığım en büyük zorluklardan biri sanırım ...
o ... yeni bir geliştiricimiz vardı. O genç ve ama çok akıllı bir çocuk. Ne zaman bir hata bulsam, geliştiriciyi
gergindi ve bunu kabul etmeyi kabul etmedi ve çoğu zaman onu sakinleştirmek zorunda kaldım ve bazen BA'ya sormak zorunda kaldık
netlik için
o Sonra gereksinimin kendisinin yeterince spesifik olmadığını fark ettim, bu yüzden onu geliştiriciden farklı bir şekilde anladım
o Sprint Retro'da, gereksinimlerin açıklığa kavuşturulması için daha fazla zaman harcamamız gerektiğini söyledim çünkü bunun anahtar olduğunu biliyorsunuz
proje başarısına. Biz de öyle yaptık ve bu sorun çözüldü.
• Bu pek çok şirkette pek yaygın değildir ancak şirketimizde platin paket kullanıcısı müşterilerimiz bulunmaktadır. Sahipler
bir özellik hakkında gündüzden geceye veya tam tersi 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 için 2-3 ve sabah 4'e kadar birçok kez çalışmak zorunda kaldım.
özellikleri. (gerekirse bir örnek verin)
• Sonuç üzerinde daha çok çalışın. Bence en önemli sorun yanlış anlaşılma ve iletişim eksikliğidir.
iş hayatı. Grup olarak bir araya gelir ve tartışırsak çözemeyeceğimiz hiçbir şey yoktur. Gerçekten minnettarım ve kutsandım
İşbirliği yapabildik ve sorunu çözmek için bir araya gelebildik.
• Ve karşılaştığım teknik zorluk, web sayfalarından doğru HTML kodunu alarak dinamik öğeleri bulmaktır.
Kimlikler problemdi; bazen sayfada görünür, bazen de kaybolur. Sonra ya örtük olarak koymak zorunda kaldım

7. Sayfa

veya onları bulmak için açıkça bekleyin.

TEKNİK OLMAYAN Zorluklar:

A. Çok fazla işiniz varsa ve son teslim tarihini bitiremiyorsanız ne yapmalısınız?

• Geliştiriciler kodlarını zamanında dağıtmadıklarında, test ekibimizin tamamlama için yeterli zamanı olmaz. Ve üst
yönetim bizden tamamlanmamızı istiyor. - Ekip üyelerimden bazıları sadece görevi tamamlamaya odaklanıyor,
test kapsamı ve iş kalitesi. - Bu yüzden, Sprint Bakım Toplantısında, çok çalışmamız gerektiğini önerdim.
geliştiriciyle yakından ve günlük olarak iletişim kurduğumuzdan emin olun. - Ayrıca geliştiriciler,
önemli görevler ve önce bunlar üzerinde çalışın. Geriye kalan herhangi bir senaryo, bir sonraki sprint'e gönderilecektir çünkü bu,
diğerleri kadar önemli. - Son olarak, işime öncelik vermeye ve test liderimi ve müdürü ne görürlerse görsünler takip etmeye çalışıyorum.
bununla başlıyorum daha önemli.
• Son zamanlarda karşılaştığım zorluklardan biri, aynı zamanda QA olan başka bir iş arkadaşımın kişisel nedenlerle şirketten ayrılmak zorunda kalmasıydı.
Muhtemelen bazı vize sorunları için ülkesine geri dönmesi gerekiyordu ve oradan beri görevini kendi başıma halletmek zorunda kaldım.
sadece iki otomasyon adamıydık, ben ve Jason. - Ve üretim tarihi ertelenemez. - Bu yüzden sorumluluk aldım, ben
yeni bir plan yapmaya başladım ve elimden gelenin en iyisini yapacak olan SM'imle iletişime geçtim, ancak ulaşmak için biraz yardıma ihtiyacım var.
zamanında üretim hedefi. Her zaman olduğu gibi, daha iyi ekip çalışması için birbirimizi anlamanın anahtarı iletişimdir. - Ve ayrıca,
geliştiriciler de test uygulamasına katıldı. Sonunda zamanında başardık. - Gurur duyduğum sebeplerden biri de bu
Scrum ekibinin bir üyesi olun. Herkes aynı amaç için çalışır ve sorumluluğu paylaşır.

• Gerçekten teknik zorluklarla karşılaşmıyorum çünkü teknik olan herhangi bir konuyu işimin bir parçası olarak görüyorum. Orada olacak her zaman üstesinden gelinmesi gereken bir zorluk olur ve HER ZAMAN üstesinden gelir ve öğrenirim. bence gerçekten zor olan ne ve düzeltmesi her zaman o kadar kolay değildir: PEOPLE. Yine aynı kişilerle 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 öğrenmek. - Bunu nasıl yaparım? - dikkat ederek, empati kurarak ve daha fazlasını yaparak hepsi, etkili bir şekilde iletişim kuruyor. bu, birçok insanla çok iş yapmak anlamına gelir, çünkü çatışmalar genellikle iki kişiyi içerir. veya daha fazla katılımcı ve ekibin geri kalanını olumsuz etkileyebilir, böylece üretkenliği düşürür. - mutlu çalışanlar üretken çalışanlar demektir. - Meslektaşlarımı gerçekten anlamak ve yapabilmek için deneme yanılma yoluyla öğrendim ofis siyasetinin zorlu sularında gezin. örneğin, [belirli bir örneğe gidin]

B. Çatışmayla nasıl başa çıkıyorsunuz?

• Hiçbir şey kişisel değildir. Herkes şirketin faydasını düşünüyor, bu yüzden endişemi ve onun açıklamasını açıklamak istiyorum benim için mantıklı. - Tabii ki, şirketime en çok yardımcı olan şeyleri yapabilirim. Bu yüzden iletişim kurmaya çalışıyorum onun / onun ve ben endişeyi anlamaya çalışırdım. Çünkü herkes aynı hedefe sahip ve işi bitirmek istiyor başarıyla.

12. Stresle nasıl başa çıkarsınız?

• Sprintlerimizden biri geliştiricim kodu çok geç dağıttı ve işi bitirmek için fazla zamanım olmadı. Ama ben çok çalışmak, ekstra saatler ve özellikle geceleri çok çalıştım ve görevimi zamanında bitirdim.

• İlk yaklaşımım sakinleşmeye ve fazladan saatler çalışmaya çalışmaktır. Stres yerine durumlara tepki vermeye çalışıyorum. Bu şekilde durum ele alınır ve stresli hale gelmez. Çoğunlukla bunlar stresi idare etmeme yardımcı oldu. Ben de çikolata yiyorum.

• Scrum ortamında da ekip olarak çalışıyoruz. Benimle her zaman iyi bir iletişim ve ilişki sürdürürüm meslektaşlar. Bu yüzden bana güveniyorlar ve benimle çok kolay iletişim kurabiliyorlar. Her zaman yanlış iletişimden kaçınırım ve benim takım her zaman bana inanın.

• Bazen bazı gereksinimler anlaşılabilir değildir, bu yüzden anlamaya ve gereksinimleri anlamaya çalışırım. İçinde uygulamanın başlangıcında işlevselliği anlamak için ekstra çaba harcıyorum. Bazen anlamak zaman alır.

8. Sayfa

13. İş yükünüze nasıl öncelik veriyorsunuz?

• "Zaman zaman, neyin en önemli ve acil olduğunu bilmenin zor olabileceği çok sayıda birbiriyle çelişen önceliğe sahibiz. Neyin en yüksek önceliğe sahip olduğunun açık olması için, derecelendirme görevleri için önemli / acil bir ölçek oluşturuyorum. Bazen alırım değerlendirme için takım liderimizden veya PO'dan yardım.

• Bir şey hem önemli hem de acilse, en yüksek önceliği alır. Önemli ama acil değil, sıradaki ve acil olan ama değil Sırada önemli, sonra önemli değil ve acil değil sonuncu.

• Ayrıntılara girmem gerekirse, aşağıdaki adımları takip ederim;

  o Tüm görevlerimin bir listesini toplayın ve bir yapılacaklar listesi yapın

  o ACİL ve ÖNEMLİ'yi belirleyin → Görevlerin çoğu önemlidir, ancak yalnızca bazıları zamana duyarlıdır.

  o Değeri değerlendirin → Her görev için gereken zamanı, çabayı ve kaynakları tahmin edin

  o Ne zaman keseceğinizi bilin → Çoğu zaman listemdeki her şeye ulaşamıyorum. Görevlerime öncelik verdikten ve tahminlerime baktıktan sonra, Kalan görevleri listemden çıkarıyorum ve gün için tamamlamam gereken ve tamamlayabileceğim önceliklere odaklanıyorum. Sonra ben derin bir nefes alın, dalın ve her şeye hazır olun. ☺

14. Baskı altında çalışabilir misiniz?

• Çalıştığım hiçbir projenin baskısı olmadığını hatırlamıyorum. Baskı bazen iyidir. Seni çalışmaya zorluyor daha sert ve daha akıllı.

• Üzerinde çalışacak çok sayıda görevin olması veya yaklaşan bir son tarih gibi iyi bir baskı, motive olmama ve 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 büyük projeler yapılması çok büyük bir baskı oldu. Ancak, bunun nasıl yapılacağını ayrıntılı bir program oluşturduğum için Her projeyi küçük görevlere ayırırdım, her üç projeyi de vaktinden önce tamamladım ve kaçındım gereksiz stres.

15. Ne zaman başlayabilirsiniz?

• Teklif mektubu ve başlangıç tarihini aldıktan sonra 1-3 hafta içinde başlayabileceğimi düşünüyorum.

16. Yarın başlayabilir misin?

• Mevcut şirketim için adil olmayacak ve yarın ayrılırsam ekibim benimle mutlu olmayacak ve ben öyle olduğunu düşünmüyorum profesyonel ve bunu daha önce hiç yapmadım. Çok üzgünüm yarın başlayamayacağım.

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

17. Bizden ne kadar bekliyorsunuz? veya ne kadar için bizimle çalışmaya razısınız?

• Beklentim 90-100K arasında. Bir miktar artışa sahip olmak benim için büyük motivasyon olacak. *(unutmayın 48 güven seviyenize bağlıdır. İsterseniz artırabilirsiniz. Size sormalarının nedeni, oranı onaylamak istemeleridir.*

Belirli bütçeleri olacak ve danışmanlık firmasının biraz para kazanmak için birkaç dolar kâr etmesi gerekiyor. Yani, her zaman tartışılabilir. Çok düşük derseniz, kendinize güvenmediğiniz anlamına gelir.)

18. Eğer işe alınırsanız, ne kadar kalmayı planlıyorsunuz?

- Çalışacak bir proje olduğu sürece, mümkün olduğu kadar uzun süre kalmaya hazırım. Şirketlerimle uzun vadeli çalışmayı seviyorum.

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

19. Seni işe alırsam ne yaparsın?

- İlk hafta, biliyorsunuz, tüm kağıt işlerini halledeceğim, makineleri alacağım ve projeye gerekli erişimi sağlayacağım,

  veritabanları vb.
- O zaman şirket kültürünü öğrenmem gerekecek. Biniş süreci.
- Projelerim ve takım arkadaşlarım hakkında da daha çok şey öğrenmem gerekiyor.
- Projenin ne yaptığını anlamak, daha üretken olmak istiyorsam çok önemli olduğunu düşünüyorum.

Sayfa 9

20. Referans için mevcut işvereninizle iletişime geçebilir miyim?

- Özgeçmişimi göndermek istiyorsanız lütfen devam edin (işe alım görevlilerine). Ama diğerlerinden çok fazla telefon araması alıyorum

  işe alımcılar. Özgeçmişimi gönderip göndermeyeceğinden emin olmayan kimsenin işverenimi aramasına izin vermek istemiyorum. *(eğer söylerlerse özgeçmişinizi gönderecek ve işten hemen sonra göndereceğinizi söyleyecektir.)*

21. Testle ilgili en çok neyi seviyorsunuz?

- Test yapmak benim için eğlenceli bir iş çünkü müşteri ve son kullanıcılar için çok önemli bir kişisiniz. Test etmeyi seviyorum çünkü sonunda

  kullanıcı Ben sanat bezelyesi olan ve hatasız daha iyi bir ürün satın almak istiyorum. Ayrıca, ürünlerinden emin olmaları için başkalarına yardım ediyorum.
  en iyi kaliteye sahiptir. Toyota Camry'nin güvenlik özelliklerini test ediyorsanız neredeyse 100 hayat kurtardığınızı düşünün.
  işini yapmak ve işini sevmek.

22. Neden sizi işe almalıyız?

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

  Bu röportaj sırasında bana çok yardımcı bir şekilde sağladığınız bilgiler için, GÜVENLİ BİR ŞEKİLDE,
  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 için** .
- Steve J., **"** *Ne kadar akıllı olursan ol, harika insanlardan oluşan bir takıma ihtiyacın var* **" dedi** ve bana inanın, harika bir takım oyuncusu benim.
- Ancak, bu benim için temel gereksinimdir ve en nitelikli olduğuma ve beklentileri aşacağıma gerçekten inanıyorum

  sadece not ettiğim şeyin ötesinde, çünkü her zaman ölçülemeyen ve ölçülemeyen çok önemli yumuşak becerilere sahibim. Ve ben
  İş tanımına veya ne yaptığımıza bakılmaksızın günün sonunda çalışanlarınızın kişiliği olduğuna inanıyorum.
  gerçekten önemli. Bu konuda benzersizim ve öne çıkıyorum çünkü sosyal becerilerin değerini yıllar önce öğrendim ve
  bu becerileri geliştirmede kapsamlı deneyim.
- Çalıştığım şirkete atlamamıza yardımcı olacak her zaman yeni teknikler ve araçlar getiriyorum.
- Birçok kişi kısa bir zaman dilimi içinde herhangi bir teknik beceride eğitim alabilir, ancak birini iletişim kurması için eğitmek olabilir.

  inanılmaz derecede zor. Bolca bu tür becerilere sahibim: Mükemmel bir iletişimciyim, son derece motive VE motive ediciyim.
  ve her şeyden önce, bir problem çözücünün tam tanımıyım. İşimi başarmak için yapılması gereken her neyse ve
  daha fazla, YAPACAĞIM.
- Bu pozisyon için en iyi niteliklere sahip adayı işe almanız gerektiğini düşünüyorum.
- Diğer adayları tanımadığım için sadece kendimi temsil edebilirim.
- Tecrübemin ve teknik uzmanlığımın şirkete ve projeye birçok değer ve fayda getireceğini düşünüyorum. Bence

  bu yüzden beni işe almalısın.

23. Bize sorunuz var mı?

- Etrafımın iyi / nazik, çalışkan, zeki insanlarla çevrili olması benim için çok önemli.

  çok çalışıyorum ve kendimi geliştirmeye devam ediyorum. İş arkadaşlarımızla yaptığımızdan daha fazla zaman geçirdiğimizi düşünerek
  arkadaşlarımla veya ailemle bile, iş yerimin kültürüne uyum sağlayabileceğimi ve bir
  orada profesyonel. Peki, şirket kültürünüz nasıl? Takım nasıl biri?
  (Eğer araştırma yaptıysanız, şirket hakkında ilginç bir şey ortaya atabilir ve onlardan detaylandırmalarını / nasıl çalıştığını vb. isteyebilirsiniz.
- Konumumda sürekli olarak gelişmem ve mükemmelliğe ulaşmaya çalışmam ve bunu yapmanın en iyi yolu benim için de önemli

  bu sürekli öğrenmektir. Her zaman yeni şeyler öğrenmeye veya eski şeyleri daha iyi öğrenmeye çalışıyorum. Eğitim veriyor musunuz,
  çalışanlarınızın eğitimini destekleyecek seminerler veya herhangi bir şey?

Sayfa 10

ÇERÇEVENİZİ SÖYLEYİN

1. Bana çerçevenizden bahseder misiniz?

Versiyon 1

• Son projemde çerçevemi tasarlamak için **Eclipse IDE** olarak farklı yönetim ve otomasyon araçları kullandım /

IntelliJ , **Tarayıcı otomasyonu** için **Selenium WebDriver** , bağımlılıklar için **Maven** , **Cucumber** ve **Jenkins** . Ben de kullandım
Kodumu düzenli ve temiz tutmak için **POM** yapısı. Bu nedenle, temelde her sayfa için ayrı bir **Java sınıfı** oluşturdum
o sayfanın tüm öğelerini ve ilgili yöntemleri depoladığım uygulamamın Ayrı sınıflarım var
uygulanan **adım tanımlarımı koruyun** . Ayrıca, her senaryo için (pozitif veya negatif) **Salatalık özellik dosyaları** oluşturdum
Test senaryolarımı açıklamak için **GHERKIN** dilini kullandığımda , bunu yaparak test durumlarımın
teknik olarak güçlü olmayanlar için bile ekibimin her bir üyesi için anlaşılabilir. Başka bir ayrı yarattım
tüm sürücülerimi, sayfa dosyalarımı veya çalıştırabileceğim yardımcı programları saklayacağım yardımcı programlarım için bir paket. Raporlama yapıldı
salatalık ve Jenkins'te. Aslında, Jenkins test her çalıştırıldığında salatalık tarafından oluşturulan Json dosyasını kullanıyor.
Tabii ki, tüm kodum GIT'de saklanıyor, böylece gerektiğinde ekip üyelerimle paylaşabilirim. Seveceğim son şey
Ek olarak, GIT'den (Sürüm Kontrolü) kodu alarak duman ve regresyon testlerimi çalıştırmak için Jenkins kullanıyorum.

Versiyon 2

• Bağımlılıklarımı ve **eklentilerimi pom.xml'de yönetiyorum** , testleri komut satırından çalıştırıyorum. **Java** ile yazılmıştır . kullanırım

Eclipse IDE / IntelliJ şirketimde. Benim çerçeve kullanan **Selenyum WebDriver** için **tarayıcı otomasyon** ve **JUnit** için
Benim testleri başlayan ve iddialar için, My çerçevesidir **BDD** kullanır, **Salatalık** özelliği dosyasında yazma testlere organize
@smoke, @api gibi test paketleri. Özellik dosyalarımız, **olmayanların** anlaşılmasını kolaylaştırmak için **Gherkin** dilinde yazılmıştır.
teknik insanlar. Çerçevem, özellik dosyasından tam adımlarla **HTML** raporları oluşturur .
Benim çerçevemde; Özellik dosyaları dışında, testlerimi çalıştıran ve adım için kodlar oluşturmaya yardımcı olan bir koşucu sınıfım var.
özellik dosyalarımdan tanım. Ayrıca, özellik dosyalarımın ve **StepDefs'in** (Glue) nerede olduğunu gösteren konumları da içerir .
Özellik dosyalarını yürütmek için yöntemlerimin bulunduğu adım tanımlama sınıfları, Driver sınıfı **Singleton** olarak tasarlanmıştır .
Yapılandırma Okuyucu ve özellikler dosyası. Ayrıca her sayfa için öğelerimi bulduğum sayfa sınıflarım var. dışında
bunlar, **yeniden kullanılabilir kodlarımı** sakladığım yardımcı program sayfalarım var . Son olarak, kodlarımı uygulayan kanca sınıfım var.
tüm testlerimden önce ve sonra çalıştır , her hata için **TakeScreenShot'ımı** çalıştırdığım yer **burasıdır** . Çerçevem destekler
Apache POI , **Scenario Outline, excel ve csv dosyalarını** kullanarak **Veriye Dayalı test .**
Benim çerçevemde, JDBC sürücüsü aracılığıyla Veritabanı testi de yapabilirim. **JIRA'yı** proje yönetimi olarak kullanıyorum ve
hata izleme aracı. Sürekli Entegrasyon ve test planlaması için **Jenkins kullanıyorum** . Ve çoklu kullanım için **Selenium Grid** kullanıyorum
tarayıcı paralel testi.

2. Çerçevemin özeti?

ÇERÇEVE

Veriye Dayalı Çerçeve

Testlerin veri setine dayalı olarak yürütüldüğü bir çerçeve, çerçeve gibi dış kaynaklardan gelen verileri okumak için tasarlanmıştır.
verilere göre testler yapın ve çalıştırın. Veriye dayalı çerçevede, aynı testi birden çok kez gerçekleştirebiliriz.
farklı veri kümeleri.

Sayfa Nesne Modeli Çerçevesi

Uygulamanın her sayfası için ayrı bir java sınıfı oluşturduğumuz sayfa nesnesi tasarım modelini kullanır

Davranış Odaklı Çerçeve

Anahtar Kelime Odaklı Çerçeve

KDF'de, testlerimizi çalıştırmak için dış kaynaktaki (excel. Csv) anahtar kelimeleri kullanırız. Çerçeve, verileri ve adımları okumak için tasarlanmıştır
excel'den alın ve buna göre eylemler yürütün.
KDF kurulduktan sonra, teknik olmayan test uzmanları bile otomatik testler yazabilir ve yürütebilir.

Sayfa 11

Hibrit Çerçeve

hibrit çerçeve, yukarıda verilen türlerden en az ikisini kullanan bir çerçevedir

ARAÇLAR

Java : Çerçevem Java dili kullanılarak yazılmıştır

Maven : Bir maven projesi olarak oluşturulan çerçevem, maven bağımlılıkları yönetmek ve ayrıca testlerimizi mvn olarak çalıştırmak için kullanılıyor
terminalden hedefler

Selenium WebDriver : tarayıcıyı otomatikleştirmek için kullanılan bir kitaplık / araç / api, tarayıcıyla etkileşim kurar.

TestNG : xml dosyalarını kullanarak testleri gruplamak, yumuşak ve zor iddialar yapmak, test yöntemleri oluşturmak, belirli bir sırayla çalıştırmak için kullanılır
Kapsam : benim çerçevem, teknik olmayan ekip tarafından okunması ve anlaşılması kolay olan ayrıntılı HTML raporları oluşturur
üyeler. Raporlarım, meydana gelebilecek arızalar için ayrıntılı test adımlarını ve ekran görüntülerini içerir. Ayrıca neye ilişkin metrikler de yapabilir

yüzde geçiyor, başarısız oluyor, atlanıyor vb.

IDE : Mevcut çerçevemde IntelliJ kullanıyorum, ancak daha önce kullandığım Eclipse ile de oldukça rahatım

TASARIM

Sayfa Nesne modeli : çerçevem, sayfaların sayfaları için ayrı bir sınıf oluşturduğum sayfa nesnesi modelini kullandı.
benim başvurum.

PageFactory tasarımı : Sayfa nesnesi sınıfına erişimi kolaylaştıran bir tasarım.
Sayfa fabrika tasarımının altında değil. PageFactory tasarımıyla aynı adı taşıyan sınıftır:

PageFactory.initElements (sürücü, bu)

Singleton Driver : Çerçevelerim, webdriver örneğini farklı sınıflar arasında paylaşmak için tekli bir model kullanıyor

TestBase : Benim çerçevem, testlerimin kapsamına giren bir testbase sınıfına sahip. testbase sınıfı, tüm testlerim için ortak adımlara sahiptir.

Configuration.feature dosyası : önemli test verilerini saklamak için kullanılır

Yardımcı programlar: çerçevemin farklı sınıflarında kullanılabilen yeniden kullanılabilir yardımcı programlara sahip

Faydaları :

Bakımı kolay:

• Çerçevem, bakımı kolaylaştıran sayfa nesne modeli kullanıyor. örneğin, herhangi birini güncellemem gerekirse bulucu, sadece bir kod değişikliği yapmam gerekiyor.

• Testlerimi birbirinden bağımsız yapmaya çalışırım. bu, bir testi güncellersem, diğerlerini etkilemeyeceği ve ayrıca biri başarısız olursa, diğerleri etkilenmeyecektir.

Genişletmesi kolay:

• Çerçeveme yeni test senaryoları eklemek kolaydır.

Yeniden kullanımı kolay:

• Tüm testler için yeniden kullanabileceğim sayfa nesne modeline, yardımcı programlara sahibim.

Çoklu tarayıcı testi : Çerçevem, minimum kod değişikliğiyle farklı tarayıcılarda aynı testleri çalıştırabilir. Junit,
TestNG, çoklu tarayıcı testi için kendi yöntemlerine sahiptir. Onlar dışında Selenium Grid kullanıyoruz

Test **türleri** : Çerçevem, uygulamanın kullanıcı arayüzünü, veritabanını ve API'sini test edebilir.

Sayfa 12

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 adlandırma kurallarına çok dikkat ediyoruz. Sınıflar,
yöntemler değişkeni, yaptıkları işe göre adlandırılır ve bir standardı izler

Sayfa nesnesi sınıfı :
ana sayfa, giriş sayfası

değişken :
loginButton, signOutLink

yöntemler :
login (): bu yöntemler yalnızca oturum açmak için kullanılır, diğer işlevler için kullanılmaz.

3. Ürün Yapısı? Web Uygulama Yapısı

- Başlangıç aşaması
  - o **SmartClient** Framework → San Francisco tabanlı bir istemciden bir **JavaScript (Js)** UI Framework'tür. % 100 Microsoft uyumlu
  - o Terminoloji
    - **HTML yapısı** : yalnızca sorumlu verilerdir
    - **CSS sunumu / görünümü** : tasarım için
    - **JavaScript dinamizmi / eylemi** : işlev için
- Arka Uç → **ServerSide Projesi**
  - o Tüm dünyada pek çok **ürünümüz** (makinemiz) var. Özellikle ABD, Avustralya, Singapur, Fransa'da
  - o Her makinede üç sunucu açıyoruz
    - **Uygulama Sunucusu**
      - **IIS Uygulaması** → Kodun -kulaklı görevleri çalıştırdığı mimari. İçin genel URL adresleri veriyoruz her müşteri. Arka uç ve Ön uç projeleri, bu Uygulama sunucularında **IIS Uygulaması** tarafından çalıştırılır .
      - Test çerçevemiz de bu sunucuda barındırılıyor.
      - Geliştirici ekibimiz uygulamaları geliştirmek için **C ++** kullanıyor ve biz testçiler Testing projesi için **JAVA** kullanıyoruz
    - **Rapor Sunucusu**
      - Bu, **Windows SQL Hizmetinin** bir hizmetidir .
      - Arayüzden Raporlar oluşturan ve çağıran Rapor hizmetlerimiz mevcuttur.
    - **Microsoft SQL Server** → SQL tablolarımız bu sunuculardadır. Verileri Microsoft SQL Server'da barındırıyoruz.

4. Mobil Uygulama Yapısı

- **Android Uygulamaları → Uygulamaları** Android Studio üzerinde geliştiriyoruz. Ve Push gibi birçok AWS hizmetini kullanıyoruz Bildirim, Mesajlaşma ve E-posta Gönderme
- **IOS Uygulamaları →** XCode 9 üzerinde geliştiriyoruz
- **FireBase →** Mobil Uygulama kullanımını, hataları ve çökmeleri izlemek için (Crashlytics)
- **Appium →** Mobil uygulamaları test etmek için (yeni başladık)
- **Selendroid → Gözlemliyor** ve kullanmayı düşünüyoruz.

Sayfa 13

BİR DEĞERLENDİRMEDE SORULACAK BÜYÜK SORULAR

Kendi kendinize sormak için bazı sorular sormak önemlidir. Pozisyon hakkında ne bilmek istersiniz? Şirket? The Bölüm? Takım? Öyleyse, bize sorunuz var mı? röportajın bir kısmı geliyor mu? Emin olmak için bu listeyi kullanın tüm üslerinizi kapladı.

1-11 İş Hakkında Sorulacak Sorular

Öncelikle, işin günlük sorumluluklarının tam olarak ne olacağından emin olun - hem şimdi hem de gelecek.

1. Tipik bir gün nasıl görünür?
2. Ele alınması gereken en acil projeler nelerdir?
3. Üzerinde çalışacağım projelerden örnekler gösterebilir misiniz?
4. İdeal bir adayda aradığınız beceriler ve deneyimler nelerdir?
5. Bu pozisyonda gerçekten başarılı olmak için birinin sahip olması gereken özellikler nelerdir?
6. Yeni bir işe almak istediğiniz takımda ne tür beceriler eksik?
7. Bu pozisyondaki birinin karşılaşacağı en büyük zorluklar nelerdir?
8. Çalışma saatleri ve fazla mesai beklentileriniz nelerdir?
9. Bu oluşturulmuş yeni bir rol mü?
10. Bu yeni bir rol mü yoksa işte daha önce birisi var mıydı?
11. Rol daha önce doldurulmuşsa, o kişi neden devam etti?
12. Bu pozisyonun temel sorumluluklarının önümüzdeki altı aydan bir yıla kadar değişmesini bekliyor musunuz?

Eğitim ve Mesleki Gelişim Hakkında Sorulacak 13-20 Soru

Her yeni işi sadece bir iş olarak değil, kariyerinizde başarıya giden yolda bir sonraki adım olarak düşünün. Bu pozisyon oraya ulaşmanıza yardımcı oluyor mu?

13. Nasıl eğitim alacağım?
14. Yeni işe aldığınız kişi ne tür bir eğitim alacak?
15. Çalışanlarınız için hangi eğitim programları mevcut?
16. İlerleme veya mesleki gelişim için fırsatlar var mı?
17. Şirket, çalışanlarına herhangi bir mesleki gelişim fırsatı sunuyor mu?
18. Şirketi endüstri konferanslarında temsil edebilir miyim?
19. Bu işi yapan son kişi nereye gidiyor?
20. Bu pozisyondaki başarılı çalışanlar daha önce nereye ilerledi?

Performansınız Hakkında Sorulacak 21-25 Soru

Potansiyel yeni yöneticinizin başarınızı nasıl ölçeceğini anlamak, hem şirketi anlamanın anahtarıdır. öncelikleri ve yönetim tarzları.

21. İşteki ilk 30, 60 ve 90 günde birinin başardığını görmek isteyeceğiniz en önemli şeyler nelerdir?
22. Bu pozisyonun ilk 12 ayda performans beklentileri nelerdir?
23. Çalışanlar performansları hakkında nasıl geri bildirim alıyor?
24. Buradaki gibi performans inceleme süreci nasıldır? Ne sıklıkla resmi olarak incelenirim?
25. Performansım hangi ölçütlere veya hedeflere göre değerlendirilecek?

26-30 Görüşmeci Hakkında Sorulacak Sorular

Görüşmeciye sorular sormak, onlarla bir kişi olarak ilgilendiğinizi gösterir ve bu, geliştirmenin harika bir yoludur. uyum.

26. Şirkette ne kadar süredir çalışıyorsunuz?
27. Buraya geldiğinizden beri rolünüz değişti mi?
28. Bundan önce ne yaptınız?
29. Bu şirkete neden geldiniz?
30. Burada çalışmanın en sevdiğiniz yanı nedir?

Sayfa 14

31-37 Şirket Hakkında Sorulacak Sorular

Nerede çalışabileceğinizi biraz öğrenmeye ne dersiniz? Çünkü iş, günlük yapılacaklar listenizden ibaret değildir.

31. Şirketin kuruluşunu okudum, ama bana daha fazla bilgi verebilir misiniz ...?
32. Bölümünüzde kıyafet yönetmeliği var mı?
33. Önümüzdeki birkaç yıl içinde bu şirketi nerede görüyorsunuz?
34. Şirketin uzaktan çalışmaya ilişkin politikaları nelerdir?
35. Yeni ürünleriniz veya büyüme planlarınız hakkında bana ne söyleyebilirsiniz?
36. Şirketin odaklandığı mevcut hedefler nelerdir ve bu ekip bu hedeflere ulaşmayı desteklemek için nasıl çalışıyor?
37. Şirketin geleceği konusunda sizi en çok heyecanlandıran şey nedir?

Takım Hakkında Sorulacak 38-44 Soru

Her gün birlikte çalıştığınız insanlar iş hayatınızı gerçekten yapabilir veya bozabilir. Ortaya çıkarmak için bazı sorular sorun sizin için doğru ekip olup olmadığı.

38. Bana birlikte çalışacağım takımdan bahseder misiniz?
39. En yakın kiminle çalışacağım?
40. Doğrudan kime rapor edeceğim?
41. Bana doğrudan raporlarımdan bahseder misiniz? Güçlü yönleri ve takımın en büyük zorlukları nelerdir?
42. Önümüzdeki altı ay içinde bu departmanda daha fazla kişiyi işe almayı bekliyor musunuz?
43. Başka hangi departmanlar bununla en yakın şekilde çalışıyor?
44. Bu departmandaki ortak kariyer yolları nelerdir?

Kültür Hakkında Sorulacak 45-54 Soru

Ofis düğmeli muhafazakar mı yoksa pantolonunun yanından uçup giden bir yer mi? İnce olanı öğrenin, ama oh-so-şirket kültürünün önemli yönleri.

45. Şirket ve ekip kültürü nasıldır?
46. Buradaki çalışma ortamını nasıl tanımlarsınız - çalışma tipik olarak işbirlikçi mi yoksa daha bağımsız mı?
47. Bana birlikte yaptığınız son takım etkinliğinden bahseder misiniz?
48. Resmi bir misyon beyanı veya şirket değerleri var mı? (Not: Bunun Google'a uygun olmadığından emin olun!)
49. En sevdiğiniz ofis geleneğiniz nedir?
50. Siz ve ekip öğle yemeğinde genellikle ne yaparsınız?
51. Ekipte ofis dışında takılan var mı?
52. Hiç başka şirket veya departmanlarla ortak etkinlikler yapıyor musunuz?
53. Burada çalışmanın, çalıştığınız herhangi bir yerden farklı olan nedir?
54. Siz katıldığınızdan beri şirket nasıl değişti?

Sonraki Adımlar Hakkında Sorulacak 55-58 Soru

Ayrılmadan önce, görüşmecinin ihtiyaç duyduğu tüm bilgilere sahip olduğundan ve sonraki adımlarda net olduğunuzdan emin olun. bu soruları sormak.

55. Geçmişimin bu role uygun olduğu konusunda sizi endişelendiren herhangi bir şey var mı?
56. Görüşme sürecinde sonraki adımlar nelerdir?
57. Size sağlayabileceğim, yardımcı olabilecek başka bir şey var mı?
58. Sizin için son soruları cevaplayabilir miyim?

Sayfa 15

SDLC ve AGILE

1. Yazılım Testi nedir?

- İşlevsel ve otomasyon araçlarını kullanarak yazılım hatalarını bulma amacıyla bir program veya uygulamayı yürütme süreci
- Bir yazılım programını / uygulamasını doğrulama / doğrulama süreci
- Test uzmanları, geçmek için test değil, yaklaşımı kırmak için test etmelidir.

2. Yazılım Gereksinimleri Belirtimi nedir?

• Yazılım gereksinimleri belirtimi, müşteri ile tedarikçi arasında bir sözleşme görevi gören bir belgedir.
• Bu SRS, son kullanıcının o uygulamayla ilgili tüm gereksinimlerini içerir. SRS iletişim olarak kullanılabilir

müşteri ve tedarikçi arasındaki ortam.
• Geliştirici ve test uzmanı, 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 **oluşturma** aşamalarını tanımlar .
o Proje Planlama
o Gereksinim Toplama (Projeyi planlamak için kullanılan bilgilerin toplanması, Risklerin belirlenmesi)
o Tasarım (Uygulama nasıl inşa edilecek)
o Kodlama (geliştirme) (Gereksinimlere göre geliştiriciler uygulamayı yazacaktır)
o **Test**
o Üretim (dağıtım) (Ürünün piyasaya sürülmesi)
o Bakım (Ürünün stabil olduğundan emin olmak, hatalarla ilgili müşteri raporlarına bakmak ve onu 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 Gereksinim analizi
o Test Planlama
o Test Tasarımı
o Test Ortamı Kurulumu
o Test Yürütme
o Test Raporlama

5. STLC ve SDLC arasındaki fark nedir?

• STLC, SDLC'nin bir parçasıdır. STLC'nin, SDLC setinin bir alt kümesi olduğu söylenebilir.
• STLC, yazılım veya ürün kalitesinin sağladığı test aşamasıyla sınırlıdır. SDLC'nin eksiksiz ve hayati bir rolü vardır.

bir yazılımın veya ürünün geliştirilmesi.
• Bununla birlikte, STLC, SDLC'nin çok önemli bir aşamasıdır ve nihai ürün veya yazılım,

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

yazılıma yeni işlevler eklendi.

6. Gereklilik nedir?

• Gereksinimler, kullanıcıların yazılım veya ürünle ilgili beklentilerini iletir.
• Müşteriden gereksinimleri toplama, analiz etme ve belgeleme süreci, gereksinim mühendisliği olarak bilinir.
• Gereksinim mühendisliğinin amacı, sofistike ve açıklayıcı SRS Sistem Gereksinimlerini geliştirmek ve sürdürmektir.

Şartname 'Dokümanı

Sayfa 16

7. Gereksinim nereden geliyor?

• Müşteriler uygulama için şartlar verir
• Son kullanıcılarla konuşun → bu uygulamayı en çok kullanacak kişiyle
• Ortaklarla konuşun -
• Etki Alanı Uzmanlarıyla Konuşun - bu uygulamayı daha önce benzer şekilde geliştirmiş olan kodlayıcılar ve geliştiriciler veya

bir uzman inşa edilmekte olan ürünün türü
• Sektör Analistleri ve rakipler hakkında bilgiler

8. Test ne zaman başlar?

• Test, gereksinimleri test etmekle başlar (en olası cevap gibi görünen kodlama aşamasından sonra değil.)
• İlk etapta gereksinimin doğru olduğundan emin olmalıyız. Yanlış gereksinimle hata oluşturmak imkansızdır

ücretsiz uygulama.

9. Gereksinimin iyi mi kötü mü olduğu nasıl anlaşılır?

• Gereksinim (SMART) olmalıdır
o **S** pecific → Kullanıcı giriş gerekir. Geçerli kullanıcı adı ve şifreye sahip yetkili kullanıcı giriş yapabilmelidir
o **M** kolay → Kullanıcı çok hızlı giriş yapabilmelidir (giriş düğmesine tıkladıktan 2 saniye sonra).
O **bir** ttainable
o **R** ealistik
o **T** 2 saniyede (çok hızlı makbuz indirmek için estable → Kullanıcı gerektiği mümkün

10. Neden test ediyoruz?

• Hatasız uygulama oluşturmak için.
• Son kullanıcıyı ve müşteriyi memnun etmek için.
• Daha fazla gelir elde etmek için harika ürünler oluşturmak.
• Test etmeyi seviyorum ve test etmek benim tutkum.

11. Test uzmanının ana sorumluluğu nedir?

- Hatayı olabildiğince erken bulmak için. Hatanın çoğunun düzeltildiğinden emin olun.
- Hatasız ve kullanıcı dostu bir uygulama sunarak son kullanıcıyı ve müşteriyi satın almak için.

12. Test görevindeki bir test uzmanının veya Yazılım Geliştirme Mühendisliğinin iş sorumluluğu nedir?

- Test otomasyonunu yazın ve web veya mobil gibi çeşitli platformlar için aynısını kurun.
- Hata raporunu yönetme ve işleme.
- Geliştirici ve müşteri arasında uygun iletişim kanalını sürdürmek.
- Test senaryolarının hazırlanması ve teslim edilmesi.

13.% 100 test mümkün müdür?

- Hayal bile edemeyeceğimiz sınırsız sayıda senaryo olduğu için uygulamayı% 100 test edemiyoruz.
- Yazılım testi, mümkün olduğunca test edebileceğimiz **işlevselliğin önceliğine** dayalı risk temelli bir faaliyettir .
- % 100 test mümkün olmasa da,% 100 müşteri memnuniyetinin kesinlikle mümkün olduğuna inanıyorum.

14. Pozitif test nedir? Mutlu Yol testi?

- Uygulamanın geçerli girdilerle test edilmesi. " **Mutlu Yol** " Testi olarak da adlandırılır .

  Örn. Geçerli bir kullanıcı adı ve parola ile oturum açarsanız, bu olumlu bir testtir.

Sayfa 17

15. Test hiyerarşisi nedir?

- Birim testi → Geliştiriciler, geliştirme sırasında her modülü veya kod bloğunu test eder.
- Bileşen Testi → Bileşen, kendi kendine çalışabilen bağımsız bir işlevdir. Örn. Amazon Alıcı İşlevselliği,

  Satıcı İşlevselliği, Prime Video İşlevselliği.
- Entegrasyon Testi → Tüm İşlevleri birleştirin. Bunları entegre ettiğimde, yine de tüm fonksiyonları kullanabilir miyim? Yapmak

  elbette hepsi hala çalışıyor.
- Sistem Testi → Uçtan Uca test. Baştan sona her şeyi test edin.
- Kabul Testi → Bir UAT (Kullanıcı Kabul Testi) Ekibi veya İş Analisti de Kabul Testi yapabilir.

  Test tamamlandıktan sonra, QA ekiplerini onaylayabilmeleri için başka bir ekibin kabul testi yapması gerekir.

  test başarılı oldu ve ürünü müşteri için hazır hale getirdi.

16. 508 Uygunluk testi nedir?

- *(Mülakatta biri 508 testinin ne olduğunu sorarsa, sadece ne olduğunu söyleyin. 508'de 5-10 yıllık tecrübem olduğunu söylemeyin.*

  Uyum testi.)
- **Devlet web siteleri için bir gerekliliktir.**
- Devlet tarafından ve devlet için kullanılan tüm web siteleri. **Engellilerin kullanabileceğinden emin olmalılar.**
- 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.

17. Risk temelli test nedir?

- Risk Temelli test, bir ürünün işlevlerinin, çıktıların önceliğine göre test edilmesi olarak tanımlanır.
- Risk Temelli test, bir ürünün ticari etkisi ve olasılığı olan önemli özelliklerinin test edilmesini içerir.

  bu özelliklerin başarısızlığı çok yüksektir.
- Bir ürünün tüm işlevlerinin önceliği, iş gereksinimine ve ardından yüksek önceliğe göre belirlenir.

  önce işlevsellikler sonra orta ve sonra düşük öncelikli işlevler test edilecektir.
- Risk Bazları testi, bir ürünün tüm işlevlerini test etmek için yeterli zaman olmadığında yapılacaktır.
- % 100 test yapılamadığından risk analizi yapmalıyız. Analize dayanarak testimize öncelik vermeliyiz

  önce yüksek riskli alanı test edin. Örneğin:

  o En kritik işlevler

  o En sık kullanılan işlevler

  o En karmaşık işlevler vb ...

18. Bu regresyon paketini inşa etmek ne kadar sürdü?

- 3 yıl sürdü; 2 test cihazı 1 manuel test cihazı + 1 otomasyon test cihazı
- koştuğumuzda:

  o yayınlanmadan önce

  o büyük hata düzeltmesinden sonra

  o büyük yeni işlevlerden sonra
- Test senaryolarını nerede tuttuğumuz ve takım olarak tek bir sprintte birden fazla uygulanacak kararı nerede aldığımız

  bazı senaryoları test edersiniz.

19. Regresyon paketini çalıştırırken bize bir zorluk söyler misiniz?

• Arızalar. Çünkü regresyon paketi çok uzun zaman önce geliştirildi ve neyin değiştiğini bilmiyorsunuz. Bir düğmesi değişmiş olabilir.

Sayfa 18

20. Kaç çevreniz var?

• Geliştirme Ortamı
  o Birim testi
  o Test ortamından daha az kararlı
• Test ortamı
  o Manuel test burada gerçekleşir
  o Üretim ortamını aynen kopyalar
  o Değişiklikler aralıklarla dağıtılır
  o Otomatik **duman testleri** burada yapılır
    ▪ Uygulamanın diğer önemli testleri gerçekleştirmek için yeterince kararlı olup olmadığından emin olmak için test ortamına karşı çalışır faaliyetler.
    ▪ Değişiklikler Test ortamına her dağıtıldığında çalıştır
    ▪ Geliştirici ortamında çalıştırılabilir
  o Otomasyon testleri burada yapılır
  o Otomatik Entegrasyon testleri burada çalıştırılır
• Üretim Öncesi Ortam
  o UAT ortamı
  o Demo burada gerçekleşir
  o yük / performans testi burada gerçekleşir
  o Değişiklikler büyük aralıklarla dağıtılır
  o Otomatik ana **regresyon testleri** burada (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

21. Fonksiyonel test nedir?

• Fonksiyonel test ekibi manuel test cihazı olarak da adlandırılabilir, otomasyon ekibi tarafından da yapılabilir (otomasyon fonksiyonel test yapmak). Kara kutu testi veya manuel test cihazlarına benzer. Sadece uygulamanın belirli işlevselliğini test etmek. Örn. Yapabilmek Kullanıcı Girişi? Kullanıcı çıkış yapabilir mi? Uygulama görünümünü ve hissini test etmemek.

22. Fonksiyonel olmayan test nedir?

• Performans testi, Güvenlik testi, Örn. 2000 kullanıcı aynı anda uygulamaya giriş yapabilir mi? Kullanıcı şuraya taşınabilir mi sonraki sayfa 1 saniye içinde mi?

23. Birim testi nedir? Hiç birim testi yaptınız mı?

• Beyaz kutu testinin bir parçasıdır. Geliştirme ortamından kodu dağıtmadan önce geliştiriciler tarafından yapılır QA ortamına.
• Geliştiriciler tarafından yapıldığı için henüz birim testi yapmadım. Ama bence öğrenebilirim ve gerekirse yapabilirim.

24. Bileşen testi nedir?

• Uygulamanın her bir bileşenini ayrı ayrı test etmek. Uygulamada tek bileşenli olabilir. Bir bileşende bağımsız işlevsellik. Örn. amazon.com'da Satıcı işlevi tek bileşen olabilir. Alıcı başka biri olabilir bileşen. Ayrıca Amazon ana videoları başka bir bileşen olabilir.

Sayfa 19

25. Duman Testi →

• **TEST YAPMA sırası:** Kod → Birim Testi → Entegrasyon Testi → Sağlık Testi → ***Duman Testi*** → Fonksiyonel Test
• Projemizde; oturum açma, kullanıcıyı görüntüleme, kullanıcı ayrıntıları sayfası, yeni kullanıcı oluşturma ve görev oluşturma
• Bu beş modülde, geliştirici ilk olarak duman testini gerçekleştirecektir.

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

oluşturuldu, görüntülendi ya da değil vb.

26. Regresyon testinin hangi bölümü otomatikleştirilmelidir?

- Kararlı olan testler
- Sık sık tekrarlanıyor
- Basittir ve test cihazı girişi gerektirmez, otomasyon için iyi adaylardır

27. Regresyon testlerinizin etkili olmasını nasıl sağlıyorsunuz?

- Regresyon testleri kusurların yakalanmasına izin verecek kadar geniş ve detaylı olmalıdır. Yinelenen testi de ortadan kaldırabilirsiniz

durumlarda, test senaryolarını ve otomatik testleri mümkün olduğu kadar birleştirin.

28. Yazılımda bir dizi kritik hata düzeltilmiştir. Tüm hatalar, raporlarla ilgili tek bir modüldedir. Test yöneticisi sadece rapor modülünde regresyon testi yapmaya karar verir.

- Regresyon testi diğer modüllerde de yapılmalıdır çünkü bir modülü sabitlemek diğer modülleri etkileyebilir.

29. Girdi ve çıktı kapsamına ulaşmak için hangi teknik kullanılabilir?

- İnsan girdisine, bir sisteme arabirimler yoluyla girdiye veya entegrasyon testinde arabirim parametrelerine uygulanabilir.

30. Regresyonunuzu nasıl yürütürsünüz? Ne sıklıkla, kaç sanal makine, kaç gün, kaç test?

- Gerileme her sürümden önce planlanmıştır ve yılda 4 kez yayınlıyoruz (2 İlkbahar sürümü ve 2 Sonbahar sürümü).
- Gerileme, büyük bir hata düzeltmesi olduğunda da gerçekleşir.
- 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.
- **Diğer bir cevap ise** ; Bir dizi regresyon testi oluşturdum. Regresyon etiketli özellik dosyalarıdır. Ve bir işim var

regresyon testlerini başlatan jenkins. Testi tetiklemek için maven komutunu kullanır. Maven komutu şunları içerir:
bu etiket adı: mvn test -D cucumber.options = ”- etiketler @Regression”.

- Yürütmenin sonunda, jenkins ayrıntılı test adımları ve ekran görüntüleriyle HTML raporu oluşturur.

31. Kara kutu testi nedir? Farklı kara kutu test teknikleri nelerdir?

- Kara kutu testi, yazılımı iç yapısını bilmeden test etmek için kullanılan yazılım test yöntemidir.

kodun veya programın.

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

o Eşdeğerlik Bölümleme o Sınır değer analizi o Neden efekt grafiği oluşturma

32. Eşdeğerlik bölümleme testi nedir?

- Eşdeğerlik bölümleme testi, uygulama girdi test verilerini her birine bölen bir yazılım test tekniğidir.

Test senaryolarının türetilebileceği eşdeğer verilerin en az bir kez bölümlenmesi. Bu test yöntemi ile zamanı azaltır
yazılım testi için gereklidir.

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

A, ancak 90'ın altındaki puanlar olmayacak.

- Girdi ve çıktı kapsamına ulaşmak için hangi teknik kullanılabilir? İnsan girişine, arayüzler üzerinden girişe uygulanabilir

bir sisteme veya entegrasyon testindeki arayüz parametrelerine.

Sayfa 20

33. Sınır değeri testi nedir?

- Girdi ve çıktı denklik sınıflarının kenarlarında, altında ve üstünde sınır koşullarını test edin.
- Örneğin, maksimum 1000 $ ve minimum 100 $ çekebileceğiniz bir banka başvurusu, yani sınırda

değer testi, ortada vurmak yerine yalnızca kesin sınırları test ediyoruz. Bu, maksimumun üzerinde test ettiğimiz anlamına gelir
limit ve minimum limitin altında.

- Örneğin, kredi kartımın: Etkinleştirilme tarihi alt sınırdır. 10/2019 son kullanma tarihi üst sınırdır. 0 $ daha düşük

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

34. Sınır değer analizi neden iyi test senaryoları sağlar?

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

35. Neden karar tabloları kullanıyoruz?

- Eşdeğerlik bölümleme 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.
eşdeğerlik bölümleme ve sınır değeri analizi, daha çok kullanıcı arayüzüne odaklanma eğilimindedir.

- Diğer iki spesifikasyona dayalı teknik, karar tabloları ve durum geçiş testi daha çok iş üzerine odaklanmıştır

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

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

Bazen karar tablosunun türetilmesine yardımcı olmak için kullanılan **'neden-sonuç grafiği' olarak** adlandırılan diyagram oluşturma tekniği .

36. Beyaz kutu testi nedir ve beyaz kutu testi türlerini listeler?

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

bir bileşenin veya sistemin şube kapsamı, yol kapsamı, koşul kapsamı vb.).
- Kod Tabanlı test veya Yapısal test olarak da bilinir. Farklı beyaz kutu testi türleri

o Bildirim Kapsamı o Karar Kapsamı

37. Beyaz kutu testinde neyi doğrularsınız?
- Koddaki güvenlik açıklarını doğrulayın
- Koddaki eksik veya bozuk yolları doğrulayın
- Belge özelliklerine göre yapının akışını doğrulayın
- Beklenen çıktıları doğrulayın
- Uygulamanın tam işlevselliğini kontrol etmek için koddaki tüm koşullu döngüleri doğrulayın
- Satır kodlamasını doğrulayın ve% 100 testi kapsayın

38. Gri Kutu Testi nedir?
- Gri kutu testi, kara kutu ve beyaz kutu testinin karmasıdır.
- Gri kutu testinde, test mühendisi bileşenin kodlama bölümü bilgisine sahiptir ve test senaryoları veya testi 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 edilen yazılım uygulaması bir kara kutu gibi davranır ve test cihazı uygulamayı test eder.
dışarıda.

39. Statik ve dinamik test arasındaki fark nedir?
- **Statik test** : **Statik test** sırasında kod yürütülmez ve yazılım dokümantasyonu kullanılarak gerçekleştirilir.
- **Dinamik test** : Bu testi gerçekleştirmek için kodun çalıştırılabilir bir biçimde olması gerekir.

40. Bakım testi nedir?
- Mevcut yazılımın değiştirilmesi, taşınması veya kullanımdan kaldırılmasıyla tetiklenir.

Sayfa 21

41. 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 arayüzlerin kullanılması amacıyla geliştirilmiştir.
- Gerçek sonuçlar ve beklenen sonuçlar aynı olduğunda entegrasyon testi tamamlanmış kabul edilir. Entegrasyon testi yapılır

birim testinden sonra. Entegrasyon testi yapmak için başlıca üç yaklaşım vardır:
o **Yukarıdan Aşağıya Yaklaşım** → bileşenleri yukarıdan aşağıya entegre ederek test eder.
o **Aşağıdan yukarıya yaklaşım** → Kontrol akışının altından üst düzey bileşenlere doğru gerçekleşir
o **Büyük patlama yaklaşımı → Bunda** , tam bir sistem oluşturmak için farklı modül bir araya getirilir ve ardından test yapılır.
üzerinde gerçekleştirildi.

42. Ölçeklenebilirlik Testi nedir?
- Ölçeklenebilirlik testi, işlevsellik ve performans yeteneklerini geliştirmek ve iyileştirmek için yapılan testtir.

uygulama. Böylelikle uygulama, son kullanıcıların gereksinimlerini karşılayabilir.
- Ölçeklenebilirlik ölçümleri, uygulama performansının yük ve stres koşullarında değerlendirilmesi yapılarak yapılır.

Şimdi bu değerlendirmeye bağlı olarak, uygulamanın yeteneklerini iyileştiriyor ve geliştiriyoruz.

43. Depolama Testi nedir?
- Depolama Testinde, uygulamanın, verileri DB'ye depolamaktan sorumlu olan işlevlerini test ederiz.
- GUI'de veya ön uçta son kullanıcı tarafından girilen veriler, veritabanında depolanan verilerle aynıdır.
- Depolama testi, uygulamanın ön ucundan alınan verilerin doğru yerde ve

veri tabanında doğru şekilde.

44. Stres Testi nedir?
- Stres testi, yazılımı, üzerindeki baskıyı arttırırsak uygulamanın çökmediğini kontrol etmek amacıyla test eder.

uygulama üzerinde çalışan kullanıcı sayısını artırarak uygulama.
- Uygulama tarafından halledilemeyen birçok işlemi ateşleyen uygulama üzerine de stres uygulayabiliriz.
- Uygulama yeteneklerini değerlendiren uygulama üzerinde stres testi yapıyoruz, uygulama yeteneklerini belirtilen sınırlar dahilinde veya ötesinde değerlendiriyoruz.

belirlemek için gereksinimler.
- Genel olarak bu, çok yüksek düzeyde yük ve stres koşullarında gerçekleştirilen bir tür performans testidir.

45. Test Donanımı nedir?
- Test koşum takımı, uygulamayı farklı testlerde çalıştırarak test etmek için gerekli olan bir yazılım ve test verileri koleksiyonudur.

stres, yük, veri odaklı ve davranışını ve çıktılarını izleme gibi koşullar. Test Demeti iki ana bölümden oluşur:
o Test yürütme motoru
o Test komut dosyası deposunu test edin
- Otomasyon testi, testlerin yürütülmesini kontrol etmek ve gerçek sonuçları beklenen sonuçlarla karşılaştırmak için bir aracın kullanılmasıdır.

Sonuçlar. Aynı zamanda test ön koşullarının oluşturulmasını da içerir.

46. Test kapsamı nedir?

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

47. V-Modeli nedir?

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

Sayfa 22

48. Aşağıdakilerden hangisi, test yakalama ve tekrar oynatma olanakları sağlayan test araçlarının kullanımından en çok yararlanacaktır?

• Gerileme testi

• Entegrasyon testi

• Sistem testi

• Kullanıcı Kabul Testi

49. Kabul testi nedir?

• Kabul testi QA testinden sonra yapılacaktır. Mevcut projemde UAT ekibi tarafından yapılıyor. UAT ekibinden sonra kabul testini gerçekleştirerek kod üretime gidecektir.

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

o QA ortamı (uygulamayı test ettiğimiz yer.)

o UAT ortamı (kod QA ortamı test edildikten sonra UAT ortamına konuşlandırılacaktır. UAT testi ekibi, iş gereksinimlerine uyduğundan emin olmak için testler gerçekleştirecektir. Aynı zamanda evreleme ortamı olarak da adlandırılır.

o Üretim ortamı (son kullanıcının gerçek uygulamayı görebildiği zamandır)

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

• **Sistem Testi** : Sistem testi, sistem bir bütün olarak teste girdiğinde kusurları bulmaktır, aynı zamanda **son** olarak da bilinir. **testi bitirmek için** . Bu tür testlerde uygulama baştan sona kadar devam eder.

• **UAT** : Kullanıcı Kabul Testi (UAT), bir ürünü belirleyen bir dizi özel testten geçirmeyi içerir. ürünün kullanıcılarının ihtiyaçlarını karşılayıp karşılamayacağı.

51. Sürekli entegrasyon nedir?

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

• Bir geliştirici sisteme yeni bir kod eklediğinde, **Sürekli tümleştirme (CI) sunucusu olarak** adlandırılan bir sunucu vardır .

• **CI sunucusu** sürekli olarak yeni kod arıyor. Yeni kod uygulamaya eklendiğinde, CI sunucusu, kodun teslim edildiğini hemen anlayın . *(bununla entegre bir araç var, belki Jenkins veya başka bir şey).*

• Bu araç, uygulamanın temel işlevselliğini kontrol etmek için otomatik duman testini başlatacaktır. Sonra hava diyecek bu kod, uygulamayı olumsuz yönde etkiledi veya etkilemedi.

52. Kod üretim ortamına nasıl dağıtılır?

• Yerelden

o pull ve push kullanarak kodu Git'e kontrol edin (benim şirketimde bu SVN'dir)

o Birim testleri çalıştırın

o Değişiklikleri sunucuya dağıtın

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

▪ Geliştirme ortamına dağıtım değişikliklerini geçtikten sonra

• Geliştirme ortamından

o Jenkins tarafından yapılan Test ortamındaki değişiklikleri dağıtın.

o Planlanabilir veya manuel olarak tetiklenebilir

• Testten → Jenkins tarafından yapılan değişiklikleri dağıtın

• Ön prodüksiyondan

53. Çevik Çerçeve?

• **Rol** : PO, SM, Takım

• **Törenler** : - Baskı Planlama, Günlük Scrum, Sprint İncelemesi, Sprint Retro, Bakım Oturumu

• **Eserleri** : Ürün birikim, - Sprint birikim, -Burnout grafik

Sayfa 23

54. Çevik nedir?

• Çevik, şelale metodolojisine alternatif olan **yinelemeli** ürün **geliştirme** metodolojisidir.
• Scrum: Takım, bir sonraki sprint için yapılacak iş miktarını planlar
• Kanban: Sprint planlaması yok, hikayeler olduğu gibi toplanıyor, ancak diğer her şeye hala sahipsiniz

55. Neden Agile'a ihtiyacımız var? Şelale ve Çevik mi?

• Şelale metodolojilerinin aşağıdaki dezavantajları olduğundan;
  o Gereksinim, belge imzalandıktan sonra değiştirilemez veya değiştirilmesi zor olamaz.
  o Şelalede bir aşamayı tamamlamadan bir sonraki aşamaya geçemezsiniz. Örneğin, kodlama aşamasından önce tamamlanan test başlatılamaz.
  o Müşteri, geliştirme yaşam döngüsünün çok ileri aşamalarına kadar ne elde edeceğini göremez.
  o Üretime gitmek daha uzun sürüyor. Ürün pazara girdiğinde çoktan modası geçmiş olabilir.
• Agile'ın aşağıdaki avantajları vardır:
  o Değişiklik memnuniyetle karşılanmaktadır. Örneğin, sprint demosundan sonra müşteri bir şeyi beğenmezse, geri bildirimde bulunun ve ürünü geliştirin. Gereksinim değişikliği tamam.
  o Yinelemeli geliştirme süreci olduğundan, geliştirme ekibi işlevsellik geliştirebilir, geri bildirim alabilir ve bir sonraki yinelemeyi iyileştirin. Böylece ürün sürekli olarak gelişecektir.
  o Scrum master yardımıyla atıklar çevik olarak ortadan kaldırılır. Örneğin, bloke olursam, beklemek ve zamanımı boşa harcamak zorunda kalmam. Ekip üyeleri birbirleriyle verimli bir şekilde iletişim kurduğundan, yinelenen çabaları önleyerek daha üretken olabiliriz.
  o Waterfall, C # .NET gibi araçları ve platformu vurgular, ancak Agile insanları vurgular. En iyi araca sahip olabilirsiniz ama son insanlar bu araçları kullanıyor. İlham alan insanların daha az ürüne sahip olsalar bile harika ürünler yapabileceklerine inandım. para veya daha az kaynak.

56. Önceki projelerinizde ne tür Agile metodolojisi kullandınız?

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

57. Scrum, Çevik bir çerçevedir, değil mi? Diğer birkaç Agile çerçeveyi adlandırın.

• Evet, Scrum bir Çevik çerçevedir. Çok az Çevik çerçeve –Özellik Odaklı Geliştirme Test Güdümlüdür Geliştirme, Kanban

58. Scrum'daki farklı roller nelerdir? Scrum rolleri?

• Ürün sahibi , aslında projenin paydaşıdır.
  o Ekip önünde proje gereksinimlerini temsil eder.
  o Neyin inşa edileceğine dair bir vizyona sahip olmaktan ve detaylı vizyonunu takıma aktarmaktan sorumludur.
  o Çevik bir scrum yazılım geliştirme projesinin başlangıç noktasıdır.
• Scrum takımı , belirli bir takımın başarılması için performans gösteren bireylerin kolektif katkısıyla oluşur. proje.
  o Ekip, talep edilen ürünün zamanında teslimi için çalışmak zorundadır.
• Scrum ustası - Scrum ustası, scrum takımının olup olmadığını kontrol eden scrum takımının lideri ve koçudur. taahhüt edilen görevleri düzgün bir şekilde yürütmek.
  o Ayrıca sprint hedefine ulaşabilmeleri için takımın verimliliğini ve üretkenliğini artırmaktan sorumludur. etkili bir şekilde.

59. Bir scrum takımını nasıl tanımlarsınız?

• 5 rock yıldızını bir araya getirirseniz, bu onların bir ekip oldukları anlamına gelmez veya harika ürünler geliştirebilecekleri anlamına gelmez.
• Benim için ekip, aynı hedefi paylaşan, aynı yöne hareket eden, 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.

Sayfa 24

60. Scrum Master'ın sorumlulukları nelerdir?

• İzleme ve izleme
• Gereksinimleri doğru anlamak
• Proje hedefine ulaşmak için çalışın
• Süreç kontrol ustası ve kaliteli usta
• Ekibi müfrezelerden koruyun
• Takımın performansını iyileştirmek
• Toplantıları yönetin ve sorunları çözün
• Çatışmaların ve engellerin çözümü
• İletişim ve raporlama

61. Negatif test senaryosu nedir?

• Negatif test senaryoları, yıkıcı bir şekilde test etme fikrine dayanılarak oluşturulur. Örneğin, ne olacağını test etmek Uygulamaya uygun olmayan girişler girilirse. Yanlış giriş bilgisi

62. "Scrum of Scrums" teriminden ne anlıyorsunuz?

• Halihazırda yedi ekibin üzerinde çalıştığı aktif bir proje varsayalım. Her takım kendi başına liderlik etmekten sorumludur. saldırı toplantısı. Ancak, farklı ekiplerle koordinasyon sağlamak ve iletişim kurmak için ayrı bir

saldırı toplantısı. Büyükelçi olarak bilinen her ekipten kendisini temsil etmekten sorumlu bir ekip lideri vardır.

Scrums ekibi.

• **Scrum takımları arasında koordinasyonu sağlamak için düzenlenen scrum toplantısı, scrums olarak bilinir.**

63. Sevk edilebilir ürün / artım?

• Ürünün parçası yapıldı ve her sprintten ek işlevler almaya devam ediyor

• Artış, geliştirme ekibinin *Bitti Tanımına uygun olmalıdır*

o Ürün artımı teslim edildiğinde, "Bitti Tanımı" nı karşılaması gerekir

o Kabul kriterleri yerine getirildi

o Ürün sahibi kullanıcı hikayelerini kabul eder

• Artış PO tarafından kabul edilebilir olmalıdır

64. BurnDown Grafiği nedir?

• İşin tamamlanma oranının ve yapılması gereken işin ne kadar kaldığının grafik temsili

65. Doğrulama ve Doğrulama nedir?

• Doğrulama, testçiler ve geliştiriciler tarafından geliştirme sırasında gerçekleşir; yazılımı geliştirme aşamasında değerlendirme sürecidir

aşama 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.

66. Hazırın Tanımı Nedir?

• Kabul Kriterleri temizlendi / gözden geçirildi ve Puan / saat verildi

67. Otopark nedir?

• Çevik'te bu şu anlama gelir: Diğer insanlarla gerçekten alakalı olmayan bir sorununuz olduğunda karşılaştığınızda, devam etmemeliyiz.

diğer insanların zamanını boşa harcadığımız için bu konuyu toplantıda tartışmak. <Hadi **park yeri** öğesi yapalım > anlamına gelir

bu konuyla ilgilenen herkes görüşmeden sonra konuşabilir.

Sayfa 25

68. Sprint iş akışı nedir?

• Bir hikaye nasıl yapılır ve yaşam döngüleri nasıl ilerler - bir şey engellendiğinde ne olur, vb.

69. Kullanıcı Hikayesi nedir?

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

• Normalde şu şekilde görünür: **<son kullanıcı> olarak <eylem>** yapmak istiyorum, böylece **<fayda>** yapabilirim .

o Amazon kullanıcısı olarak giriş yapabilmeliyim, böylece çevrimiçi malzeme satın alabilirim

70. "Sevk edilebilir" dediniz, bununla ne demek istiyorsunuz?

• Bir kullanıcı olarak kullanıcı adımı kullanıcı adı alanına koymak istediğimi gerçekten söyleyemezsin.

• Böylece kullanıcı adımı oraya yazabilirim. Tam işlevsellik olmalı. Kullanıcı adı girmek, gönderilebilir bir işlev değildir.

Ancak giriş yapabilmek tam bir işlevselliktir. Sevk edilebilir derken bunu kastediyorum.

71. 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 Müşteri olarak hesabımı görebilmek için giriş yapabilmek istiyorum.

o Müşteri olarak satın alabilmek için bir ürünü arayabilmek istiyorum.

o Müşteri olarak satın alacağım ürünün parasını ödeyebilmek için ödeme aşamasına geçmek istiyorum.

o Müşteri olarak hesabımı koruyabilmek için çıkış yapabilmek istiyorum.

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

her sprintte bir veya daha fazla kullanıcı hikayesi.

72. En son projenizde çevik deneyim?

• Sprintimiz 4 haftadır ve her 3 sprint'i yayın döngüsü olarak yayınlıyoruz

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

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

• Sprint Planlama Toplantısı ile bir sprint başlatıyoruz ve

o ekibin öncelikli özellikleri ve ürün iş yığını öğeleri hakkında tartışırız ve

o uygulamanın geliştireceğimiz kısmını öğreniyoruz.

o dayalı hikaye seçimi *hız* ve *kapasite*

▪ **Hız** : Bir sprintte iletilen hikaye puanı / demo sayısı. Örneğin: ekip 30 hikaye puanı planladıysa

(İş değeri); bir sprintte değerinde kullanıcı hikayesi var ve planlandığı gibi sunabiliyorsa, takımın hızı 30'dur

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

o Bu toplantı her hafta yapılır ve yaklaşık 1 saat sürer. Sprint Bakım toplantısından daha genel bir fikir ediniriz görevler için bazı tahminler vermek için.

- ▪ Ekip, SM ve PO, çalışma öğelerinin ilgili ve yararlı olmasını sağlamak için bir araya gelir
- ▪ Kullanıcı öykülerinin PO'suna sorular sorun
- ▪ Kabul kriterlerini yeniden tanımlayın
- ▪ Yeni hikayeler yazmak
- ▪ Destanları kullanıcı hikayelerine bölme
- ▪ Doğru tahminde bulunmak / gereğinden az / fazla tahminde bulunmak için hikayeyi anlayın

Nasıl tahmin edersiniz?

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

- Sprint başladıktan sonra Günlük Standup Toplantısı yaparız
  - o her gün sabah ve dün ne yaptık, bugün ne yapacağız ve herhangi bir engelleyici var mı tartışıyoruz.
  - o Sadece sprint hakkındaki bilgileri senkronize ediyoruz.

Sayfa 26

- Sprint sonu, genellikle Sprint Demo / Gözden Geçirme Toplantısı yaparız.
  - o Sadece müşteriye sprint oluşturduğumuzu göstermek içindir (PO geri bildirimde bulunabilir)
  - o Ekibimde bir SDET olarak, bazen sunum yaptım ve konferans odası.
  - o Müşteri, paydaşlar veya iş adamları bilmedikleri sorular sorarlar.
- Sprint Demosundan sonra Sprint Retrospektif Toplantısı yapıyoruz.
  - o Sprint Retro'da, son sprintte neyin iyi olduğunu, ne tür hatalar yaptığımızı konuşuyoruz.
  - o Bunların üzerinden geçer ve aynı hataları bir daha yapmayacağımızdan emin oluruz.
  - 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.

73. Kabul kriterleri nedir?

- Kabul kriterleri, kullanıcı hikayesinin başarılı bir şekilde geliştirilip geliştirilmediğini bilme şeklimizdir.
- Bir hikayenin ne zaman "tamamlandığını" belirlemek için kullanıcının bakış açısından açıklanan gereksinim beyanları ve beklendiği gibi çalışmak
- 3 bölümden örnekler
  - o Giriş → geçerli e-posta adresi
  - o İşlem → mesajlaşmayı işaretleme
  - o Sonuç → pazarlama mesajı tasarımı, pazarlama tarafından sağlanan özelliklerle eşleşiyor

74. Fare deliği nedir?

- Agile takımında çok fazla iletişim olduğu için, takımın birçok konuyu tartışması gerekir. Ama bazen tartışma bir konu için çok uzun sürecek ve gerçekten verimli değil. Bunun **<fare deliği>** olduğunu söyleyeceğiz, bu, yapmamamız gerektiği anlamına gelir bu konuyu çok uzun süre ele alın ve ilerleyin.

75. Ne tür Test senaryoları?

- Farklı senaryoları ele alıyorum
  - o Olumlu
  - o Negatif
  - o Sınır Değer Analizi

76. Test Vakası?

- Test durumu, Test Edilmekte Olan Uygulama ile karşılaştırılacak özel bir durumdur. Test adımları, ön koşullar hakkında bilgi içerir, test ortamı ve çıktılar.
- Test senaryosu, işlevselliği ve test adımlarını açıklar.
  - o Test Vakası Kimliği
  - o Adım numarası
  - o İşlevselliğin açıklaması
  - o Beklenen sonuç
  - o Gerçek Sonuç

77. Genellikle bir haftada kaç Test vakası (regresyon süitinizde) tamamlıyorsunuz?

- 10 küçük test durumu, 7-8 orta, 2-3 büyük
- VEYA Projeye bağlıdır. COOLSIS'te 2000 test vakamız var. 4Stay'de yaklaşık 700 test vakamız var.

78. Regresyon süitinizi çalıştırmak 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 testi yürütmek 2 ila 3 gün sürer durumlarda. Ayrıca, manuel test ediciler bazı manuel test durumlarını yürütecekler, ancak ne kadarını yürüttüklerinden emin değilim. inanıyorum yalnızca bazı önemli test durumlarını yalnızca önceliklendirmeden sonra yürütürler.

Sayfa 27

79. Otomatikleştirilmiş komut dosyanızı çalıştırdığınızda ne yaparsınız ya da regresyon çalıştırdığınızda ne yaparsınız?

- İlk olarak, betiğimi çalıştırmalıyım. Komut dosyası yürütme tamamlandığında, olup olmadığını görmek için çalıştırma sonucunu analiz etmeliyim.

başarısız test durumları. Başarısız test senaryoları varsa, meşru uygulama sorunu nedeniyle başarısız olup olmadığını belirlemem gerekir veya bazı komut dosyası sorunlarından kaynaklanıyor. (komut dosyası da otomasyon kodu sorunu nedeniyle başarısız olabilir) uygulama sorunundan kaynaklanıyorsa, El ile yeniden üretmeye ve yeniden üretebilirsem bir kusur kaydetmeye çalışacağım. Senaryomdan kaynaklanıyorsa, düzeltmem gerekiyor. Ama bu değil çoğu zaman durum.

80. Otomatikleştirmek için attığınız adımlar nelerdir?

- İşlevselliği öğrenin
  - o Okuma gereksinimleri
  - o BA ile bilgi aktarımı oturumu
  - o Takım arkadaşlarına sor
- Manuel olarak 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 Assertions'ı kullanarak testleri doğrulayın

81. Otomasyon ile manuel arasında konumun yüzde kaçı var?

- •% 80-85 otomasyon% 15-20 manuel

82. Manuel test yerine otomatik testi ne zaman seçersiniz?

- Test senaryoları yüksek öncelikli test durumlarıysa.
- İşlevsellik kritik işlev ise.
- Sallama veya duman testi test durumları.
- Test senaryoları çok uzun ve yürütmek çok zorsa. Önceliğe dayalı regresyon testi durumları.
- Mümkün olduğunca otomatikleştirmeliyiz.

83. Sprintinizde otomasyonu ne zaman yaparsınız?

- Geliştiriciler kendi paylarına düşeni yaptıklarında
- Kod, KG / test ortamına dağıtıldığında
- Test çerçevesi kurulduğunda
- Tüm manuel testler yapıldığında
- Duman testleri geçiyor

84. Test Planı nedir?

- Test planı, test kapsamını açıklayan bir kelime belgesidir
  - o Yüksek seviye test döngüsü
  - o Kusur yaşam döngüsü
  - o Giriş Kriterleri (teste başlamak için gerekenleri tanımlar)
  - o Çıkış Kriterleri (testin ne bittiğini tanımlar)

Sayfa 28

85. Test planlarındaki tablolar nelerdir?

- Test tasarımı, kapsam, test stratejileri, yaklaşım Test planı belgesinin içerdiği çeşitli detaylardır.
  - o Test senaryosu tanımlayıcısı
  - o Kapsam
  - o Test edilecek özellikler
  - o Test edilmeyen özellikler
  - o Test stratejisi ve Test yaklaşımı
  - o Teslimatların test edilmesi
  - o Sorumluluklar
  - o Personel ve eğitim
  - o Risk ve Beklenmedik Durumlar

86. Test planı ile QA planı arasındaki fark nedir?

- Bir test planı, ürünü test etmek için ne yapılması gerektiğini ortaya koyar ve kalite kontrolün hataları belirlemek için nasıl çalışacağını ve kusurlar.

• Öte yandan, bir kalite güvence planı, hataları ve kusurları test etmek ve düzeltmek yerine önlenmesiyle ilgilidir.

87. Akran değerlendirmesi nedir?

• Akran incelemeleri, aynı ekipte çalışan kişiler arasında yapılan incelemelerdir. Örneğin, yazılmış bir test senaryosu

bir QA mühendisi tarafından bir geliştirici ve / veya başka bir QA mühendisi tarafından incelenebilir.

88. 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 tüm tasarlanmış özelliklerinin test edilmesini sağlar.

• A2-İhtiyaç izlenebilirlik matrisine sahip olmamızın nedeni budur. Kaç gereksinimin olduğunu söyleyebiliriz

test senaryoları kapsamına girer ve kaçının hala RTM'den kaldığı.

89. Test durumlarını kim onaylar?

• Test senaryolarının onaylayıcısı bir kuruluştan diğerine değişir. Bazı kuruluşlarda, QA sorumlusu,

vakaları test ederken bir başkası onları akran incelemelerinin bir parçası olarak onaylar.

90. Test planlarını ve test senaryolarını kim yazıyor?

• Test planları tipik olarak kalite güvence sorumlusu tarafından yazılırken, test uzmanları genellikle test senaryoları yazarlar.

91. Test tasarım tekniğinin amacı nedir?

• Test koşullarının belirlenmesi ve test senaryolarının belirlenmesi.

92. Test senaryosu ile Test komut dosyası arasındaki fark nedir?

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

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

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

• Yazılım testinde bir test komut dosyası, test edilen sistemde test edilen

sistem beklendiği gibi çalışır.

93. Bir test stratejisine neler dahil edilmelidir?

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

süreçler, vb.). Test için sınırlar belirler ve manuel veya otomatik testin kullanılıp kullanılmayacağını belirtir.

Sayfa 29

94. Komut dosyası başarısız olduğunda ne yapacaksınız?

• Deneyimlerime göre, başarısızlığı tespit edeceğim,

o uygulama hatası, senkronizasyon hatası, komut dosyası sorunu veya ortamın çalışmamasından kaynaklanıyorsa, öncelikle sonucu şu şekilde analiz ederim:

Jenkins aracılığıyla yeniden üretin, yalnızca başarısız olanı çalıştırın,

o senkronizasyon sorununun kaynaklanıyorsa, örtük, açık ve bazı özel beklenenleri kullanarak ekstra süre ekleyeceğim

koşullar,

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

o eğer gerçek bir kusursa, o zaman kusuru kaydederim.

95. Test Senaryosu?

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

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

test edilen uygulama

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

• Test Senaryosu: Giriş sayfasını doğrulayın

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

o Test Senaryosu 2: Şifrenizi sıfırlayın

o Test Senaryosu 3: Geçersiz kimlik bilgilerini girin

• Her test senaryosunda, yürütme için ayrıntılı adımlar ve koşullar bulunur

96. Gereksinim İzlenebilirlik Matrisi (RTM)

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

97. İşlevsel bir özellik yoksa veya herhangi bir sistem ve geliştirme yoksa bir sistem için bir test geliştirmek için neler yapılabilir?

belgeler?

• İşlevsel özellikler veya sistem geliştirme belgeleri olmadığında, test uzmanı kendini alıştırmalıdır

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

98. Fonksiyonel test türleri nelerdir?

• Birim Testi

• Duman testi

• Sağlık testi
• Entegrasyon Testi
• Sistem Testi
• Gerileme testi
• UAT (kullanıcı kabul testi)

99. Akıl sağlığı testi ile duman testi arasındaki fark nedir?

• Akıl sağlığı testi yapıldığında, ürün, sırayla test grubu ile bir ön test turundan geçirilir.
düğme işlevi gibi temel işlevleri kontrol etmek için. Duman testi ise geliştiriciler tarafından yapılmaktadır.
müşterinin gereksinimlerine göre.

100. Akıl sağlığı testinde hangi adımlar yer alır?

• Sağlık testi, duman testine çok benzer. Yapmak için yapılan bir bileşenin veya uygulamanın ilk testidir.
en temel düzeyde çalıştığından ve daha ayrıntılı testlere devam etmek için yeterince kararlı olduğundan emin olun.

101. WinRunner ile Rational Robot arasındaki fark nedir?

• WinRunner işlevsel bir test aracıdır, ancak Rational Robot hem işlevsel hem de performans testi yapabilir. Ayrıca,
WinRunner 4 doğrulama noktasına ve Rational Robot 13 doğrulama noktasına sahiptir.

Sayfa 30

102. Kalite Güvencesi ile test arasındaki fark nedir?

• Kalite Güvencesi'nin hedefleri, test etme hedeflerinden çok farklıdır.
• QA'nın amacı hataları önlemektir, testin amacı ise hataları bulmaktır.

103. Rastgele testi açıklayın.

• Rastgele test, uygulamanın rastgele oluşturulan girdi verilerini nasıl işlediğini kontrol etmeyi içerir. Veri türleri
tipik olarak yok sayılır ve rasgele bir harf, sayı ve diğer karakterler dizisi veri alanına girilir.

104. Kalite Kontrol ve Kalite Güvencesi arasındaki fark nedir?

• Kalite kontrolü (QC) ve kalite güvencesi (QA) birbiriyle yakından bağlantılıdır, ancak çok farklı kavramlardır. QC, bir
Geliştirilen ürün, kalite güvencesinin amacı, geliştirme sürecinin,
sistem veya uygulama gereksinimleri karşılayacaktır.

105. Proje geliştirmede QA'nın rolü nedir?

• QA ekibi, geliştirme için yürütülecek sürecin izlenmesinden sorumludur.
• QA ekibinin sorumlulukları, test yürütme sürecini planlamaktır.
• QA Lideri, zaman tablolarını oluşturur ve ürün için bir Kalite Güvence planı üzerinde anlaşır.
• QA ekibi, QA sürecini ekip üyelerine iletti. QA ekibi, test senaryolarının gereksinimlere göre izlenebilirliğini sağlar.

106. İyi bir QA veya Test yöneticisi yapan nedir?

• Yazılım geliştirme süreci hakkında bilgi
• Üretkenliği artırmak için ekip çalışmasını iyileştirin
• Yazılım, test ve kalite kontrol mühendisleri arasındaki işbirliğini geliştirin
• QA süreçlerini iyileştirmek için.
• İletişim yetenekleri.
• Toplantılar düzenleyebilir ve odaklanmış halde tutabilir

107. Regresyon testi ile yeniden test arasındaki fark nedir?

• Regresyon testi, bir modül veya sistemde yapılan değişikliklerin olumsuz bir etkiye sahip olmadığından emin olmak için testler gerçekleştiriyor
önceki sürümlerde. Yeniden test sadece aynı testi tekrar çalıştırmaktır. Regresyon testi yaygın olarak sorulan kılavuzdur
mülakat sorularını test etmek ve dolayısıyla bu konuyu anlamak için daha fazla araştırmaya ihtiyaç vardır.

108. Hata önem derecesi ve hata önceliği arasındaki farkı açıklayın.

• Hata şiddeti, hatanın uygulama veya sistem üzerindeki etki düzeyini ifade ederken, hata önceliği seviyeyi ifade eder
bir düzeltmeye ihtiyaç duyan aciliyet.
• Genellikle ciddiyet, maddi kayıp, çevreye verilen zarar, şirketin itibarı ve can kaybı olarak tanımlanır.
Bir kusurun önceliği, bir hatanın ne kadar hızlı düzeltilmesi ve canlı sunuculara dağıtılması gerektiğiyle ilgilidir.

109. Sistem testi ile entegrasyon testi arasındaki fark nedir?

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

110. Fonksiyonel ve yapısal testler arasındaki farkı açıklayın.

• İşlevsel test, test edenin sistemin veya
uygulama spesifikasyona göre çalışır. Yapısal testler ise koda veya algoritmalara dayanmaktadır.
ve beyaz kutu testi olarak kabul edilir.

Sayfa 31

111. 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 test yapılır.
    geliştirme döngüsünün.
  - o Beta testinde uygulama, uygulamanın kullanıcı gereksinimlerini karşıladığından emin olmak için birkaç kullanıcıya verilir ve
    herhangi bir gösterici içermez, oysa pilot test ekibi üyesi olması durumunda kaliteyi iyileştirmek için geri bildirimde bulunur.
    uygulamanın.

112. Alfa testi nedir?

- Geliştiricinin sitesinde son kullanıcı temsilcileri tarafından ön sürüm testi.

113. Başarısızlık nedir?

- Başarısızlık, belirtilen davranıştan uzaklaşmadır.

114. Test karşılaştırıcıları nelerdir?

- Bazı yazılımlara bazı girdiler koyarsanız bu gerçekten bir test midir, ancak yazılımın bunları üretip üretmediğine asla bakmazsanız
  doğru sonuç?
- Testin özü, yazılımın doğru sonucu verip vermediğini kontrol etmektir ve bunu yapmak için karşılaştırmalıyız.
  Yazılımın üretmesi gereken şey için ne ürettiği.
- Bir test karşılaştırıcısı, bu karşılaştırmanın yönlerini otomatikleştirmeye yardımcı olur.

115. Yazılım testi sırasında Risk analizinin nasıl gerçekleştirileceğini açıklayın?

- Risk analizi, uygulamada riski belirleme ve test etmek için önceliklendirme sürecidir. Aşağıdakilerden bazıları
  riskler:
  1. Yeni Donanım.
  2. Yeni Teknoloji.
  3. Yeni Otomasyon Aracı.
  4. Kod tesliminin sırası.
  5. Uygulamanın kullanılabilirliği kaynakları test edin.
- Bunları üç kategoriye ayırıyoruz:
  - o Yüksek büyüklük: Hatanın uygulamanın diğer işlevselliği üzerindeki etkisi.
  - o Orta: Uygulamada tolere edilebilir ancak istenmez.
  - o Düşük: tolere edilebilir. Bu tür bir riskin şirket işi üzerinde hiçbir etkisi yoktur.

116. İpek Testi nedir?

- İpek Testi, uygulamanın regresyon ve işlevsellik testini gerçekleştirmek için geliştirilmiş bir araçtır. İpek Testi bir araç
  Windows, Java, web veya geleneksel istemci / sunucu tabanlı uygulamaları test ederken kullanılır.
- Silk Test, test planının hazırlanmasında ve bu test planlarının yönetiminde, testin doğrudan erişimini sağlamak için yardımcı olur.
  veritabanı ve alanın doğrulanması.

117. Ana Test Planı ile Test Planı arasındaki fark nedir?

- Ana Test Planı, uygulamanın tüm test ve riskle ilgili alanlarını içerirken, Test senaryosu dokümanı şunları içerir:
  test durumları.
- Ana Test planı, genel geliştirme sırasında çalıştırılacak her bir testin tüm ayrıntılarını içerir.
  uygulama, test planı ise test gerçekleştirmenin kapsamını, yaklaşımını, kaynaklarını ve programını açıklar.
- Ana Test planı, uygulama üzerinde gerçekleştirilecek 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 bunu test planı olarak adlandırdık.

Sayfa 32

118. Bir test ne zaman başarılı kabul edilir?

- Testin amacı, uygulamanın gereksinimlere göre çalışmasını sağlamak ve olabildiğince çok
  mümkün olduğunca hatalar ve hatalar. Bu, daha fazla işlevselliği kapsayan ve daha fazla hata ortaya çıkaran testlerin,
  en başarılı ol.

119. Kusur nedir?

- Beklenen sonuç gerçek sonuçla eşleşmediğinde, bu kusurdur.

120. Kusur yoğunluğu tanımlansın mı?

- Hata yoğunluğu, kod satırı başına toplam kusur sayısıdır.

121. Kusur Yaşam Döngüsü (DLC) nedir?
- Yeni → Atanmış → Açık → Sabit → Yeniden Test Edildi → Kapat

122. Kusur kategorileri nelerdir?
- **Yanlış** : Gereksinimler uygulamada yanlış uygulanmıştır.
- **Eksik** : Müşteri tarafından verilen gereksinim ve uygulama bu uygulamayı karşılamadığında.
- **Ekstra** : Son müşteri tarafından verilmeyen ürüne dahil edilen bir gereksinim. Bu her zaman bir farktır

  şartname ancak ürünün kullanıcısı tarafından istenen bir özellik olabilir.

123. Bir kusur bulduğunuzda ne yapmalısınız?
- Bir kusur bulursam, bunu bildirmeden önce, geçerli bir kusur olduğundan emin olmam gereken hatayı yeniden oluşturuyorum.
- Küçük bir sorunsa, geliştirici masasına gideceğim ve hemen düzeltebilir.
- Büyük bir sorunsa, JIRA'mı açar ve kusuru günlüğe kaydederim.
- Hata olup olmadığından emin değilsem KOBİ ile konuşacağım (konu uzmanı, uygulamayı bilen kişi demektir

  hiç yoktan iyidir).

124. Test nasıl yapılmalıdır?
- Test, uygulamanın teknik gereksinimlerine göre yapılmalıdır.

125. Geliştirici kusur olmadığını söylüyorsa 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 atıyorum.
- Aslında mevcut projemde karşılaştığım en büyük zorluklardan biri de bu.

126. Bir programı test edip hataların% 100'ünü bulabilir misiniz?
- Bir uygulamadaki tüm hataları bulmak imkansızdır çünkü çoğunlukla kaç tane hata olduğunu hesaplamanın bir yolu yoktur. Orada

  Programın karmaşıklığı, programcının deneyimi gibi böyle bir hesaplamayla ilgili birçok faktör var.
  ve benzeri. Bu Manuel test mülakat soruları, test uzmanları tarafından dikkate alınan en zor sorulardır.

127. Hata ayıklama ve test etme arasındaki fark nedir?
- Hata ayıklama ile test etme arasındaki temel fark, hata ayıklamanın genellikle aynı zamanda sorunu çözen bir geliştirici tarafından yürütülmesidir.

  hata ayıklama aşamasında hatalar. Öte yandan test, hataları düzeltmek yerine bulur. Bir testçi bulduğunda
  hata, genellikle bir geliştiricinin düzeltebilmesi için rapor ederler.

128. İyi bir test olarak kabul edilen nedir?
- Bir nesnenin veya sistemin işlevselliğinin çoğunu kapsayan test, iyi bir test olarak kabul edilir.

Sayfa 33

129. Test ne zaman durdurulmalıdır?
- Test edilen sistemin risklerine bağlıdır. Testi durdurabileceğiniz bazı kriterler vardır.
  - o Son Tarihler (Test, Sürüm)
  - o Test bütçesi tükendi
  - o Hata oranı belirli bir düzeyin altına düşüyor
  - o Belirli bir yüzde ile tamamlanan test senaryoları
  - o Test bitimleri için alfa veya beta dönemleri
  - o Kod, işlevsellik veya gereksinimlerin kapsamı belirli bir noktaya kadar karşılanır

130. Yukarıdan aşağıya ve aşağıdan yukarıya test arasındaki fark nedir?
- Yukarıdan **Aşağıya** test, sistemle başlar ve ünite seviyesine kadar ilerler.
- **Aşağıdan yukarıya** testi, ters yönde, birim seviyesinden genel sisteme arayüzle kontrol eder. İkisinin de değeri var ama

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

131. Oluşturduğunuz çalıştırılabilir dosyaların ortalama boyutu nedir?
- Bu, yürütülebilir dosyalar ile deneyimimiz hakkında basit bir röportaj sorusudur. Oluşturduğunuz herhangi birinin boyutunu biliyorsanız,

  sadece bu bilgiyi sağlayın.

132. Ön uç ve arka uçta testler yaptınız mı?
- Ön Ucu test ettiğimde, uygulamayı açarak ve kullanıcı arayüzünde test gerçekleştirerek aslında kullanıcı arayüzünü test ediyorum. Eğer yaptıysam

  Kullanıcı arayüzündeki herhangi bir şey varsa, değişikliğin veritabanında da yapılıp yapılmadığını görmek için arka uç testi yapmam gerekiyor. İçin
  Örneğin, bir ebeveyn iletişim bilgilerini güncellediğimde veya yeni bir uygulama oluşturduğumda, veritabanına bağlanıyorum ve
  verilere değişiklikler uygulanır veya yeni uygulama oluşturulur veya oluşturulmaz.

133. Ön Uç Testi ile Arka Uç testi arasındaki fark nedir?
- Ön Uç Testi, Grafik Kullanıcı Arayüzünde gerçekleştirilirken Arka Uç Testi, veri tabanlarının testini içerir.

• Ön uç, kullanıcının etkileşime girebileceği web sitesi görünümünden oluşurken, arka uç durumunda gerekli olan veritabanıdır.
verileri saklamak için.
• Son kullanıcı, ön uç uygulamanın GUI'sine veri girdiğinde, bu girilen veriler veritabanında saklanır. Kaydetmek

Bu verileri veritabanına SQL sorguları yazıyoruz.

134. Test sırasında bulduğunuz en zor sorun nedir?

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

Mülakat soruları, cevabınız işinizi belirleyecektir. Problem çözme becerilerinizin
ve işin. Öyle bir cevap vermelisiniz ki problem çözme becerileriniz ve yeni şeyler öğrenme hevesiniz,
ve işe olan bağlılığınız cevaplarınızla belirtilecektir.

135. Scrumda karşılaştığınız zorluk nedir?

• Scrum, çapraz fonksiyonlu takımı vurguladığından (bu, geliştiricinin test edebilmesi ve test uzmanlarının geliştirebilmesi gerektiği anlamına gelir)

geleneksel bir QA test cihazı olarak geliştirme ekibinin bir parçası olmak zordur. Çünkü genellikle KG'ler nasıl kod yazılacağını bilmiyor.
Bu yüzden kendimi çok rekabetçi tutmalıyım. Ne zaman vaktim olursa, Java gibi daha çok kodlama öğreniyorum.
• Zaman değişikliği sorunu → Veritabanına girilen tarihi bir tarih kaydettiğimde, bunlar daha erken.

136. Otomasyon Testi nedir?

• İnsan müdahalesini azaltan testin otomatik olarak gerçekleştirilmesi süreci, bu otomasyon testidir.
• Otomasyon testi QTP, Selenium, WinRunner vb. Bazı 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 raporu ve rapor oluşturur

Sayfa 34

137. Ne zaman otomatikleştireceksin?

• Çok fazla manuel çaba gerektiriyorsa. En az bir kez manuel çalıştırıyorum ve ardından otomatikleştiriyorum.
• Otomasyon, tekrar eden çoğu işlevsellik için iyidir

138. Hangi testler otomatikleştirilebilir?

• Regresyon testleri
• Duman testleri
• Fonksiyonel testler
• API
• Veri tabanı

139. Ne zaman otomatikleştirmeyeceksiniz?

• İşlevsellik değişmeye devam ederse
• İşlevsellik tüm proje boyunca yalnızca bir kez kullanılırsa
• **Ad-hoc testi** otomatikleştirilemez.

140. Bir scrum sprintinin süresi nedir? Sprint ne kadar sürüyor?

• Mevcut projemde senaryo döngümüz 4 haftadır. Sprintiniz burada ne kadar sürer? 2 hafta mı 4 hafta mı? (bazen iyidir

soru sormak için. ATM gibi davranmamanız gerektiğini unutmayın. Genellikle sadece soruya cevap veren kişileri unuturlar. Orada
bir denge olmalıdır.)
• Ekip sayımız 7 kişidir. 1 SM, 1 PO, 3 geliştirici, 1 MT, 1 AT

141. Hız nedir?

• Hız, takımın baskıya göre ilerleme hızıdır.
• İki farklı scrum takımıyla karşılaştırılamayacağını da söyleyebilirim.

142. "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 oluşturulur.
• Daha sonra test edenin sorumluluğu, hatayı ortadan kaldırarak yazılımı normal çalışma aşamasına geri döndürmektir.

143. Scrum'daki engeller hakkında ne biliyorsunuz ? Bazı engel örnekleri verin.

• Engeller, scrum ekibinin karşılaştığı, çalışma hızlarını yavaşlatan engeller veya sorunlardır.
• Scrum ekibinin işlerini "Bitti" yapmasını engellemeye çalışan bir şey varsa, bu bir engeldir.
• Engeller herhangi bir biçimde olabilir. Bazı engeller şu şekilde verilmiştir:

o Kaynak eksik veya hasta ekip üyesi
o Teknik, operasyonel, organizasyonel sorunlar
o Yönetim destek sistemi eksikliği
o İş sorunları
o Hava durumu, savaş vb. dış sorunlar
o Beceri veya bilgi eksikliği

• Çözüm: Ekip çalışması, sıkı çalışın, iyi iletişim kurun, çevrimiçi bağlantı, rehberlik ve eğitim

144. Çevik ve Scrum arasındaki fark ve benzerlik nedir?

• Çevik, geniş bir spektrumdur, proje yönetimi için kullanılan bir metodolojidir, Scrum ise sadece Çevik

Süreci ve adımlarını daha kısaca açıklar.

- Çevik bir uygulamadır, oysa scrum bu uygulamayı sürdürmek için bir prosedürdür.
- Çevik → projeleri adım adım veya aşamalı olarak tamamlamayı içeren benzerlik. Çevik metodoloji dikkate alınır

doğada yinelemeli olmak. Bir Agile formu olan Scrum, Agile ile aynıdır. Aynı zamanda artımlı ve yinelemelidir.

Sayfa 35

145. Artış nedir? Açıklamak.

- Artış, **bir sprint sırasında tamamlanan tüm ürün biriktirme kalemlerinin toplamıdır** .
- Her artış, kümülatif olduğu için önceki tüm sprint artış değerlerini içerir.
- Hedefimize ulaşmak için bir adım olduğu için sonraki sürümde mevcut modda olmalıdır.

146. Daily stand-up'tan ne anlıyorsunuz?

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

Aşağıdaki üç soruya cevap bulmak için 15 dakika -

o Dün ne yapıldı? Bugün için planın nedir?

o Görevinizi tamamlamanıza engel olan herhangi bir engel veya engel var mı?

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

147. Scrumban hakkında ne biliyorsun?

- Scrumban, yazılım geliştirme için Scrum ve Kanban tabanlı bir modeldir.
- Bu model, özellikle sürekli bakıma ihtiyaç duyan, çeşitli programlama hataları olan veya

bazı ani değişiklikler var.

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

148. Agile kalite stratejilerinden bazılarını belirtiniz mi?

- Yineleme
- Yeniden düzenleme
- Dinamik kod analizi
- Kısa geribildirim döngüleri
- İncelemeler ve inceleme
- Standartlar ve Talimatlar
- Dönüm noktası incelemeleri

149. Çevik Manifesto ve Prensiplerini biliyor musunuz? Kısaca açıklayın.

- Bu, çevik / scrum rollerinin çoğunun ipuçlarında olması gereken teoridir.
- Bu sorunun bir parçası olarak dört manifesto değeri ve 12 ilke olabildiğince açıklanmalıdır.
- % 100 doğru bir şekilde anlatılmasa bile iyi olmalı, ancak değerlerin ve ilkelerin niyetleri gelmeli

dışarı örneğin

- Bildiri

o Süreçler ve araçlardan ziyade bireyler ve etkileşimler

o Kapsamlı dokümantasyon yerine çalışan yazılım

o Sözleşme müzakeresi yerine müşteri işbirliği

o Bir planı takip etmek yerine değişime yanıt vermek

- Rehber ilkeler

o Müşteri Memnuniyeti

o Hoş Geldiniz Değişen Gereksinimler

o Çalışan Yazılım Sıkça Verilir (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 Görüşme, iletişimin en iyi şeklidir

o Çalışan yazılım, ilerlemenin birincil ölçüsüdür

o Sürdürülebilir kalkınma, sabit bir tempoyu koruyabilen

o Teknik mükemmelliğe ve iyi tasarıma sürekli dikkat

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

Sayfa 36

o En iyi mimariler, gereksinimler ve tasarımlar kendi kendini organize eden ekiplerden ortaya çıkar

o Ekip düzenli olarak nasıl daha etkili olabileceği üzerine düşünür ve buna göre ayarlanır

150. Yakma ve yanma çizelgelerinin faydası nedir?

- Yakma çizelgesi bir projede tamamlanan iş miktarını gösterirken, yakma çizelgesi miktarı gösterir

bir projeyi tamamlamak için geriye kalan iş.
• Bu nedenle, yanma ve yanma çizelgeleri bir projenin ilerlemesini izlemek için kullanılır.

151. Çevik modelin herhangi bir dezavantajı var mı? Öyleyse, açıkla.
• Evet, Agile modelinin bazı dezavantajları var, bazıları şöyle;
o Bir görevi tamamlamak için gereken çaba hakkında bir tahminde bulunmak kolay değildir. Durumda daha sorunlu hale geliyor
gereken toplam çaba hakkında bir fikir edinmek zorlaştığından büyük projeler için.
o Bazen, projenin tasarımına ve dokümantasyonuna düzgün şekilde odaklanmak mümkün değildir
o Müşterinin gereksinimlerinin doğru anlaşılmaması durumunda nihai proje müşteriyi karşılamayacaktır.
Gereksinimler. Böylece müşteri memnuniyetsizliğine yol açacaktır.
o Yalnızca Çevik metodolojiler konusunda hatırı sayılır deneyime sahip lider önemli kararlar alabilir. The
Deneyimi çok az olan veya hiç olmayan ekip üyeleri karar verme sürecine dahil olmazlar, bu nedenle ilerleme şansları olmaz.
onların bilgisi.

152. Agile'da Zero Sprint ve Spike'ı tanımlayın.
• Sıfır Sprint, Çevik'teki ilk sprint'in hazırlık adımı 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 Çiviler, tasarım veya araştırma, tasarım gibi teknik konularla ilgili faaliyetler için yaygın olarak kullanılmaktadır.
prototip oluşturma ve keşif.
o İki tür sivri uç vardır - işlevsel sivri uçlar ve teknik sivri uçlar.

153. Scrum Master'ın rolü nedir?
• Scrum ustası, Scrum takımının lideri ve koçudur.
• SM, performansı etkileyebilecek her türlü engelden takıma hizmet etmekten ve korumaktan sorumludur.
• SM'nin ana rolü, takımını sprint hedefine ulaşması için motive etmektir.
• Kendini organize eden ve motive olmuş bir ekip oluşturmaya odaklanmıştır ve her bir üye, aşağıdakilerin uygulanmasına aşinadır.
Çevik ve Scrum ilkeleri ve uygulamaları.
• SM, adanmış görevleri düzgün bir şekilde yerine getirip getirmediklerini scrum ekibini uygun bir şekilde kontrol eder.
• Sprint hedefine ulaşabilmeleri için takımın verimliliğini ve üretkenliğini artırmaktan da sorumludur.
etkili bir şekilde.

154. Scrum'daki bir hikaye noktası hakkında ne biliyorsunuz?
• Scrum'daki bir hikaye noktası, belirli bir işlemi gerçekleştirmek veya tamamlamak için gereken toplam çabanın tahmin edilmesine yönelik birimdir.
görev.

155. Sashimi'nin Scrum metodolojisindeki rolü nedir?
• Sashimi, Scrum metodolojisinde önemli bir rol oynar.
• Sashimi, geliştiriciler tarafından oluşturulan tüm işlevlerin 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.

Sayfa 37

156. Çevik test teriminden ne anlıyorsunuz?
• Çevik test, tamamen yazılım geliştirmenin çevik ilkelerine dayanan bir yazılım test uygulamasıdır. O bir
gereksinimlerin ürün sahibi ile ekip arasındaki işbirliğinin sonucu olduğu yinelemeli metodoloji.
Çevik ilkeler ve uygulamalar, müşteri gereksinimlerini karşılamak için, başarılı bir şekilde tamamlanır.
proje.

157. Scrum yerine şelale kullanılması hiç önerildi mi? Varsa, ne zaman olduğunu açıklayın.
• Evet, bazen Scrum yerine şelale modelinin kullanılması önerilmektedir.
• Müşteri gereksinimleri basit, iyi tanımlanmış, tam olarak anlaşılmış, öngörülebilir olduğunda ve tabi olmadığında yapılır.
projenin tamamlanmasına kadar değiştirmek.

158. Scrum, projeler için otomatik testlerin kullanılmasını neden teşvik ediyor?
• Scrum, en hızlısı yapmak için otomatik (otomatikleştirilmiş performans veya otomatik regresyon) testinin kullanılmasını teşvik eder
projenin olası teslimi. *otomatikleştirmek için kullandığınız bazı araçları açıklayabilirsiniz*

159. Çevik için bazı yaygın matrisleri açıklayın.
• **Hız** → Hız, son 3-4 sprintten alınan ortalama puan sayısıdır. Hepsinin toplamı ile ölçülür.
hikayelerin onaylanmış tahminleri. Kapasite, ilerleme vb. Hakkında fikir verir.
• **Kümülatif Akış Şeması** → 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** → zaman yatırımı hakkında hızlı bilgi veren önemli bir faktördür, örn.

zaman harcanıyor ve zaman faktörü olarak hangi göreve öncelik verilmesi gerekiyor.

- **Zaman Kapsamı** → Test sırasında bir koda verilen zamandır. Yüzde olarak hesaplanır.

test paketi tarafından çağrılan kod satırlarının sayısı ve ilgili kod satırlarının toplam sayısı.

- **Verilen İş Değeri** → Ekibin çalışma verimliliğini ifade eden bir terimdir. İş hedefleri

öncelik, karmaşıklık ve ROI düzeyine göre sayısal değerler 1,2,3 vb. atanır.

- **Kusur Giderme Farkındalığı** → Ekibin kaliteli bir ürün sunmasına yardımcı olan faktördür. Bir kimlik

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

- **Hata Çözüm Süresi** → Ekip üyelerinin hataları (hataları) tespit edip bir öncelik belirlediği bir prosedürdür.

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

Kusur resmini temizleme, kusur tespitini planlama, kusur tespitini tamamlama, çözüm oluşturma ve işleme

bildiri.

- **Sprint Burndown Matrix** → Sprint burndown grafiği, uygulanmayan veya uygulanmayanların sayısını temsil eden bir grafiktir.

Scrum döngüsü sırasında sprintler uyguladı. Bu matrik, sprint ile tamamlanan işin izlenmesine yardımcı olur.

160. Çevik modeli kullandığınız bazı metodolojileri ve gelişmeleri adlandırın.

- Çevik modelin kullanılabileceği metodolojilerden ve geliştirmelerden bazıları şunlardır:
  - o Kristal metodolojileri
  - o Yalın yazılım geliştirme
  - o Dinamik geliştirme ve Özellik odaklı geliştirme

161. Scrum M / Ürün O / Çevik ekip üyesi olarak deneyiminizi paylaşın ve birincil sorumluluklarınız nelerdi?

- Bu sorudaki püf noktası, açıklarken kendi kendini organize eden ve kendi kendini motive eden bir ekip gösterip göstermediğinizdir.

162. Projenizdeki sprintlerin / yinelemelerin uzunluğu neydi?

- Buradaki fikir, ne tür bir ortamda çalıştığınıza karar vermektir. Kesinlikle gibi takip soruları olacak

bu uzunluk başlangıçta sabitlendi ve hiç değişmedi mi? Bu uzunluktan daha fazlasını mı yoksa daha azını mı denediniz?

Sayfa 38

163. "Planlama Poker" tekniği hakkında ne biliyorsunuz?

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

planlama poker tekniği oturumu başlatmak, çevik kullanıcı hikayesi ürün sahibi tarafından okunur.

- Poker planlama tekniğinde gerçekleştirilen adımlar şunlardır:
  - o Her bir tahmincide 0, 1, 2, 3, 5 vb. değerlere sahip bir deste poker kartı vardır ve hikaye puanlarını belirtmek için idealdir.
    günler veya takımın tahmin için kullandığı başka bir şey.
  - o Her bir tahmincinin ürün sahibi ile bir görüşmesi vardır ve ardından müşterilere göre özel olarak bir kart seçer.
    bağımsız tahmin.
  - o Aynı değere sahip kartlar tüm tahminciler tarafından seçilirse, bu bir tahmin olarak kabul edilir. Değilse, tahminci
    tahminlerinin yüksek ve düşük değerini tartışır.
  - o Sonra yine, her bir tahminci özel olarak bir kart seçer ve ortaya çıkarır. Bu poker planlama süreci, bir hedefe ulaşmak için tekrarlanır.
    Genel Anlaşma.

164. Projelerinizdeki kullanıcı hikayesi haritalamasını ve hikayelerinin tahminini nasıl yaptınız?

- Planlama poker, tişört, boyutlandırma vb. Gibi herhangi bir tahmin tekniği kullandınız mı? Kullandığınız teknik ne olursa olsun

proje sadece çok açık bir şekilde bahsedin.

165. Çevik test metodolojisi diğer test metodolojilerinden nasıl farklıdır?

- 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.

optimum sonuçlar için stratejik değişiklikler ve diğerleri.

166. Scrum takım üyelerini idare ederken projenizde karşılaştığınız en büyük zorluk nedir?

- Dolandırıcılığın ilk aşamalarında genellikle karşılaşılan zorluklar, hızı sabitlemek, ekip üyeleri çatışır,

zaman boks vb.

o Uygulama test edilebilecek kadar stabil olmalıdır. o Her zaman zaman kısıtlaması altında test etme

o Gereksinimleri anlamak.

o Alan bilgisi ve iş kullanıcısı bakış açısı anlayışı.

167. İlk olarak hangi testler yapılmalı?

- Tam Uygulamanın Test Edilmesi. o Regresyon testi.
- Yetenekli test uzmanlarının olmaması. o Değişen gereksinimler.
- Kaynak, araç ve eğitim eksikliği

168. Scrum Master sertifikanız var mı?

- Sertifikalı bir scrum ustasıysanız, sertifika sınavı, alınan puan ve

sertifika sınavını geçtiğiniz yıl. Sertifikanız yoksa, deneyiminizden bahsedin ve vurgulayın
belirli alan. Ayrıca, yakın gelecekte sertifikasyona yatırım yapmayı planlayıp planlamadığınızı görüşmeciye bildirin.

169. Çevik sertifikanız var mı? Neden bu sertifikayı seçtiniz?

- Çevik ve Scrum metodolojileri bir projeyi en kısa sürede tamamlamak için kullanılır.
- Çevik ilkelerin uygulanması müşteri memnuniyetiyle sonuçlanırken, scrum, esnek özelliği ile bilinir.

Gereksinimler.

170. Daha önce offshore ekibiyle çalıştınız mı?

- Hayır, bilmiyorum. (*Offshore, temelde ekibin farklı bir ülkede bulunduğu ancak yine de şirketiniz tarafından istihdam edildiği anlamına gelir*)

Sayfa 39

171. Yaygın UI test otomasyon araçları nelerdir?

- Selenyum
  - o Salatalık
  - o TestNG
- Appium
- Açıölçer
- Şarap
- UFT / QTP
- **Katalon Stüdyo**

172. Test Yazılımı nedir? Test gereçleri?

- Uygulamanın test edilmesine yardımcı olan yazılımın alt kümesidir.
- Testleri planlamak, tasarlamak ve yürütmek için test yazılımı gerekir. Belgeleri, komut dosyalarını, girdileri, beklenen sonuçları, kurulumu içerir

ve testte kullanılan ek yazılım veya yardımcı programlar.

- Test yazılımı, bir yazılım paketini test etmek için gerekli olan tüm yardımcı programların ve uygulama yazılımının birleşimine verilen bir terimdir.

Özel olduğu için;
o Farklı amaç
o Kalite için farklı ölçütler ve
o Farklı kullanıcılar

173. İstemci veya sunucu ortamı testi nasıl etkiler?

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

sunucu vb. istemci veya sunucu teknolojileri ile çalışırken, testler kapsamlı olacaktır.

- Zaman sınırımız olduğunda entegrasyon testini yapıyoruz. Çoğu durumda yükü, stresi ve performansı tercih ederiz.

istemci veya sunucu ortamı için uygulamanın yeteneklerini incelemek için test.

174. Bir test uzmanının tipik deneyimleri veya aşırı yük çalışma günü veya
test (SDET) kaynaklarında yazılım geliştirme mühendisi?

Test uzmanı için herhangi bir günde üç temel görev her zaman çok fazla zaman alır:

- Projenin gereksinimlerini anlamak.
- Hazırlama ve yürütme, müşterinin beklenen işlevlerine dayalı test senaryoları gerektirir.
- Müşteri için geliştirilen bireysel işlevlerde tanımlanan hataların geliştiriciye raporlanması ve yeniden test edilmesi

Beklenen işlevselliğin herhangi bir yaygın hata olmadan düzgün bir şekilde sunulmasını sağlamak için geliştirici tarafından yeniden teslim edildikten sonra aynı.

175. Bir test uzmanının, sağlanan ürünün gerçekten taşınmaya hazır olduğuna nasıl karar verebileceğine ilişkin bazı uzman yorumlarını açıklayın.
canlı ortamda?

Bu kritik kararlardan biridir, bu yüzden asla tek kişi veya genç adamlar tarafından alınmamıştır. Yalnızca geliştirici ve test kullanıcısı
Bu kararın alınmasına dahil olmadığında, üst yönetim periyodik olarak buna dahil olur. Yönetim testi esas olarak aşağıdakileri sağlar:
Ürün teslimatının hatasız olduğundan emin olmak için aşağıda doğrulama:

- Test uzmanı tarafından sağlanan hata raporlarını doğrulama. Hatanın nasıl çözüldüğü ve test cihazı tarafından yeniden test edilip edilmediği.
- Test eden tarafından bu belirli işlev, dokümantasyon ve alınan onay için yazılan tüm test senaryolarının doğrulanması

aynı test cihazından.

- Yeni işlevlerin mevcut herhangi bir işlevi bozmamasını sağlamak için otomatik test senaryolarını çalıştırın.
- Bazen, gelişen tüm bileşenin test senaryoları tarafından kapsanmasını sağlayan test kapsamı raporunun doğrulanması

yazılı.

Sayfa 40

176. Sağlanan ürünün gerçekten taşınmaya hazır olduğuna bir test uzmanının nasıl karar verebileceğine ilişkin bazı uzman yorumlarını açıklayın.

canlı ortamda?

Bu kritik kararlardan biridir, bu yüzden asla tek kişi veya genç adamlar tarafından alınmamıştır. Yalnızca geliştirici ve test kullanıcısı

Bu kararın alınmasına dahil olmadığında, üst yönetim periyodik olarak buna dahil olur. Yönetim testi esas olarak aşağıdakileri sağlar:

Ürün teslimatının hatasız olduğundan emin olmak için aşağıda doğrulama:

- Test uzmanı tarafından sağlanan hata raporlarını doğrulama. Hatanın nasıl çözüldüğü ve test cihazı tarafından yeniden test edilip edilmediği.
- Test eden tarafından bu belirli işlev, dokümantasyon ve alınan onay için yazılan tüm test senaryolarının doğrulanması

  aynı test cihazından.
- Yeni işlevlerin mevcut herhangi bir işlevi bozmamasını sağlamak için otomatik test senaryolarını çalıştırın.
- Bazen, gelişen tüm bileşenin test senaryoları tarafından kapsanmasını sağlayan test kapsamı raporunun doğrulanması

  yazılı.

177. Hangi Test Teknikleri var ve bunların amacı nedir?

Test Teknikleri esas olarak iki amaç için kullanılır: a) Kusurların belirlenmesine yardımcı olmak için, b) Test senaryolarının sayısını azaltmak.

- **Eşdeğer bölümleme** , esas olarak, aşağıda belirtilen farklı veri kümelerini tanımlayarak test senaryolarının sayısını azaltmak için kullanılır.

  aynı değildir ve her veri kümesinden yalnızca bir test yürütmek
- Sınır Değer Analizi , izin verilen verilerin sınırlarında sistemin davranışını kontrol etmek için kullanılır.
- Durum Geçiş Testi , izin verilen ve verilmeyen durumları ve bir durumdan diğerine geçişleri doğrulamak için kullanılır.

  çeşitli giriş verileri
- Çift veya Tüm Çiftler Testi çok güçlü bir test tekniğidir ve esas olarak test senaryolarının sayısını azaltmak için kullanılır.

  özellik kombinasyonlarının kapsamını artırmak.

178. Bir kusur veya hata raporuna hangi bilgiler dahil edilmelidir?

- Kusurun kısa bir özeti
- Yeniden üretme adımları da dahil olmak üzere kusurun tam açıklaması
- Gerekirse ekran görüntüsü ekleri
- Kusurun bulunduğu ve ortaya çıktığı tarih
- Kusuru kim ihbar etti.
- Kusurun ciddiyeti ve / veya Önceliği
- Kusurun hangi bileşen atandığı.

Sayfa 41

TEST TÜRLERİ

Kodunuzdaki değişikliklerin beklendiği gibi çalıştığından emin olmak için kullanabileceğiniz birçok farklı test türü vardır. Değil

Yine de tüm testler eşittir ve burada ana test uygulamalarının birbirinden nasıl farklı olduğunu göreceğiz.

Manuel ve otomatikleştirilmiş test

- Yüksek düzeyde, manuel ve otomatik testler arasında ayrım yapmamız gerekiyor. Manuel testler şahsen yapılır,

  Uygulamaya tıklayarak veya uygun araçlarla yazılım ve API'lerle etkileşime girerek. Bu çok

  Birisinin bir ortam kurmasını ve testleri kendisinin gerçekleştirmesini gerektirdiği için pahalıdır ve insanlara eğilimli olabilir

  hata, test eden kişi test komut dosyasında yazım hataları yapabilir veya adımları atlayabilir.
- Öte yandan, otomatik testler, içinde yazılmış bir test komut dosyasını çalıştıran bir makine tarafından gerçekleştirilir.

  ilerlemek. Bu testler, bir sınıftaki tek bir yöntemi kontrol etmekten,

  kullanıcı arayüzündeki bir dizi karmaşık eylem aynı sonuçlara yol açar. Otomatik testlerden çok daha sağlam ve güvenilirdir

  - ancak otomatik testlerinizin kalitesi, test komut dosyalarınızın ne kadar iyi yazıldığına bağlıdır.
- Otomatik test, sürekli entegrasyon ve sürekli teslimatın önemli bir bileşenidir ve işletmenizi ölçeklendirmenin harika bir yoludur.

  Uygulamanıza yeni özellikler eklerken KG süreci. Ancak, ne olduğu ile bazı manuel testler yapmanın hala değeri var

aşağıda göreceğimiz gibi keşif testi olarak adlandırılır.

Farklı test türleri

Duman testi

- Duman testleri, uygulamanın temel işlevselliğini kontrol eden temel testlerdir. Hızlı bir şekilde uygulanmaları amaçlanmıştır ve Amaçları, size sisteminizin temel özelliklerinin beklendiği gibi çalıştığına dair güvence vermektir.
- Duman testleri, yeni bir yapı yapıldıktan hemen sonra daha pahalı testler yapıp yapamayacağınıza karar vermek için yararlı olabilir veya uygulamalarının yeni dağıtılan ortamda düzgün çalıştığından emin olmak için bir dağıtımdan hemen sonra.

Entegrasyon testleri

- Entegrasyon testleri, uygulamanız tarafından kullanılan farklı modüllerin veya hizmetlerin birlikte iyi çalıştığını doğrular. Örneğin, veritabanıyla etkileşimi test etmek veya mikro hizmetlerin beklendiği gibi birlikte çalıştığından emin olmak. Bu tür Uygulamanın birden çok parçasının çalışır durumda olmasını gerektirdiğinden testleri çalıştırmak daha pahalıdır.

Gerileme testi

- Gerileme, uygulamanın değişmemiş kısımlarının yeniden test edilmesi anlamına gelir. Regresyon testi, doğrulamak için yapılan bir testtir. Yazılımdaki bir kod değişikliğinin ürünün mevcut işlevselliğini etkilemediğini.
- Bu test, ürünün yeni eklenen işlevlerle veya üründeki herhangi bir değişiklikle daha önce olduğu gibi iyi çalıştığından emin olur. mevcut özellik veya hata düzeltmesi yapıldıktan sonra. Daha önce yürütülen test senaryoları, etkisini doğrulamak için yeniden yürütülür. değişiklik.
- Regresyon Testi, önceki test durumlarının olup olmadığını kontrol etmek için test senaryolarının yeniden yürütüldüğü bir Yazılım Testi türüdür. uygulamanın işlevselliği iyi çalışıyor ve yeni değişiklikler herhangi bir yeni hataya yol açmadı. Bu test olabilir tek bir hata düzeltmesinde bile orijinal işlevsellikte önemli bir değişiklik olduğunda yeni bir derlemede gerçekleştirilir.
- Bir uygulamanın bir bütün olarak herhangi bir modülde veya işlevde değişiklik için test edilmesi, Gerileme Testi olarak adlandırılır. Bu Regresyon Testinde tüm sistemi kaplamak zordur, bu nedenle bu türler için tipik olarak otomasyon test araçları kullanılır. test yapmak.

Sayfa 42

Fonksiyonel testler

- İşlevsel testler, bir uygulamanın iş gereksinimlerine odaklanır. Yalnızca bir eylemin çıktısını doğrularlar ve bu eylemi gerçekleştirirken sistemin ara durumlarını kontrol edin.
- Her ikisi de birden fazla bileşen gerektirdiğinden, entegrasyon testleri ile fonksiyonel testler arasında bazen bir karışıklık olur birbirleriyle etkileşim kurmak. Aradaki fark, entegrasyon testinin veritabanını sorgulayabildiğinizi doğrulayabilmesidir. işlevsel bir test ise, ürün gereksinimleri tarafından tanımlanan veri tabanından belirli bir değer almayı bekler.

FONKSİYONEL TEST	FONKSİYONEL OLMAYAN TESTLER
• Birim Testi	• Performans testi
• Duman testi	• Yük Testi
• Gerileme testi	• Hacim Testi
• Sağlık Testi	• Stres testi
• Entegrasyon Testi	• Güvenlik Testi
• Kabul testleri	• Kurulum Testi
• GUI Testi	• Penetrasyon testi
• Kullanılabilirlik testi	• Uyumluluk Testi
• Sistem Testi	• Göç Testi
• Alfa Testi	• Kurtarma testi
• Beta testi	• Güvenilirlik Testi
• Kullanıcı Kabul Testi	• Kullanılabilirlik testi

Birim testleri

- Birim testleri, uygulamanızın kaynağına yakın, çok düşük seviyelidir. Bireysel yöntemleri test etmekten oluşurlar ve yazılımınız tarafından kullanılan sınıfların, bileşenlerin veya modüllerin işlevleri. Birim testleri genellikle otomatikleştirmek için oldukça ucuzdur ve sürekli bir bütünleştirme sunucusu tarafından çok hızlı bir şekilde çalıştırılabilir.

Uçtan uca testler

- Uçtan uca test, eksiksiz bir uygulama ortamında yazılımla bir kullanıcı davranışını kopyalar. Bunu doğrular çeşitli kullanıcı akışları beklendiği gibi çalışır ve bir web sayfası yüklemek veya oturum açmak kadar basit veya çok daha karmaşık olabilir e-posta bildirimlerini, çevrimiçi ödemeleri vb. doğrulayan senaryolar ...
- Uçtan uca testler çok kullanışlıdır, ancak gerçekleştirilmesi pahalıdır ve otomatikleştirildiklerinde bakımı zor olabilir. Birkaç önemli uçtan uca test yapılması ve daha düşük seviyeli test türlerine (birim ve entegrasyon) daha çok güvenilmesi önerilir. testler) kırılan değişiklikleri hızlı bir şekilde belirleyebilmek için.

Kara Kutu Testi

• Davranışsal, opak kutu, kapalı kutu, spesifikasyona dayalı veya göz göze test olarak da bilinen kara kutu testi,

hakkında fazla bir şey bilmeden bir yazılımın / uygulamanın işlevselliğini analiz eden bir Yazılım Test yöntemidir.

test edilen öğenin iç yapısı / tasarımı ve girdi değerini çıktı değeriyle karşılaştırır.

Kara kutu testindeki ana odak, bir bütün olarak sistemin işlevselliğidir.

• 'Davranış testi' terimi, kara kutu testi için de kullanılır. Davranışsal test tasarımı siyahtan biraz farklıdır.

kutu testi tasarımı, çünkü dahili bilginin kullanılması kesinlikle yasak değil, ancak yine de önerilmez.

Blackbox testi türleri :

- o Fonksiyonel Testler,
- o İşlevsel olmayan,
- o Eşdeğerlik Bölümleme,
- o Sınır Değer Analizi,
- o Karar Tablosu Testi,
- o Hata Tahmin Etme,
- o Grafik Tabanlı Yöntemler,
- o Karşılaştırma,
- o Durum Geçiş testleri

Sayfa 43

Beyaz Kutu Testi

• Beyaz Kutu testi, bir uygulamanın kodunun dahili mantığı hakkındaki bilgilere dayanır.

• Cam kutu Testi olarak da bilinir. Bu tür bir işlemin gerçekleştirilmesi için dahili yazılım ve kod çalışması bilinmelidir.

test yapmak. Bu testler kapsamında, kod ifadelerinin, dalların, yolların, koşulların vb. Kapsamına dayanır.

• Sonuç

• Yukarıda belirtilen Yazılım Test Türleri, testin yalnızca bir parçasıdır. Ancak yine de 100'den fazla

test türleri, ancak tüm test türleri tüm proje türlerinde kullanılmamaktadır. Bu nedenle, yaygın olarak kullanılan bazı Türleri ele aldım.

Çoğunlukla test yaşam döngüsünde kullanılan Yazılım Testi.

• Ayrıca, farklı organizasyonlarda kullanılan alternatif tanımlar veya süreçler vardır, ancak temel kavram aynıdır

her yerde. Bu test türleri, süreçleri ve uygulama yöntemleri proje ne zaman ve ne zaman proje

gereksinimler ve kapsam değişiklikleri.

Whitebox testi türleri :

- o Birim Testi,
- o Yürütme Testi,
- o Mutasyon Testi,
- o Operasyon Testi
- o Bildirim Kapsamı,
- o Şube Kapsamı,
- o Yol Kapsamı,
- o Güvenlik Testi,
- o Grafik Tabanlı Yöntemler,
- o Karşılaştırma,
- o Durum Geçiş testleri,

Kabul testleri

• Kabul testleri, bir sistemin iş gereksinimlerini karşılayıp karşılamadığını doğrulamak için yapılan resmi testlerdir. Hepsini gerektirirler

uygulama hazır ve çalışır durumda olmalı ve kullanıcı davranışlarını kopyalamaya odaklanmalıdır. Ancak daha da ileri gidebilir ve

sistemin performansı ve belirli hedefler karşılanmazsa değişiklikleri reddeder.

Performans testi

• Performans testleri, önemli yük altında olduğunda sistemin davranışlarını kontrol eder. Bu testler işlevsel değildir ve

platformun güvenilirliğini, kararlılığını ve kullanılabilirliğini anlamak için çeşitli biçime sahip olabilir. Örneğin, olabilir

çok sayıda isteği yerine getirirken yanıt sürelerini gözlemlemek veya sistemin önemli ölçüde nasıl davrandığını görmek

veri.

• Performans testlerinin uygulanması ve çalıştırılması doğaları gereği oldukça maliyetlidir, ancak yeni değişiklikler olup olmadığını anlamanıza yardımcı olabilirler

sisteminizi bozacak.

Ad-hoc Test

• İsmin kendisi, bu testin amaca yönelik olarak gerçekleştirildiğini, yani test senaryosuna atıfta bulunmadığını ve ayrıca

bu tür testler için herhangi bir plan veya belge olmadan. Bu testin amacı kusurları bulmak ve

uygulamanın herhangi bir akışını veya herhangi bir rastgele işlevselliği çalıştırarak uygulamayı bozun.

• Ad-hoc test, kusurları bulmanın gayri resmi bir yoludur ve projedeki herkes tarafından yapılabilir. Tanımlamak zor

test senaryosu olmayan kusurlar, ancak bazen geçici test sırasında bulunan kusurların olmamış olması mümkündür.

mevcut test senaryoları kullanılarak tanımlanır.

Sınır Değer Testi

• Bu tür testler, uygulamanın sınır düzeyindeki davranışını kontrol eder.

• Sınır değer Test, sınır değerlerinde kusur olup olmadığını kontrol etmek için gerçekleştirilir. Sınır değer testi aşağıdakiler için kullanılır:

farklı bir sayı aralığını test etmek. Her aralık için bir üst ve alt sınır vardır ve test yapılır.

bu sınır değerleri.

• Test, 1'den 500'e kadar bir test aralığı gerektiriyorsa, Sınır Değer Testi 0, 1, 2'deki değerler üzerinde gerçekleştirilir,

499, 500 ve 501.

Sayfa 44

Keşif testi

- Keşif Testi, test ekibi tarafından gerçekleştirilen gayri resmi testlerdir. Bu testin amacı,

  uygulama ve uygulamada var olan kusurları aramak. Bazen bu test sırasında büyük

  keşfedilen kusur sistem arızasına bile neden olabilir.
- Keşif testi sırasında, hangi akışı test ettiğinizi ve daha önce hangi aktiviteyi yaptığınızı takip etmeniz önerilir.

  belirli akışın başlangıcı.

Testlerinizi nasıl otomatikleştirebilirsiniz?

- Bir kişi yukarıda bahsedilen tüm testleri uygulayabilir, ancak bunu yapmak çok pahalı ve verimsiz olacaktır. Gibi

  insanlar, çok sayıda eylemi tekrarlanabilir ve güvenilir bir şekilde gerçekleştirme kapasitemiz sınırlıdır. Ama bir makine

  bunu kolayca yapabilir ve giriş / şifre kombinasyonunun 100. kez şikayet etmeden çalıştığını test eder.
- Testlerinizi otomatikleştirmek için, önce testlerinizi kendinize uygun bir test çerçevesi kullanarak programlı olarak yazmanız gerekir.

  uygulama. PHPUnit , Mocha , RSpec , PHP, JavaScript ve Ruby için kullanabileceğiniz test çerçeveleri örnekleridir.

  sırasıyla. Her dil için birçok seçenek var, bu yüzden biraz araştırma yapıp sormanız gerekebilir

  sizin için en iyi çerçevenin ne olacağını öğrenmek için geliştirici toplulukları.
- Testleriniz terminalinizden komut dosyası aracılığıyla yürütüldüğünde, bunların bir

  Bamboo gibi sürekli entegrasyon sunucusu veya Bitbucket Pipelines gibi bir bulut hizmeti kullanın. Bu araçlar

Sayfa 45

JAVA

1. Java Sanal Makinesi

- JVM, derlenmiş java sınıfı dosyalar için bir çalışma zamanı ortamı olan Java Sanal Makinesi anlamına gelir.

2. JavaScript ve Java aynı mı?

- Java bir OOP programlama dilidir, Java Script ise bir OOP kodlama dilidir.
- Java, JavaScript kodu yalnızca bir tarayıcıda çalıştırılırken sanal bir makinede veya tarayıcıda çalışan uygulamalar oluşturur.
- JavaScript kodunun tamamı metin içindeyken Java kodunun derlenmesi gerekir.
- Farklı eklentiler gerektirirler.

3. Java Runtime Environment

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

• JRE = JVM + Kitaplık Sınıfları

4. Java Geliştirme Kiti

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

• JDK = JRE + Geliştirme araçları

5. Java Interview için popüler konular nelerdir?

Java röportajı için popüler konulardan bazıları şunlardır:

• OOPS Kavramları
• Java Dizesi
• Koleksiyon Çerçevesi
• Çoklu kullanım
• Jenerikler
• İstisna işleme
• Akış API'si
• Lambda İfadeleri
• En Son Sürüm Özellikleri
• Java EE Frameworks - Spring, Hibernate vb.

6. Gelişmiş Java Konuları nelerdir?

Java röportajı için popüler konulardan bazıları şunlardır:

• Yığın ve Yığın Bellek
• Çöp toplama
• Reflection API
• İş Parçacığı Kilitlenme
• Java ClassLoader
• Java Günlük API'si
• Java'da uluslararasılaşma
• Java Modül Sistemi

7. Tüm sınıfların üst sınıfı hangi sınıftır?

• java.lang.Object , tüm java sınıfları için kök sınıftır ve onu genişletmemize gerek yoktur.

8. Yöntem nedir?

• Bir işlemi gerçekleştirmek için bir arada gruplandırılmış ifadelerin toplanması. **System.out.println () 'i** aradığınızda

yöntem, örneğin, sistem konsolda bir mesaj görüntülemek için gerçekte birkaç deyimi yürütür.

• Bir yöntem, adla atıfta bulunulan ve bir programın herhangi bir noktasında yalnızca aşağıdaki yöntemlerle çağrılabilen (çağrılan) bir kod kümesidir.

yöntemin adını kullanarak. Bir yöntemi, verilere göre hareket eden ve genellikle bir değer döndüren bir alt program olarak düşünün. Her yöntem kendi adı var.

9. Yapıcı nedir?

• Java'daki bir kurucu, nesneleri başlatmak için kullanılan özel bir yöntemdir. Yapıcı, bir nesnenin bir nesnesi olduğunda çağrılır.

sınıf oluşturulur.

• new () anahtar sözcüğü kullanılarak bir nesne her oluşturulduğunda, en az bir yapıcı (varsayılan kurucu olabilir) için çağrılır

aynı sınıfın veri üyelerine başlangıç değerleri atayın.

Sayfa 46

10. Oluşturucu ve Yöntem arasındaki fark?

• Oluşturucunun bir dönüş türü yoktur ve kurucunun adı, sınıf adıyla aynı olmalıdır.

o Oluşturucu, yeni bir nesne oluşturulduğunda otomatik olarak çağrılır. Oluşturucu örtük olarak çağrılır.
o Herhangi bir kurucumuz yoksa Java derleyicisi varsayılan bir kurucu sağlar.
o Oluşturucular 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 sağlanmamaktadır.
o Yöntemler alt sınıflar tarafından miras alınır.

11. Yerel bir değişken ile bir durum değişkeni arasındaki fark nedir?

• **Yerel bir değişken** tipik olarak bir yöntem, yapıcı veya blok içinde kullanılır ve yalnızca yerel kapsama sahiptir. Böylece bu değişken,

yalnızca bir blok kapsamında kullanılabilir.

• Yerel bir değişkene sahip olmanın en iyi yararı, sınıftaki diğer yöntemlerin o değişkenden haberdar olmayacak olmasıdır.

Misal

```
eğer (x> 100) {
        Dize testi = "Alberto";
}
```

• Java'daki bir **örnek değişkeni** , nesnesinin kendisine bağlı bir değişkendir. Bu değişkenler bir

sınıf, ancak bir yöntemin dışında. Bu sınıfın her nesnesi, onu kullanırken değişkenin kendi kopyasını oluşturacaktır. Böylece herhangi biri Değişkende yapılan değişiklikler, o sınıfın başka herhangi bir örneğini yansıtmayacak ve o özelliğe bağlı olacaktır.
yalnızca örnek.
Misal

```
class Test {
        public String EmpName;
        public int empAge;
}
```

12. Nesneye Yönelik Programlama (OOP)

• OOP, eylemler (mantık ve işlevler) yerine nesne etrafında düzenlenen bir programlama dili modelidir.

• Diğer bir deyişle, OOP esas olarak mantık yerine manipüle edilmesi gereken nesnelere odaklanır. Bu yaklaşım programlar için ideal, büyük ve karmaşık kodlar ve aktif olarak güncellenmesi veya bakımı yapılması gereken;
• Geliştirme ve bakımı kolaylaştırır - Veri gizleme sağlar - Gerçek dünyayı simüle etme yeteneği sağlar.

OOP dili 4 prensibi takip eder :

o **Kapsülleme** : Özel anahtar kullanarak verilere doğrudan erişimi gizleyebiliriz ve alıcı kullanarak özel verilere erişebiliriz ve ayarlayıcı yöntemi.

o **Soyutlama** : Uygulama detaylarının gizlenmesi ve kullanıcıya sadece işlevselliğin gösterilmesi işlemidir. Soyutlama nasıl yaptığı yerine nesnenin ne yaptığına odaklanmanızı sağlar.

o **Kalıtım** : İki sınıf arasındaki ilişkiyi tanımlamak için kullanılır. Bir çocuk sınıfı tüm özellikleri edindiğinde ve miras olarak bilinen ebeveyn sınıfın davranışları. Çocuk sınıfı, ebeveyn sınıfında yazılan tüm kodları yeniden kullanabilir. Sağlar kodun yeniden kullanılabilirliği.

o **Çok biçimlilik** : Nesnenin çoklu biçimde davranma yeteneğidir. Java'da polimorfizmin en yaygın kullanımı değişkenin bir üst sınıf başvuru türü, bir alt sınıf nesnesine başvurmak için kullanıldığı zamandır.

Misal

WebDriver sürücüsü = yeni ChromeDriver ();

Polimorfizme ulaşmak için yöntem aşırı yükleme ve geçersiz kılma kullanıyoruz.

Sayfa 47

13. Kapsülleme nedir ve nasıl kullandınız?

• Değişkenleri özel yaparak ve genel alıcı ve ayarlayıcı yöntemleri sağlayarak veri gizleme.
• Projemde, test verilerini ve gerçek verileri yönetmek için birden fazla POJO / BEAN sınıfı oluşturdum.
• EX: JSON'u API yanıtından alıp POJO sınıfımın nesnesine dönüştürüyorum, tüm değişkenler alıcılarla özeldir ve ayarlayıcı.

14. Soyutlama kavramı nedir ?

• OOP'de, soyutlama, uygulama ayrıntılarını kullanıcıdan gizleme sürecidir, yalnızca işlevsellik sağlanacaktır. kullanıcıya.
• Başka bir deyişle, kullanıcı nesnenin nasıl yaptığı yerine ne yaptığı bilgisine sahip olacaktır.
• Java'da soyutlama, Soyut sınıflar ve arayüzler kullanılarak gerçekleştirilir.
• Örneğin: banka hesabınızda çevrimiçi oturum açtığınızda, kullanıcı kimliğinizi ve parolanızı girip oturum açma tuşuna basarsınız. Ne daha sonra, sunucuya gönderilen giriş verilerinin nasıl doğrulandığı, tümüyle sizden soyutlanır.

15. Soyutlama ve Kapsülleme Arasındaki Fark ?

• **Soyutlama** , nesnenin **nasıl** yaptığı yerine **ne** yaptığına odaklanmanızı sağlar .
o **Kapsülleme** , nesnenin bir şeyi nasıl yaptığına dair dahili ayrıntıları gizlemek anlamına gelir.
• **İstenmeyen** verileri gizlemek ve ilgili verileri vermek için **soyutlama** kullanılır .
o **Kapsülleme** , kodu ve verileri gizlemek ve verileri dışarıdan korumak anlamına gelir.
• **Soyutlama** , Soyut sınıf ve Arayüzler kullanılarak elde edilebilir
o **Kapsülleme** , "özel" anahtar kelime kullanılarak elde edilebilir.

16. Soyut Sınıf ve Arayüz arasındaki fark ?

• Temel fark, bir Java arayüzünün yöntemlerinin dolaylı olarak soyut olması ve uygulamalarının olmamasıdır. Java özeti sınıf, varsayılan bir davranışı uygulayan örnek yöntemlerine sahip olabilir.
• Abstract anahtar sözcüğüyle bildirilen bir sınıf, soyut sınıf olarak bilinir. Soyut ve soyut olmayan yöntemlere sahip olabilir.
• Bir Arayüz, bir sınıfın taslağıdır. Bu bir şablondur ve interface anahtar sözcüğü ile bildirilmiştir. Soyut yöntemlere sahip olabilir, varsayılan yöntemler, statik yöntemler ve genel nihai statik değişkenler
• Abstract sınıfını kullanmak istediğimizde, " **extended** " anahtar sözcüğünü kullanırız. Arayüzü kullanmak istediğimizde, " uygulama " kullanıyoruz anahtar kelime.
• Soyut sınıf ve arayüz, soyutlama elde etmek için kullanılır Her ikisi de somutlaştırılamaz; bir nesne yaratamayız.

17. Polimorfizm nedir?

• Polimorfizm, OOP'de çok önemli bir kavramdır çünkü;
o çağrının yapıldığı nesneye bağlı olarak çalışma süresindeki uygulamaların davranışını değiştirmeyi sağlar. olur.
o Polimorfizm ile; bir nesnenin farklı biçimleri olabilir
• İki tip → Statik olan **Derleme Zamanı** ve alt ve üst sınıf ile ilgili olan **Çalışma Zamanı** Polimorfizmi.
• Çok biçimlilik, Yöntem aşırı yükleme ve yöntemi geçersiz kılma kavramı kullanılarak gerçekleştirilir. Bu sadece olabilir sınıflar, miras kullanarak üst ve alt ilişki altında olduğunda.

18. Kalıtım nedir?

• Kalıtım, ebeveyn-çocuk ilişkisi olarak da bilinen **IS-A** ilişkisini temsil eder .
• Java'da bir sınıfın başka bir sınıfın özelliklerini (alanları ve yöntemleri) devralmasına izin verilen mekanizmadır.
• Java'da kalıtımın arkasındaki fikir, mevcut sınıflar üzerine inşa edilmiş yeni sınıflar oluşturabilmenizdir.
• Var olan bir sınıftan miras aldığınızda, üst sınıfın yöntemlerini ve alanlarını yeniden kullanabilirsiniz.

• Ayrıca, mevcut sınıfınıza yeni yöntemler ve alanlar da ekleyebilirsiniz.

Sayfa 48

• Kodun yeniden kullanımı, kalıtımın en önemli yararıdır çünkü alt sınıflar,

süper sınıf.

19. Önemli terminoloji Kalıtım ?

• **Sınıf:** ortak özelliklere sahip nesneler grubu. Nesnelerin oluşturulduğu bir şablon veya plandır.

• **SuperClass** : miras alınan sınıf (veya bir temel sınıf veya bir üst sınıf).

• **Alt** Sınıf: başka bir sınıftan (veya türetilmiş bir sınıftan, genişletilmiş sınıftan veya alt sınıftan) miras alan sınıf.

o Alt sınıf, üst sınıf alanlara ve yöntemlere ek olarak kendi alanlarını ve yöntemlerini ekleyebilir.

• **Yeniden kullanılabilirlik** : bir sınıf oluşturduğunuzda mevcut sınıfın alanlarını ve yöntemlerini yeniden kullanmanızı kolaylaştıran bir mekanizma.

yeni sınıf. Önceki sınıfta zaten tanımlanmış olan aynı alanları ve yöntemleri kullanabilirsiniz.

20. Çok Biçimlilik ve Kalıtım Arasındaki Fark

• Gerçek dünyada olduğu gibi, Miras, iki sınıf arasındaki ilişkiyi tanımlamak için kullanılır. Baba-Oğul'a benzer

ilişki. Java'da, Ebeveyn sınıfımız (süper sınıf olarak da bilinir) ve alt sınıfımız (alt sınıf olarak da bilinir) vardır. Benzer

gerçek dünyada Çocuk, Ebeveynlerin niteliklerini, yöntemlerini ve kodlarını miras alır.

o Bir alt sınıf, Parent sınıfında yazılan tüm kodları yeniden kullanabilir ve yalnızca

Ebeveyn.

o Kalıtım aslında kodun yeniden kullanılması 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 mirası,

iki sınıf arasındaki herhangi bir ilişki. Polimorfik kod yazmak mümkün değildir.

o Dinamik Polimorfizm → Geçersiz Kılma

o Statik Polimorfizm → Aşırı Yükleme

21. Metot Aşırı Yüklemesi ile Metodu Geçersiz Kılma arasındaki fark ?

• Aşırı yükleme ve geçersiz kılma arasındaki ilk ve en önemli fark şudur:

o aşırı yükleme durumunda, yöntem adı aynı olmalı, ancak parametreler farklı olmalıdır;

o geçersiz kılma durumunda, yöntem adı ve parametreleri aynı olmalıdır

• Metot aşırı yükleme ile geçersiz kılma arasındaki ikinci büyük fark şudur;

o Metodu aynı sınıfta aşırı yükleyebiliriz ancak metodu geçersiz kılma kalıtımı olan iki sınıfta gerçekleşir.

ilişki.

• Java'da static, final ve private metodu geçersiz kılamayız, ancak Java'da static, final ve private metodu aşırı yükleyebiliriz.

• Yöntem aşırı yüklemesinde dönüş tipi aynı veya farklı olabilir. Yöntemin geçersiz kılınmasında, dönüş türü aynı veya eşdeğişken olmalıdır

yazın.

22. Değişmez nedir?

• Değişmez, bir nesnenin yapıcısı yürütmeyi tamamladığında, örneğin değiştirilemeyeceği anlamına gelir.

• Bu, başka birinin gideceğinden endişe etmeden, etrafınızdaki nesneye referansları iletebileceğiniz anlamına geldiği için kullanışlıdır.

içeriğini değiştirin.

Özellikle eşzamanlılıkla uğraşırken, asla değişmeyen nesnelerle ilgili kilitleme sorunları yoktur.

```
class Foo {
    özel final String myvar ;
    public Foo (final String initialValue) {
    bu . myvar = initialValue ;
}
    Genel Dize getValue () {
    bunu geri ver . myvar ;
    }
```


Sayfa 49

23. Dinamik / çalışma zamanı bağlamasına karşı statik bağlama nedir?

• Statik bağlama, aşırı yükleme ve dinamik bağlama, yöntem aşırı yüklemesidir

24. Erişim değiştirici nedir ve farklı erişim değiştiriciler nelerdir?

• Java, sınıflar, değişkenler, yöntemler ve yapıcılar için erişim düzeylerini ayarlamak için bir dizi erişim değiştirici sağlar.

o Paket tarafından görülebilir, varsayılan. Değiştiriciye gerek yoktur.

o Yalnızca sınıfa görünür (özel).

o Dünyaya görünür (halka açık).

o Paket ve tüm alt sınıflar (korumalı) tarafından görülebilir.

25. Java'da Genel, Özel ve Korumalı değiştirici arasındaki fark nedir?

- Java'da sınıf, yöntem ve değişkenlerin erişilebilirliğini belirten erişim değiştiricisi. İçinde dört erişim değiştirici vardır

  Java, yani Genel, Özel, Korumalı ve Varsayılan.
- Bu erişim modifikasyonları arasındaki fark şudur;
  - o En önemlisi erişilebilirlik düzeyidir.
  - o Herkese her yerden erişilebilir
  - o Özel, yalnızca beyan edilen aynı sınıfta erişilebilir
  - o Varsayılana yalnızca aynı paket içinde erişilebilir
  - o Korumalıya aynı paket içinde ve aynı zamanda paketin dışında ancak sadece alt sınıflar erişilebilir.
- Üst düzey bir sınıfla özel veya korumalı değiştirici kullanamayız.
- Ayrıca, erişim değiştiricinin genel, özel veya Java'da korunan yerel değişkenler için uygulanamayacağını unutmamalıyız.

26. Java'da Set, List ve Map arasındaki fark nedir?

- Set, List ve Map, Java toplama çerçevesinin 3 önemli arabirimidir.
  - o Liste, *çoğaltma içerebilecek sıralı* ve indekslenmiş koleksiyon sağlar .
  - o Set, benzersiz nesnelerin *sırasız bir şekilde* toplanmasını sağlar . Set *, kopyalamaya izin vermiyor* . Liste ve Küme her ikisi de genişletilir

    koleksiyon arayüzü.
  - o Harita, Anahtar Değerine dayalı bir veri yapısı sağlar. Anahtar her zaman benzersizdir, değer çift olabilir.

27. Liste, Set ve Harita ne zaman kullanılır?

- İndeks kullanarak öğelere sık sık erişmemiz gerekirse, List, gitmenin bir yoludur ArrayList, index ile daha hızlı erişim sağlar.
- Elemanları saklamak istiyorsak ve bir sırayı korumalarını istiyorsak, Liste sıralı bir koleksiyondur ve sırayı korur.
- Herhangi bir Set uygulamasını seçmektense, yinelemesiz benzersiz öğeler koleksiyonu oluşturmak istiyorsak. (HashSet ...)
- Verileri Anahtar ve Değer biçiminde depolamak istiyorsak, gitmenin yolu budur. HashMap, Hashtable arasından seçim yapabiliriz ...

28. Dizi nedir?

- Dizi, tek bir türden sabit sayıda değeri tutan bir kap nesnesidir. Bir dizinin uzunluğu belirlenir

  dizi oluşturulduğunda. Oluşturulduktan sonra uzunluğu sabittir. Halihazırda ana dizinin bir örneğini gördünüz.

  "Merhaba Dünya!" uygulama. Bu bölümde diziler daha ayrıntılı olarak tartışılmaktadır.
- Bir dizideki her öğeye öğe adı verilir ve her öğeye sayısal dizini ile erişilir. Yukarıda gösterildiği gibi

  örnekleme, numaralandırma 0 ile başlar. Bu nedenle, örneğin 9. öğeye indeks 8'den erişilebilir.
- **Java Dizisinin Avantajı**
  - o Kod Optimizasyonu: Kodu optimize eder, verileri kolayca alabilir veya sıralayabiliriz.
  - o Rastgele erişim: Herhangi bir indeks konumunda bulunan herhangi bir veriyi alabiliriz.
- **Java Dizisinin Dezavantajı**
  - o Boyut Sınırı: Dizide yalnızca sabit boyuttaki öğeleri saklayabiliriz. Çalışma zamanında boyutunu büyütmez. Bunu çözmek için

    problem, java'da koleksiyon çerçevesi kullanılıyor.

Sayfa 50

29. ArrayList'in yineleme içerip içermediğini nasıl anlarsınız?

- Kullanılabilecek birkaç yol vardır. En kısa olanı .stream (). Differt (). Count () yöntemidir

  list.size ()! = list.stream (). farklı (). count ()
- Diğer yöntemler:

```
// YÖNTEM 1
public static <T> boolean containsUnique ( Liste <T> listesi) { Set <T> set = new HashSet <> ();
return list.stream (). allMatch (t -> set.add (t));
}

// YÖNTEM 2
public static <T> boolean containsUnique ( List <T> listesi) { return list.stream (). allMatch ( yeni
HashSet <> () :: ekle);
} // sadece saf akışları işleyebildiği için değil, aynı zamanda durduğu için de en iyisi gibi görünüyor
ilk kopya (# 1 ve # 2 her zaman sonuna kadar yinelenirken)

// YÖNTEM 3
public static <T> boolean containsUnique ( Liste <T> listesi) {
        Takım <T> grubu = Yeni HashSet <> ();
        for (T t: list) {
                eğer (! set.add (t))
yanlış dönüş ; }
```

30. Java'da Diziler ve ArrayList arasındaki fark nedir?

- Dizi, temel Java programlamasının bir parçasıdır ve özel sözdizimine sahiptir ArrayList, koleksiyon çerçevesinin bir parçasıdır ve uygular

  Liste arayüzü
- En büyük fark şudur; Dizi sabit uzunlukta bir veri yapısıdır, bu nedenle oluşturulan Array uzunluğunu değiştirebiliriz, ArrayList

yeniden boyutlandırılabilir.

- Diğeri ise 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. Bir uzunluk için kullanıyoruz Array, bir ArrayList için size () yöntemini kullanıyoruz.

31. İş parçacığı güvenli veya Senkronize nedir?

- İş parçacığı güvenliği çok önemlidir ve programımızı çok iş parçacıklı ortamda güvenli hale getirme sürecidir, program iş parçacığımızı güvenli hale getirmenin farklı yolları vardır.
- **Senkronizasyon** , iplik güvenliği için en kolay ve en yaygın kullanılan araçtır.

Sayfa 51

- JVM, senkronize kodun bir seferde yalnızca bir iş parçacığı tarafından yürütüleceğini garanti eder.
- **Senkronize edilmiş** JAVA anahtar sözcüğü, **senkronize** edilmiş kod oluşturmak için kullanılır ve dahili olarak nesne veya Sınıf üzerindeki kilitleri kullanır. sadece bir iş parçacığının eşitlenmiş kodu çalıştırdığından emin olun.
- Java senkronizasyonunun kaynağın kilitlenmesi ve kilidinin açılması üzerinde çalıştığını, bu nedenle senkronize edilen iş parçacığına girmediğini söylüyorum kodu.
- Senkronize edilmiş anahtar kelimeyi iki şekilde kullanabiliriz, biri tam bir yöntemi senkronize etmek, diğeri ise senkronize blok oluşturun.

32. Oluşturduğunuz bir nesneyi nasıl sınıflandırırsınız?

- Sıralayabilecektir.
- Ayrıca, nesnelerimi bir TreeSet veya TreeMap'e depolayabilirim → Örn: SONRAKİ SAYFA
- Java, bir listeyi sıralamak için birkaç yol sağlar.
  - o KARŞILAŞTIRILABİLİR - KARŞILAŞTIRICI arayüzleri sıralama için kullanılabilir. Bu durumlarda, CompareTo öğesini geçersiz kılmalıyız yöntem.
- Diğer bir yol, bir karşılaştırıcı kullanabilen Liste arabirimi sıralama yöntemidir. Bu yöntemle artan veya artan şekilde sıralayabiliriz Azalan.

```
users.sort (Karşılaştırıcı.comparing (Kullanıcı :: getUserID));
```

- Orijinal listeyi değiştirmek istemiyorsak, ancak yeni bir sıralanmış liste döndürmek istiyorsak; daha sonra sıralı () yöntemini kullanabiliriz. Akış arayüzü…

```
List <Kullanıcı> sıralanmışUsers = users.stream ()
  .sorted (Karşılaştırıcı.comparing (Kullanıcı :: getUserID))
  .collect (Collectors .toList ());
```

33. Java'da Hashtable ve HashMap arasındaki fark nedir?

Java'da HashMap ve Hashtable arasında birkaç fark vardır:

- Hashtable senkronize edilirken HashMap senkronize değildir. Bu, HashMap'i iş parçacıklı olmayan uygulamalar için daha iyi hale getirir. senkronize edilmemiş Nesneler tipik olarak senkronize edilmiş 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 sırasıdır), HashMap'i LinkedHashMap için kolayca değiştirebilirsiniz. Bu olamaz Hashtable kullanıyorsanız kolay.

Senkronizasyon benim için bir sorun değilse, HashMap kullanmayı tercih ederim. Bir sorun haline gelirse, tercih ederim Collections.synchronizedMap () veya ConcurrentHashMap.

- Hem Hashtable hem de HashMap, Harita arabirimini uygular ve her ikisi de Anahtar ve Değer'dir.
- HashMap iş parçacığı güvenli değildir, Hashtable iş parçacığı açısından güvenli bir koleksiyondur.
- HashMap senkronize olmadığı için ikinci önemli fark performanstır. Hashtable'dan daha iyi performans gösterdi. → Collections.synchronizedMap (... Harita ...);

34. İstisna ile nasıl başa çıkarsınız?

İstisnayı ele almak için dene-yakala yaklaşımını kullanırdım

• 1- Bir dene-yakala bloğunun içine bir istisna oluşturabilecek kodumu koyardım. Try-catch bloğu ile bir istisna veya kurtarma adımlarımı gerçekleştirmeyi deneyin. Ayrıca gerekirse multi veya Union Catch blokları kullanabilirim

• 2- throws anahtar sözcüğünü de kullanabilirim. ANCAK bu, benim yöntemimi arayan herkesin artık onu da halletmesi gerektiği anlamına geliyor!

• 3- Diğer bir yol da AutoCloseable'dır: Try bildiriminde AutoCloseable olan referansları yerleştirdiğimizde, kaynağı kendimiz kapatmamız gerekiyor. Yine de, istediğimiz başka türden bir temizlik yapmak için yine de bir nihayet bloğu kullanabiliriz. Deneyin-ile

Sayfa 52

35. TreeSet ve TreeMap

• TreeSet: Yalnızca benzersiz değerler içerebilir - artan düzende sıralanır

• TreeMap: yalnızca benzersiz anahtarlar içerebilir. - anahtarlar artan düzende sıralanır

36. final vs final vs nihayet?

• **final** → bir anahtar sözcüktür ve sınıf, yöntem ve değişken üzerinde kısıtlamalar uygulamak için kullanılır.

o final Sınıfı Devralınamaz

o son Yöntem Geçersiz Kılınamaz

o son Değişken değeri DEĞİŞTİRİLEMEZ.

• **nihayet** → bir bloktur ve önemli kodu yerleştirmek için kullanılır, istisna ele alınsın veya işlenmesin çalıştırılır

• **finalize** → bir yöntemdir ve Object is Garbage toplanmadan önce temizleme işlemini gerçekleştirmek için kullanılır.

37. Java'da Hata ve İstisna Arasındaki Fark?

• Hem Hata hem de İstisna, Java'da Throwable'dan türetilmiştir.

• Hata, genellikle ele alınamayan hataları temsil eder.

Örnekler için: OutOfMemoryError, NoClassDefFoundError

• Öte yandan, İstisna, yakalanabilen ve dağıtılabilen hataları temsil eder.

Örnekler için> IOException, NullPointerException

• İstisna, işaretlenmiş ve işaretlenmemiş İstisna olmak üzere iki kategoriye ayrılır. Kontrol Edilmiş İstisna zorunlu gerektirir bunu işlemek için dene-yakala kod bloğu. Denetlenmemiş İstisna çoğunlukla programlama hatalarını temsil eder (NullPointerException veya Çalışma zamanı istisnası)

• Hatalar kontrol edilmeyen istisnalardır ve geliştiricinin bunlarla herhangi bir şey yapması gerekmez

• **Tüm Hatalar İstisnadır, ancak tersi doğru değildir.**

• Genel olarak Hatalar, hiç kimsenin ne zaman olduğunu kontrol edemediği veya tahmin edemediği durumlardır, diğer yandan İstisna tahmin edilebilir ve ele alınabilir

38. Java'da RuntimeException ve CheckedException arasındaki fark nedir?

• İstisna, Çalışma Zamanı (işaretlenmemiş) İstisna ve CheckedException olmak üzere iki kategoriye ayrılır.

• RuntimeException ve CheckedException arasındaki temel fark, işlemek için try-catch sağlamanın zorunlu olmasıdır. CheckedException durumunda RuntimeException zorunlu değildir.

• NullPointerExceptio, ArrayIndexOutOfBound, ClassNotFoundException gibi en yaygın İstisnalardan bazıları, IOException.

Öncelikle, Java İstisnalarının kontrolsüz olarak da bilinen RuntimeException iki kategoriye ayrıldığını hatırlatmak istiyorum.

Sayfa 53

İstisna ve kontrol edilen (derleme zamanı) İstisna.

RuntimeException ile kontrol edilen İstisna arasındaki temel fark, try catch veya try catch sağlamak zorunludur.

kontrol edilen İstisnayı işlemek için blok ve bunu yapmamak derleme zamanı hatasıyla sonuçlanırken, RuntimeException olması durumunda bu zorunlu değildir.

NullPointerException, ArrayIndexOutOfBoundException gibi en yaygın İstisnalardan bazıları işaretli değildir ve bunlar

java.lang.RuntimeException'dan türetilmiştir.
Kontrol edilen İstisnalar için popüler bir örnek ClassNotFoundException ve IOException'dur ve yapmanız gereken budur.
Birçoğu IOException attığı için Java'da dosya işlemleri gerçekleştirirken bir try catch nihayet bloğu sağlayın.
Kişisel fikrimi sorarsam, Kontrol Edilmiş İstisnalar'ın, kazan plakası kodunu.
try-catch nihayet blok.

39. Java'da fırlatma ve fırlatma arasındaki fark nedir?

- throw ve throws, Java programlama dilinin Exception özelliğiyle ilgili iki anahtar sözcüktür.
- throw anahtar sözcüğü açıkça bir istisna atmak için kullanılır, diğer yandan, bir istisna bildirmek için throws anahtar sözcüğü kullanılır

  bu, dene-yakala bloğuna benzer şekilde çalıştığı anlamına gelir.
- Sözdizimini atmadan daha iyi görürsek, ardından bir Exception sınıfı atışı örneği gelir ve ardından istisna sınıfı adları gelir.
- yeni ArithmeticException ("Aritmetik İstisna") oluşturun; ArithmeticException oluşturur;
- throw anahtar sözcüğü yöntem gövdesi için kullanılırken, özel durumu bildirmek için yöntem imzasında atmalar kullanılır.

Her ikisi de Java'nın İstisna özelliği ile ilgili iki anahtar kelimedir. Atmak ve atmak arasındaki temel farkı hatırladığım kadarıyla
atar, kullanım ve işlevselliğindedir.

- throws, yöntem imzasında muhtemelen herhangi bir yöntem tarafından atılan İstisnayı bildirmek için kullanılır, örneğin

```
public void shutdown (), IOException {
        yeni IOException ("Kapatılamıyor");
 }
```

Ancak fırlatma, aslında Java kodunda İstisna atmak için kullanılır.

```
Yeni İstisna atın ("başlatılamıyor");
```

Diğer bir deyişle; throws anahtar sözcüğü herhangi bir yerde kullanılamaz istisna yöntemi imzası atma anahtar sözcüğü içinde kullanılabilir
yöntem veya statik başlatıcı bloğu, yeterli istisna işleme sağladı.
Oh, throw ile ilgili bir şey daha hatırlıyorum, throw anahtar sözcüğü break kullanmadan bir switch ifadesini kırmak için de kullanılabilir.
anahtar kelime

40. Nesne ve Sınıf Arasındaki Fark?

- Sınıf, istediğiniz kadar nesne oluşturabileceğiniz bir plan veya şablondur. Nesne, bir sınıfın üyesi veya örneğidir
- Sınıf, 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 bir varsayılan değer dahil olmak üzere nesne özelliklerini tanımlar. Bir sınıf
ayrıca nesne davranışını açıklar. Bir nesne, bir sınıfın üyesi veya "örneği" dir ve tümünün 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 pek çok farklılık vardır, ancak bunlardan bazılarının bilinmesi önemli olduğunu söyleyeyim;

- Java'da yeni anahtar kelime, newInstance () yöntemi, clone () yöntemi, fabrika gibi nesne oluşturmanın birçok yolu vardır.

  yöntem ve seriyi kaldırma. Java'da class anahtar sözcüğünü kullanarak sınıfı tanımlamanın tek bir yolu vardır.
- Nesne, ihtiyaca göre birçok kez oluşturulur. Sınıf bir kez ilan edilir.
- Nesne, bir sınıfın örneğidir. Sınıf, nesnelerin oluşturulduğu bir plan veya şablondur.
- Nesne fiziksel bir varlıktır. Sınıf, mantıksal bir varlıktır.

Sayfa 54

Örneğin :

Sınıf : İnsan	**Nesne** : Erkek, Kadın	**Sınıf** : Cep telefonu	**Nesne** : iPhone, Samsung, Moto
Sınıf : Meyve	**Nesne** : Elma, Muz, Mango, Guava	**Sınıf** : Yiyecek	**Nesne** : Pizza, Burger, Samosa

41. StringBuffer ve StringBuilder?

- Temel fark, StringBuilder'ın senkronize edilmediği sırada StringBuffer'ın senkronize edilmesidir. Yani, StringBuilder olabilir

  eşzamanlı olarak aranır. Ve bu, StringBuilder'ı daha verimli hale getirir.
- StringBuffer senkronize edildi, StringBuilder senkronize değil
- StringBuilder, StringBuffer'dan daha verimlidir
- Yapıcı;
  - o StringBuilder () → **16** başlangıç **kapasitesine** sahip boş bir dize oluşturdu .
  - o StringBuilder (str str) → belirtilen dizeyi bir StringBuilder oluşturdu.
  - o StringBuilder (int length) → uzunluk olarak belirtilen kapasiteye sahip boş bir dizge oluşturdu.
- Yöntem;
  - o StringBuilder str = new StringBuilder ("Merhaba");
  - o str.append ("Java"); → // Merhaba Java
  - o str.insert (1, "Java"); → // HJavaello
  - o str.replace (1,3, "Java"); → // HJavalo
  - o str.delete (1,3); → // Merhaba
  - o str.reverse (); → // olleH

```
string str = "Merhaba";
ters dize = “”;
```

```
for (int i = str.length () - 1; i> = 0; i -) {
    ters + = str.charAt (i);
}
sysout (tersine çevrilmiş);
```

42. Kesinleştirme () nedir?

- finalize () yöntemi, java.lang.Object sınıfının korumalı ve statik olmayan bir yöntemidir.
- Bu yöntem, java'da oluşturduğumuz tüm nesnelerde mevcuttur.
- Bu yöntem, nesneden kaldırılmadan önce bir nesne üzerinde bazı son işlemleri veya temizleme işlemlerini gerçekleştirmek için kullanılır.

  hafıza.
- Bir nesne yok edilmeden önce gerçekleştirmek istediğimiz işlemleri korumak için finalize () yöntemini de geçersiz kılabiliriz.

  Çağrılabilir. object.finalize ();

43. Son anahtar kelime nedir?

- final anahtar sözcüğü Class ile birlikte başka hiçbir sınıfın onu genişletemeyeceğinden emin olmak için kullanılır, örneğin String sınıfı sondur ve yapamayız

  uzat.
- Çocuk sınıflarının onu geçersiz kılamayacağından emin olmak için final anahtar kelimesini yöntemlerle birlikte kullanabiliriz.
- final anahtar sözcüğü, yalnızca bir kez atanabildiğinden emin olmak için değişkenlerle birlikte kullanılabilir. Ancak, durumu

  değişken değiştirilebilir, örneğin, bir nesneye yalnızca bir kez son bir değişken atayabiliriz, ancak nesne değişkenleri

  daha sonra değiştirin.
- Java arayüz değişkenleri varsayılan olarak nihai ve statiktir.

44. Statik anahtar kelime nedir?

- static anahtar sözcüğü sınıf düzeyi değişkenlerle birlikte kullanılabilir ve genel hale getirilebilir, yani tüm nesneler aynı değişkeni paylaşacaktır.
- static anahtar sözcüğü yöntemlerle de kullanılabilir. Statik bir yöntem yalnızca sınıfın statik değişkenlerine erişebilir ve yalnızca

  sınıfın statik yöntemleri.

Sayfa 55

45. system.gc () nedir?

- Belleği boşaltmak için Çöp toplayıcıyı çalıştırması için JVM'ye bir istek
- Her zaman işe yaramıyor

Java.lang.System.gc () yöntemi çöp toplayıcısını çalıştırır. Bunu aramak, Java Sanal Makinesi'nin
Şu anda kapladıkları hafızayı hızlı bir şekilde yeniden kullanıma hazır hale getirmek için kullanılmayan nesneleri geri dönüştürme çabası.
Bu bir komut değil, bir rica. Bu isteği yerine getirmek çöp toplayıcıya kalmıştır.

46. Önemli String Metotları?

47. IS-A ve HAS-A ilişkisi arasındaki fark nedir?

- **IS-A** , kalıtıma dayanır → Bu şey, o şeyin bir türüdür
- **HAS-A** ilişkileri kullanıma dayalıdır
  - o Örn: A Sınıfı HAS -AB, Sınıf A'daki kodun B sınıfının bir örneğine referansı varsa

```
public Horse {
    özel Halter myHalter;
    public void jump () {
                Sysout "atlıyorum"
```

• Horse sınıfından gelen atlama yöntemini kullanmak için bir Halter örnek değişkenini çağırıyorsunuz - bunun yaptığı şey şu ki

Atın HAS-A Halter olduğu anlamına gelir

• Horse sınıfının Halter'ı vardır çünkü Horse, Halter türünde bir örnek değişkeni bildirir. Kod, üzerinde tie () işlevini çağırdığında

Horse nesnesinin Halter örnek değişkeni -}

• Soyut sınıfın yapıcıları varken arayüzün yapıcıları yoktur

Sayfa 56

48. Yineleyici nedir ve her döngü için arasındaki fark nedir?

• Yineleyici, dizi ile değil ArrayList ile çalışır.

• Öğeleri yinelememize yardımcı olacaktır.

• Fark, yineleyici ile yineleme sırasında listede değişiklikler (öğeyi kaldır) yapabilmenizdir.

• her döngü içinde listemizde değişiklik yapamayız

49. Java Collection Framework

İki tür Koleksiyon (Bunları karıştırmamaya dikkat edin)

✪ **java.util.Collection** - Set ve List genişlemesinden arayüz (uygulama değil)

✪ **Set** *(Eşsiz şeyler)* - ÇİFTLERE İZİN VERMEZ. Seti Uygulayan Sınıflar;

♦ **HashSet** → Yinelemeleri istemediğinizde ve yinelediğinizde siparişi önemsemediğinizde kullanın

vasıtasıyla

o Sırasız ve Sıralanmamış

♦ **LinkedHashSet** → **HashSet'in sıralı** sürümü ve yineleme sırasını önemsediğinizde HashSet üzerinden kullanın

♦ **SortedSet**

♦ **TreeSet** → Öğeler, öğelerin doğal sırasına göre artan sırada olacaktır.

o Doğal düzenin kendi kurallarını uygulamak için kurucuyu da özelleştirebilir

✪ **Liste** *(şeylerin listesi)* - dizini önemsiyor. List'i uygulayan sınıflar;

♦ Bağlantılı **Liste** → Dizin konumuna göre **sıralanır** ve öğeler birbirine çift bağlıdır

o Yığın ve kuyruğu uygulamak için iyi bir seçimdir

o ArrayList'ten daha yavaş yineler, ancak hızlı ekleme ve silme

♦ **Vektör** → ArrayList ile aynı AMA vektör yöntemleri senkronize edilir (iş parçacığı güvenli)

♦ **Dizi Listesi** → Hızlı yineleme ve Hızlı rastgele erişim ve sıralı (dizine göre)

o Ayrıca sıralanmamış (ancak sıralamak için Collections.sort () işlevini çağırabilir)

✪ **java.util.Collections** - koleksiyonlarla kullanılmak üzere statik yardımcı program yöntemlerini tutan bir sınıf; Ekle, kaldır, içerir, içerir

boyut ve yineleyici vb.

• **Harita** *(benzersiz kimliği olan şeyler)* → Önemli: Harita ile ilgili sınıfların ve arayüzlerin hiçbiri Koleksiyon formunu genişletmez.

Haritanın uygulama sınıfları Koleksiyon değil, "koleksiyonlar" olarak düşünülür. Map'i uygulayan sınıflar;

♦ **Hashtable**

o HashMap ile aynı ANCAK HashTable yöntemleri senkronize edilir (UNUTMAYIN. YALNIZCA YÖNTEMLER

SENKRONİZE EDİLMİŞ, SINIFLAR VEYA DEĞİŞKENLER DEĞİL)

o Hashtable hiçbir şeyi BOŞ (BOŞ) almanıza izin vermez

♦ **LinkedHashMap**

o Kampanya siparişini (veya isteğe bağlı olarak erişim sırasını) korur

o Öğe eklemek / kaldırmak için HashMap'ten daha yavaş, ancak DAHA HIZLI İTERASYON

♦ **HashMap** → Sıralanmamış ve Sıralanmamış & Bir koleksiyonda bir boş ANAHTAR ve birden çok boş değere izin verir

o KeySet ()

o Map.keySet () - bir dizi Anahtar döndürür

o Map.keySet (). size - anahtar sayısını döndürür

♦ **SortedMap** → TreeMap

• Set, List ve Map uygulama sınıfları ASLA hem sıralanamaz hem de sıralanamaz, tüm diğer kombinasyonlar olabilir.

50. float'ı String'e nasıl dönüştürebilirim?

```
float f = Float.parseFloat ("25");
Dize s = Float.toString (25.0f);
```


Sayfa 57

51. 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 temp = a; // geçici değişken
a = b; b = sıcaklık;
```

52. Yazmayı biliyor musunuz? Oyuncu seçimi nedir?

• **Otomatik kutulama** → ilkel bir değer alıp sarmalayıcı sınıf nesnesine int i = 10 atadığınız bir süreçtir;

```
Tamsayı n = i;
Tamsayı num = 200;
Tamsayı num2 = new Integer (400); // BOXING YOK
```

• **Un-boxing** → Wrapper sınıf nesnesini alıp ilkele dönüştürdüğünüz bir süreçtir.

```
Tamsayı num2 = yeni Tamsayı (400);
Tamsayı num = 200;
int i = num2;
```

• Bir türden bir değerin başka türden bir değişkene atanması, Type Casting olarak bilinir.

53. 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);
    }
}
```

Çıkış: 1 0 2 3 2 1 2 0

54. Projenizde soyut bir sınıfı nasıl kullanıyorsunuz bana bir örnek verin?

• Bu kavramlar genellikle çerçeve geliştirmede kullanılır. Soyut sınıf, ortak bir süper sınıfı tanımlamada kullanılır

çerçevenin Sayfa Nesne Modeli katmanını yazarken. Genellikle hepsine sahip olmak için BasePage adında soyut bir sınıf oluştururuz.
Bu sınıf örneğinde yazılan her sayfa için ortak üyeler **getPageTitle () .**

• Daha sonra her Sayfa sınıfı (HomePage, LoginPage, DashboardPage vb.) BasePage'den devralır. Bazen ihtiyaç duyulabilir

üst sınıfta uygulanan yöntemlerin davranışını değiştirir. Dolayısıyla, alt sınıf, bu yöntemi geçersiz kılma özgürlüğüne sahiptir.
polimorfizm kullanın. Abstract sınıfını gerçek projelerde böyle kullanıyoruz.

55. Değere göre geçirme ile referansla geçirme arasındaki fark nedir? değere göre geçmek ve referans ile geçmek?

• Değere göre geçiş, işlev parametresinin değerinin belleğinizin başka bir konumuna kopyalandığı anlamına gelir ve

İşlevinizdeki değişkene erişirken veya değiştirirken, yalnızca kopyaya erişilir / değiştirilir ve orijinal değer
dokunulmadan bırakılır. Değere göre geçiş, değerlerin çoğu zaman nasıl aktarıldığıdır.

• Başvuru yoluyla geçiş, değişkenin bellek adresinin (bellek konumuna bir işaretçi),

işlevi. Bu, bir değişkenin değerinin aktarıldığı değere göre geçişten farklıdır. Örneklerde hafıza adresi
/ myAge 106'dır. myAge işlevini artırırken, işlev içinde kullanılan değişken olan artışAgeByRef (yaş
bu örnek) hala orijinal myAge değişkeni ile aynı bellek adresine işaret etmektedir (İpucu:
fonksiyon parametresi, bir değişkenin referansını / işaretçisini elde etmek için birçok programlama dilinde kullanılır).

Sayfa 58

SELENYUM

1. Selenyum nedir ve nelerden oluşur?

• Selenium, otomatik web testi için bir araç paketidir. Tarafından bestelendi;

o Selenium IDE (Entegre Geliştirme Ortamı); kayıt ve oynatma için çalışan bir Firefox eklentisi.
o Selenium RC (Uzaktan Kumanda) (1.0); bir test aracıdır ve web uygulamasını otomatikleştirmek için JS üzerinde çalışmak için kullanılır. (2004)
o WebDriver (2.0); bir web otomasyon çerçevesidir ve testlerinizi farklı tarayıcılarda yürütmenize olanak tanır. (2011)
o Selenyum Izgara; testlerin birden çok makinede paralel olarak yürütülmesine izin verir.

2. Selenium'un avantajları nelerdir?

• Selenium açık kaynak kodludur ve herhangi bir lisans maliyeti olmaksızın kullanımı ücretsizdir
• Java, Ruby, Python, C # gibi birden çok dili destekler ...
• Çoklu tarayıcı testini destekler
• İyi miktarda kaynağı vardır ve topluma yardım eder
• Windows, Mac, Linux gibi birçok işletim sistemini destekler ...
• Web uygulamasıyla etkileşim kurun

3. Selenium'un 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 konum belirleyicileri yazmak zor
- Senkronize etmek zor

4. Selenium'un sınırlamaları nelerdir?

- Masaüstü uygulamasını test edemiyoruz
- Web hizmetlerini test edemiyoruz
- Test çerçevesi (TestNG, JUnit),

  harici dosyalar (excel için Apache POI)
- Captcha'yı Selenium kullanarak otomatikleştirmek mümkün değildir
- Dosya yükleme özelliğini desteklemez.

5. Selenium ile hangi tür testleri otomatikleştiriyorsunuz?

- fonksiyonel testler (pozitif / negatif, UI)
- duman testleri
- regresyon testleri
- entegrasyon testleri
- uçtan uca test
- veri tabanlı

6. Selenyum ile ne yapmıyoruz?

- Performans, yük, stres testi, manuel geçici test (Bu testler, bu araçlarda eğitim almış uzmanlar tarafından yapılır)
- Saf veritabanı testi (yalnızca veritabanının kendisini test edersek),
- Birim testleri ..., görünüm ve his temelli testler (renk, şekiller, vb.),
- statik test

Sayfa 59

7. Selenium araç setinde neler var?

- Selenium IDE → bir Chrome ve Firefox uzantısı olarak uygulanır ve testleri kaydetmenize, düzenlemenize ve hata ayıklamanıza olanak tanır.
- Selenium RC → herhangi bir programlama dilinde otomatik web uygulaması UI testleri yazmak için
- Selenium WebDriver → testlerinizi farklı tarayıcılarda gerçekleştirin
- Selenium GRID → testlerinizi farklı makinelerde paralel olarak farklı tarayıcılarda çalıştırın.

8. Şu anda Selenium'un hangi sürümünü kullanıyorsunuz?

- JDK (JAVA) - 1.8 → → Lambda exp nedeniyle beğendim. ve hata işlemeyi deneyin, birden çok yakalama ekleyebilirsiniz.
- IntelliJ - 2018.03.04
- Salatalık - 4.2.6
- Selenyum - 3.141.59
- Maven - 3.6.0
- TestNG - 6.14.3
- GIT - 2.17.2

9. Örtülü Bekle mi Açık Bekle mi?

- **Örtülü bekleme** , fırlatmadan önce bir öğeyi bulurken belirli bir süre bekleyen bir beklemedir

  "NoSuchElementException". Varsayılan olarak selenyum, öğeleri beklemeden hemen bulmaya çalışır. Bu yüzden iyi

  örtük bekleme kullanmak için. Bu bekleme, geçerli sürücü örneğinin tüm öğelerine uygulandı.
- **Açık bekleme** , belirtilen Beklenen Koşul karşılanıncaya kadar belirli bir web öğesine uygulanan beklemedir.
- Örtük bekleme basittir; koşul, zaman aşımından önce karşılanırsa, koşul içinde karşılanmazsa bir sonraki adıma geçer.

  zaman aşımı "Böyle Bir Öğe Yok" istisnasını atar.
- Açık bekleme bazen öğenin görünür, tıklanabilir, etkin olması gibi belirli bir olayı / koşulu beklememiz gerekir ...

```
driver.manage (). timeouts (). örtük olarak Bekleme (5, TimeUnit.SECONDS);
webDriverWait wait = new WebDriverWait (sürücü, 5);
wait.until (ExpectedConditions.visibilityOf (element);
```

10. fluentWait nedir?

- Diyelim ki bazen sadece 1 saniyede görünen ve bir süre sonra ortaya çıkması dakikalar alan bir öğeniz var. Şöyle

  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

```
<WebDriver> bekleyin = yeni
FluentWait <Webdriver> (sürücü) .withTimout (5, timeUnit.seconds) .pollingEvery (100, timeunit.
milisaniye) .ignoring (NoSuchElementException.class);
```

11. Selenium'da bir elementi bulmanın çeşitli yolları nelerdir?

- Selenium Konumlandırıcılar → Kimlik ve ad
- Selenyum bulucuda, html'de bir öğe bulmanın bir yolu:
- Id, name, className, xpath, css, linkText, partialLinkText, tagName

12. Elementi neden bulamıyorum?

- Konum belirleyici değiştirildi
- Bir iframe var
- Bekleme süresi :: sayfa yavaş yükleniyor veya Öğe dinamik :: locator
- Sayfa tam olarak yüklenmemiş / açılmamış
- Sayfa değişir ve bu öğe artık mevcut değildir

Sayfa 60

13. Bir öğe nasıl vurgulanır?

- Selenium WebDriver'ın vurgulama eylemi yoktur.
- Ancak bunu yapmak için JavaScript kullanabiliriz

```
JavascriptExecutor js = ((JavascriptExecutor) sürücüsü);
Dize bgcolor = element.getCssValue ("backgroundColor");
için (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 gezinebilirdik.
- buton. checkbox, Image ext ... in web Page

15. Mutlak (/) ve Göreli (//) Xpath?

- Sözdizimi → // etiketadı [@ özellik = "değer`]
- Mutlak xpath, kök öğeden başlayarak öğeye kadar tek eğik çizgiyle (/) başlar.
- Göreli xpath, belgenin herhangi bir yerinde seçimle eşleşen çift eğik çizgi (//) ile başlar.

16. Dinamik unsurları nasıl idare ediyorsunuz?

- id'nin statik kısmını bulun ve bir konum belirleyici yazın (xpath veya css) → Ve ardından Startswith, contains, EndsWith kullanın
- içerir () → // * [içerir (@ name = `btn`)]
- ile başla( ) → // etiket [startwith (@id, `message`)]
- Metin( ) → // td [text () = `usedId`]
- veya & ve → // girdi [@type = `submit` AND @name =` login`]

17. Dinamik web sayfası nasıl test edilir?

- Bu soruna tüm çözümlere uyan tek bir çözüm yoktur. Uygulamayı çok iyi anlamalıyız
    - o Gerektiğinde açık beklemeler kullanın.
    - o Özel xpath'leri ve css bulucularını kullanın
        - ▪ Xpath: içerir, şununla başlar, ile biter, metin içerir.
        - ▪ Ebeveyn, çocuk, kardeş ilişkilerini kullanarak başka bir kararlı öğeye ilişkin öğeyi bularak

18. Dinamik tablo nasıl test edilir?

- Özel xpath'ler ve css bulucuları kullanın
    - o Xpath: içerir, şununla başlar, ile biter, metin içerir.
    - o Ebeveyn, çocuk, kardeş ilişkilerini kullanarak öğeyi başka bir kararlı öğeyle ilişkili olarak bularak
- Tablo ile çalışan yardımcı yöntemlerim var. Bir tablo web elemanını alan ve tüm sütunu döndüren bir yöntemim var

  isimler. Bir tabloyu, numarayı alan ve o satırdaki tüm verileri döndüren bir yöntemim var.

19. xpath kullanarak üst öğeye nasıl geçebiliriz?

- xpath'de (..) ifadesini kullanarak, üst öğeye geçebiliriz

20. close () ve quit () komutu arasındaki fark nedir?

- driver.close () → mevcut tarayıcıyı kapatmak için kullanılır
- driver.quit () → tüm tarayıcı örneklerini kapatmak için kullanılır

Sayfa 61

21. xpath kullanarak n'inci alt elemana nasıl geçebiliriz?

- **İki yol vardır:**
  - o dizin konumunda köşeli parantez kullanarak
    Örneğin: div [2], ikinci div öğesini bulacaktır
  - o position () yöntemini kullanarak
    Örneğin: div [position () = 2], ikinci div öğesini bulacaktır

22. xpath ve css seçici arasındaki fark nedir?

- xpath ile öğeleri geriye veya ileriye doğru arayabiliriz ...
  css yalnızca ileri yönde çalışır
- Xpath metinle çalışabilir, css çalışamaz
- Xpath'in daha fazla kombinasyonu vardır ve dizine göre arama yapabilir
  css dizine göre arama yapamaz, ancak css xpath'den daha hızlı çalışır.

23. Çerçeve nedir?

- Test otomasyonunda çerçeve, test otomasyonunun taslağıdır.
- İşlev kitaplığınızı, test sonuçlarını, test verilerini, kaynakları nereye kaydedeceğiniz klasör yapılarınızı içerir.
- Bu çok önemlidir çünkü bir otomasyon projesi üzerinde çalışırken herkesin takip etmesi gereken bir kılavuz olacaktır ve bizim
  komut dosyasının bakımı daha kolay olacaktır.

24. Görüşme sırasında HTML haberciliğinden mi bahsediyorsunuz?

- Çerçevemde birden fazla raporlama yöntemi kullanıyorum, sürücü betiği test durumları excel sayfasına geçti / kaldı,
- Reporter yardımcı programı nesnesi UFT raporuna yazıyor, ayrıca özel bir HTML raporlama motoru geliştirdim.
- Not Defterine 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, maximize () çalışmaz> öyleyse yol nedir?

- İşlemler yapın veya sürümü değiştirin.

```
ChromeOptions seçenekleri = yeni ChromeOptions ();
options.addArguments ("startmaximized");
```

27. Selenium'daki anahtar sınıf nedir?

- Bize klavyeden tuşlara basma seçeneği sunar
- Anahtar.ENTER
- SendKeys () yöntemine GEÇİLMELİDİR
- Örn; .sendkeys ("şarj cihazı" + keys.ENTER)

28. Ya rastgele açılan dinamik bir açılır pencere varsa

- Uyarı ile dene / yakala özelliğini kullanın

29. Thread.sleep () nedir?

- Yakalamak için selenyum yavaşlatır
- İstisna fırlatır, bu yüzden onu ele almalı veya fırlatmalıdır

Sayfa 62

30. Selenium Çerçeve nedir?

- Kod bakımını kolaylaştırmaya, kod okunabilirliğine ve kodun yeniden kullanılmasına yardımcı olan bir kod yapısıdır.
- Selenium WebDriver tarafından test durumlarını otomatikleştirmek için oluşturulmuş başlıca 3 tür çerçeve vardır:

Veriye Dayalı Çerçeve

- Piyasadaki en popüler otomasyon çerçevelerinden biridir
- Tüm test verilerimiz bazı harici dosyalardan oluşturulur;
  - o excel
  - o veya özellik dosyasında senaryo taslağı
  - o veya TestNG Veri Sağlayıcısı
- Selenium WebDriver, web tabanlı uygulamaları otomatikleştirmek için harika bir araçtır. Ancak okuma ve yazmayı desteklemiyor
  excel dosyalarındaki işlemler. Bu nedenle, **Apache POI** gibi üçüncü taraf API'leri kullanıyoruz

Anahtar Kelime Odaklı Çerçeve

• Anahtar kelimeye dayalı test, uygulamayla ilgili anahtar kelimeleri içermek için veri dosyalarını kullanan bir komut dosyası oluşturma tekniğidir

test edilmek.

• Anahtar sözcükler excel dosyası gibi bazı harici dosyalara yazılır ve Java kodu bu dosyayı çağırır ve test durumlarını yürütür.

HybridDriven Çerçeve

• DDF ve KDF'nin bir kombinasyonunun genellikle HDF olduğu söylenir.

• Hem test verileri hem de test eylemi harici dosyalarda saklanır.

31. Selenium'da aşırı yüklenmiş Yöntemleri nasıl kullandınız?

• İki değerin eşit olup olmadığını iddia ederken, → TestNG'den Assert.assertEquals (gerçek, 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 gizli olup olmadığını kontrol edin

33. js uyarılarını nasıl ele alıyorsunuz?

• Tarayıcıdaki uyarı JavaScript'ten geliyorsa, Alert sınıfını kullanırız.

```
Uyarı uyarısı = driver.switchTo.alert ();
alert.accept ();
alert.dismiss ();
alert.sendKeys ();
alert.getText ()
```

34. Birden çok çerçeve nasıl kullanılır?

• 4 kare varsa, belirli bir kareye ulaşmak için arka arkaya geçmeniz gerekir. 3. kareye atlayamıyorum

1. kareden.

35. driver.get () ve driver.navigateto () arasındaki fark nedir?

• driver.get () → Bir URL'yi açmak için ve tüm sayfa yüklenene kadar bekleyecektir

• sürücü. () 'E gidin → Bir URL'ye gitmek için ve tüm sayfanın yüklenmesini beklemeyecektir

Sayfa 63

36. Selenium'da çerçeveler nasıl kullanılır?

• Bir html sayfasını başka bir sayfaya yerleştirmek için kullanılan çerçeveler

• Adımlar

o iframe'i bulun

o driver.switchTo (). frame () ile başka bir iframe'e geçin;

.frame () → string, Integer, webElement, name veya id'yi doğrudan parametre olarak alır

```
driver.switchTo (). çerçeve (webElement);
driver.switchTo (). frame ();
```

• 2. karenin dışında (şu anda bulunduğunuz) bir öğe bulmak istiyorsanız, şimdi 2. karedesiniz

NosuchElementException oluşturur

• Önceki kareye geri dönmeniz gerekirse

o driver.switchTo (). parentFrame () → Bir seviye yukarı gider

o driver.switchTo (). defaultcontent () → En üste gider

• Count kullanarak geçiş yapabilir

o driver.switchTo (0) → Varsayılan çerçeve olmayan her şeyi sayar

Bu yöntemler, kullandığınız tarayıcıya bağlı olarak size farklı sonuçlar verebilir.

37. Tarayıcı açılır pencerelerini nasıl idare ediyorsunuz?

• **void dismiss ()** → açılır pencere görünür görünmez "İptal" düğmesine tıklar.

• **geçersiz kabul et ()** → açılır pencere görünür görünmez "Tamam" düğmesine tıklar.

• **String getText ()** → uyarı kutusunda görüntülenen metni döndürür.

• **void sendKeys (String stringToSend)** → belirtilen dize modelini uyarı kutusuna girer.

38. Windows / OS açılır pencerelerini nasıl kullanıyorsunuz?

• Selenium, Windows tabanlı uygulamaları desteklemez, yalnızca web uygulaması testini destekleyen bir otomasyon test aracıdır.

• AutoIT, Robot sınıfı gibi bazı üçüncü taraf araçları kullanarak Selenium'da Windows tabanlı açılır pencereleri işleyebilirdik

• **driver.getWindowHandle ();** Bu, onu bu sürücü örneğinde benzersiz bir şekilde tanımlayan mevcut 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

o WebDriver = yeni htmlunitdriver ()

o Çok kararlı değil

• Phantonjsbrowser

o Daha kararlı

o tarayıcı = yeni phantomjsbrowser ()

40. findElement vs findElements?

• FindElement> bu yöntem ilk WebElement'i döndürür!

o öğe bulunamazsa İstisna verir

• FindElements> List <WebElement> döndürür;

Sonuç listesinde bulunmayan eleman boş değerlere sahipse o istisna vermez

Sayfa 64

41. Birden çok pencere / sekme nasıl kullanılır?

• Selenyum tek pencerede kalır

• Bir pencere açarsanız ve ardından 5 sekme açılırsa, selenyum ilk pencereye odaklanır

• Yeni bir penceredeyseniz ve selenyum'a varsayılan pencerede bir öğe yazdırmasını söylerseniz, yine de çalışacaktır.

kullanıcının odağı yeni penceredir

• Yeni pencereye geçmeli

o windowHandle () kullan

o Driver.getWindowHandle ()

▪ Selenium her tarayıcı açtığında, Kollar adlı sayfa için bir dizin kimliği verecektir.

▪ Geçerli sayfanın tutamacını / kimliğini döndürür (bir dize olarak)

o driver.switchTo (). window (dize tutamacı)

o Birden çok pencere için driver.getWindowHandles ()

▪ Bir dizi pencere tutacağı döndürür

o Başlıkları kullanarak geçiş yapın

```
for (string handle: driver.getWindowHandles ()) {
    driver.switchTo (). Pencere (tanıtıcı)
    eğer (driver.getTitle (). equals (targetTitle) {
    kırmak;
    }
}
```

42. Sayfadaki tüm bağlantılar nasıl bulunur?

```
List <WebElement> list = driver.findElements (By.tagName ("a"));
```

43. isDisplayed (), isEnabled () arasındaki fark. Selenium WebDriver'da Selected () yöntemi mi?

• isDispalyed () → web sayfasındaki bir web öğesinin varlığını doğrulayın. Bulunursa → doğru, Bulunmazsa → yanlış

• isDisplayed () → mevcut her türlü web öğesinin varlığını kontrol edin

• isEnabled () → web öğesinin web sayfasında etkin mi yoksa devre dışı mı olduğunu doğrulayın.

• isEnabled () → öncelikli olarak düğmelerle kullanılır

• isSelected () → web öğesinin seçilip seçilmediğini doğrular

• isSelected () → radyo düğmeleri, açılır menüler ve onay kutuları ile kullanılır.

44. Nasıl Sürükleyip Bırakılır?

```
Eylemler eylem = yeni Eylemler (sürücü);
action.clickAndHold (driver.findElement (By.id ("öğe")))
.moveToElement (driver.findElement (By.id ("hedef")))
.release (). build ()
.perform ();
```

45. Aşağı kaydırmak için:

```
WebDriver sürücüsü = yeni ChromeDriver (); JavascriptExecutor jse = (JavascriptExecutor) sürücüsü;
jse.executeScript ("window.scrollBy (0,250)", "");
```

• VEYA, aşağıdaki gibi yapabiliriz:

```
jse.executeScript ("kaydırma (0, 250);");
```

46. Yukarı Kaydırma için:

```
jse.executeScript ("window.scrollBy (0, -250)", ""); VEYA, jse.executeScript ("kaydırma (0, -250);");
```


Sayfa 65

47. Tanımlama bilgileri nasıl kullanılır?

- Selenium Webdriver'da, aşağıdaki yerleşik yöntemle çerezleri sorgulayabilir ve bunlarla etkileşim kurabiliriz:

```
driver.manage (). getCookies (); // Tüm Çerezlerin Listesini Döndür
driver.manage (). getCookieNamed (arg0); // İsme göre belirli bir çerez döndür
driver.manage (). addCookie (arg0); // Çerezi oluşturun ve ekleyin
driver.manage (). deleteCookie (arg0); // Belirli çerezleri silin
driver.manage (). deleteCookieNamed (arg0); // Belirli bir çerezi Ada göre silin
driver.manage (). deleteAllCookies (); // Tüm çerezleri sil
```

48. Tanımlama bilgilerini neden işlememiz gerekiyor?

- Her tanımlama bilgisi bir ad, değer, etki alanı, yol, son kullanma tarihi ve güvenli olup olmadığına ilişkin durumla ilişkilendirilir. Sırayla

  Bir istemciyi doğrulamak için, bir sunucu tüm bu değerleri bir tanımlama bilgisinde ayrıştırır.
- Selenium web sürücüsünü kullanarak bir web uygulamasını test ederken, bir tanımlama bilgisi oluşturmamız, güncellememiz veya silmemiz gerekebilir.
- Örneğin, Çevrimiçi Alışveriş Uygulamasını otomatikleştirirken, birçoğumuz yer siparişi gibi test senaryolarını otomatikleştirmemiz gerekir,

  Sepeti Görüntüle, Ödeme Bilgileri, sipariş onayı vb.
- Çerezler depolanmazsa, yukarıda listelenen test senaryolarını yürütmeden önce her seferinde oturum açma eylemi gerçekleştirmemiz gerekecektir.

  Bu, kodlama çabanızı ve yürütme sürenizi artıracaktır.
- Çözüm, tanımlama bilgilerini bir Dosyada saklamaktır. Daha sonra, çerez değerlerini bu dosyadan alın ve ona mevcut tarayıcınızı ekleyin.

  oturum, toplantı, celse. Sonuç olarak, her Test Durumunda oturum açma adımlarını atlayabilirsiniz çünkü sürücü oturumunuzda bu bilgiler bulunur.
- Uygulama sunucusu artık tarayıcı oturumunuzu kimliği doğrulanmış olarak kabul eder ve sizi doğrudan istediğiniz URL'ye götürür.

49. Çerez çerez bilgilerini saklayın

```
public class cookieRead, BasePage'i genişletir {
    public static void main (String [] args) {
        driver.get ("http://demo.guru99.com/test/cookie/selenium_aut.php");

        // E-posta kimliği ve Parola Girin Zaten Kaydolduysanız
        driver.findElement (By.name ("kullanıcı adı")). sendKeys ("abc123");
        driver.findElement (By.name ("şifre")). sendKeys ("123xyz");
        driver.findElement (By.name ("submit")) tıklayın ();

        Dosya dosyası = yeni Dosya ("Cookies.data");                    // çerezleri saklamak için dosya oluştur

        Deneyin {
            file.delete (); // Varsa eski dosyayı silin
            file.createNewFile ();
            FileWriter fileWrite = new FileWriter (dosya);
            BufferedWriter Bwrite = new BufferedWriter (fileWrite);

            // çerez bilgilerini almak için döngü
            için (Çerez ck: driver.manage (). getCookies ()) {
                Bwrite.write ((ck.getName () + ";" + ck.getValue () + ";" + ck.getDomain () + ";" + ck.getPath ()
                                                    + ";" + ck.getExpiry () + ";" + ck.isSecure ()));

                Bwrite.newLine ();
            }

            Bwrite.flush (); Bwrite.close (); fileWrite.close ();
        } catch (Exception ex) {
            ex.printStackTrace ();
        }
    }
}
```


Sayfa 66

50. Oturum açma bilgileri için saklanan çerezleri kullanın

```
public static void main (String [] args) {
    Deneyin{
        Dosya dosyası = yeni Dosya ("Cookies.data");
        FileReader fileReader = new FileReader (dosya);
        BufferedReader Buffreader = new BufferedReader (fileReader);

        String strline;
        while ((strline = Buffreader.readLine ())! = null) {
            StringTokenizer belirteci = new StringTokenizer (strline, ";");
            while (token.hasMoreTokens ()) {
```

```
                        Dize adı = token.nextToken ();
                        Dize değeri = token.nextToken ();
                        Dize etki alanı = token.nextToken ();
                        Dize yolu = token.nextToken ();
                        Son kullanma tarihi = boş;

                        Dize değeri;
                        eğer (! (val = token.nextToken ()). equals ("null")) {
                            süre sonu = yeni Tarih (val);
                        }
                        Boolean isSecure = new Boolean (token.nextToken ()). BooleanValue ();
                        Çerez ck = yeni Çerez (ad, değer, etki alanı, yol, süre sonu, isSecure);
                        System.out.println (ck);
                        driver.manage (). addCookie (ck); // Bu, saklanan çerezi mevcut oturuma ekleyecektir
                    }
                }
            } catch (Exception ex) {
                ex.printStackTrace ();
            }
            driver.get ("http://demo.guru99.com/test/cookie/selenium_aut.php");

    }
```

ÇIKTI: Giriş kullanıcı kimliği ve şifresini girmeden doğrudan oturum açma başarılı ekranına yönlendirilirsiniz.

NOT: Yukarıdaki komut dosyasını çalıştırdıktan sonra oturum açma sayfasını görürseniz, tam yenilemeyi kullanın.

51. JavaScriptExecutor kullanıyor musunuz?

- Bu, kendi JavaScript'imi yazmama yardımcı oluyor. JS'nin selenyumdan çok daha fazla kontrolü var.
- bu sınıfı kullanarak tarayıcıya JS komutları gönderebiliriz

  JavaScriptExecutor jsExecutor = (JavaScriptExecutor) sürücüsü;

  o executeScript (); komutu yerine getirir

  o Parametrenin içinde JS kodunu koyduğunuz yerdir
- jsExecutor.executeScript ("alert ('UYARI: Bu gereksiz bir mesajdır');" → Bu kod bir JS açılır penceresi açar
- Ayrıca 2 parametre de koyabilirsiniz: .executeScript (“js kodu”, öğe);

  o Kaydırma için kullanılır (selenyum kaydırmada iyi değildir, terimler üzerinde çalışırken bir meydan okuma olduğunu söyleyebilirsiniz. ve devam düğmesine tıklamadan önce sayfayı okumanız gereken durum sayfası.

  o Selenyum ve eylemler sınıfını kullanmayı denediğimde işe yaramadı, bu yüzden javaexecutor kullandım) ve bir öğeye tıkladım;

Sayfa 67

52. Öğenin mevcut / görünür / etkin / olup olmadığı ve metnin mevcut olup olmadığı nasıl kontrol edilir?

- Eleman Var olup olmadığını kontrol etmek için:

```
eğer (sürücü.findElements (By.xpath ("değer")). size ()! = 0) {
    System.out.println ("Öğe Mevcut");
}Başka{
    System.out.println ("Öğe Yok");}
```

- veya

```
eğer (driver.findElement (By.xpath ("değer"))! = null) {
    System.out.println ("Öğe Mevcut");
}Başka{
    System.out.println ("Öğe Yok"); }
```

- Görünür'ü işaretlemek için:

```
eğer (driver.findElement (By.cssSelector ("a> yazı tipi")). isDisplayed ()) {
    System.out.println ("Öğe Görünür");
}Başka{
    System.out.println ("Öğe Görünmez"); }
```

- Etkinleştir'i işaretlemek için:

```
if (driver.findElement (By.cssSelector ("a> yazı tipi")). isEnabled ()) {
    System.out.println ("Öğe Etkin");
}Başka{
    System.out.println ("Öğe Devre Dışı Bırakıldı"); }
```

- Mevcut metni kontrol etmek için

```
eğer (driver.getPageSource (). içerir ("Kontrol edilecek metin")) {
    System.out.println ("Metin mevcut");
}Başka{
    System.out.println ("Metin yok"); }
```

53. Açılır menüde birden çok seçili değeri nasıl kontrol edebilirim?

- ArabalarListesi seçin = yeni Seç (el)
- carList.getSelectedOptions (): // seçilen seçenekleri bir liste olarak döndürür (List <webelement>)
- her biri için: carList.getSelectedOptions ()

54. Eylemler sınıfı nasıl kullanılır?

- Eylemler sınıfı, gelişmiş fare ve klavye işlemleri yapmamızı sağlar:
- Fareyi kontrol edin
- Gelişmiş kullanıcı etkileşimleri için yöntemler sağlayan sınıf
  - o Gezinme
  - o Çift tıklama
  - o Sağ tıklama
  - o Kaydırma
  - o Sürükle ve bırak
  - o mix / match operatörleri
- Eylemler eylemi = yeni Eylemler (sürücü)
- Eylem yöntemleri
  - o tıklayın ()
  - o tut ()
  - o moveToElement (öğe)
  - o gerçekleştirmek ()
  - o keydown () build ()
  - o dragAndDrop (kaynak, hedef) .perform ()
  - o sendKeys () genellikle kullandığımızdan farklı

Sayfa 68

- ▪ Sendkeys işlemini farklı elemanlar üzerinde yapalım
- ▪ Webelement'tan gelen normal gönderme anahtarları, metin girişi olmayan bir şeye bir istisna atar.
- ▪ Uzun yol;

```
actions.moveToElement (kaynak) .clickAndHold (). moveToElement (hedef) .release (). perform ();
```

- perform () kullanılmadığı sürece eylemler çalışmayacaktır
- Zincirleme yöntemleri kullanıyorsanız, perform () işleminden önce build () kullanmanız gerekir.

55. 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 eylemleri = yeni Eylemler (sürücü). Eylemleri gerçekleştir.doubleClick (el) .perform ()
actions.moveTo (el) .perform actions.doubleClick.perform
actions.moveTo (el) .doubleClick (). build.perform ()
```

56. Dosya indirme ve yükleme

- **İndir**
  - o Selenium'un kendisi dosya indirmelerini doğrulayamaz, indirme bağlantısına tıklayabilir ancak tarayıcının dışına çıkamaz ve indirilmiş dosya
  - o Robot ve AutoIT için diğer araçların kullanılması gerekir
- **Yükle**
  - o Yüklemeyi Selenium yönetir, ancak gerçek kullanıcıya göre farklı bir şekilde yapar
  - o Adımlar
    - Yükleme penceresini tetikleyen öğeyi bulun
    - Yüklemek istediğiniz dosyanın yolunu bulun
      - o Bir Dizede Saklayın
        - Örn: Dize → dosya = ”C: \\ Kullanıcılar \\ Andy \\ Masaüstü \\ klasör1 \\ dosya.key”;
        - Sonra driver.findElement (yükleme düğmesi) .sendkeys (dosya);

57. Açılır menüden seçilen değer nasıl kontrol edilir?

```
ArabalarListesi seç = yeni Seç (el)
carList.getFirstSelectedOption ()
assertequals ("bir metin", carList.getFirstSelectedOption (). getText ())
```

58. Select etiketi olmadan açılır menü ile nasıl çalışılır?

- Açılır listede seçme etiketi yoksa, seçili sınıfı kullanamayız
- Açılır listeyi ve seçeneklerini ayrı öğeler olarak değerlendirin, her öğeyi ayrı ayrı bulun
- Bir seçenek seçmek için:
  - o 1. Listeyi bulun ve tıklayın
  - o 2. Seçeneği bulun ve tıklayın

59. Ya seçme etiketi yoksa?

• Açılır liste için etiketi bir web elemanı olarak ayrı ayrı seçmelisiniz.
• Daha sonra manuel olarak tıklama yöntemini kullanın

60. Bazen sendKeys çalışmıyor

• Robot veya AutoIT
• kitaplık == jar dosyası == bağımlılık

Sayfa 69

61. Çerçeve değiştirmek için sözdizimi nedir?

• Çerçeve, başka bir html belgesinin içindeki bir html belgesidir.
• Web sürücüsü bir seferde bir sayfa / html belgesini işler. Başka bir kareyi kontrol etmek için her zaman geçiş yapmalıyız
• Driver.switchTo.frame (webelement) → iframe'i bulun ve param olarak geçirin
• Driver.switchTo.frame (string) → iframe'in kimliğini veya adını bulun ve param olarak geçirin
• Driver.switchTo.frame (int) → dizini bulun ve param olarak geçirin

62. 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
• Değiştirebilmek için önce pencere tutamacını kullanmalıyız

```
getWindowHandles () yöntemi sürücüsü.switchTo.window (Dize) // → pencere tutamacı
// her döngü için: driverGetWIndowHandles:
Driver.switchTo.window ("tutamaç")
Ifdriver.getTitle == beklenen başlık;
Kırmak;
```

63. Bir dosya karşıya yüklemek için sözdizimi nedir?

```
Public void fileUpload (Stirng yolu) {
    WebELement karşıya yükleme = sürücü.findELement; Upload.sendKeys (yol)
}
```

• Yükle düğmesini html'de bulmamız gerekiyor.
• Elemanın etiket girişi olacaktır.
• Daha sonra yüklemek istediğimiz dosyanın yolunu ileterek anahtarları göndeririz

64. Bazen sendKeys / yol çalışmıyor

• Projemizin içindeki bir dosya için dinamik bir yol oluşturma Proje konumuna giden yol:

```
String projectDir = System.getProperty (“user.dir”) // proje dizini
Dize dosyası = "src / test / kaynaklar / test_data / dosyam.txt";
Element.sendKeys (projectDir + dosya);
```

65. sendKeys () çağrılmadan metin kutusuna metin nasıl girilir?

```
    // Kullan                                                    javascriptExecutor
JavascriptExecutor JS = (JavascriptExecutor) web sürücüsü;
    // için                                          giriş                          Kullanıcı adı
JS.executeScript (“document.getElementById ('Kullanıcı'). Value = 'www.google.com'”);
    // Parola girmek için
JS.executeScript (“document.getElementById ('pass'). Value = 'tester'”);
```

66. Selenium WebDriver'da metin kutusunda ENTER tuşuna nasıl basılır?

• Selenium WebDriver kullanarak Enter tuşuna basmak için,
• Selenium Enum anahtarlarını sürekli Enter ile kullanmamız gerekiyor
• Driver.findElement (By.xpath (“xpath”)) sendKeys (Keys.ENTER);

67. Çapraz tarayıcı testi yaptınız mı? tarayıcılar arası test

• Tarayıcı türü, ana url, kullanıcı adı, şifre, ortam gibi anahtar kelimeler için her zaman bir kontrol dosyanız olduğunu belirtin.

Sayfa 70

68. Sertifika sorununu nasıl çözersiniz?

• CHROME, IE → DesiredCapabilities yeteneği = DesiredCapabilities.chrome ();
• Jenkins'e → eklememiz gerekiyor . relaxedHTTPSValidation

```
Yanıt yanıtı = RestAssured.given ()
                                .contentType (ContentType.JSON)
```

```
                .relaxedHTTPSValidation ()
                .get ("https://api.got.show/api/continents");
System.out.println (response.asString ());
```

69. Web Öğesinin sayfadaki konumunu nasıl doğrularsınız?

- element.getLocation ();
- WebElement sınıfı, öğenin sol üst köşesini döndüren bir get Location yöntemine sahiptir

70. Page Factory sınıfı?

- Page Factory sınıfı Selenium ile birlikte gelir.
- Ve sayfa nesnesi sınıfları oluşturduğumuzda kullanılır.
- Amacı, sınıfta tanımlanan webElement'leri başlatmaktır.

71. Bana salatalıkla yaptığınız test yürütme akışınızı açıklayın.

- Runner> Feature file> Scenario> Steps> Step def> POM kullanarak Selenyum kodu

72. UX ve Restful webServices'i test etmek için hangi araçları kullanıyorsunuz?

- UX → Kullanıcı Deneyimi. Öncelikle 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

73. Selenium WebDriver kullanarak tarayıcı penceresi nasıl yeniden boyutlandırılır?

- Tarayıcı penceresini belirli boyutlara yeniden boyutlandırmak için, tarayıcı penceresini yeniden boyutlandırmak için 'Dimension' sınıfını kullanırız.
- // Boyutlar sınıfının nesnesini oluşturun

```
Boyut d = yeni Boyut (480,620);
```

- // Geçerli pencereyi give n boyutuna göre yeniden boyutlandır

```
driver.manage (). window (). setSize (d);
```

74. Selenium'da hangi istisnaları biliyorsunuz?

- Sık sık **NoSuchElementException yaşıyorum**
- **StaleElementException**
  - o Öğe tamamen silindi.
  - o Öğe artık DOM'a eklenmez.
  - o StaleElementException'ı nasıl ele alırız;
    - ▪ Öğe DOM'a eklenmemiş → 'for döngüsü' içinde 'yakalama bloğunu dene'

      Veya
    - ▪ 1. Sayfayı yenileyin ve aynı öğe için tekrar deneyin.
    - ▪ 2. Öğenin kullanılabilir hale gelmesini bekleyin
- **TimeOutException**

75. ASSERT (sert iddia) DOĞRULA KARŞISI (yumuşak iddia)

- Hard assert, bir assert ifadesi başarısız olduğunda hemen bir AssertException oluşturur ve test paketi bir sonraki ile devam eder

  @Ölçek. Assert adımları başarısız olursa, testin yürütülmesi bu noktada durur! ve varsa bir sonraki teste gidecek!

  - o (Örnek: sadece basit Assert.assertTrue (boolean);)
- Soft assert, @Test sırasında hataları toplar Soft Assert, bir onaylama başarısız olduğunda ve devam ederse bir istisna atmaz

  assert ifadesinden sonraki adımla. Adımları doğrula başarısız olursa, bir başarısızlık bildirecek, ancak yürütmeye devam edecektir!
  - o Örnek: SoftAssert soft = new SoftAssert (); // yumuşak nesne oluşturma için
  - o soft.assertTrue (boolean);
  - o soft.assertAll (); // sonunda neyin başarısız olduğunu rapor edecek!

76. Selenium'da bulunan doğrulama noktası nedir?

- Selenyum IDE'de, Selenium Doğrulama ve Onaylama Komutlarını Doğrulama noktaları olarak kullanıyoruz
- Selenium WebDriver'da, doğrulama noktaları için yerleşik özellikler yoktur, tamamen bizim 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.)

77. Metnin var olduğunu doğrulayın mı?

- VerifyTextPresent → belirtilen metin dizesi sayfada bir yerde BULUNDU ise DOĞRU döndürür; Aksi takdirde YANLIŞ.
- VerifyTextNotPresent → belirtilen metin dizesi sayfanın herhangi bir yerinde BULUNAMADI ise DOĞRU döndürür; YANLIŞ, eğer öyleyse

  bulundu.

78. Bir web sayfasında bir metni nasıl bulursunuz?

• // etiket adı [içerir (metin (), 'metin')] belirli testi içerir
• //tagname[.='text '] tam metni içeriyor bazen Selenium çalışmıyor

79. Apple'ın önceki tüm kardeşlerine nasıl sahip olunur?

• Xpath: "// ul / li [içerir (text (), 'Apple Mobiles')] / precedingsibling :: li"

• Bu, "Samsung Mobiles" verecektir

80. Apple'ın aşağıdaki kardeşlerinin tümü nasıl edinilir?

• Xpath: "// ul / li [içerir (text (), 'Apple Mobiles')] / followingsibling :: li"

• Bu, önceki tüm kardeşleri verecektir (Nokia Mobiles, HTC Mobiles, Sony Mobiles, Micromax cep telefonları)

81. Web Tabloları / ızgaraları nasıl kullanılır?

• Tablo verileri için kullanılan tablo etiketi bir ızgara biçiminde düzenlenmiştir

Sütun adı için etiket Örnek -

```
<tr>
    En üst satırda <th> FirstName </th> sütun adları
    <th> Soyadı </th>
    <th> Yaş </th>
</tr>
```

o </tr> bir satırı belirtmek için kullanılan tr etiketi, bir satırdaki bir sütunu belirtmek için tüm sütuna uygulanır td etiketi Örnek

```
<tr>
    <td> Danny </td> actual_data_on_the_very_first_row
    <td> Smith </td>
    <td> 29 </td>
</tr>
```

• Bazı tablolarda tablonun verilerini göstermek için kullanılır, genellikle sütun adlarını içermez (th)

Sayfa 72

82. Excel nasıl kullanılır?

```
FileInputStream ExcelFile = new FileInputStream (yol);
excelWBook = yeni XSSFWorkbook (ExcelFile);
excelWSheet = excelWBook.getSheet (sayfaAdı);
hücre = excelWSheet.getRow (rowNum) .getCell (colNum);
```

83. Selenium sürüm 3'ü nasıl buluyorsunuz? Selenium 3, Selenium 2'den büyük ölçüde farklı mı?

• Selenium 3, selenium 2'den hata düzeltmelerine sahiptir ve 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 düzeltmesi ve 2 yerine bırakma işlemidir.

o Selenium Grid hata düzeltmeleri de yapıldı.

o Selenium projesi aktif olarak yalnızca WebDriver API'yi desteklemeyecektir.

o Zamanlamayla ilgili olarak Mozilla, Firefox'ta değişiklikler yaptı, bu da Firefox 48'den bunları kullanmanız gerektiği anlamına gelir.
geckodriver, Selenium 2 kullanıp kullanmadığınıza bakılmaksızın o tarayıcıyı kullanmak için

o Bildiğimiz gibi Selenium 3.0, Selenium Jar'ın en son sürümüdür

Sayfa 73

UZMAN

1. Maven nedir?

- Runner sınıfımı çağıran ve bağımlılıklarımı yöneten POM xml dosyası olarak adlandırılan bir oluşturma aracı ve komut istemi aracı.
- Maven, bir derleme otomasyon aracı veya bir proje yönetim aracıdır. Maven ile tüm kitaplıkları içe aktarabilir ve ayrıca

  proje yapıları. Maven'de birçok dahili şablonumuz var. Bu şablonlara arketip denir. Bir Maven temelde

  uygulamalarımızı derlemek için kullanılan bir araç.
- Komut İstemi mvn arketipi; oluşturmak
  - o Proje oluşturur
- Bir # seçin ve enter tuşuna basın
- Bir # seçin ve enter tuşuna basın
- Grup kimliği; com.nameOfProject (genellikle com.example.foo gibi ters çevrilmiş bir alan adı)
- ArtifactID; testmavenproject
  - o Sürüm girişi
  - o Paket girişi
  - o Y; giriş

2. Neden Maven? Projenizi etkili bir şekilde geliştirmenize nasıl yardımcı olur?

- Proje yapısını veya dağıtım, temizleme, paketleme, kavanoz ve çok daha fazlası gibi uygulamaları geliştirmeye ve yönetmeye yardımcı olur

  Java tabanlı proje için özellikler.
- Başka bir deyişle, bir Java aracıdır. Örnek bir proje veya iskelet proje oluşturmak istiyorsanız Maven'i kullanabilirsiniz. O bir

  otomatik inşa aracı. Maven, akıllı başlangıçlar ürettiği ve zeka varsaydığı basitliğe odaklandı

  varsayılanlar. Ayrıca, Uygulama Yaşam Döngüsü Yönetimindeki derlemeye yönelik aşamaları da kapsar, yani test etme, devreye alma, derlemeler

  yönetimi ve sürüm oluşturma.
- Projenin çok hızlı bir şekilde kurulmasına **yardımcı olur** ve build.xml gibi karmaşık derleme dosyalarından kaçınır. Maven, POM.xml gibi dosyaları gerektirir;

  yalnızca Maven'in amacına hizmet eder. POM.xml, Java Projenizin belirleyebileceği bir bağımlılıklar koleksiyonudur.

  Maven'e ve daha sonra Maven hepsini internetten indirecek ve daha sonra bir depoda, yani yerel

  depo, merkezi depo ve uzak depo.

3. Maven Artifact nedir?

- Artefakt, bir Maven deposuna dağıtılan ve genellikle bir JAR olan bir dosyadır.
- Bir Maven derlemesi, derlenmiş bir JAR ve bir "kaynaklar" JAR gibi bir veya daha fazla yapı üretir.
- Her yapının bir grup kimliği (genellikle com.example.foo gibi ters çevrilmiş bir etki alanı adı), bir yapay kimliği (yalnızca bir ad) ve bir

  sürüm dizesi. Üçü birlikte eseri benzersiz bir şekilde tanımlar. Misal:

```
<groupId> org.seleniumhq.selenium </groupId>
<artifactId> seleniumjava </artifactId>
<version> 3.11.0 </version>
```

- Bir projenin bağımlılıkları yapay olarak belirtilir.

4. Bana maven yaşam döngüsünü açıklayın?

- Komutlar yalnızca belirli **pom xml** dosyasının bulunduğu dizinde çalışabilir
- 3 yerleşik derleme yaşam döngüsü
  - o Varsayılan → Proje dağıtımınızı yönetir
  - o Temiz → Proje temizliğini yönetir
  - o Site → Projenin site belgelerinin oluşturulmasını yönetir

Sayfa 74

5. Bir derleme yaşam döngüsü aşamalardan oluşur

- Doğrula → Projenin doğru olduğunu ve gerekli tüm bilgilerin mevcut olduğunu onaylayın
- Derleme → Projenin kaynak kodunu çalıştır (hata olup olmadığını kontrol etme, yoksa → derleme başarısı)
  → Hedef klasör oluşturulur ve Raporlar burada saklanır
- Ölçek
  - o Derlenen kaynak kodunu uygun bir birim test çerçevesi kullanarak test edin.
  - o Kodun paketlenmesini veya konuşlandırılmasını gerektirmemelidir
  - o Mvn D (VariableName) = testname → Parametreye göre özel testler çalıştırın
- Paket → Derlenen kodu alın ve JAR gibi dağıtılmış bir biçimde paketleyin
- Doğrula → Kalite kriterlerinin karşılandığından emin olmak için entegrasyon testlerinin sonuçları üzerinde her türlü kontrolü çalıştırır
- Yükle → Bağımlılık olarak kullanmak için paketi yerel depoya yükleyin
- Dağıt → Derleme ortamında Bitti, son paketi diğer geliştiricilerle paylaşmak için uzak depoya kopyalar
  ve projeler

6. maven projesini tutulma projesine nasıl dönüştürürsünüz?

- Mvn tutulması

7. Java projeleri nasıl yapılır?

1. Klasörler / paketler oluşturun
2. Kitaplıklar / bağımlılıklar ekleyin
3. Sınıf dosyaları oluşturun
4. Derleyin
5. Testleri çalıştırın
6. Dağıtın

8. Bağımlılıklarınızı / kitaplıklarınızı nerede buluyorsunuz?

- Mvnrepository.com
- maven çalışmıyorsa projeyi güncelleyin
  - o Pom dosyanızın içinde bağımlılıklar olduğunda ve güncellemeyi kullandığınızda, maven JAR dosyalarını internetten çekecek ve
    projenize ekleyin

9. .m2 klasörü nedir?

- Jar dosyalarınızın / havuzlarınızın bilgisayarınızda kaydedildiği yer

10. POM xml dosyası nedir?

- Tüm projeyi yöneten bir dosya
- Bir maven komutunu çalıştırdığınızda, her şey pom.xml aracılığıyla yapılmalıdır

11. Araçların sürümleri?

- RestAssured 3.3.0 yayın tarihi: 2019-01-11

Sayfa 75

12. Log4j?

- Herhangi bir uygulama tarafından kullanılır
- Örnek: LOG4J2 → Apache'den
- Aktiviteyi kaydeder
- Dev, günlüklere bakacak, saate bakacak, IP adresine gidecek ve bir hata varsa neler olup bittiğini görecek
- Kaydediciler uygulamaların çok önemli bir parçasıdır ve gerçekleşen her adımı / olayı zaman damgasıyla birlikte tutar
- Normalde günlükler programlı olarak .log dosyasına yazılır
- Herhangi bir çerçeveye veya uygulamaya eklenebilecek hazır araçlar / kitaplıklar vardır.
- Java'da, en ünlü günlük kaydı kütüphanesi / çerçevesi apache'den LOG4J'dir.

13. Günlüklerin amacı?

- Uygulamayla ilgili olabilecek sorunları gidermemize yardımcı olun.
- Bazen uygulamada bir hata bulunduğunda, geliştiriciler öncelikle günlükleri kontrol ederler. Kullanıcının hangi adımlar olduğunu 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

Sayfa 76

TESTNG & JUNIT

1. TestNG nedir?

• 500 test vakanız var → Her test vakası için bir Java Paketi ve 500 Sınıfı oluşturuyoruz

Müşteri sizden sadece 40 tanesini duman testi için çalıştırmanızı istedi → Blokları ve raporlama mekanizmasıyla Jasmine'de hallediyoruz.
• TestNG bir test çerçevesidir
• Merkezi kontrolör: farklı test senaryolarının çalıştırılmasını yönetin ve ardından raporlar, günlükler oluşturun
• Toplu yürütme: 100 test durumu ve bunları birer birer çalıştırın
• İsteğe bağlı yürütme: bazı test durumlarını atlayabiliriz

2. TestNG'deki iddialar nelerdir?

• Testi çalıştırdık ve başlık testi durumu başarısız oldu. Diğer test durumlarını etkilemeyeceği için betiğimizin durmasını istemiyoruz.
  o Kritik → dur / başarısız İddiası
    ▪ Bir boole bağımsız değişkeni ve String mesajı alır. Bir koşulun doğru olduğunu iddia eder. Değilse, bir Onay Hatası,

    verilen mesajla birlikte atılır.
  o Kritik değil → hata / devam SoftAssert
    ▪ Soft Assert, bir iddia başarısız olduğunda bir istisna atmaz ve sonraki adımla devam eder.

    ifade iddia.

3. JUnit ve TestNG arasındaki fark

• Ek açıklamalar; **JUnit:** @Test, @BeforeClass, @AfterClass, @Before, @After, @Ignore
TestNG : @Test, @BeforeTest, @BeforeClass, @BeforeSuite, @BeforeMethod,
@AfterTest, @AfterClass, @AfterSuite, @AfterMethod
• Her ikisi de bize yardımcı olmak için çerçeve test ediyor

otomasyon komut dosyalarını çalıştırma.
• TestNG, html raporu sağlar
• TestNG vardır @Hayalhanemersin

ek açıklama Salatalık ile aynı
Veriye Dayalı Senaryo Özeti
Test yapmak.
• TestNG'de paralel testler yapabiliriz,

ancak JUnit paralel olmayı desteklemiyor
testi, bu yüzden bunun için SauLab kullanıyoruz.
• TestNG destek grubu testi ancak JUnit

desteklemiyor

• TestNG ve JUnit her ikisinde de

testi parametreleştirmek ancak TestNG
parametreli test yapılandırması
yapılandırması çok kolay. İki tane
parametreleştirme elde etmenin yolları
TestNG;
o @Parameters ve TestNG xml dosyası
o @DataPrivider

Sayfa 77

4. Çapraz Tarayıcı ve Paralel Test

• Mevcut projemde, çapraz tarayıcı testi için SauLab kullanıyoruz. Ama önceki projemde testng.xml dosyasını kullandım.

• Temel olarak, süitin içinde 3 anahtar var (ad, iş parçacığı sayısı, paralel) ve bunlardan biri için 2 farklı test oluşturdum.

• Chrome ve diğeri Firefox içindir.

• Parametre açıklaması da vardır ve adı ve değeri içerir; ad tarayıcıdır ve değer Chrome'dur.

```
<? xml version = "1.0" encoding = "UTF 8" ?>
<! DOCTYPE suite SİSTEMİ ...>
< süit ... >
    < test name = "ChromeTest" ... >
            < parametre adı = "tarayıcı" value = "chrome" />
                    < sınıflar >
                            < class name = "testsuite ..." />
                    </ sınıflar >
    </ test > <! İlk Test>
    < test adı = "FireFox" ... >
            < parametre adı = "tarayıcı" value = "FireFox" />
                    < sınıflar >
                            < class name = "testsuite…." />
                    </ sınıflar >
    </ test > <! İkinci Test>
</ süit > <! Süit>
```

Sayfa 78

SALATALIK & GHERKIN

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

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

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

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

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

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

3. Raporlarınızı salatalıkta nasıl görebilirsiniz?

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

4. Kornişon nedir?

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

Sayfa 79

5. Cucumber BDD çerçevesinin bileşenleri nelerdir?

1. Özellik dosyaları

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

2. Cukes Koşucusu

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

3. Adım tanımı

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

- TEKNİK OLMAYAN PPL'NİN ANLAŞILMASI İÇİN

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

6. @CucumberOptions ne işe yarar?

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

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

7. JUnit ile Salatalık nasıl çalıştırılır?

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

8. Salatalık TestNG ile nasıl çalıştırılır?

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

9. Koşucu sınıfınızı etiket olmadan çalıştırırsak ne olur?

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

  "Özellikler ="

Sayfa 80

10. Salatalıktaki Kanca nedir?

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

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

11. Salatalıkta nasıl ekran görüntüsü alıyorsunuz?

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

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

12. DDT ile Salatalık nasıl çalıştırılır?

- Salatalık sofraları kullanıyorum:

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

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

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

- Liste sırasına göre yazdırır
- Harita için sipariş yok

13. Arka Plan nedir?

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

14. Senaryo Taslağı nedir? Senaryo mu?

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

Sayfa 81

15. Geçebileceğim değişken türlerini nasıl sınırlayabilirim?

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

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

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

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

17. DDT için salatalık senaryosunu nasıl kullanırım?

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

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

20. Veriye dayalı

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

Sayfa 82

18. Haritalar salatalıkta nasıl kullanılır?

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

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

21. POJO salatalıkta nasıl kullanılır?

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

22. TestNG kullanarak bir grup test senaryosu nasıl çalıştırılır?

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


Sayfa 83

23. Veriye Dayalı Test

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

24. TestNG kullanarak veriye dayalı çerçeveyi nasıl oluşturabiliriz?

- @DataProvider ek açıklamasını kullanarak Veriye Dayalı Çerçeve oluşturabiliriz

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

25. TestNG'de Grup Grubu nasıl oluşturulur?

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

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

26. TestNG kullanarak test senaryoları paralel olarak nasıl çalıştırılır?

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

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

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

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

- Test.xml dosyasına dışlama etiketini ekleyerek

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


Sayfa 84

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

- Test.xml dosyasına dışlama etiketini ekleyerek

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

30. TestNG sonuçları için rapor oluşturmanın farklı yolu nedir?

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Sayfa 85

37. Test zaman aşımı testte ne anlama geliyor?

- Bir test senaryosunun alması gereken maksimum milisaniye sayısı

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

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

38. @Factory ve @DataProvider ek açıklaması nedir?

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

39. ek açıklamalar - öncelik

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

40. testNG'de paralel yürütme

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

41. Çerçeve Araçları: Hıyar BDD çerçevesi

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

42. Çerçeve Araçları: TestNG + Selenium

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

43. Çerçeveniz nasıl raporlar üretiyor?

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

44. Seçici olarak salatalık testleri nasıl yapılır?

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

Sayfa 86

45. Günlük kaydı için ne kullanıyorsunuz?

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

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

46. ÖZELLİK DOSYASI NASIL ÇALIŞIR?

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

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

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

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

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

49. Başarısız testler Hıyar'da nasıl yeniden yapılır?

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

50. Jenkins'te başarısız testler nasıl yeniden çalıştırılır?

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

Sayfa 87

51. SALATALIK TESTLERİNİ PARALEL OLARAK KOŞUYOR MU?

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

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

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

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

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

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

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

aynı anda kaç testin çalıştığını belirleyebiliriz

2. Salatalık 4.x paralel seçeneği

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

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

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

pom dosyası.

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

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

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


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

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

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

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

Nasıl koşulur?

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

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

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

Paralelleştirmenin faydaları:

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

Paralelleştirmenin zorlukları?

o uygulaması zor -> yapması kolay değil.

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

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

o projemde bazı test durumları paralel olarak çalışmadı.

Sayfa 89

API

1. API nedir?

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

2. API ve Web Hizmetleri mi?

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

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

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

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

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

5. HTML Durum Kodları?

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

6. Yanıt aldığınızda ilk kontrol ettiğiniz şey nedir?

- Durum teklifi (200 her zaman Tamam anlamına gelir)
- Her zaman 404'ün bulunamadı anlamına geldiğini kontrol ederiz
- rest-assured.io ==> ECS makinesini uzak Masaüstü arama türünde bulmaya yönelik otomasyon için

7. HTTP yöntemleri ve istek türleri

- Get vücut gerektirmez
- Yerleştirme gerekli gövde, GÜNCELLEME bilgisi anlamına gelir
- Gönderi gerekli gövde, CREATE bilgi anlamına gelir
- Silme , gövde gerektirmez
- GET -> OKU, YAYINLA -> OLUŞTUR, PUT -> GÜNCELLEME, SİL -> SİL
- POST VS PUT

Sayfa 90

8. Parametreler api

- 2 TİP:
  - YOL PARAMETRESİ (DEĞER URL'NİN BİR PARÇASI OLACAKTIR) SORGU / TALEP
  - PARAMETRELER (ANAHTAR + DEĞER FORMATI)

9. Hamcrest Matcher ne içindir?

- Hamcrest, 'eşleşme' kurallarının bildirimsel olarak tanımlanmasına izin veren eşleştirici nesneler yazmak için bir çerçevedir.

```
import org.junit.jupiter.api.Test ;
statik kuruluşu içe aktar . hamcrest . MatcherAssert . assertThat ;
statik org içe aktar . hamcrest . Eşleştiriciler . *;

public class BiscuitTest {
    @Ölçek
    public void testEquals () {
        Biscuit theBiscuit = new Biscuit ( "Ginger" );
        Biscuit myBiscuit = new Biscuit ( "Ginger" );
        assertThat ( theBiscuit , equTo ( myBiscuit ));
      assertThat ( "çikolata parçaları" , theBiscuit . getChocolateChipCount (), equTo ( 10 ));
      assertThat ( "fındık" , theBiscuit . getHazelnutCount (), equTo (3));
    }
}
```

```
// ilk bağımsız değişkenin ikinciye eşit olup olmadığını doğrulayın
assertThat ( str1 , is ( "Kunkka" )) ;
assertThat (str1 , olduğu (str2)) ;

// ilk bağımsız değişkenin ikinciye eşit OLMADIĞINI doğrulayın
assertThat (str1 , bir ( değil ( "Tidehunter" ))) ;

// büyük / küçük harfleri görmezden gelin
assertThat ( str1 , equToIgnoringCase ( "kunkka" )) ;

// öncesi ve sonrası boşluğu yok saymayı karşılaştırın
assertThat ( str1 , equToIgnoringWhiteSpace ( "Kunkka" )) ;

// sayıları karşılaştır
assertThat ( 10 , daha büyükThan ( 9 )) ;
assertThat ( 10 , lessthan ( 11 )) ;
assertThat ( 10 , lessThanOrEqualTo ( 11 )) ;

// boş olmadığını doğrula
assertThat ( str1 , notNullValue ()) ;

List <String> list = Diziler. asList ( "bir" , "çok" , "ağaç" ) ;
    assertThat (list , hasSize ( 3 )) ;
    assertThat (list , containsInAnyOrder ( "çok" , "ağaç" , "bir" )) ;
    assertThat (liste , hasItems ( "bir" , "çok" ) ) ;

<Tamsayı> sayıları listele = Diziler. asList ( 11 , 12 , 13 ) ;
assertThat (sayılar , her öğe (daha büyükThan ( 9 ))) ;
```


Sayfa 91

10. RestAssured Günlük Kayıt Günlükleri

- **Günlük Kaydı İste**

```
given (). log (). all () // Parametreler, başlıklar ve gövde dahil tüm istek belirtimi ayrıntılarını günlüğe kaydedin
given (). log (). params () // Yalnızca isteğin parametrelerini günlüğe kaydedin
given (). log (). body () // Yalnızca istek gövdesini günlüğe kaydedin
given (). log (). headers () // Yalnızca istek başlıklarını günlüğe kaydedin
given (). log (). cookies () // Yalnızca istek çerezlerini günlüğe kaydedin
given (). log (). method () // Yalnızca istek yöntemini günlüğe kaydedin
given (). log (). yol () // Yalnızca istek yolunu günlüğe kaydedin
```

- **Yanıt Günlüğü**

```
get ("/ x"). sonra (). log (). body ()
get ("/ x"). sonra (). log (). ifError ()
get ("/ x"). sonra (). log (). tümü ()
get ("/ x"). sonra (). log (). statusLine () // Yalnızca durum satırını günlüğe kaydet
get ("/ x"). sonra (). log (). headers () // Yalnızca yanıt başlıklarını günlüğe kaydet
get ("/ x"). sonra (). log (). cookies () // Yalnızca yanıt çerezlerini günlüğe kaydet
get ("/ x"). sonra (). log (). ifStatusCodeIsEqualTo (302)
      // Yalnızca durum kodu 302'ye eşitse günlüğe kaydedin
get ("/ x"). sonra (). log (). ifStatusCodeMatches (eşleştirici)
      // Yalnızca durum kodu sağlanan Hamcrest eşleştiriciyle eşleşirse günlüğe kaydedin
```

11. Serileştirme ve Seriyi Kaldırma

- Serileştirme; bir Java nesnesini API JSON formatına eşleştirdiğimizde (JAVA NESNESİNİ JSON'A DÖNÜŞTÜR);
    - Java nesnesi (POJO (Düz Eski Java Nesnesi), FASULYE) → API JSON / XML ile eşleştirin
    - Bir sınıftan bir nesneye sahip olduğumuzda ve onu RESTful API'mizde bir JSON formatına eşleştirdiğimizde

```
{make: "Toyota",
Model: "Camry"}
Araba arabası = yeni Araba ();
car.setMake ("Toyota");
car.setModel ("Camry");
verilen (). body (car) .when (). post (uri)
```

- Seriyi kaldırma; API JSON / XML → Java Nesnesine (JSON'dan JAVA NESNE) EŞLEME

```
Araba araba2 = yeni Araba ();
car2 = ne zaman (). get (uri) .body.as (car.class);
car.setMake ("Toyota");
car.setModel ("Camry");
```

Sayfa 92

12. RestAssured Kitaplığı ile API / Web Hizmetleri?

```
import static io.restassured.RestAssured. *;
URI uri = yeni URI ("... / yöntemler (alma, yayınlama)")
```

- GET;

```
Yanıt yanıtı = verildi (). Kabul et (ContentType.JSON) .when (). Get (URI);
response.then (). assertThat (). statusCode (200).
      ve (). assertThat (). ContentType (ContentType.JSON);
```

- POST;

```
Yanıt yanıtı = verildi (). ContentType (ContentType.JSON) .with (). Accept (ContentType.JSON)
. ve (). body (JSONbody) .when (). post (URI);
response.then (). assertThat (). statusCode (200);
```

-

statik org.hamcrest.Matchers. *;
sonra (). assertThat (). body ("Id", Matchers.equalTo (123));

•

JsonPath json = JsonPath (JSONbody);
json.getString ("anahtar");
json.getInt ("anahtar");
json.getList ("key1.key2");

13. EndPoint nedir?

• <protokol>: // <hizmet adı> / <Kaynak Türü> / Kaynak Kimliği → URI (Tekdüzen Kaynak Tanımlayıcı)
Temel URI / kaynak? Parametreler
(http://www.google.com/search?source=book…) →? → sorgu parametreleri

14. Yetkilendirme ve Kimlik Doğrulama

• kimlik doğrulama -> sen kimsin
• yetkilendirme -> hangi haklara sahipsiniz
• Kimlik doğrulama, kullanıcı ve paroladır
• Yetkilendirmenin türleri vardır:
  ○ Yetki yok
  ○ Temel Yetkilendirme
  ○ Taşıyıcı Jetonu
  ○ Yetkilendirmeyi ebeveynden devral

15. RESTful Web Hizmeti / API

• REST, Temsili Devlet Transferi anlamına gelir
• RESTful, REST mimari konsepti uygulanarak yazılan web servisleri için yönlendirilir.
  ○ RESTful'da, CRUD işlemlerini gerçekleştirmek için GET, POST, PUT, DELETE gibi web hizmeti http yöntemleri kullanılabilir.
  ○ CRUD = Oluştur → Oku → Güncelle → Sil

Sayfa 93

16. Yanıt bedeninizdeki bir değeri nasıl doğrularsınız?

• Exp için: kimliğin doğru numarayı içerdiğini doğrulayın
  o *Hamster Matcher* iddia kitaplığıdır.

sonra (). 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:

String body = ... thenReturn (). Body (). AsString ();
JsonPath json = new JsonPath (gövde);
assertEquals (123, json.getInt ("Id"));

  o Bir (POJO) nesnesine (veya Nesne Eşleme) serileştirme

POJO myPojo =… ne zaman (). 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 bilgileri içermiyorsa, **401 Yetkisiz** durum kodunu döndürür .
  ○ Meydan okuma
    ■ İstek API'ye ulaştığında, API kimlik bilgileri gerektirdiğini söyleyecek ve ardından müşteri sağlayacaktır
kimlik bilgileri.
  ○ oauth -> kimlik doğrulamak için üçüncü taraftan anahtarların ve belirteçlerin kullanıldığı kimlik doğrulama türleri. Var
2 tür oauth:
    ■ oauth1 → uygulaması zor
    ■ oauth2 → daha güvenli

- sindirmek
  - Temelden daha şifrelenmiştir. https ...

18. SABUN kullanmanın avantajı nedir?

- REST, çok çeşitli veri formatlarına izin verirken, SOAP yalnızca XML'e izin verir.
- JSON ile birleştirildiğinde (genellikle verilerle daha iyi çalışır ve daha hızlı ayrıştırma sağlar), REST genellikle dikkate alınır çalışmak daha kolay.
- JSON sayesinde, REST tarayıcı istemcileri için daha iyi destek sunar.
- REST, özellikle değiştirilmemiş ve dinamik olmayan bilgileri önbelleğe alarak üstün performans sağlar
- Yahoo, Ebay, Amazon ve hatta Google gibi büyük servisler için en sık kullanılan protokoldür.
- REST genellikle daha hızlıdır ve daha az bant genişliği kullanır. Mevcut web siteleriyle 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, ek işlevler ekleyebilirler.

Sayfa 94

19. SOAP ve RESTful web servisleri arasındaki fark nedir?

- RESTful JSON, XML, TEXT'i destekler, ancak SOAP yalnızca XML'i destekler
- REST, SOAP tabanlı web hizmetlerinden daha hızlıdır

20. URI nedir, amaç ve biçim?

- URI, Tekdüzen Kaynak Tanımlayıcısı anlamına gelir
- URI'nin amacı, web hizmetini barındıran sunucuda bir kaynak bulmaktır.
- Bir URI aşağıdaki biçimdedir:
  - <protokol>: // <hizmet-adı> / <KaynakTürü> / <KaynakKimliği>

21. Projenizde hangi Web Hizmetlerini kullanıyorsunuz?

- Temsili Aktarım Durumu olan Restful kullanıyorum ve XML ve JSON ile iletişim kuruyor, ancak şu anki proje JSON kullanıyor

22. XML nedir?

- Bilgi işlemde, Genişletilebilir Biçimlendirme Dili (XML), kodlama için bir dizi kural tanımlayan bir biçimlendirme dilidir hem insan tarafından okunabilen hem de makine tarafından okunabilen bir biçimde belgeler.

23. JSON nedir?

- JavaScript Nesne Gösterimi (minimum, verileri yapılandırmak için okunabilir bir biçimdir.)
- XML'e alternatif olarak, öncelikle bir sunucu ile web uygulaması arasında veri iletmek için kullanılır.
- Temel olarak, XML'in basit bir sürümü
- Anahtar: Değer biçimi
- Anahtar her zaman çift tırnak içindedir ve eğer dize çift tırnaklarsa ve sayılar tırnak işareti yoksa değer
- Tamamen http protokolüne dayalıdır - bu nedenle tarayıcıdaki bağlantıya ulaşır ve sonuçları görür

24. Swagger'ı biliyor musunuz? Swagger nedir

- Swagger, geliştiricilerin tasarım yapmasına yardımcı olan geniş bir araç ekosistemiyle desteklenen açık kaynaklı bir yazılım çerçevesidir. RESTful Web hizmetlerini oluşturun, belgeleyin ve kullanın.
- Swagger, makinelerin okuyabilmesi için API'lerinizin yapısını tanımlamanıza olanak tanır.
- API'lerin kendi yapılarını tanımlama yeteneği, Swagger'daki tüm mükemmelliklerin köküdür
- xml şemasına benzer ancak Json için

25. json vs gson

- JSON, anahtar ve değerlere sahip bir biçimdir
- GSON, bir dönüştürme sürecidir
  - java'dan json'a (serileştirme),
  - json'dan java'ya (seriyi kaldırma)

26. Nasıl ve nereye talep gönderiyorsunuz?

- Rest kullandığım için uç noktaları var. Geliştiricilerim genel URL'ler oluşturur ve bu URL'ye istekler gönderilir

27. Web hizmetleri olmayan 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 açıklamaları ve yönergeleri burada

Sayfa 95

29. API'yi projenizde nasıl test ediyorsunuz?

- 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.
- Test kullanıcısı olarak bir API isteği gönderiyor ve api'nin durum kodunu, yanıt gövdesini ve uç noktalarını kontrol ediyoruz.
  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 doğruluyorum
- Negatif - Durumun 200 olmamasını bekliyorum, geçersiz istekler, başlıklar, parametreler ve gövde gönderiyorum

30. Rest 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 RESTASSURED kitaplığı kullanıyorum.
- POST, PUT, GET, DELETE tür istekleri gönderiyorum ve yanıt durum kodunu ve yanıt gövdesini, başlığı doğrularım.
- API'nin pozitif ve negatif testlerini de yapıyorum.
- Pozitif test yaptığımda, geçerli istek parametreleri, geçerli başlıklar, geçerli istek json gövdesi gönderiyorum ve
  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 gönderiyorum 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?

- Buna bağlıdır, API geliştiricim bu URL'nin GET, POST, PUT veya DELETE istekleriyle çalışıp çalışmayacağına karar verir

32. API'nizi manuel olarak nasıl test edersiniz?

- Postman kullanıyorum → bu, REST API URL'sini test eden bir REST API istemci aracıdır

33. API testi için hangi araçları kullanıyorsunuz?

- Manuel test için postacı
- Kendinden Emin Kitaplık

34. Rest API'deki İstek türleri nelerdir?

- Al, Gönder, Koy ve Sil istekleri vardır
  - Okunan verileri alın
  - Gönderi veri oluşturur
  - Güncelleme verilerini koy
  - Sil verileri siler

35. REST API'deki başlıklar nelerdir?

- Kabul Et (İçerik Türü.JSON) türünü kullanıyorum - aldığım şeyin JSON veya XML biçiminde olması gerektiğini kontrol eder
- Ve ContentType. (Contenttype.Json) - gönderdiğim şeyin JSON biçiminde olması gerektiğini kontrol eder

36. RestAssured Kütüphane nedir?

- BDD biçiminde olan ve seriyi kaldırma ve serileştirmeyi kullanarak java kodunu entegre etmeye yardımcı olan web dışı bir hizmet api
  verileri saklamak, doğrulamak ve doğrulamak için Json'dan verileri 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

Sayfa 96

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 ()
  - equTo ()

40. Yanıt arayüzünü nasıl kullanırım?

✪ Raporlama

- Mvn Verify başarısız olsa bile testleri çalıştırır (hatayı yok sayar)

- Tüm testin bitmesini bekler
- Başarısızlığı yok sayar, bc bu bizim derleme yapılandırmamızda var

<testFailureIgnore> doğru </testFailureIgnore>

- Doğrulama, testten sonra gelen bir Maven yaşam döngüsüdür

• Mvn testi, bir şey başarısız olursa testi çalıştırmayı durdurur
• Aldığımız orijinal html raporu o kadar iyi değil, istatistiksel verilere ihtiyacımız var
• Örn; "html: hedef / salatalık-raporu" → Kaç testin başarılı / başarısız yüzdeleri olduğu gibi
• Daha fazla istatistiksel veri raporlama için *Cucumber Sandwich'i* (bu, pom xml'de bir bağımlılık dosyasıdır) kullanacağız
• cukesrunner'da "json: target / cucumber.json" ekleyin

- Bu, JSON dosyasından alınan bir html raporu → Bu rapor nasıl çalışır? Json dosyası, raporu oluşturmak için kullanılır
- Sürüm 3.15 (vid'dan)
- pom'a yeni bir yapı xml ekleyin (zaten pom dosyanızda, TestProject adında)
  - Yalnızca bu json raporlamasını alacaksınız (grafikler ve istatistikler içeren salatalık raporu. Bu rapor,

    Yalnızca SİZİN görebileceğiniz yerel olun, Jenkins için değil) YALNIZCA MVN Doğrulaması'nı çalıştırırsanız
  - ANCAK ÇALIŞTIKTAN SONRA DAİMA JSON DOSYASI (salatalık raporundan farklı) ALACAKSINIZ

    TEST, BİLE DOĞRULAYACAK
    - Bu JSON dosyası, salatalık raporu eklentisi için Jenkins için çok önemlidir

• TestProject derlemesi:

- <id> yürütme </id>
- <phase> doğrula </phase> - bu nedenle html (json) raporu yalnızca doğrulama kullanılırken oluşturulur
- <hedefler>
- <goal> oluştur </goal> </goals>
- Rapor ayrıca size bir json dosyası verecektir
- Doğrulamayı kullanarak testleri çalıştırmak için, pom dosyasına sağ tıklayın ve maven oluştur seçeneğine 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 doğrula yazın
- Sözdizimi mvn <yaşam döngüsü / hedef> şeklindedir

• mvn doğrulaması kullanılarak Yürütme Sırası

1. Pom dosyasına karşı koşun
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?

Sayfa 97

41. 100 limitli parametreye ve çalışan id = 100 yol parametresine ihtiyacım olan bir yöntemi nasıl yazabilirim?

• Yazardım;

- . Ve (). Params ("limit", 100)
- . Ve (). PathParams ("employee_id", 110)

42. Backend-API nedir?

• Uygulama mantık kodunun bulunduğu yerdir. Koşullarınız vb.
• Nasıl test edilir?

1) El ile → Postacı vb. Araçları kullanma İstekler göndererek ve yanıtları doğrulayarak.
2) Otomasyon → Java + RestAssured Kitaplığı

43. HTTP İsteği ve HTTP Yanıtı nedir?

HTTP istek yöntemi dört bileşenden oluşur:

- **İstek Yöntemi** → Al, Gönder, Koy, Sil (bunlar yaygın olanlardır)
- **URI talep et** → kaynağın URL'si
- **Üstbilgi İste** → Dil Kabul Et, Kodlama Kabul Et, Kullanıcı Aracısı, Ana Bilgisayar
- **Talep Gövdesi** → bu kaynağa gönderilecek veridir

HTTP yanıt yöntemi üç bileşenden oluşur:

- **Yanıt Durum Kodu** → 200, 301, 404, 500 (bunlar en yaygın olanlardır)
- **Yanıt Başlığı Alanları** → Tarih, Sunucu, Son Değiştirilen, İçerik Türü
- **Yanıt Gövdesi** → bu, sunucudan istemciye geri gelen verilerdir

Sayfa 98

HTML ve CSS

1. HTML kodunda etiketi nedir?

- öğeler → etiketi, diğer sayfa öğelerini ve
  HTML belgesini bölümlere ayırır.
- Web geliştiricileri, HTML öğelerini bir arada gruplandırmak ve sitedeki birçok öğeye CSS stilleri uygulamak için <dib> öğelerini kullanır.
  bir Zamanlar.

2. Web Uygulama Mimarisi ??

- Başlangıç aşaması
  - o HTML yapısı: yalnızca sorumlu verilerdir
  - o CSS sunumu / görünümü: tasarım için
  - o JavaScript dinamizmi / eylemi: işlev için
- Arka Uç → Veritabanı

3. HTML nedir?

- Web sayfalarını tanımlamak için kullanılan bir dildir. (1991'de Tim Berners Lee tarafından oluşturuldu)
- Hiper Metin Biçimlendirme Dili anlamına gelir.
- Bir programlama dili değildir; biçimlendirme dilidir.
- Biçimlendirme dili, biçimlendirme etiketlerinden oluşan bir koleksiyondur
- Web sayfalarını tanımlamak için işaretleme etiketlerini kullanır.
- Html, her şeyi küçük harfle yazmanız önerilir

4. ETİKETLER nedir?

- Web öğelerinin oluşturulması
- Kaplar gibi davranırlar. Açılışları ve kapanışları arasında kalan bilgiler hakkında size bir şeyler söylerler.
  etiketleri.
- Etiketlerin içinde öznitelikler var

5. ÖZELLİKLER nedir?

- Bir elemanın içeriği hakkında ek bilgi sağlar.
- Öğenin açılış etiketinde görünürler.
- değer özelliği, gerçek değerdir
- isim özniteliği bu değerin tanımlayıcısıdır

6. HTML dosyaları nasıl yazılır?

Index.html

Bir HTML sayfasının yapısı:

```
<! DOCTYPE html>
    <html>
        <head>
            <title> Yakala </title>
        </head>

        <body>
                İçerik
```

</html> </body>

Sayfa 99

7. DOCTYPE HTML nedir?

- <! DOCTYPE html>
- Web tarayıcısına sayfanın hangi HTML (HTML5) sürümüyle yazıldığına dair bir talimattır.
  - o Belgenin en üstünde olmalı
  - o Beyan, büyük / küçük harfe duyarlı değildir

8. Başlıklar?

- HTML, altı seviyeli başlık yazı tipini tanımlar.
- Başlık öğeleri h1, h2, h3, h4, h5 ve h6'dır; h1 en yüksek (veya en önemli) düzeydir ve h6, en az.
  - o <h1> > <h2> > <h3> > <h4> > <h5> > <h6>

9. Satır sonları?

- **
** paragraflar arasına bir satır ekler, sonraki satıra gider, bu etiketin kapatılmasına gerek yoktur
- **
** Paragrafın ortasına bir satır sonu eklemek için
- ** ** Web sayfasına hoş geldiniz **.
** Bilmek istediğiniz her şey bu web sitesinin içindedir

10. Yatay Kurallar?

- **<hr />** Temalar arasında ara oluşturmak için (paragrafı bitirdikten sonra ve yeni başlıklardan önce)

11. Kalın İtalik Altı Çizili Yorum Paragrafları?

- **BOLD**
- **ITALIC**
- **UNDERLINED** <u> </u>
- **YORUM** <! - ->
- **PARAGRAFLAR**

12. Hizalama ve Stil?

- <h1 style = "text-align: center; background: black; color: yellow;"> Bu Cybertek! </h1>
- <h2 style = "color: orange"> Biz kimiz? </h2>

13. Arka Plan Rengi?

<body style = "background: url (https://images7.alphacoders.com/373/thumb-1920-373253.jpg);"> </body>

14. GÖRÜNTÜLER?

<h4> Bu bir kedi! </h4>
<img src = "kedi1.jpg"> </img>

img => etiket
src => özellik
cat1.jpg => değer

Sayfa 100

15. LİSTE?

Her satır bir liste öğesidir

```
<ul> <! - kapanış etiketi ->
    <li> Java </li>
    <li> Selenyum </li>
    <li> Çevik </li>
    <li> API </li>
</ul>
```

16. BAĞLANTILAR?

 KENDİNİZİ DENEYİN

a : Bağlantı anlamına gelen etiket
href : Bağlantı uygulamasının hedefini belirten **nitelik** .
html Bağlantının gideceği gerçek URL

17. TABLO oluşturmak?

```
<table> <! - bir tablo oluşturur ->
<kafa> <! - başlık satırı ->
        <th> Ad </th> <! - sütun adı ->
        <th> Soyadı </th>
        <th> Konu Adı </th>
</thead>
<tbody> <! - tablonun içeriği ->
        <tr> <! - tablodaki satır ->
                <td> Nadir </td> <! - hücre ->
                <td> Shafiev </td>
                <td> JIRA </td>
        </tr>
                <tr>
                <td> Akbar </td>
                <td> Smith </td>
                <td> OCA </td>
        </tr>
</tbody>
</table>
```

18. Değişiklikleri nasıl görebilirim?

- html dosyasında değişiklik yapın
- CTRL + S veya CMD + S dosyasını kaydedin
- kromdaki html dosyasını yenileyin

19. Nasıl yenilenir?

- F5
- CMD + R veya CTRL + R
- Tarayıcıdaki yenile düğmesini tıklayın
- dosyayı tekrar açın
- URL'yi seçin

Sayfa 101

20. FORMLAR nedir?

- Bilgileri doldurmanız için boşluklar içeren basılı bir belgeye atıfta bulunmuşsa

1- **Metin Eklemek:**

o Metin Girişi <input type = "text" name>

```
<input type = "text" name = "username" size = "15" maxlength = "30" />
```

o Parola Girişi <input type = "text" name>

```
<input type = "text" name = "password" size = "15" maxlength = "30" />
```

o Metin Alanı (Çok Satırlı) <textarea> </textarea>>

```
<textarea name = "comments" cols = "20" rows = "3"> Yorumlarınızı girin </textarea>
```

2- **Seçim Yapmak**

o Radyo Düğmesi

```
<input type = "radio" name = "gender"> Kadın <! - yalnızca bir seçenek ->
```

o Kontrol Düğmesi

```
<input type = "checkbox"> Superbowl ticari <! - birden çok seçenek ->
<input type = "checkbox" işaretlendi> Superbowl reklamı
```

o Açılır Liste Kutusu

```
<label> </label>
    <seç>
            <option> İlkbahar 2019 </option>
```

o Çoklu Seçim Kutusu

```
<label> </label>
    <isim seçin = "" size = "2" çoklu = "çoklu">
            <option> İlkbahar 2019 </option>
            <option> Sonbahar 2019 </option>
```

3- **Form Gönderme**
Gönder Düğmesi
<input type = "submit" value = "Uygula"> </input>

Sayfa 102

SQL

21. SQL biliyor musunuz?

- Evet, SQL Sorguları ve DDL ve DML komutları yazma konusunda çok rahatım.
- Şu anda AMAZON CLOUD SERVER içinde çalışan Oracle veritabanıyla çalışıyor.
- DDL (Veri tanımlama dili): CREATE, ALTER, DROP, TRUNCATE
- DML (Veri işleme dili): SEÇ, SİL, EKLE, GÜNCELLEME

22. SQL?

- Yapılandırılmış Sorgu Dili. Veri tabanındaki 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 kontrol etme
  - Veritabanı tutarlılığı ve bütünlüğü

23. SQL ifadelerinin kategorileri nelerdir?

ben. DML (Veri İşleme Dili)
- DML ifadeleri bir tablodaki kayıtları etkiler. Bunlar temel
  birkaç kayıt seçme gibi veriler üzerinde gerçekleştirdiğimiz işlemler
  bir tablodan, yeni kayıtlar eklemek, gereksiz silmek
  kayıtlar ve mevcut kayıtları güncelleme / değiştirme.

ii. DDL (Veri Tanımlama Dili)
- DDL ifadeleri 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 nesnelerine sahip.

iv. TCL (İşlem Kontrol Dili)
- TCL beyanları, kontrol etmenize ve yönetmenize olanak tanır
  SQL içindeki verilerin bütünlüğünü korumak için işlemler
  ifadeler.

24. Bana TCL'den bahseder misin?

- SQL dili, dört tür birincil dil ifadesine bölünmüştür: DML, DDL, DCL ve TCL.
- Bu ifadeleri kullanarak, veritabanı nesneleri oluşturarak ve değiştirerek bir veritabanının yapısını tanımlayabiliriz ve
  güncellemeler veya silmelerle bir tablodaki verileri işleyin.
- Tek bir iş birimi oluşturmak için hangi kullanıcının verileri okuyabileceğini / yazabileceğini veya işlemleri yönetebileceğini de kontrol edebiliriz.

25. Sürümler

- Java 8 → 2014 mevcut Java 7 → 2011 - 2014 Java 6 → 2006 - 2011
  Selenyum 3.5.3

26. Veritabanı Şeması?

- Tüm tabloları ve sütun adlarını, veri türlerini ve PK, FK ve tabloların birbiriyle nasıl ilişkili olduğunu içeren bir şema gibidir.

27. SQL cümlesi?

- SEÇ ve Kimden

Sayfa 103

28. Ne tür bir Veritabanı testi yapıyorsunuz?

- Çoğunlukla Veritabanı doğrulamaları yapıyorum.
- Ön uçta değişiklikler yapıyorum veya veri ekliyorum (ödünç oluşturuyorum) ve veritabanında doğruluyorum. Ön uçtaki veriler, DB
- Ayrıca RESTapi kullanarak değişiklikler yapıyorum ve Veritabanında da değişikliklerin başarılı olduğunu doğruluyorum.
- Ayrıca DB geçiş sürecini de destekliyorum. Kodum, JDBC'yi kullanarak Sybase'e (eski veritabanı) bağlanıyor ve ardından Oracle'a Bağlanıyor (YENİ DB) daha sonra verilerin taşındığından emin olmak için kayıtları karşılaştırın

29. 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 sistemiyle ilgili uzmanlığınız var?
    - SQL ve Oracle gibi RDBMS

30. Kısıtlamalar nelerdir?

- Tablo sütununun uyması gereken özellikler.
- Sütunların, verilerin nasıl depolanacağını tanımlayan kısıtlamaları vardır.
  - Birincil Anahtar: benzersiz ve BOŞ DEĞİL
  - Yabancı Anahtar: yinelenen ve NULL olan ve PK'de olmayan verileri ekleyemez
  - Benzersiz Anahtar: yalnızca benzersiz değer
  - Null: boş olabilir
  - null değil: null olamaz

31. SQL'de veri türleri?

- Numara
- Tamsayılar
- karakter → char (20): 20 yıllık boşluklar bellekten alınır
- varchar → varchar (30): varchar2 belleğinden 5 boşluk
- boole
- tarih
- para birimi

32. SQL select deyimleri için yetenekler

- Projeksiyon → Tablodaki sorgu tarafından döndürülen sütunları seçin
- Seçim → Bir sorgu tarafından döndürülen tablodaki satırları seçer
- Katıl → Farklı tablolarda depolanan verileri aralarındaki bağlantıyı belirterek bir araya getirir

Sayfa 104

33. DML (Veri İşleme Dili) - DDL (Veri Tanımlama Dili)

DML komut eylemleri geri yüklenebilir. DDL komutu eylemleri **olamaz** / çözülmek restore edilecek.

Komutlar :
- Tablo adından **SEÇİN** ; (okuyun)
- tablo adı değerlerine **INSERT** (...); (ekle)
- **GÜNCELLEME** tablename SET değeri NEREDE konum;
- tablename WHERE konumdan **SİL** ; (satırlar)
- **BİRLEŞTİR**

Komutlar :
- **CREATE** tablo tablename (sütun1, sütun2…);
- **ALTER** tablo tablo adı değiştirme değeri;
- **TRUNCATE** tablo tablo adı; (tüm tablo verilerini silin)
- **DROP TABLE** ; (yapısıyla birlikte tüm tabloyu silin)
- **YENİDEN ADLANDIR**
- **COMMENT**

34. Oracle'da PL / SQL işlevleri ve Tetikleyiciler?
- Bir DML ifadesi yürütüldüğünde tetikleyici otomatik olarak tetiklenir.
- 2 tür işlev vardır:
  - Prosedür (Java'daki bir void yöntemi gibi)
  - İşlev (Java'da void olmayan (dönüş) bir yöntem gibi)

35. SQL JOIN BİRLEŞİMLERİ

JOIN (INNER) JOIN
birden çok tablodan veri alırken kullanılır
ve yalnızca eşleşen verileri döndürür
SOL DIŞ KATILMA
birden çok tablodan veri alırken kullanılır
ve soldaki tablo ile eşleşen sağdaki tüm tabloları döndürür
tablo kayıtları.
SAĞ (DIŞ) KATIL
birden çok tablodan veri alırken kullanılır
ve sağdaki tabloyu ve eşleşen sol tabloyu döndürür
tablo kayıtları.
TAM (DIŞ) KATIL
birden çok tablodan veri alırken kullanılır
ve hem tablo kayıtlarını hem de eşleşen ve
eşleşmeyen

36. BİRLİK
- Union, iki sorgunun sonuç kümelerini birleştirir

```
table_name'den column_names'i seçin {UNION} table_name'den column_name'i seçin)
```

37. En yüksek maaşlı ilk 3 çalışan nasıl bulunur?
- mySQL'de ** oracle'da ROWNUM var

```
Çalışanlardan maaş, ad, soyadı SEÇİN
MAAŞ İLE SİPARİŞ AÇIKLAMA SINIRI 3;
```


Sayfa 105

38. Çalışanlarda yinelenen isimleri bulun

```
Çalışanlardan ad, COUNT (ad_adı) SEÇİN
GROUP BY first_name
HAVING (COUNT (ilk_ad> 1);
```

39. Maaşları ortalamanın altında olan çalışanlar nasıl bulunur?

```
Çalışanlardan birinci isim, maaş SEÇİN
NEREDE maaş <= (Çalışanlardan ORTALAMA (maaş) SEÇİN);
```

40. Her departmanda maksimum maaş nasıl bulunur?

```
İlk_adı SEÇİN, MAX (maaş)
Bölümden d DIŞ ÜZERİNDEN ÇALIŞAN ÜZERİNDEN SOL (d. Bölüm_kimliği = e. Bölüm_kimliği)
GROUP BY depart_id;
```

41. En düşük maaş nasıl bulunur?

```
Çalışanlardan ad, soyad, maaş, iş kimliği SEÇİN
NEREDE maaş = (MIN (maaş) Çalışanlardan);
```

42. Çalışanların ikinci en yüksek maaşı nasıl bulunur?

Çalışanlardan MAKS (maaş) SEÇİN
Maaş NEREDE DEĞİLDİR (Çalışanlardan MAKS (maaş) SEÇİN);

43. SQL Geliştirici

- Geliştirme ortamı (sorguları kullanarak veritabanının manuel olarak test edilmesi)
  - Sürüm 2.1 -2009 → 3.0 - 2011
  - Sürüm 4.0 - 2013 (en son)
- Migration 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
- Window, Linux ve Mac OS x'i destekler

44. SQL İfadeleri Yazmak

- Anahtar kelimeler büyük harfle yazılırken, sütunlar ve tablo adları küçük harflidir
- İfadeler büyük / küçük harfe duyarlı değildir
- Maddeler genellikle ayrı satırlara yerleştirilir
- Anahtar kelimeler kısaltılamaz veya satırlara bölünemez

45. Aritmetik İfadeler

- Operatörleri herhangi bir maddede kullanırsınız (From maddesi hariç)
- Tarih ve Zaman Damgasıyla - yalnızca toplama ve çıkarma kullanılabilir
- Topla (+), Çıkar (-), Çarp (*), Böl (/)

46. Tarihlerle Çalışma

- Varsayılan tarih görüntüleme biçimi DD-MON-RR şeklindedir

Sysdate işlevi
İade tarih ve saati
Seç SysDate itibaren çift ;

47. SQL tecrübeniz var mı?

- Evet, ilişkisel veritabanlarıyla çalıştım ve DDL ve DML komutları konusunda çok rahatım

Sayfa 106

48. Arka uç test çerçevesi

- Serileştirme ve serileştirme sürecinin bu şekilde gerçekleşmesi için çerçevemde RESTASSURED'i kullanmak
  Json yanıtını bir java toplama veri yapısında depolayın ve verileri beklenen değerle (ayrıca java'da saklanır) onaylayın
  veri yapısı)
- ÇERÇEVİMİ DAVETMEDEN ÖNCE manuel test için postacı kullanırım

49. Arka Uç Testi

50. Herhangi bir arka uç / veritabanı testi yaptınız mı?

- Evet, veritabanları ile çalışma konusunda çok deneyimim var.
- Ve SQL sorguları yazma konusunda çok rahatım.
- Oracle, MySQL, SQL Server gibi İlişkisel Veritabanları üzerinde çalışma deneyimim var
- İlişkisel olmayan veritabanlarıyla çalıştınız mı?
- Uygulamalı deneyime sahip değilim, ancak JSON formatı gibi olduğunu biliyorum
- Veritabanı ve JSON dosyalarıyla çalışma konusunda iyi deneyime sahibim.
- Ve hızlı öğrenen biriyim

51. Veritabanı test çerçevesi

- Manuel test için SQL sorguları üretmek için SQL geliştiricisini kullanıyorum
- OTOMASYON İÇİN; oracle veritabanından bir BAĞLANTI alıp sonra oluşturarak java'yı entegre etmek için JDBC kitaplığını kullanıyorum
  SQL sorgularını kullanan ve ardından verileri RESULTSET nesnesine depolayan STATEMENTS.

52. Üst veri nedir?

- Verilerle İlgili Meta Veri Verileri

```
ResultSetMetaData rsmd = rs.getMetaData ();
int columNum = rsmd.getColumnCount ();
```

53. Veri Yapıları ve Neden İhtiyacımız Var?

- Veri yapıları, verileri verimli bir şekilde düzenlemek için
- manipülasyon: Veri ekleme, arama, okuma, silme.
- Verileri okumak ve uygulamamızdan, veritabanımızdan veya API'den veri depolamak için her zaman java veri yapılarını kullanırım.

Sayfa 107

54. db'yi nasıl bağlayabiliriz?

```
Bağlantı bağlantısı = DriverManager.getConnection (URL, kullanıcı, parola);
İfade ifadesi = connection.createStatement ();
ResultSet resultset = deyim.executeQuery ("sorgu");

resultset.close ();
ifade.close ();
connection.close ();

// Bağlantıdan sonra;
DatabaseMetaData db = connection.getMetaData ();

// resultSet'ten sonra;
ResultSetMetaData rs = resultSet.getMetaData ();
```

- Bağlanamama bir istisna oluşturacaksa:
  - SQLException (hatalı URL veya kimlik bilgileri)
  - ClassNotFoundException (JDB sürücüsü sınıf yolunda değil)

55. JDBC için Bağımlılık

```
<bağımlılık>
    <groupId> oracle </groupId>
    <artifactId> ojdbc6 </artifactId>
    <version> 11.2.0.3 </version>
</dependency>
```

56. Prosedür nedir?

- Depolanan yordam, veritabanında oluşturulan ve depolanan bir grup SQL deyimidir.
- Depolanan bir prosedür, giriş parametrelerini kabul eder, böylece tek bir prosedür ağ üzerinde birkaç kişi tarafından kullanılabilir. farklı giriş verilerini kullanan istemciler.
- Depolanan bir prosedür, ağ trafiğini azaltacak ve performansı artıracaktır. Depolanan bir yordamı değiştirirsek, istemciler güncellenmiş saklı yordamı alacaktır. Depolanan yordam oluşturma örneği

```
PROSEDÜR OLUŞTUR test_display AS
SEC ad, soyadı DAN tb_test ;
EXEC test_display ;
```


Sayfa 108

57. SQL Kısıtlamaları nedir?

- SQL Kısıtlamaları, bir tabloya girebilecek veri türlerini sınırlamak, doğruluğu ve bütünlüğü korumak için kullanılan kurallardır Tablo içindeki verilerin
- Kısıtlamalar aşağıdaki iki türe ayrılabilir:
  - **Sütun düzeyi kısıtlamaları** : Yalnızca sütun verilerini sınırlar.
  - **Tablo düzeyinde kısıtlamalar** : Tüm tablo verilerini sınırlar.
- Kısıtlamalar, veri bütünlüğünün veritabanında muhafaza edilmesini sağlamak için kullanılır. Aşağıdakiler en çok kullanılanlardır bir tabloya uygulanabilecek kısıtlamalar.

- **NOT NULL** → kısıtlama bir sütunun **NULL** değerine sahip olmasını kısıtlar

```
CREATE TABLE Student (s_id int NOT NULL, Name varchar (60), Age int);
```

- **UNIQUE** → kısıtlama, bir alanın veya sütunun yalnızca **benzersiz değerlere** sahip olmasını sağlar **.**
  o Tablo oluştururken kısıtlama (Tablo seviyesi)

```
TABLO OLUŞTUR Öğrenci (s_id int NOT NULL UNIQUE, Name varchar (60), Age int);
```

  o Bir tablo oluşturduktan sonra kısıtlama (Sütun seviyesi)

```
DEĞİŞİKLİK TABLOSU Öğrenci ADD UNIQUE (s_id);
```

- **BİRİNCİL ANAHTAR** → kısıtlama, bir veritabanındaki her kaydı benzersiz şekilde tanımlar. Benzersiz ve boş değil
  o Tablo seviyesinde Kısıtlama

```
TABLO OLUŞTUR Öğrenci (s_id int PRIMARY KEY, Name varchar (60) NOT NULL, Age int);
```

  o Sütun seviyesinde Kısıtlama

```
DEĞİŞTİRME TABLOSU Öğrenci EKLE BİRİNCİL ANAHTAR (s_id) ;
```

- **YABANCI ANAHTAR** → iki tabloyu ilişkilendirmek için kullanılır. **FOREIGN KEY** kısıtlaması, aynı zamanda tablolar arasındaki bağlantıları yok edin. Aşağıdaki tabloların yardımıyla kullanımını görelim:

Müşteri_Detay Tablosu

c_id	Müşteri_adı adresi	
101	Adam	Noida
102	Alex	Delhi
103	Stuart	Rohtak

Sipariş_Detay Tablosu

Sipariş Kimliği	Order_Name	c_id
10	Sıra1	101
11	Sıra2	103
12	Sıra3	102

*Customer_Detail tablosunda **c_id** , Order_Detail tablosunda yabancı anahtar olarak ayarlanan birincil anahtardır. Herhangi bir yanlış veri eklemeye çalışırsanız, DBMS hata verir ve verileri eklemenize izin vermez .*

o Tablo Düzeyinde FOREIGN KEY kısıtlamasını kullanma

```
CREATE table Order_Detail (
    order_id int BİRİNCİL ANAHTAR,
    order_name varchar (60) NOT NULL,
    c_id int YABANCI ANAHTAR REFERANSLAR Customer_Detail (c_id)
) ;
```

Bu sorguda, Order_Detail tablosundaki c_id, yabancı anahtar olarak yapılır, bu da c_id sütununun referansıdır. Customer_Detail tablosu.

o Sütun Düzeyinde FOREIGN KEY kısıtlamasını kullanma

```
ALTER tablosu Order_Detail ADD YABANCI ANAHTAR (c_id) REFERANSLAR Müşteri_Ayrıntıları (c_id);
```


- **CHECK** → kısıtlama, bir aralık arasındaki bir sütunun değerini kısıtlamak için kullanılır. Değerleri kontrol eder, bunları veritabanına kaydetmeden önce. Verileri bir sütuna kaydetmeden önce durum kontrolü gibidir.

o Tablo Düzeyinde **CHECK** kısıtlamasını kullanma

```
Tablo Oluşturma Öğrenci (
    s_id int NULL DEĞİL KONTROL (s_id> 0),
    İsim varchar (60) NOT NULL,
    Yaş int
) ;
```

Yukarıdaki sorgu, s_id değerini sıfırdan büyük olacak şekilde kısıtlayacaktır.

o Sütun Düzeyinde **CHECK** kısıtlamasını kullanma

```
ALTER tablosu Öğrenci ADD CHECK (s_id> 0);
```

- **VARSAYILAN**

Sayfa 110

Git ve GitHub

1. GitHub nedir?

- Sürüm kontrol sistemi
- Dokümanların yeni / eski versiyonunu takip eder
- 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 ana bilgisayar (GITHUB) Değişikliklerimiz yerelden uzak depoya gider
- *Yerel Depo:* Genellikle bilgisayarınızda -Değişikliklerimiz burada Çalışma Dizini, dizin ve HEAD'den oluşur.

4. Git komutları nelerdir?

- **Ekle** : hazırlık alanına ekleyin
- **Kaydetme** : çalışma dizininden ve yerel depodan ekleyin
- **Gönder** : uzak depoya ekle
- **Pull** : değişiklikleri uzaktan çalışma dizinine alın
- url ile klonla: url'yi dizine kopyalar
- Git sürümü: size git sürümünü verir
- Git durumu: hangi dalda olduğunuzu ve izlenmeyen tüm değiştirilmiş dosyaları gösterir
  - Menşei: uzaktan kumanda adı
  - Master: şubenin adı
- Git ekle:
  - Evreleme alanına ekleme
  - Özyinelemeli ekleme
  - Her şeyi ekler
- **git commit -m** : "mesaj tüm dosyalar için geçerli olacak"
- **git push** : origin nameOfBranch
- **git ignore** :
  - Notepad.gitignore → Not defterinde, hazırlama alanına eklemek istemediğiniz dosyaları ekleyin
  - YOK ETMEK İSTEDİĞİNİZ DOSYALARIN SİPARİŞ EDİLMESİ İÇİN .GITIGNORE DOSYASINI BASMALISINIZ GIT'TE YOK EDİLDİ
  - Bazı dosyalar önemli değildir ve git'e gönderilmemelidir

• Dosyayı kaldır → GIT EKLE POM'U KALDIR → BUNU İŞLEMLE → VE İTME Kendi şubesini oluştur
• checkout branch -git → 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 commit"
git uzaktan kaynak ekle
https://github.com/Andylam224/SqlMentor.git git push -u kaynağı
usta
• mevcut bir repo-git'i aktar
○ git uzaktan kaynak ekle https://github.com/Andylam224/SqlMentor.git
git push -u kaynak yöneticisi
• Varsayılan düzenleyici

Sayfa 111

6. GIT Komutları?

git init
git ekle.
git commit -m "yorumum"
---------- ----------
git bilgisi
git günlüğü
git push -u kaynak yöneticisi
git itme
---------- ----------
git init
git uzaktan kaynak URL ekle // kopyala yapıştır https: // url'yi URL yerine
git add src / /// sadece bu klasörü eklemek istersem
git commit -m "yorumum"
git günlüğü
git push -u kaynak yöneticisi

7. En son sürüme mi dönüyorsunuz?

// ikisini de yazmamız gerekiyor
git getirme kaynağı
git sıfırlama - sabit kökeni / ana

8. Bir seferde çift dosya mı ekliyorsunuz?

git dosya1 dosya2 dosya3 ekle //

9. GIT Şube şubeleri?

git rm file.java
git commit -m "kaldırma"
git itme kaynağı yöneticisi

git şube BranchName : - Dal oluşturma
git şubesi : - şube yöneticisini kontrol etme
git checkout BranchName : - ad, geçiş yapmak istediğiniz şube adıdır
git şube -d BranchName : - yerelde ayraç siliniyor
git itme kaynağı: deleteBranchName : - Uzak (gitHub WebSite) yerel (intellij) üzerinde şube deteled siliniyor
git şube -a : - Yerelde silinmiş olsa bile tüm şubeleri kontrol etme (ancak uzaktan değil)
git checkout -b BranchName : - Dal oluşturma ve yeni şubeye geçiş
git merge BranchName : - Birleştirme şubesi
git push --set-upstream origin BranchName: - Şubeyi yerelden uzak (gitHub WebSite) (intellij) itme
git kaynak BranchName getir : - Uzaktan yerel (intellij) şubeye çekme (github WebSite)

git itme kaynağı branch1: branch2
git çekme kaynağı branch1: branch2

Sayfa 112

10. Şubeyi ana ile birleştirme

--ikinci şubenize gidin sonraki adımları uygulayın

git ekle.

git commit -m "yorumunuz"

- ana şubenize gidin

git merge "branchName"

- eğer birleşmiyorsa git commit -m "yorum" u ana şubeden tekrar yapmalıyız

11. Kod GIT Komutlarındaki Değişikliklerle GitHub deposundan Yerel ana sunucuya yeni Branch birleştiriliyor mu?

git kaynak BranchName getir
git checkout BranchName
git şubesi // yeni şubede olmalısın
git ödeme yöneticisi
git şubesi // ana dalda olmalısın
git merge BranchName // birleşecek. herhangi bir çatışma varsa düzeltmeniz gerekiyorsa, birleştirme çatışmanız yoksa geçecek

git şube -a // hem yerel hem de uzak dalları görebiliyoruz
klonlamak istediğiniz şeyin git clone URL'si // ve bağlantıyı kopyaladıktan sonra
git getir // ctobi obnovit obnovleniya v glavnom
git birleştirme
git log - grafik // neyin taahhüt edildiğini ve ne olduğunu göstermek
git log --graph - çevrimiçi // neler olduğunu tek satırda gösterme
- Çakışmanız varsa projeye gidin sağ tıklayın -> git -> çakışmayı çöz -> birleştir:
wq ve kaç

12. GITHUB URL'SINI KONTROL EDİN?

git uzaktan -v
git config --get remote.origin.url
git uzak gösteri kaynağı
git config --get remote.origin.url

13. Çekme talebi nedir?

git merge fetch_head --allow-unrelated-geçmişleri

- İleri sarma sorunu olmayan bir sorunu çözdü
- Escape tuşuna ve ardından
  - ": x!" → Kaydeder ve çıkar
  - ": q!" → Kaydetme ve çıkma yok

14. Çekme talebi nedir?

- Git merge fetch_head --allow-unrelated-geçmişleri
  - İleri sarma olmayan bir sorunu çözme sorunu çözüldü

15. Git'teki çatışmayı nasıl çözersiniz?

- deponuz → cd ~ / <repo_directory>
- Son sürüm deposunu çek → git çekme
- Kaynak şubeye göz atın → git checkout <feature_branch>
- Hedef dalı kaynak dala çekin → git çekme orijini <hedef_branch>
- Çakışmaları düzeltin ve ardından sonucu kesin.

Sayfa 113

16. Git ve SVN komutları

Git-Subversion komutlarının karşılaştırma tablosu

Komut	Operasyon	Yıkım
git klon	Depoyu kopyala	**svn** ödeme
git commit	Dosya geçmişindeki değişiklikleri kaydedin	**svn** commit
git göster	Kaydetme ayrıntılarını görüntüleyin	**svn** kedi
git durumu	Durumu onaylayın	**svn** durumu

git fark	Farklılıkları kontrol edin	**svn** fark
git günlüğü	Günlüğü kontrol et	**svn** günlüğü
git ekle	İlave	**svn** ekle
git mv	Hareket	**svn** mv
git rm	Silme	**svn** rm
git checkout	Değişikliği iptal et	**svn** geri döndürme [1]
git sıfırla	Değişikliği iptal et	**svn** geri döndürme [1]
git şubesi	Bir şube yap	**svn** kopyası [2]
git checkout	Şube değiştir	**svn** anahtarı
git birleştirme	Birleştirmek	**svn** birleştirme
git etiketi	Bir etiket oluşturun	**svn** kopyası [2]
git çekme	Güncelleme	**svn** güncellemesi
git getir	Güncelleme	**svn** güncellemesi
git itme	Uzaktan kumandaya yansıtılır	**svn** commit [3]
git görmezden gel	Dosya listesini yoksay	**svn** görmezden gel

1. SVN'de geri dönme, değişikliğin iptalidir, ancak Git'te Geri Döndür, olumsuzlamanın kesinleştirilmesidir. Revert'in anlamları farklıdır.
2. Dal ve etiket SVN'deki yapıda aynıdır, ancak Git'te açıkça farklıdırlar
3. SVN, yerel depo / uzak depo kavramına sahip değildir, bu nedenle commit doğrudan uzaktaki depoya yansıtılır.
   Ancak Git, yerel depoya yansıtma ve uzak depoya yansıtma için farklı yansıtma yöntemlerine sahiptir.

Sayfa 114

Sayfa 115

JIRA

1. JIRA nedir?

- Proje yönetimi aracı ve kusurların izlenmesine yardımcı olur
  - Planlama ve zaman yönetimine izin verir
  - Son tarihleri / ödevleri takip eder
- Yalnızca iş yığını ve aktif sprintlerdeki test cihazı

2. Hata izleme için hangi araçları kullanıyorsunuz?

- JIRA, içindeki tüm çalışmaları bir Sorun olarak ele alır
- Dolayısıyla, JIRA'da bir kusur yaratmak, "Hata" türünde bir sorun yaratmak olacaktır.
- Hata bildirimi:
  - Kusur Kimliği
  - Kusur başlığı
  - Kusur açıklaması (yeniden oluşturma adımları)
  - Çevre bilgisi
  - Ekran görüntüsü (ek)
  - Önem
  - Geliştiriciye atayın

3. Aktif Sprint Board nedir?

- *İş Akışı:* Yapılacaklar> Devam Ediyor (burada da engellenebilir)> İncele (Teknik lider, şu adrese geçmeden önce kodu gözden geçirin) bitti> Bitti
- *Engellendi:* Hikayeye gidin ve seçeneklere tıklayın ve daha fazla seçeneğe tıklayın> engellendi
  - Neden engellendiğine dair bir yorum yazın
  - Scrum master, ASAP ile ilgilenmek zorunda kalacak
  - Bir günden fazla blokta hiçbir şey kalmamalıdır
  - Düzeltildikten sonra engelleyici çözüldü olarak değiştirebilirsiniz

4. Jira şartları nelerdir?

- Sorun → Yapmanız ve düzeltmeniz gerekiyor
- Sorun Türleri
  - Hikaye
  - Görev
  - Hata
  - Destansı

5. Destan ile biletler arasındaki fark nedir?

- Epic, BA tarafından yazılır, biletler test kullanıcıları tarafından oluşturulur
- Açıklama kutusu
  - Hata bildirme örneği
  - Kutuya yazarsın
    - Hata neyle ilgili?
    - Hangi işlevsellik bozuluyor
    - Hatayı yeniden oluşturma adımları nelerdir (gerekli verilerle)
    - Hatanın raporunu ve ekran görüntüsünü ekleyin

- Beklenen sonuçlar
- Gerçek sonuçlar

Sayfa 116

6. JIRA'dan Kullanıcı Hikayelerini nasıl otomatikleştiriyorsunuz?

- Açıklamaya bakın - Çevik hikaye
- Özellik dosyası oluşturun ve dosyayı Jira story.feature olarak kaydedin
  - Kabul kriterlerinde bulunan senaryo ekleyin
- cukesRunner'ı dryRun = true ile çalıştırın
- Yöntemleri uygulayın
- JIRA'DAKİ TEST DURUMLARINI OTOMATİKLEŞTİRMEDEN ÖNCE HER ZAMAN ÖNCE ELLE TEST EDİN

7. Selenium'u Jira ile nasıl bütünleştirirsiniz?

- 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)

Sayfa 117

JENKINS

1. JENKINS nedir?

- Sürekli Entegrasyon ve Dağıtım aracı. derlemeleri, dağıtımları vb. planlamak ve otomatikleştirmek için kullanılır.
  geliştiriciler ve testçiler tarafından.
- Jenkins'in 3 bileşeni
  - 1. Kod değişikliği → Devs, uygulama kodunda değişiklik yapar

- 2. Test → CI aracı değişiklikleri otomatik olarak alır ve uygulamayı test eder
- 3. Dağıt → CI aracı, uygulamayı değişikliklerle birlikte dağıtır

2. Jenkins işi nedir?

- Her şey bir iş oluşturarak yapılır
- Jenkins'in programına göre gerçekleştirdiği bir görev
- Birkaç adımdan oluşur
- Ne zaman çalışacağını belirleyen bir tetikleyiciye sahip olabilir
- Çalıştırmanın sonuçlarını otomatik olarak raporlar

3. Sürekli Entegrasyon nedir?

- CI, geliştiricilerin kodu paylaşılan bir havuza günde birkaç kez 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 sürüm için hazırlanır
- Daha sonra her kontrol, ekiplerin sorunları erken tespit etmesine olanak tanıyan otomatik bir yapı tarafından doğrulanır.

5. Jenkins'in bakımını yapıyor musunuz?

- Özel Ekip -DevOps Ekibi-, çevre ekibi, mimarlık ekibi tarafından yapılır.
- Ancak onlara testlerim ve yapılandırma bilgilerimin bilgilerini ve ayrıca bildirim göndermek için e-postaları da sağlıyorum.
  - Git yolu
  - Mvn kodu; hedefler - derleyin veya doğrulayın -drunner = xml, vb.
  - Belirli testler için zaman çizelgesi

6. Jenkins'i kim kurar?

- Şirketimde, jenkinlerin bakımından sorumlu DevOps / Operasyon Destek ekibimiz var.
- Yüklenen ve yapılandırılan jenkinler. dağıtımlar vb. için geliştiricilerle birlikte çalışarak inşa etmek ve uygulamaları dağıtmak.
- Duman testleri oluşturmak için otomasyon mühendisleriyle birlikte çalışırlar.

7. Jenkins'te Duman Testlerini kim kurar?

- Bir otomasyon mühendisi olarak duman testlerimi oluşturmak için devop'lar / operasyon destekleri ile çalışıyorum.
- Testlerimi çalıştırmak için, Jenkins'te belirli eklentilerin kurulu olması gerekiyor, ayrıca java'ya, jenkins üzerinde yapılandırılmış maven'e ihtiyacım var, ayrıca testlerin çalışacağı sunucuda yüklü tarayıcılar.
- Projemde, yalnızca devops / operasyon destek ekibi üyeleri yukarıdaki yapılandırmaları yapma hakkına sahip. bu yüzden yapmalıyım onlarla çalışın.
- Jenkins'in yapılandırması tamamlandığında duman testi işini oluşturabilir ve çalıştırabilirim.

Sayfa 118

8. Jenkins üzerinde duman testleriniz nasıl yapılandırılıyor?

- Jenkins işimiz, otomatik testlerimi GitHub'dan alacak ve Maven kullanarak her gün çalıştırılacak şekilde yapılandırıldı.
- Yapılandırmada önce yeni bir iş oluşturdum ve " ***duman testleri*** " adını verdim .
- Daha sonra Kaynak Kod Yönetimi bölümünde git seçeneğini seçtim ve GitHub'da çerçevemin yolunu girdim. Ve, git kimlik bilgilerini de girdi.
- Sonraki yapılandırma, periyodik olarak oluştur seçeneğini belirlediğim ve çalıştırmak istediğim zamanı girdiğim derleme tetikleyicileri hakkındadır testlerim.
- Build bölümünde, otomasyon çerçevem maven kullanılarak oluşturulduğundan, üst düzey maven komutunu çağır seçeneğini seçiyorum.
- Bölümde maven komutunu giriyorum (mvn kısmı olmadan).
- Bu nedenle, ***testimi*** terminal kullanarak çalıştırmak için normal komutu ***'mvn test'*** , ***jenkins'e*** sadece ***'test'*** giriyorum .
- Buraya komut satırını kullanarak çalıştırmak istediğim etiketi de giriyorum.

```
test -Dcucumber.options = "- etiketler @smoke"
```

- regresyon testleri çalıştırmak istersem, komut şu şekildedir:

```
test -Dcucumber.options = "- etiketler @regression"
```

- Derleme Sonrası İşlemlerde, çift yapılandırma yapıyorum,
  1. Salatalık raporları seçeneğini ekliyorum. Bu, jenkinsimize salatalık html raporları eklentisi yüklediğimiz için kullanılabilir. Salatalık raporları eklentisi, her derleme için html raporu oluşturacaktır. Jenkins üzerinde her duman testi yaptığımızda, yeni bir bildiri. ve tüm derleme için tüm raporlar kaydedilir.
  2. Sonraki Oluşturma Sonrası Eylemler olarak e-posta seçeneğini ekliyorum. Burada bunu, Agile ekibimdeki herkesin Test sonuçları.

9. Regresyon testinizi jenkins üzerinde çalıştırırsanız, otomatik olarak çalışırsa nasıl **çalıştırırsınız** ?

10. Regresyon testinde neler var?

- Ekibe, test planına, kapsamına ve iş değerine bağlıdır

11. Jenkins'te kaç iş var? Jenkins üzerinde hangi testlerin var? Şirketinizde jenkins test için nasıl kullanılıyor?

- Otomatik testler için kişisel olarak 2-3 iş ayarladım
  - 1 duman için → Duman günde 2,3 kez akıyor ve tüm ortamların çalışır durumda olmasını sağlıyor
  - Tam regresyon (manuel ve otomatik testler çalıştırılıyor)?
    - Her üretim sürümünden önce (3 sprintten sonra)
    - Yalnızca çok kararlı test durumları tam regresyonda saklanır
    - İşlevselliğin güncellenmesi
  - Küçük gerileme
    - Sprint sonunda çalışır
    - Belirli modüller ve işlevlerle ilgili testler
    - Hangi modülün çalıştırılacağını belirtmek için etiketleri kullanıyorum

12. Jenkinlerde ne tür testler yapılabilir?

- Jenkins herhangi bir otomatik testi çalıştırabilir.
- Örneğin, sahip olabiliriz
  - birim testleri,
  - duman testleri,
  - entegrasyon testleri,
  - regresyon testleri,
  - akıl sağlığı

Sayfa 119

13. Jenkins kullanılarak hangi test katmanları test edilebilir?

- Uygulamanın farklı katmanlarının test edilmesi test kodumuz ile yapılır.
- Jenkins, kullanıcı arayüzünü veya veritabanını veya API'yi test etmemizi umursamıyor. sadece testleri başlatır ve raporlar gönderir.
- Dolayısıyla, otomatik testim bir UI testiyse, bu, Jenkins'in UI testleri çalıştırdığı anlamına gelir. veya bu bir API testiyse, Jenkins API testleri çalıştırıyor.

14. Kod ortamınıza nasıl dağıtılır?

- Devs kodu yazar, test eder, ardından geliştirmeden test ortamına jenkins'e dağıtılır
- **Ya olmazsa?**
  - Geliştiricinizle konuşun ve ondan onu dağıtmasını isteyin

15. Etiketlere göre nasıl arama yaparsınız?

- Arama aracını Eclipse'de açmak için **ctrl + H tuşlarını** kullanabilir ve etiket adını oraya koyabilirsiniz ve

16. Jenkins'te bir derlemeyi nasıl planlar?

- Jenkins'te, iş yapılandırması altında çeşitli derleme tetikleyicileri tanımlayabiliriz.
- Basitçe "Tetikleyicileri Oluştur" bölümünü bulun ve "Düzenli Olarak Oluştur" onay kutusunu işaretleyin.
- Periyodik derleme ile derleme tanımını tarihe veya haftanın gününe ve yürütme saatine göre planlayabilirsiniz.
  yapı.
- 'Plan' metin kutusunun biçimi aşağıdaki gibidir:
  - DAKİKA (0-59), SAAT (0-23), GÜN (1-31), AY (1-12), HAFTANIN GÜNÜ (0-7)
- Jenkins'te nasıl planlama yaparsınız? Testin her 3 saatte bir uygulanmasını nasıl planlayacaksınız?
  - H 3 *** Yapınızı her gün saat 7: 00'da planlamak istiyorsanız, şu işi yapacaktır: 0 7 * * *

17. Jenkins'te derleme işlem hattı nedir?

- Jenkins'teki iş zinciri, bir işin yürütülmesinden sonra diğer işleri otomatik olarak başlatma sürecidir.
- Bu yaklaşım, çok adımlı derleme ardışık düzenleri oluşturmanıza veya bağımlılıklarından biri ise bir projenin yeniden oluşturulmasını tetiklemenize olanak tanır.
  güncellenmiş.

18. Jenkins üzerindeki duman testi işinizi nasıl sürdürüyorsunuz?

- İşi, kodu git'ten alacak, bir mvn komutu ileterek testleri çalıştıracak, raporlar oluşturacak ve gönderecek şekilde yapılandırdım
  e-posta. İş yapılandırmasına gidip değiştirmiyorum. Bir kez kurulduktan sonra her zaman çalışır.

19. JENKINS SALATALIK RAPORU

- Jenkins, Kullanıcı Dostu rapor oluşturabilen Salatalık rapor eklentisine sahiptir
- Yalnızca Hıyar raporu eklentisinin ihtiyaç duyduğu veri Json formatlı rapordur
- Salatalık raporu oluşturulduktan sonra, raporun URL'sini raporu isteyen herkese gönderin
- Otomasyon testlerinin tüm geçmişi Jenkins'te saklanır. Tarihleri ve saatleri ve diğer ayrıntıları gösterir.
- Geçmişteki otomasyon raporunu görmeniz gerekiyorsa, onu Jenkins'te bulabilirsiniz.

20. Jenkins İşini Planlama

- CRON Job → Planlanmış otomatik görev
  - Jenkins kullanıcılarının CRON saat formatını kullanmalarının nedeni budur.

Sayfa 120

SELENIUM IZGARA & Sosluklar

1. Selenium Grid nedir?

- Otomatik testlerinizi farklı tarayıcılarda (ve bunların farklı sürümlerinde) ve platformlarda çalıştırmanıza olanak tanır (temelde İşletim sistemleri ve sürümleri. Window, Linux, Mac) VISGRID
- Bu araç, çok sayıda Testiniz varsa (500'ün üzerinde) kullanışlıdır
- Yapmak yerine

```
WebDriver sürücüsü = yeni Chromedriver ()
```

- Do

```
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 çalıştırmak için kullanılabilir, böylece dağıtılmış test yürütmesi elde etmek için

3. Selenium Grid nasıl çalışır?

- Izgara, Hub ve düğümlerden oluşan bir kurulumdur
- Hub, tüm düğümlerin bağlandığı merkezi bir makinedir
  - IP adresi ve bağlantı noktası numarası vardır, Ardından Hub'ı Düğümlere bağlarsınız
- Hub olarak adlandırılan bir ana makine ve birden çok düğüm var (testlerinizi gerçekten çalıştıran makineler)
- İcra emri;
  - Kodunuz> uzak sürücü> Selenium Hub> Selenium düğümleri (birden fazla olabilir)
    - Selenium Hub'ınızdan testlerinizi çalıştırmasını istediniz
      - Daha sonra selenyum hub, hub'a bağlı bir düğüm bulacak ve oradan testinizi çalıştıracaktır.
    - İstediğiniz kadar düğüme sahip olabilirsiniz, ancak yalnızca bir hub

4. Selenyum ızgaranızı nasıl kurarsınız?

- Hub farklı bir sunucu makinesidir ve her düğüm ayrı bir sunucu makinesidir
- Hub ve düğümlerimiz Amazon AWS Ec2 makinelerinde kurulacak (ideal olarak)
- Hub'ınızdan testleri çalıştırmasını isteyebilirsiniz, ardından hub bir düğüm bulur ve oradan testinizi çalıştırır
- Aynı makinede veya sunucuda hub ve birden fazla düğüme de sahip olabiliriz
- Her düğüm, belirli bir yapılandırmayla HUB'a kaydolur ve HUB, düğümde bulunan tarayıcının farkındadır
- HUB'a belirli bir tarayıcı (İstenen yetenekler nesnesi ile) için bir istek geldiğinde, HUB, istenen tarayıcı, çağrıyı * o * belirli GRID Düğümüne yönlendirir ve ardından çift yönlü olarak bir oturum kurulur ve infaz başlar

5. Çapraz tarayıcı / platform nerede çalışıyor?

- Jenkins'in kurulu olduğu Amazon AWS makinesinde çalışıyor. Ancak normalde şirket jenkins'i geliştiriciler tarafından kullanılır. ekip, devops, dağıtım ekibi ve QA otomasyon ekibi
  - Orada tavsiye edilmez
- İdeal olan, Jenkins> GitHub> Maven> Runner sınıfı> Selenium Hub'dır ve farklı şekilde yapılandırılmış düğümlerden birinde çalışır. sunucu
  - Hooks sınıfınızda, webDriver'ın yerel bir sürücüyü göstermesini sağlamak yerine, bunu bir RemoteDriver () olarak değiştirin ve hub'a sahip bulut makinesini işaret edin

Sayfa 121

6. Çoklu tarayıcı testindeki zorluklar nelerdir?

- Bir şey tıklanmıyor
- Görünmez

- Bazı öğeler bir tarayıcıda başka bir tarayıcıdan farklı görünüyor

7. Hangi tarayıcıları test ediyorsunuz?

- Chrome - Firefox - IE / Edge - Safari - Opera

8. Selenium Grid'de hub nedir?

- Hub, farklı makinelerdeki test yürütmelerini kontrol eden bir 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 otomatik hale getirirsiniz?

- Tarayıcıyı çerçevemdeki özellikler dosyamda başka bir şeye değiştirin
  - Testlerimi farklı bir tarayıcıda çalıştırmak istediğimde
  - Örn; "Tarayıcı = chrome" dan "Internet Explorer" a
  - Bu yöntem, testleriniz 500 testten azsa işe yarar
- Çerçevemde Selenium Grid'i uyguladım ve farklı tarayıcılar kullanarak farklı bulut makinelerinde testler çalıştırabiliyorum

11. Başarısız testlerle ne yapılmalı?

- Otomasyon yürütme raporuna bakın
- Başarısızlığın nedenini öğrenin
- Adımları manuel olarak yapmayı deneyin,
  - Manuel geçerse, otomasyon sorunu → böylece düzeltin, yeniden çalıştırın ve geçip geçmediğini görün
  - Uygulama sorunu varsa
    - Bir kusur oluşturun
      - Hata giderilirken, Ad-hoc testi kullanarak manuel olarak test yapıyorum
  - Kusur bir engelleyici değilse, diğer testleri çalıştırır ve
    - Eğer öyleyse, beklemelisiniz, 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 gerçekleştirilir
- Bu senaryolar için hizalanmış test senaryoları olmadığından, bu yöntem kullanılarak bulunan kusurların kopyalanması zordur
- Resmi testin yürütülmesinden sonra gerçekleştirilir

13. SauceLabs - bulut Izgara hizmeti. Birden çok tarayıcı ile birden çok Platforma erişim

- Bulut makinesi sağlar, böylece çok sayıda
- Paralel testler iyi sonuç verir

14. SauceLabs'ı kullanarak nasıl bildirimde bulunursunuz?

- JIRA sunucuma bağlanıyorum

Sayfa 122

15. SauceLabs ile nasıl test edilir?

- Genellikle, "WebDriver sürücüsü = yeni FireFoxDriver ();
- Şimdi yapacağız;

```
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 ps)
```

16. Testlerinizi soslu laboratuvarlarda / Hazır herhangi bir selenyum ızgarasında çalıştırma adımları

- İstenenCapabilities nesnesini oluşturun ve testlerinizin selenyum 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);
```

- SauLabsDemo sınıfı oluşturun

```
// @BeforeTest içinde
// DesiredCapabilities (Selenium'dan gelir)
caps = DesiredCapabilities.firefox (); // (hangi tarayıcıyı seçer)
```

```
caps.setCapability ("platform", "Windows 7");
caps.setCapability ("sürüm", "38.0");
```

- Dize URL'si = ” http: // YOUR_USERNAME: YOUR_ACCESS_@ondemand.saucelabds.com: 80 / wd / hub ”
  - Bu Selenium Hub adresidir
  - **URI** = birleştirilmiş kaynak tanımlayıcı
  - **URL** = birleşik kaynak bulucu

17. Testlerinizi birden çok aşamada paralel olarak nasıl çalıştırırsınız?

- İş parçacığı, çalıştırılan uygulamanın bir işlemi veya örneği gibidir
- 4 yollu
  - 1. Farklı etiketlerle birden fazla cukesrunner oluşturabiliriz
    - Örn. cukerunner'ın "@Test" etiketi var
    - Smokerunner'da @smoke var
    - Regresyon koşucusunda @Regression var
  - 2. testng xml oluşturun ve bu koşucu sınıfını tek bir testin altına ekleyin
    - 3 koşucunun tümünü bir xml'ye ekleyin
    - Ardından, verbose = 2'nin yanına (xml dosyasının üstüne) “parallel =” classes ”thread-count =” 10 ”ekleyin>
  - 3. Ardından, sürücümüzün hub'ı gösteren uzak bir WebDriver'ı açtığından emin olun
    - Kodu sürücü sınıfına ekleyin
  - 4. Testng xml dosyasını tek başına veya maven kullanarak çalıştırın.

Sayfa 123

18. Selenium Grid'i AWS'de 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 kullanın
      - HUB'ı kurun;

        java -jar selenium-server-standaline-3.5.3.jar - rol hub http: // localhost: 444 / grid / console
      - Düğüm1'i kurun

        java -jar selenyum-sunucu-bağımsız-3.5.3.jar -role düğümü - hub http: // locahost: 4444 / grid / register
      - Düğüm2

        java -jar selenyum-sunucu-standaline-3.5.3.jar -role düğümü -hub http: // localhost: 4444 / gird / register -port 7777
- (3) Visgrid'i de kullanabilirsiniz
  - Makine 1'de indir
  - Jar dosyasını açın
    - Maksimum oturumu 10 olarak ayarlayın
    - Hub'ı başlat
    - 4444 numaralı bağlantı noktası (hub'ınızın olduğu yer. Değiştirebilirsiniz ancak unutmayın)
  - Düğüm oluştur'a tıklayın
    - Düğüm için bir tarayıcı seçin
    - Örnek sayısını yazın
    - Ekle'yi tıklayın
      - ec2 makinesinde tarayıcı açın
      - Tür: localhost: 4444 (bu seleniumGrid merkezidir) (Şimdi HUB'ımıza güç veriyoruz)
      - Konsolu tıklayın
      - Sayfayı yenileyin
      - Şimdi düğümleri gösteriyor (hepsi aynı makinede)
      - Başka bir düğüm ekleyin (şimdi 2 düğüm gösterecek)
  - Şimdi başka bir makineye gidin ve oradan düğümler oluşturun ve onu HUB'a bağlayın
  - Makineye gitmeden önce 2
    - aws konsoluna gidin
    - Hub'ınızı tutan örneğe gidin
      - Güvenlik gruplarına tıklayın = sihirbazı başlat
      - Gelen aramaya git
        - Hub port numarasını (4444 olan) ekleyin ve kaydedin

- 2. makineye git
  - 1. makine için kullandığınız Visgrid'in aynı jar dosyasını indirin
  - jdk'yi indirin (dosyayı açmak için)
  - jar> hub'ı başlat> düğüm oluştur> HUB'ı Geçersiz Kıl'ı tıklayın (bu hub'a ihtiyacımız yok çünkü ilk makine)> makine 1 ip adresi ve bağlantı noktası numarası ekleyin: 4444> ekle

Sayfa 124

19. Linux Komutları (büyük / küçük harfe duyarlı)

- yeniden başlat → sistemi yeniden başlatır
- adam → size komutun talimatını verir - Örn: "man reboot"
- mikdir → Dizin (klasör) oluşturur
- cd → Dizini değiştir
- Ls → Dizin içeriğini listeleyin
- pwd → Mevcut çalışma dizininin adını yazdırın. Size tam konumu verir; Örn: / home / Andy / Desktop
- ll → Uzun liste biçimi
- ls-la → Dosyaları ve gizli dosyayı yazdırır
- temizle → Ekranı temizle
- cd .. → Ana dosyaya gider (kök dosyaya değil)
- cd / → Ana kök dosyasına 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)

- **Nasıl hesap oluşturulur?**
  - Kullanıcı → useradd Andy
  - Grup → groupadd Cybertek

- **Gruba bir kullanıcı ekleme**
  - useradd -G Cybertek Andy
  - Andy kimliği → bu kişinin ayrıntılarını yazdırır (Andy'de cybertek olduğunu gösterir)

- **Yapılandırma / ağı değiştirme**
  - vi / etc / sysconfig / network
  - crontab → Dosyanızın çalışması için bir zamanlayıcı ayarlar (jenkins gibi program oluşturun)

- **İzinleri ayarlama**
  - chmod → Dosya modu bitlerini değiştir
    - Sipariş sahip, grup ve diğerleri
      - Dosya klasörse, d ön taraftadır
      - Klasör değilse d yok
  - chmod 777 → Sahip, grup ve diğerlerine erişim sağlar; Çok tehlikeli; KULLANMAYIN
    - r- oku
    - W-yaz
    - X-execute
    - rwxrwxrwx (777)
  - chmod 644 → Sahibine (okuma ve yazma), gruba (salt okunur), diğerlerine (salt okunur) erişim izni verin
    - -rw-r - r--
    - Varsayılan erişim ve Standart
- grep kullanarak bir dosya nasıl bulunur (dosyanın adını biliyorsanız)
  - grep 'test başarısız dosyanın adı' / home / Andy / Test1 / TestScenario (konum)> / home / Alex / AutomationFile
    - Artık dosya bu konumdadır; / home / Alex / AutomationFile
  - 'Adı' olan herhangi bir dosyayı bulun
  - grep 'başarısızlık senaryosu' *

Sayfa 125

AWS

1. AWS ile mi çalışıyorsunuz?

- EC2 bulut sunucularıyla çalışıyorum.
- Temel olarak, bu benim sanal makinelerim.
- Selenium Grid'e sahip olduğumda, farklı sanal makinelerim oluyor ve her makinem ayrı çalışıyor.
- Örneğin, regresyon testleri için gereken zamanı en aza indirmek için gerçekten etkilidir, şirketimize çok zaman kazandırır.

2. AWS nedir?

- AWS, bulut sanal makinesi sağlıyor. Bir EC2 örneği oluşturun.
- Bu örneği uzak masaüstü ile kullanabilirim. Aslında, öğle yemeğimi yedikten sonra, normal bir bilgisayar gibi kullanıyorum.

3. Temel sayfa nedir?

- Ortak işlevlerimizi bir temel sınıfta depolar ve daha sonra bu temel sınıfı genişletir ve diğer sınıfta kullanırız.

Sayfa 126

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

- Selenium Java
- Salatalık Sandviç
- <özellikler> içinde JRE Sistem kitaplığı 1.8
- TestNG
- WebDriverManager_BoniGarcia

- Salatalık Java
- Salatalık TestiNG
- Apache POI.XML
- APACHE POI
- JDBC
- Huzurlu
- Gson
- Log4j

4. Çerçeve Yapısı (paketler) oluşturun

- Sayfalar
  - ben. Web öğeleri ve yöntemler
- POJO / Fasulye
  - ben. Özel sınıflar
- Koşucu
  - ben. Cuckesrunner - kodlar üretir ve html raporunu hedefe kaydeder
  - ii. Sigara içmek
  - iii. Regresyon
- StepDefinitions
  - ben. Gerçek kodlar ve kanca sınıfı
- Testler
  - ben. Veriye dayalı testler
- JDBC
- API
- Yardımcı Programlar
  - ben. ConfigurationReader
  - ii. Sürücü sınıfı (Singleton)
  - iii. browserUtils
  - iv. DBUtils
  - v. ApiUtils
- Configuration.properties
- Testng_runner.xml
  - ben. Runner sınıfından biri Paketleri com.app.utilities ile çağırın

Sayfa 127

5. Kaynaklarda unsur dosyası (.feature) ile unsur klasörü oluşturun

- Kornişon bir dilde yazılmış yürütülebilir dosya özelliği

6. Senaryonuzu yazmayı bitirdikten sonra , cukesRunner'ınızı dryRun = false ile çalıştırın, bu size 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'yi kopyalayın ve tutulmaya gidin
- git deposunu yapılandırın ve url'yi ekleyin
- Projeyi sağ tıklayın, ekibi tıklayın, tamamlamayı tıklayın ve git aşamasına geçeceksiniz
- ARTIK KODUNUZ GITHUB'DA

9. Sırada JENKINS ENTEGRASYONU var

- Jenkins
  - ben. Açık kaynak otomasyon sunucusu
  - ii. Yazılım geliştirme sürecinin insan dışı kısmını otomatikleştirmeye yardımcı olur
  - iii. Sürekli entegrasyona izin verir
  - iv. Düzen aralıklarında kodu paylaşılan bir depoya entegre etmek için geliştirici gerektiren geliştirme uygulaması
  - v. Bağlantı noktası 8081 localhost'tur
- Projenizi jenkins'te çalıştırmak için
  - ben. Jenkins hesabına giriş yapın
  - ii. Proje oluştur - serbest stil
  - iii. Eklentileri yükleyin - salatalık raporu ve git
  - iv. Kaynak kodu yönetimi altında git ve geçmiş git url'sini seçin
  - v. Tetikleyici oluştur - düzenli aralıklarla oluşturmayı seçin
  - vi. Üst düzey maven'i çağırın
    1. Maven sürümü; 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 günlüğünü ekleyin; derleme günlüğünü seçin
2. Gelişmiş ayarları tıklayın

ix. Arıza-Herhangi
1. Gelişmiş'i tıklayın
2. Alıcı listesi - raporu alacak e-posta adresi. Birden çoksa virgül ekleyin
3. Her zaman hata gibi tetikleyici ekle'yi tıklayın
4. Derleme günlüğünü ekleyin; derleme günlüğünü ekle'yi seçin
5. Kaydet

x. Son adım
1. Şimdi oluştur'a tıklayın ve test çalışacak ve salatalık raporunuzu verecektir

Sayfa 128

Yapımdan sonra salatalık raporu hakkında konuşun

1. Sayfa Nesne Modeli nedir

a. Kod fazlalığını azaltır ve kodu düzenler
b. Öğeleri tanımlamaya ve bir sayfa nesnesi değişkeni olarak depolamaya yardımcı olur
c. Saklandığı yere bağlayabilirsiniz
d. PageFactory tasarım deseni eklendi

2. Otomasyon aracım olarak Selenium WebDriver

e. Önce manuel olarak test edin:
ben. Başlangıç aşaması
- İşlevsel test

ii. Arka uç
- Veritabanı - SQL Developer IDE
- API - Postacı

f. Selenium'u aşağıdakilerle entegre edin:
ben. Uzman
- Paketi test edin
- Yardımcı program paketi
  a. UI
  b. DB
  c. API
- Yapılandırma dosyası
  a. Özellikleri
- Sürücü sınıfı
  a. Singleton tasarım deseni
    a. Özel bir kurucuya sahip olmak

ii. Salatalık BDD
- BDD süreci sırasında işbirliğini kolaylaştırın
- Hikayeyi ve kabul kriterlerini kolay bir dille açıklamayı sağlar.

iii. Git - kaynak denetimi
iv. Jenkins
v. Java
- Koleksiyon Çerçevesi
- Apache İÇN
- JDBC
- Huzurlu

3. Davranış Odaklı Geliştirme

- Müşteri için doğru standartları karşıladığından emin olmak için müşteri ile birlikte gelişmek

4. Veriye Dayalı Geliştirme

- Farklı veri kümelerine karşı aynı test senaryosunun yürütülmesi
- Test akışı verilere göre değişmemelidir

5. Salatalık raporlama

- Hedef klasör
- Jenkins

Sayfa 129

Çerçevenizi ve Testlerinizi tanımlayın

ÇERÇEVEM ve görüşmeci için nasıl açıklanacağı - Andy Lam ve biraz Alex

// **Veriye Dayalı ve Davranış Odaklı** - Karma çerçeveye dayalı

// **Maven** - yalnızca bağımlılık yönetimi için değil, aynı zamanda pom xml dosyasını kullanarak bir komut istemi aracı olarak oluşturma aracı, I ayrıca **dumanımı** 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 Koleksiyonu çerçevesini** kullanıyorum veri ve karşılaştır

// Ayrıca hassas / yeniden kullanılabilir verileri depolayan bir **özellikler** dosyam var - URL, şifre, tarayıcı

// ve hataları bulmak için verileri java olarak biçimlendirdikten sonra akışı kontrol etmek ve verileri doğrulamak için **TestNG** test aracını kullanıyorum

Ayrıca, yalnızca bir evrensel web sürücüsü oluşturmak ve kullanmak için **tekli desen** kullanan yardımcı program paketinde **Sürücü sınıfını** da oluşturdum.

FRONT END ; **selenium webdriver** ve çerçevemde tasarım **modelim** olarak **Sayfa nesne modelini** kullanıyorum ; - sayfa oluştur nesneler; webElements'i tanımlayın ve bir webelement değişkeni olarak depolayın,

POM = ELEMENT / YÖNTEM TABANLI SAYFA HEDEFİN YENİDEN KULLANILABİLİRLİĞİ

// Web öğelerimi @FindBy kullanarak somutlaştırmak için **Page Factory tasarım modelini** de kullanıyorum - daha kolay / kullanışlı

// yardımcı program; tarayıcı kullanımı - tarayıcıyı otomatikleştirmek için kodları kolaylaştırmak için hayatınızı kolaylaştıran statik yeniden kullanılabilir kod;

GERİ SONU (Api)

// kullanarak **RESTASSURED** süreci için sırayla çerçevesinde **deserialization** ve **serileştirme** oluşmaya

// **Json yanıtını** bir java toplama veri yapısında depolamanın (/ i yüksek seviyeli Pojolar ve harita nesneleri üretir) ve 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

// ÇERÇEVİMİ DAVETMEDEN ÖNCE manuel test için **postacı** kullanıyorum

VERİTABANI testi

// SQL sorguları üretmek için manuel I **sql geliştiricisi**

// OTOMASYON İÇİN; oracle veritabanından **BAĞLANTI** alarak java'yı entegre etmek için **JDBC kitaplığını** kullanıyorum

// daha sonra SQL sorgularını kullanarak **STATEMENTS** oluşturmak ve ardından verileri bir **RESULTSET** nesnesine depolamak .

// İçindeki verileri depolamak ve karşılaştırmak için java veri yapılarını kullanıyorum

// ve **DATA DRIVEN ve CUCUMBER BDD** çerçevesini kullandığım için, bu testlerin tümü **özellik dosyalarının** içinde saklanıyor

// **ÖZELLİK DOSYASINDAN** kodlar oluşturmaya ve bunları b adlı bir dosyaya uygulamaya yardımcı olan **RUNNER sınıflarım** var

// ayrıca tüm testlerimden önce ve sonra çalışan kodlarımı uygulayan **HOOK sınıfı** da var - burası benim **Senaryo** arayüzünü kullandığımda tetiklenen **TAKESCREENSHOT** arayüzü (senaryo başarısız olduğunda)

// başarısız olan adımdayken fotoğraf çek

// SD - bu, **kornişon dilinin** beklenen değerini temel alan kodlarımı sakladığım yer

DDT

// az miktarda test verisiyle çalışıyorsam, **senaryo ana hatlarıyla** çalışacağım , burada **örnekler** oluşturuyorum ve ardışık düzen kullanarak verileri depolayın

// test verileri büyük miktarda i kullanma böylece onun genellikle harici dosyası (Excel) varsa **Apache POI** için **INVOKE DDT EXCEL OTOMASYON** ve excel dosyasından oku ve verileri sakla

// java veri yapısına

Sayfa 130

// Temelde yüksek riskli **kodlarımı** günlüğe kaydetmek için **log4j2** adında bir günlüğe kaydetme **aracım** da var

// ve son olarak raporlarım için, **çerçevemde** salatalık tarafından oluşturulan "rerun: target / rerun.txt" **salatasında Rerun.txt kodunu** kullanıyorum sandviç kütüphanesi

// bu başarısız salatalık özellik dosyalarımı depolayacak

// daha sonra başarısız senaryoların (rerun.txt) konumuna sahip olan **failScenario runner sınıfım** da var

// başarısız bir Senaryo xml dosyası oluşturuyorum

// bu yüzden başarısız özellik dosyalarım olduğunda mvn komutunu kullanıyorum; **mvn -Drunner = failScenarios.** Başarısız testlerimi çalıştırmak için **xml** dosyası

// raporlama - salatalık raporları adı verilen hedef klasörde bulunan html raporu kullandım - "html: hedef / salatalık-raporu"

Paralel test - koşucular oluşturmak için **salatalık jvm-paralel eklentisi** ve testleri çalıştırmak için **hata korumalı eklenti** kullandı

İçin **sürekli entegrasyon (Jenkins)**
// devops yapılandırmayı halleder
// github yolu var
// ancak araç bir mvn komutum çağrıldı - mvn doğrulayın -drunner = smoketest.xml test cihazı tarafından sağlanan - xml dosyası
// raporlar için her yapıda, testin grafik bilgilerini ve ekran görüntüsünü veren bir salatalık raporu olacaktır

Kaç test durumu / senaryosu / özellik dosyası: Sprint başına 23 test durumu, 150 özellik dosyası, 400 senaryo Duman testim
2 özellik dosyası, her biri 2 senaryo, günde bir kez 5 dakikalık duman testi yapılır.

Benim gerilemem
150 özellik dosyası
400 senaryo
3 yıl ; 1 test kullanıcısı 3 yıl çalıştı, ikincisi 2 yıl çalıştı

Duman testiniz ne sıklıkla, ne kadar sürer ve neyi test eder? Her sabah saat 5'de, 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 sürer ve neyi test eder? Haftada bir 1am, yarım saat, tüm ilçeler her gün,
sadece SIMS. Hafta sonları doğrulama regresyon testleri, 5 ayrı EC2 makinesi, her okul için paralel test, başvuru,
doğrulama, çapraz doğrulama ve sertifikasyon. Testimiz SCS ile doğrulanıyor.

Örnek Senaryo: SIMS raporunun indirildiği ve dosya türünün bir Excel Dosya türü olduğu göz önüne alındığında, ilk sütun başlığı
"LASID" ve ilk sütundaki her hücre alfanümerik olmalı ve ilk sütundaki her hücre 9 basamaklı ve tümü LASID olmalıdır
sayılar benzersiz olmalıdır.

Örnek Test Vakası: Kullanıcının oturum açtığı göz önüne alındığında, SIMS raporu yürütüldüğünde, ilk sütundaki her hücre 9 olmalıdır.
rakamlar ve ayrıca alfanümerik.
Olmalıdır: 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)

Risk temelli test: Tam regresyon testi yapmak için zaman olmadığında, yalnızca önemli olan, ilgili kısımları test edersiniz.

Gereksinimler olmadan test etme: Üretim kusurlarının genellikle herhangi bir gereksinimi yoktur ve
durumu daha iyi ve sonra test edin.

Sayfa 131

Aşırı yükleme yöntemi için örnek : Beklemeler için BrowserUtils'te çeşitli aşırı yüklenmiş yöntemler. Konum belirleyiciye göre açık bekler veya
WebElement.
Geçersiz kılma yöntemi örneği: Aşağıda

Miras / 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.

131

Sayfa 132

MARUFJAN Sınıflarından

Jenkins nedir?

- Sürekli entegrasyon aracı.
- Geliştiriciler ve testçiler tarafından kullanılan derlemeleri, dağıtımları vb. Planlamak ve otomatikleştirmek için kullanılır.

Jenkins kullanılarak hangi tür işlemler otomatikleştirilebilir?

- birim testleri çalıştırma
- uygulamayı oluşturma
- otomatik kullanıcı arayüzü testleri çalıştırma
- farklı ortamlara dağıtım

Şirketinizde jenkins test için nasıl kullanılıyor?

- Projemde, otomatik duman testlerini planlamak ve yürütmek için jenkins kullanıyoruz.
- duman testlerimiz her gün yapılmaktadır.

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, mantıklı olabiliriz.

Jenkins kullanılarak hangi test katmanları test edilebilir?

- Uygulamanın farklı katmanlarının test edilmesi test kodumuz ile yapılır.
- Jenkins, kullanıcı arayüzünü veya veritabanını veya api'yi test etmemizi önemsemez. sadece testleri başlatır ve raporlar gönderir.
- Dolayısıyla, otomatik testim bir UI testiyse, bu, jenkins'in UI testleri çalıştırdığı anlamına gelir. veya otomatik testim bir API testiyse, jenkins, API testleri çalıştırıyor.

Jenkins'i kim kurar?

- Şirketimde, jenkinlerin bakımından sorumlu DevOps / Operasyon Destek ekibimiz var. kurulu olanlar ve yapılandırılmış jenkins.
- Konuşlandırmalar vb. İçin, uygulamaları oluşturmak ve dağıtmak için işler yaratmak üzere geliştiricilerle birlikte çalışırlar.
- Duman testleri oluşturmak için otomasyon mühendisleriyle birlikte çalışırlar.

Jenkins'te duman testlerini kim kurar?

- Bir otomasyon mühendisi olarak duman testlerimi oluşturmak için devop'lar / operasyon destekleri ile çalışıyorum.
- Testlerimi çalıştırmak için Jenkins üzerinde kurulu bazı eklentilere ihtiyacım var, ayrıca javaya, jenkins üzerinde yapılandırılmış maven'e ihtiyacım var, ayrıca testlerin çalışacağı sunucuda yüklü tarayıcılar.
- Projemde, yalnızca devops / operasyon destek ekibi üyeleri yukarıdaki yapılandırmaları yapma hakkına sahip. bu yüzden sahibim onlarla çalışmak için.
- Jenkins'in yapılandırması tamamlandığında duman testi işini oluşturabilir ve çalıştırabilirim.

Jenkins üzerinde duman testleriniz nasıl yapılandırılıyor?

- Jenkins işimiz, otomatik testlerimi GitHub'dan alacak ve her gün maven kullanarak çalıştırılacak şekilde yapılandırıldı.

- yapılandırmada önce yeni bir iş oluşturdum ve "duman testleri" adını verdim. sonra Kaynak Kodda git seçeneğini seçtim
  Yönetim bölümüne gidin ve GitHub'daki çerçevemin yolunu girdim. ve ayrıca git kimlik bilgilerini girdi.
- bir sonraki yapılandırma, düzenli olarak oluştur seçeneğini belirlediğim ve testlerin ne sıklıkla çalıştırılacağını girdiğim tetikleyicilerle ilgili.

Sayfa 133

Build bölümünde , otomasyon çerçevem maven kullanılarak oluşturulduğundan, üst düzey maven komutunu çağır seçeneğini seçiyorum.

- bölümünde maven komutunu giriyorum (mvn kısmı olmadan). bu yüzden testimi terminal kullanarak çalıştırmak için normal komut
  'mvn test', jenkins'e sadece 'test' giriyorum. burada komut satırını kullanarak çalıştırmak istediğim etiketi de giriyorum.
  test -Dcucumber.options = "- etiketler @smoke"
- regresyon testleri çalıştırmak istersem, komut şu şekildedir:
  test -Dcucumber.options = "- etiketler @regression"

Derleme Sonrası İşlemlerde , çift yapılandırma yapıyorum,

- Salatalık raporları seçeneği ekliyorum. Bu kullanılabilir çünkü bizim jenkins'imize salatalık html raporları eklentisi yükledik.
  Salatalık raporları eklentisi, her derleme için html raporu oluşturacaktır. Jenkins üzerinde her duman testi yaptığımızda, yeni bir
  bildiri. ve tüm derleme için tüm raporlar kaydedilir.
- Sonraki Oluşturma Sonrası Eylemler olarak e-posta seçeneği ekliyorum. Burada bunu, Agile ekibimdeki herkesin hakkında bilgilendirilmesi için yapılandırıyorum.
  test sonuçları.

Testler için e-posta raporlarını kim alır?

- Agile ekibimdeki herkes test sonuçları hakkında bilgilendirilir.

"DevOps ardışık düzeninde çalıştınız mı?"

- Devops boru hattının parçası olan jenkins üzerinde duman testleri yaptım. devops ardışık düzeni tarafından oluşturulur ve yönetilir
  DevOps / İşlem Desteği. Ancak duman testimiz boru hattının bir parçası.
- Ben de jenkins üzerinde duman testleri oluşturup yapılandırarak katıldım.

Entegrasyon testi ile uçtan uca test arasındaki fark nedir?

- **Google genellikle 70/20/10 ayrımı önerir:% 70 birim testleri,% 20 entegrasyon testleri ve% 10 uçtan uca testler.**
- **Birim Testleri**
  - En küçük işlevsellik birimini, tipik olarak bir yöntem / işlevi test eder (örneğin, belirli bir duruma sahip bir sınıf verildiğinde, x
    sınıftaki yöntem y'nin olmasına neden olmalıdır). Birim testleri belirli bir özelliğe odaklanmalıdır
- **Entegrasyon Testleri**
  - Entegrasyon testleri, kod birimlerini birleştirerek ve ortaya çıkan kombinasyonun
    doğru çalışıyor. Bu, bir sistemin içgüdüleri olabilir veya birden fazla sistemi bir araya getirerek yapabilir.
    kullanışlı bir şey. Beyaz kutu test yaklaşımıdır. Geliştirici tek tek birim geliştirir ve bunu entegre eder
    birbirinizi ve test edin.
- **Fonksiyonel Testler**
  - Fonksiyonel testler, belirli bir girdiye ait sonuçları,
    Şartname. Fonksiyonel testler ara sonuçlarla veya yan etkilerle ilgilenmez, sadece sonuçla (
    x'i yaptıktan sonra y nesnesinin z) durumuna sahip olması umrumda değil.
- **Kabul Testi**
  - Bu, yazılım müşteriye teslim edilmeden önce gerçekleştirilen son testtir. Sağlamak için yapılır.
    geliştirilen yazılım tüm müşteri gereksinimlerini karşılar. İki tür kabul testi vardır - biri
    dahili kabul testi (Alfa testi) olarak bilinen geliştirme ekibinin üyeleri tarafından gerçekleştirilir ve
    (Beta testi) olarak bilinen müşteri veya son kullanıcı tarafından gerçekleştirilen diğer
- **Uçtan Uca test**
  - o, a noktasından z'ye gittiğiniz yerdir ve oraya giderken çeşitli farklı noktalara dokunursunuz. Bir tek için olabilir
    bir e-posta gönderme işlemi gibi bir sistem veya birden fazla sistemin dahil olduğu durumlarda kullanılabilir, örneğin
    öğrencinin bir sınava kaydolması, sınava girmesi ve sonunda puanlarını alması.
  - Gerçek dünya sistem testi. Uygulama tüm entegre donanım, veritabanı, ağ ve diğer
    arayüzler.

Sayfa 134

Bunları Jenkins üzerinde nasıl çalıştırıyorsunuz?

Uzun hikaye…

Bana çerçevenizden bahseder misiniz?

- En son Book-IT otomasyon çerçevemde, otomatik testler için Cucumber BDD çerçevesini kullandık. bu çerçeve
  çok esnek bir çerçeve. Birçok farklı otomasyon testi konseptini entegre eden karma 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 Hıyar BDD çerçevesidir. Salatalık, otomatik testi teknik olmayanlar tarafından anlaşılır hale getirmek için kullanılan bir araçtır.

  takım üyeleri. Salatalık, otomasyon mühendisleri ile teknik olmayan ekiplerin birbirine bağlanmasında köprü görevi görür.

  üyeler. salatalık versiyon 4 kullanıyoruz
- Junit'i test aracı olarak kullanıyoruz. Junit, salatalık testlerini başlatmak ve aynı zamanda iddialarda bulunmak 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. Cucumber, senaryo ana hatlarını kullanarak veriye dayalı testleri yerel olarak destekler. The

  framework ayrıca apache POI kitaplıklarını kullanarak excel'den Veriye dayalı testler yapabilir.
- Çerçevem sayfa nesne modeline dayanıyor. sayfa nesnesi modeli, sayfadaki uygulamadaki sayfaları temsil ettiğimiz zamandır

  nesne sınıfları.
- bir web sürücüsü oluşturmak için fabrika modeli kullanıyoruz. webdriver sınıfımız, ne tür bir sürücüye bağlı olarak

  istemek. webdriver nesnesi için tekli desen kullanır
- URL, tarayıcı türü, oturum açma bilgileri vb. Gibi çerçevemizle ilgili önemli bilgileri depolamak için özellikler dosyasını kullanırız.
- Şirketimizde IntelliJ kullanıyoruz, ancak tutulmada da oldukça iyiyim.

Hangi uygulama katmanını test ediyor?

Çerçevemizi kullanarak, uygulamanın kullanıcı arayüzünü, veritabanını ve API'sini test ediyoruz.

- **UI** → UI'yi test etmek için selenyum web sürücüsü kullanıyoruz
- **Veritabanı** → veritabanını test etmek için JDBC kitaplıklarını kullanıyoruz
- **API** → **API'yi** test etmek için REST Assured kitaplıkları kullanıyoruz

Test türleri:

- **fonksiyonel** , **kabul testleri** , **duman testleri** , **regresyon testleri** , **entegrasyon testleri** yapıyoruz .
- Projemde sürüm kontrol aracı için 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 testleri çalıştırabilir 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, runners için paketlerimiz var

  farklı yardımcı programlar kullanıyoruz, WebDriver için yardımcı programlarımız, tarayıcı yardımcı programları, excel yardımcı programı, yapılandırma yardımcı programı, Veritabanı yardımcı program, api ile ilgili yardımcı programlar, tarih yardımcı programı (takvimle ilgili uygulamayı test ettiğimiz için, tarihle ilgili çok şey yapıyoruz).
- özellikleri:

  birçok özellik dosyamız var ve bunları düzenlemek için klasörler ve etiketler kullanıyoruz. her ana bileşen için klasörlerimiz var

  uygulama: rezervasyonlar, harita, hesap bilgileri ...

  Jira'daki sorun numarasını özellik dosyasındaki etiketlerden biri olarak kullanırız, böylece onu Jira ile eşleştirebiliriz.
  - özellikleri rezervasyon

    hesap harita

Sayfa 135

```
Özellik: Kullanıcı rolleri
  # verilen içinde herhangi bir boş yer olup olmadığını öğrenmenin bir yolunu bulmanız gerekir

  @login @ BRIT_3521 @smoke
  Senaryo Özeti: <kullanıcı> türü olarak oturum açın
    Kullanıcının açık taraf <kullanıcı> olarak oturum açtığı göz önüne alındığında
    Ve zamanlama için uygun noktalar var
    Ne zaman bir nokta için kullanıcı avı
    Ardından kitap düğmesi <beklenilen> görüntülenecek

    Örnekler:
      | kullanıcı | beklenen |
      | takım üyesi | olmamalı |
      | takım lideri | gerekir |
      | öğretmen | gerekir |
```

Paralelleştirme → testlerimizi maven-surefire-eklentisini kullanarak paralel olarak çalıştırıyoruz. paralel testi destekleyen salatalık 4 kullanıyoruz doğal olarak.

Test verileri → test verilerimizi test / kaynaklar paketinde saklamak için özellik dosyalarını, excel dosyalarını kullanın

Yürütme akışı veya testleri nasıl çalıştırıyoruz

1. **cukes koşucusundan koşmak**

   cukes koşucusuna etiket koyarız ve sağ tıklayıp dosyayı çalıştırırız. cukes runner tüm eşleşen etiketleri çalıştırır.
2. **terminalden maven komutu olarak çalıştırın**

   testleri başlatmak için mvn doğrulama veya mvn test komutunu giriyoruz. ve maven, pomda gösterilen cukes runner dosyalarını çalıştıracak

dosya. cukes runner dosyası, sahip olduğu etiketlerle eşleşen özellikleri çalıştırır.

GIT dallanma

git dallanmasını şimdi nasıl yapıyorsun?

Şu anda GRUP PROJELERİNDE NASIL YAPILIR:

Her ekip üyesi için ana şube ve ayrı şubeler vardır. birisi işi bitirdiğinde,
kendi şubelerini inceledikten sonra ustalaşmak için birleştirilir.

NASIL YAPTIK?

projemde kişi için ustalık, geliştirme ve şubemiz vardı. bu nedenle, 2 otomasyon test cihazımız varsa,

usta
geliştirmek
test cihazı1
test cihazı2

her test kullanıcısı kendi şubelerine giriş yapar. incelendikten sonra şube geliştirmek için birleştirilir. ustayı birleştiriyoruz ve sadece bir sprint geliştirir.

✪ görüşmede dallanma hakkında sorduklarında, otomasyon projesi dallanma stratejiniz hakkında konuşun.

Projenizde, kodunuz uygulama kod 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.

Sayfa 136

JAVA Mülakat Teknik Soruları

Ters Dize - Döngü İçin

```
public static String reverseString (String str) {
    Dize ters = "";
    for (int i = str.length () - 1; i> = 0; i--) {
        ters + = str.charAt (i);
    }
tersine dönüş;
}
```

EN İYİ YOL:

```
Dize adı = "Alper Aslan";
String reversed = new StringBuilder (isim) .reverse (). ToString ();
System.out.println (tersine çevrilmiş);
```

Ters Dize - StringBuilder

```
public static void main (String [] args) {
String str = "Merhaba Dünya";
StringBuilder sb = new StringBuilder (str);

Sistem. out .println (sb.reverse ()); }
```

Ters Dize - Karakter Dizisi

```
public void printReverse (karakter [] harfler, int boyut) {
    for (int i = letters.length-1; i> = 0; i -) {
            System.out.print ([i] harfleri);
    }
}
```

Asal sayı

```
public static boolean checkPrime (int n) {
    eğer (n <= 1) {
```

```
                yanlış dönüş;
    }
    for (int i = 2; i <Math. sqrt (n); i ++) {
                eğer (n% i == 0) {
                        yanlış dönüş;
                }
    }
doğruya dön;                    }
```


Sayfa 137

Palindrom

```
public static boolean isPalindrome (String str) {
    eğer (str == null)
                yanlış dönüş;
                StringBuilder sb = new StringBuilder (str);
return sb.reverse (). toString (). equals (str); }
```

Palindrome - charAt () ile

```
public static boolean isPalindrome (String s) {
        int head = 0;
        int kuyruk = s.length () - 1;
        while (baş <kuyruk) {
                if (s.charAt (head)! = s.charAt (kuyruk)) {
                        yanlış dönüş;
                }
                kafa ++;
                kuyruk--;
        }
doğruya dön;                    }
```

Ters Dize - ArrayList - Itirator

```
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);

   Nesne [] arr2 = arrList.toArray ();

    for (int i = 0; i <arr2.length; i ++) {
                ters + = dizi2 [i];
    }
    Sistem. out .println (ters);
}
```

DİĞER YOL

```
Dize ters = arrList.stream ()                                   // Akış <Karakter>
                        .map (String :: valueOf) // Akış <String>
                        .collect (Collectors.joining ());
```


Sayfa 138

Faktöriyel

```
int sayı = 10;
int factorialSum = 1;

for (int i = 1; i <= sayı; i ++) {
                factorialSum = factorialSum * i;
}
Sistem. out .println ("" + sayı + "faktöriyeli" + faktör toplamı "dır); }
```

Basamakların Toplamı

```
int numarası = 1346;
int toplam = 0;

while (sayı> 0) {
    toplam + = sayı% 10;
    sayı = sayı / 10;
}
Sistem . out .println (toplam);                    }
```

Armstrong

```
int toplam = 0;
int rakam;
int temp;
int sayı = 370;
temp = sayı;
while (temp> 0) {
    basamak = sıcaklık% 10;
    toplam = toplam + (basamak * basamak * basamak);
    sıcaklık = sıcaklık / 10;
}
eğer (sayı == toplam)
    System.out.println (sayı + "sağlam bir sayıdır");
}Başka
```

Merdiven

```
for (int x = 1; x <= 5; x ++) {
for (int y = 1; y <= x; y ++) {
    System.out.print (y + "");
}
System.out.println ();                    }
```


Sayfa 139

Fibonacci

```
int a = 0;
int b = 1;
System.out.print (a + "" + b + "");

for (int i = 2; i <= 10; i ++) {
    int c = a + b;
    a = b;
    b = c;
    System.out.print (c + "+");
}
```

Geçici Bir Değişken Kullanmadan İki Numarayı Değiştirin

```
    // tek satırlı yöntemler
a = a ^ b ^ (b = a);                    //Yöntem 1
```

```
b = (a + b) - (a = b); // yöntem 2
a + = b - (b = a); // yöntem 3

// geçici değişken
int temp = a;
a = b;
b = sıcaklık;
```

Dizileri Kullanan Dizideki En Büyük Sayı

```
int [] dizi = {5, 6, 76, 31, 43, 1};
Arrays.sort (dizi);
System.out.println (arr [arr.length-1]);
```

Koleksiyonları Kullanan Dizideki En Büyük Sayı

```
public static int returnLargest (Tamsayı [] b, int toplam) {
    List <Integer> list1 = Diziler. asList (b);
    Koleksiyonlar. sort (liste1);
    int en büyük = list1.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));
```


Sayfa 140

Koleksiyonları Kullanan Dizideki En Büyük Sayı

```
public static int getLargest (int [] a, int toplam) {
    int temp;
    for (int i = 0; i <toplam; i ++) {
        for (int j = i + 1; j <toplam; j ++) {
            eğer (a [i]> a [j]) {
                temp = a [i];
                a [i] = a [j];
                a [j] = sıcaklık; } } }
bir [toplam-1] döndürür; }
```

Ters Sayı

```
public static int reverse (int sayı) {
    int ters = 0;
    int kalan = 0;
    yapmak{
                kalan = sayı% 10;
                ters = ters * 10 + kalan;
                sayı = sayı / 10;
    } while (sayı> 0);

tersine dönüş; }
```

Bir Dizideki İlk İki Maksimum Sayıyı Bul

```
    public void GetTwoMaxValues (int [] numaralar) {

                int maxOne = 0;
                int maxTwo = 0;

                Arrays.sort (nums);
                System.out.println ("Max1 -" + (nums [nums.length-1]));
                System.out.println ("Max2 -" + (nums [nums.length-2]));
```

```
}
```

Bölme veya Mod İşleci kullanmadan bölme

```
public static String division (int divident, int divisor) {
        int quotient = 0;
        int kalan = 0;
        while (bölünen> = bölen) {
                bölünen = bölünen - bölen;
                bölüm ++;
                kalan = bölünen; }
return "bölüm =" + bölüm + "kalan =" + kalan;}
```


Sayfa 141

İkili Arama (Doğrusaldan Daha Hızlı) -

Sıralanmış bir dizi içindeki bir hedef değerin konumunu bulur. İkili arama, dizinin orta elemanına hedef değer. Eşleşiyorsa, değilse true döndür ulaşana kadar tekrar bölüyorsun.

```
public static void binarySearch (int arr [], int first, int last, int key) {
        int orta = (ilk + son) / 2;
        while (ilk <= son) {
            eğer (arr [mid] <key) {
                ilk = orta + 1;
            } else if (arr [mid] == key) {
                System.out.println ("Öğe dizinde bulunur:"
                                                                    + orta);
                kırmak;
            }Başka{
                son = orta - 1;
            }
            orta = (ilk + son) / 2;
        }
        eğer (ilk> son) {
                System.out.println ("Öğe bulunamadı!"); }
}

public static void main (String args []) {
        int arr [] = {10,20,30,40,50};
        int anahtar = 30;
        int son = dizi. uzunluk-1;
binarySearch (arr, 0, last, key); }
```

Kabarcık Sırala

Mevcut eleman bir sonraki eleman ile karşılaştırılır. Geçerli öğe daha büyükse sonraki öğe değiştirilir.

```
public static void bubbleSort (int [] arr) {
    int n = dizi uzunluk;
    int temp = 0;
     for (int i = 0; i <n; i ++) {
            for (int j = 1; j <(ni); j ++) {
                eğer (dizi [j-1]> dizi [j]) {
            temp = dizi [j-1]; // öğeleri değiştir
            dizi [j-1] = dizi [j];
            arr [j] = sıcaklık;
     }}}}
public static void main (String [] args) {
            int arr [] = {3,60,35,2,45,320,5};
    bubbleSort (arr); // bubble sort kullanarak dizi öğelerini sıralama
public static void main (String [] args) {
System.out.println ("Kabarcık Sıralamasından Sonra Dizi");
            for (int i = 0; i <arr.length; i ++) {
System.out.print (arr [i] + "");
```


Sayfa 142

Bir Dizide Yinelenenleri Filtreleme

```
public static void main (String [] args) {
                ArrayList <String> list = new ArrayList <String> ();

                // 0-9 arası bir sayı listesi oluşturun.
                for (int i = 0; i <10; i ++) {
                        list.add (String.valueOf (i));
                }
                // 0-5 arasında yeni bir sayı kümesi ekleyin.
                for (int i = 0; i <5; i ++) {
                        list.add (String.valueOf (i));
                }

                System.out.println ("Giriş listesi:" + liste);
                System.out.println ("\ nFiltre edilmiş kopyalar:" + processList (liste));                                    }

    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) {
                        eğer (! tempSet.add (yourInt)) {
                                resultSet.add (yourInt);
                        }
                }
                return resultSet;                       }
```

En Büyük Palindrom

```
public static String en büyükPalindrome (String str) {
    str = str.toLowerCase ();
    String majorPalindrome = "";
    Dize [] arr = str.split ("");

    for (String each: arr) {
                Dize ters = "";
                for (int i = each.length () - 1; i> = 0; i--) {
                        ters + = each.charAt (i);
                }
    eğer (each.equals (ters) && each.length ()>
                                                                    mostPalindrome.length ()) {
                en büyükPalindrome = her biri;                  }
    }
en büyükPalindrome'a dön; }
```


Sayfa 143

Bir Dizeden Ekstra Boşlukları Kaldırma

```
public class removeExtraSpaces {
        public static void main (String args []) {

          String input = "Fazla boşlukları kaldırmayı deneyin.";
          StringTokenizer substr = new StringTokenizer (input, "");
          StringBuffer sb = new StringBuffer ();

          while (substr.hasMoreElements ()) {
              sb.append (substr.nextElement ()). append (""); }

          System.out.println ("Gerçek dize:" + girdi);
       System.out.println ("İşlenen dize:" + sb.toString (). Trim ()); }
```

DİĞER YOL

```
String input = "Fazla boşlukları kaldırmayı deneyin.";
String inputNew = input;

while (inputNew.contains ("")) {
    inputNew = inputNew.replace ("", "");
};

System.out.println (giriş);
System.out.println (inputNew);
```

Ünlülerin ve Ünsüzlerin sayısını sayın

```
    public static String numaraları (String word) {
                int countVowels = 0;
                int countConsonants = 0;

    <Karakter> ünlüleri listele = Diziler. asList ('a', 'e', 'i', 'o', 'u');
    <Karakter> ünsüzleri Listele = new ArrayList <> ();

                for (char i = 'a'; i <= 'z'; i ++) {
                        eğer (! vowels.contains (i)) {
                                ünsüzler.add (i); }
                }
                kelime = word.toLowerCase ();

                for (int i = 0; i <word.length (); i ++) {
                        if (vowels.contains (word.charAt (i))) {
                                countVowels ++;
                        }Başka {
                                countConsonants ++; }
                }
                return "ünlülerin sayısı" + countVowels + "ve ünsüzlerin sayısı" +
countConsonants; }
```


Sayfa 144

Dinamik - Çalışma Zamanı Polimorfizmi

```
class Animal {
    void eat () {
                System.out.println ("yeme"); }
}
class Dog, Animal {
    void eat () {
                System.out.println ("meyve yeme");}
}
class BabyDog, Dog {
    void eat () {
                System.out.println ("içme sütü");
}
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 (); // "yeme meyveleri" yazdırır
                a3.eat (); }} // "içme sütü" yazdırır
```

Çok Düzeyli Kalıtım ile Dinamik Polimorfizm

```
class Animal {
    void eat () {
                System.out.println ("hayvan yiyor ...");                                    }
}
class Dog, Animal {
    void eat () {
                System.out.println ("köpek yiyor ..."); }
}
class BabyDog1, Dog'u genişletir {
```

```
    public static void main (String args []) {
                Hayvan a = yeni BabyDog1 ();
                a.eat ();                    }
}
```

Xpath - Kardeşin İzinde

Aşağıdaki-sibling :: siblingName [1];
Parent :: parentTag
// etiket [ile başlar (@ id, 'mesaj')]

Sayfa 145

Zaman Polimorfizmini Derleyin

```
class Overload {
        void demo (int a) {
             System.out.println ("a:" + a); }

        void demo (int a, int b) {
             System.out.println ("a ve b:" + a + "," + b);
        }
        çift demo (çift a) {
             System.out.println ("double a:" + a);
             dönüş a * a; }
}

class MethodOverloading {
        public static void main (String args []) {
              Aşırı Yük Nesnesi = yeni Aşırı Yük ();
              çift sonuç;
              Obj. Demo (10);
              Obj. Demo (10, 20);
              sonuç = Obj .demo (5.5);
              System.out.println ("O / P:" + sonuç); }}
```

1-100 arası tüm çift sayılar (i% 2 == 0)
Hepsi tek sayı (i% 2! = 1)

Veri üyeleri tarafından çalışma zamanı polimorfizmi elde edilemez.

String'i Double'a dönüştürmek için

```
Double doubleString = Double.parseDouble (toBeDouble);
Double doubleStr = Double.valueOf (toBeDouble);

Double'yi String'e dönüştürmek için
String strDouble = String.valueOf (toBeString);
String stringDouble = toBeString.toString ();

    public static char returnFirstChar (String str) {

                dönüş str.charAt (0);
    }

    public static char returnLastChar (String str) {
                dönüş str.charAt ((str.length () - 1));
    }
```

Sayfa 146

Uyarılar

Uyarı uyarısı = driver.switchTo.alert ();
alert.accept (); alert.dismiss (); alert.sendKeys () alert.getText ()

Örtülü Bekleme

driver.manage.timeouts.implicitlyWait (5 saniye)

Açık bekleme

WebDriverWait wait = new WebDriverWait (sürücü, 30);
WebElement öğesi = wait.until (
ExpectedConditions.elementToBeClickable (By.id (öğe)));
visibilityOf (), alertIsPresent ()

Excel - Apachi POI

```
public void readExcel () atar IOException {
        Dize DosyasıP a th = "D: \\ sampledoc.xls";
        FileI n put Stream fis = new FileInputStream (FilePath);
        Çalışma kitabı wb = WorkbookFactory.create (fis);

        Sayfa sh = wb.getSheet ("Sayfa1"); // getSheetAt (1)
        // Sayfadaki mevcut satır sayısını almak için
        int totalNoOfRows = sh.getRows ();
        // Sayfada bulunan sütun sayısını almak için
        int totalNoOfCols = sh.getColumns ();
for ( int satır = 0; satır <totalNoOfRows; satır ++) {
        for ( int col = 0; col <totalNoOfCols; col ++) {
System.out.print (sh.getCell (sütun, satır) .getContents () + "\ t");}
        System.out.println ();}
```

Row row = workSheet.getRow (0);
Hücre hücresi = row.getCell (0);
System.out.println (cell.toString ());

Hücre hücresi = row.getCell (0);
cell.setCellValue ("kahve");
FileInputStream outStream = new FileInputStream (filePath);
workbook.write (outStream);

Sayfa 147

JDBC - Veritabanı Bağlantısı

Bağlantı bağlantısı = DriverManager.getConnection (Url, Kullanıcı, Geçiş);
İfade ifadesi = connection.createStatement
(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
ResultSet resultSet = statement.executeQuery ("ülkelerden * seçin");
sonuçSet.last (); // sonuç kümesinde kaç kayıt bulun
int rowsCount = resultSet.getRow ();
System.out.println ("Satır sayısı:" + rowsCount);

resultSet.first ();
while (resultSet.next ()) {
System.out.println (resultSet.getString (1) + "-" + resultSet.getString ("ülke_adı") + "-

"+ resultSet.getInt (" region_id "));}

Sınıf Seçin

ArabalarListesi **seçin** = yeni Seç (el);
carList.getFirstSelectedOption ();
assertequals ("bir metin", carList.getFirstSelectedOption (). getText ())

İframe

Driver.switchTo.frame (webElement veya Dize veya dizin olarak kimlik)

Pencere tutacağı

getWindowHandles ()
for (Dize tutamacı: driver.get.getWindowHandles ())
driver.switchTo.window ("handle") :: bir dizge alır
Eğer (driver.getTitle (). Equals (beklenen başlık)
kırmak;

Eylem Sınıfı

WebElement el = driver.findElement
Eylemler eylemler = yeni Eylemler (sürücü) .perform
actions.doubleClick (el) .perform

Dosya yükleme

Genel void fileUpload (Dize yolu) {
WebELement karşıya yükleme = sürücü.findELement;
Upload.sendKeys (yol)

Sayfa 148

SQL - İç Birleştirme

SEÇ e.FIRST_NAME, e.SALARY, j.JOB_TITLE
DAN ÇALIŞANLARIN e **INNER JOIN** İŞLERİ j **AÇIK** e.JOB_ID = j.JOB_ID;

İstenen Yetenekler - Sertifika Verilmesi

DesiredCapabilities yeteneği = DesiredCapabilities.chrome ();
// SSL sertifikasını kabul etmek için
capability.setCapability (CapabilityType.ACCEPT_SSL_CERTS, true);
System.setProperty ("webdriver.chrome.driver", "E: /chromedriver.exe");
// Google Chrome örneği oluşturun ve maksimize edin
sürücü = yeni ChromeDriver (yetenek);

5 Ekran inci tablodan Row

(EMPLOYEE_ID, FIRST_NAME, ROWNUM OLARAK ROWNUM SEÇİN.
ÇALIŞANLARDAN) RN = 5 NEREDE;

Sayfa 149

B10 - GERÇEK HAYAT ÇİFTİ MÜLAKAT SORULARI:

Shahin G.
- Bize projenden bahset.
- Java'yı nerede kullanıyorsunuz?
- String ve Stringbuilder arasındaki fark nedir?
- Yığın ve yığın arasındaki fark nedir?
- Birleştirmeler (SQL) nedir?
- Senaryo Taslağı'nın kullanımı nedir?

İbrahim S.
- Arka uç testini hangi araçları kullanıyorsunuz ve projenizde Arka Uç Testinde neyi test ediyorsunuz?
- Selenyum tanımlayıcı nedir?
- Açık ve üstü kapalı bekleme arasındaki fark, bana bir örnek verin?
- Java ile senkronize edilmiş 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 uyumluluk testi nedir?
- Test otomasyon çerçevesinin oluşturulmasına dahil olan adımlar nelerdir?

Enes S.
- AWS sertifikasına sahip olduğunuzu görüyorum,
    - o hakkında biraz konuşabilir misin?
    - o EC2'de jenkins nasıl yapılandırılır
- Günlük aktiviteler
- bir google doc belgesi açtı ve harita için bir liste ve anahtar değer olarak bir değer yazmamı istedi -> nasıl koyarsınız harita içi
- selenyumda öğe görünmüyorsa nasıl aşağı kaydırılır -> jsexecuter aşağı kaydırma yöntemi yazmamı istedi
- Java'da akış nedir?

Feruk C.
- 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?
- Takım yapınızı açıklayabilir misiniz?
- Neden bu pozisyonla ilgileniyorsunuz ve neden sizi işe alalım?
- Hiç sıfırdan bir çerçeve oluşturdunuz mu?
- BDD çerçevenizi ve mevcut projenizde veri odaklı çerçeveye ihtiyaç duymanızın nedenini açıklayabilir misiniz? ve lütfen açıklar mısınız?
- Mevcut projenizde kullandığınız araçları açıklayabilir misiniz?
- JMeter ile PostMan arasındaki fark nedir?
- Entegrasyon testinin nasıl yapılacağını biliyor musunuz?

Merve O.
- Çerçevemi detaylı olarak anlattım.
- Ve bana duman ve gerileme testini sordu
    - o duman veya regresyon testi olduğunu nasıl anlarsınız?
- OOP'yi sordu
- Özet ve arayüz?

: Kalıtım(Daha çok insan dilinde bir örnek veriyorum)
Açık bekleme ve Örtük bekleme
• Jira ve bitbucket hakkında konuştu

Sayfa 150

Toufiq N.
• Bana günlük aktivitelerinizden bahseder misiniz?
• Java açısından Webdriver nedir?
• Java'da bir dizeyi nasıl tersine çevirirsiniz?
• Önemsiz ek açıklamalar mı yoksa testNG ek açıklamaları mı?
• Selenium ile tarayıcıdaki açılır pencereleri nasıl işlersiniz?
• Bir tarayıcıdaki bir açılır pencereyi nasıl idare edersiniz?
• Bir tarayıcıda olduğumuzu ve bizi farklı bir bağlantıya yönlendiren 5 farklı öğe olduğunu varsayalım. Bir kez tıkladığımızda eleman ve yeni bir pencere açılır, bu pencereye nasıl geçersiniz?
• Selenium'daki hangi yöntem, onay kutusu düğmesinin seçili olup olmadığını sağlar?
• XSSF ile HSSF arasındaki fark nedir?
• Örtülü ve açık bekleme arasındaki fark nedir?
• Set ve Liste arasındaki fark nedir?
• Haritalar çerçevesinde bir HashMap nedir?
• PUT ve POST http protokolü ile API arasındaki fark nedir?
• API'de sorgu parametresi nedir?
• Bana çerçevenizden bahseder misiniz? (sadece çerçevenizde hangi araçları kullandığınızı belirtmeyin, NASIL sahip olduğunuzu açıklayın bu araçları çerçevenizde 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 ile Geçersiz Kılma arasındaki fark nedir?
• Bir alt sınıf sınıfı, Java'da son olarak başlatılan bir üst sınıfın işlevlerini çağırabilir mi?
• FindElement () ve FindElements () arasındaki fark nedir?
• Jenkins'e aşina mısınız? Onu ne için kullanıyorsun?
• Java'da çöp toplayıcı nedir?
• SQL'de iç birleştirme nedir?
• SQL'de sol birleştirme nedir?
• SQL'de doğru birleştirme nedir?
• SQL'de Birlik nedir?
• SQL'deki kısıtlamalar nelerdir?
• URI 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 verin? (if-else, ternary, switch ifadesi…)
• Selenium'da bir elementi bulmanın farklı yolları nelerdir?
• Bir öğeyi bulma açısından, tek eğik çizgi (/) ve çift eğik çizgi (//) nedir?
• OOP'de Kapsülleme nedir?
• SQL'deki Birincil ve Yabancı anahtar arasındaki fark nedir?
• Bir Dizi ve Vektör (Koleksiyon çerçevesi) arasındaki fark nedir?

Tarık K.
WEBEX RÖPORTAJI
• Son projenizden ve özellikle Selenium ve Hıyar deneyiminizden kısaca bahsedebilir misiniz?
• Otomasyon çerçeveniz hakkında biraz bilgi verebilir misiniz?
• Ekip yapınızdan bahsedebilir misiniz?
• Size bazı test durumları verildiğini varsayalım; otomatikleştirip otomatikleştirmemelerine nasıl karar veriyorsunuz? Bana belirli bir şey söyleyebilir misin faktörler?
• Regresyon paketiniz ne kadar büyük?
• Bunun için takip S'si -> Bunların tümü (test senaryoları) daha çok UI tabanlı mı yoksa REST tabanlı mı?
• Regresyon paketinin bir yürütme döngüsünü tamamlamak ne kadar sürer?
• Onları uyguladıktan sonra, herhangi bir arıza görüyor musunuz yoksa mükemmel bir şekilde çalışıyorlar mı?
• Karşılaştığınız bir otomasyon sorununu bana anlatır mısınız ve nasıl çözdünüz?
• Bir web sayfasındaki dinamik öğeleri nasıl işlersiniz?
• API dışında arka uç doğrulamaları yapıyor musunuz? (Veritabanı doğrulamasını ve bunu nasıl yaptığımı sordu)
• Jira ve ALM gibi test yönetimi araçlarıyla ilgili deneyiminiz var mı?

Sayfa 151

• Java kodunuz Jira'daki kusurları otomatik olarak tanımlar ve günlüğe kaydeder mi yoksa bunu manuel olarak mı yapıyorsunuz?
• Bana raporlama yapınızdan bahseder misiniz? Raporlarınızı nasıl düzenliyorsunuz?
• UFT, Protractor gibi diğer test otomasyon araçlarıyla ilgili deneyiminiz var mı?
• Bana kaynak kontrol yapınızdan bahseder misiniz?

SKYPE RÖPORTAJI
• Kendinden biraz bahseder misin?
• Bana projenizden, çerçevenizden ve kullandığınız araç / teknolojilerden bahseder misiniz?

- DDD ile de çalıştığınızı söylediniz, test verilerinizi nereden alıyorsunuz? Örnekler tablosundan mı yoksa dışardan mı kaynak?
- API kullanırken, aldığınız farklı durum kodları nelerdir?
- Salatalık özellik dosyasındaki bileşenler nelerdir?
- Arka planı tek bir senaryo için kullanabilir miyiz?
- UI otomasyonu için ne tür tarayıcılar kullandınız?
- Duman testlerinizi ne sıklıkla yapıyorsunuz? Ne tür bir CI / CD sürecini takip ediyorsunuz?
- Regresyon paketinizde kaç tane test vakası var?
- Regresyon paketinizi ne sıklıkla çalıştırıyorsunuz?
- Sprintiniz ne kadar sürüyor?
- Çerçevenizde OOP'yi nasıl uyguluyorsunuz?
- Maven'i nasıl kullanıyorsunuz? (Ona testlerimi çalıştırması için maven komutlarından bahsettim ve bu komutların ne olduğunu sordu)
- Projenizde Koleksiyonlar çerçevesini kullandınız mı?
- Otomatikleştirirken bana bir zorluk veya senaryo söyleyebilir misiniz? (Meydan okumayı açıkladıktan sonra daha fazlasını sordu bu meydan okumayla ilgili ayrıntılar)
- Tarayıcı açılır pencerelerini nasıl idare ediyorsunuz?

TELEKONFERANS GÖRÜŞMESİ

- 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 takıma katkınız nedir?
- SDET ekip üyesi görüşmeye katıldı ve sordu: Çerçevenizde bana rehberlik edebilir misiniz? Teknolojiler nelerdir ve kullandığınız araçlar? Test yürütmeniz nasıl gidiyor?
- Ayrıca duman ve regresyon süitlerini koruduğunuzu söylediniz, bunları nasıl organize ediyorsunuz?
- Bir iş yaratmaya yönelik tüm adımlarda bana rehberlik edebilir misiniz? (Jenkins)
- Jenkins'iniz raporları nasıl gönderiyor? Raporları nasıl depoluyorsunuz? Ne tür raporlar kullanıyorsunuz?
- Rapor e-postasını kim alır?
- Salatalığın teknik olarak nasıl çalıştığını bana anlatır mısınız? Salatalık testlerini yürütme adımlarını açıklayın.
- Runner sınıfınızda bana rehberlik edebilir misiniz? Sınıfın içinde ne var?
- Sınıfınızda Sayfa Nesne Modeli kullandığınızı da belirttiniz, sayfanızın yapısından bahseder misiniz? nesne sınıfları?
- Bana Kredi Birliği ile Banka arasındaki farkı söyler misiniz?
- Kredi Birliğine kimler üye olabilir?
- Üzerinde çalıştığınız uygulamanın bir veritabanı var mı?
- Kendi sorgularınızı mı yürütüyorsunuz yoksa birisi sizin için mi yapıyor?
- İç Birleştirme ve Dış Birleştirme arasındaki farkı bana söyleyebilir misiniz?
- Bu teknik olabilir veya olmayabilir, bana X'te çalışırken öğrendiğiniz bir şeyi söyleyin (çalıştı)?
- Çerçevenizde bir API testi uygulaması var mı? Nasıl kullandınız ve hangi araçları kullandınız?
- Bana POJO'larınızdan bahseder misiniz?
- Günlük olarak çalıştığınız HTTP durum kodları nelerdir?

Hazar M.

- Regresyon paketinizde kaç tane test vakası var?
- Regresyon paketinizi ne sıklıkla çalıştırıyorsunuz?
- Bana kendinden bahset.

Sayfa 152

- *C: BT kariyerime 2013 yılında manuel test cihazı olarak başladım. Sonra Java'yı öğrendim ve bir yıl sonra aynı şirkette Java kullanarak otomasyon yapmak. Otomasyon konusunda uzmanlaşmama rağmen şu ana kadar Sağlık, Sigorta ve Hem manuel hem de otomasyon tekniklerini kullanarak alanları finanse edin. Java ve OOPS konseptlerinde çok rahatım. Yani şimdiye kadar Maven ile çalıştım, Gradle POM tasarım modelini kullanarak BDD, DDF ve Hibrit çerçeveler oluşturdum ve TestNG, JUnit, Cucumber, Selenium WebDriver gibi araçlar ve Apache POI, Rest- Assured, Jackson gibi kütüphaneler, İçinde JDBC. SQL konusunda deneyimli, Oracle, PostgreSQL ve MySQL gibi RDBS'ler ile Veri Bütünlüğü Testi gerçekleştirme. ben Farklı tür çerçeveler oluşturma konusunda çok rahatım ve mevcut olanları kolayca uyarlayabilir ve sürdürebilirim. ben Duman, Regresyon, Fonksiyonel ve Arka Uç testlerinde deneyimli. Şimdiye kadar Waterfall ve Agile / Scrum'da çalıştım metodolojiler ve SDLC, STLS ve Hata Yaşam Döngüsünün tüm aşamalarında iyi deneyim. Tüm scrumlara yakından katıldı törenler ve çapraz fonksiyonlu bir ekip üyesi olarak önemli bir rol oynadı. Çok pozitifim, sonuç odaklıyım, uyarlanabilirim takım oyuncusu ve arkadaşlarım, kişilerarası ve iletişim becerilerinizin harika olduğunu söylüyorlar, baskı altında çalışabilirim son teslim tarihine uymak için sipariş verin. Hızlı öğrenen ve esnek kişi.*

- Test senaryoları, testler, süre, sprint, sürüm:
  - *Duman: Yaklaşık 20 test durumu ve 16 dakika + -20 saniye boyunca yürütülür, her gün saat 7'de yürütülür*
  - *Regresyon: 500'den fazla test senaryosu, 200'den fazla özellik dosyası her biri 2-4 senaryo ve her biri yaklaşık 5,5-6 saat boyunca yürütülür sprint, her sürüm, yeni işlevler eklendikten sonra, büyük hata düzeltildikten sonra*
  - *Her sprint 3-5 kullanıcı hikayesi*
  - *Günde 1-2 test vakası, sprint başına 15-20 test vakası*
  - *Her 2 ayda bir (4 sprint), yaklaşık 14-18 kullanıcı hikayesi, sürüm başına 60-70 test vakası yayınlayın .*

- Bana çevik sürecinizden bahsedin:
  - *C: Çevik süreçler genellikle, sık sık denetlemeyi ve Takım çalışmasını, kendi kendini örgütlemeyi ve hesap verebilirliği teşvik eden bir liderlik felsefesi olan adaptasyon, bir dizi yüksek kaliteli ürünün hızlı bir şekilde teslim edilmesini sağlamayı amaçlayan en iyi mühendislik uygulamaları. Sprintim 2 hafta. Hepsini yapıyoruz scrum törenleri ve ek olarak başka toplantılarımız da var: sprint ortası (2 Salı), yayınlanmadan önce (her 2 ay), tımar ve bilgi aktarımı (her ay) toplantıları. Sprint planlama toplantısı ile sprintimize başlıyoruz, ve uygulamanın geliştirip test edeceğimiz bir kısmını öğreniyoruz. Hakkında genel bir fikir edindikten sonra projesinde, hikayeler için bazı tahmin noktaları ve zaman vermek için sprint tımarı yapıyoruz. Bülteni 2 ayda bir yayınlıyoruz.*

Sprint başladığında, her sabah her gün standup toplantısı yapıyoruz ve dün ne yaptığımızı bugün yapacaklar ve herhangi bir engelleyici varsa. Sprint sonunda genellikle sprint Demosu yaparız. SDET olarak, ekibimde sunumlar yaptı ve işlevlerin üzerinden geçin. Müşteriler, paydaşlar veya iş adamları aşağıdakiler hakkında sorular soracaktır: Anlamadıkları teknik kısım. Sprint demosundan sonra sprint retrospektif toplantısı yapıyoruz. sprint retro'da konuşuruz son sprintte neyin iyi gittiği hakkında. iyi ve kötü ne tür hatalar yaptık! ve onların üzerinden geçiyoruz ve aynı hataları tekrar yapmayacağımızdan emin olun. devam edeceğimiz ve daha iyi olmaya çalışacağımız iyi kısımlar hakkında sprint geliyor.

- Bana ekibinizden bahsedin:
  - *A: Ekibim, motivasyonu yüksek ve kendi kendine organize olan, uyarlanabilir, işlevler arası ve kendi kendini organize eden bireylerden oluşuyor. bilgili. Biz 4 geliştiriciyiz, 3 test ediciyiz - 1 kılavuz, 2 otomasyon, 1 BA, 1 scrum ustasıyız.*
- Mevcut projenizdeki rolünüz ve sorumluluklarınız nelerdir?
  - *C: Bir otomasyon mühendisi olarak, POM tasarım modeline dayanan "test çerçevemi" geliştiriyorum / değiştiriyorum.*
  - *Fonksiyonel test, duman testi, regresyon testi, arka uç testi gibi çeşitli test türleri gerçekleştiriyorum.*
  - *Uygulamada veya sprint sonunda yeni işlevsellik olduğunda regresyon testi yapmaktan sorumluyum veya herhangi bir büyük hata düzeltildikten sonra.*
  - *Ayrıca, ortamın ilk iş olarak çalışır durumda olduğundan emin olmak için duman testi raporlarını kontrol etmekten sorumluyum. sabah. Herhangi bir sorun varsa bunları analiz edeceğim. sunucu sorunuysa, hemen geliştiricilerle iletişime geçeceğim. eğer ilgiliyse betiklerim, onların hatalarını ayıklayacağım. Eğer bir kusursa, onu yeniden oluşturacağım ve kusuru günlüğe kaydedeceğim.*
  - *Şimdi, otomatik test senaryolarının yaklaşık% 80'ine sahibiz. 50 / 50'den önce ana başarımdı*
  - *Ayrıca, Back-end otomasyon faaliyetlerine dahil oldum, bu yüzden otomasyon için huzur dolu kitaplık kullanıyordum, Postman-Manuel API testi için istemci. PostgreSQL DB'lerle API testi yapıyordum*
  - *Jira'yı hata izleme aracı olarak kullanıyorum. Hata geliştiriciler tarafından giderildikten sonra tekrar test ediyorum ve geçerse kapatıyorum. kusur ise sabit değil, biletimi yeniden açacağım. Ayrıca, Agile-Scrum ekibinin bir parçası olarak, çeşitli adım adım açıklamalarına katılıyorum. gereksinim incelemeleri için toplantılar ve İD'ye değerli geri bildirimler sağlar. Son olarak, işlevler arası bir ekip üyesiyim bu, sprint hedefimize ulaşmak için ekibimin her şekilde yardımına her zaman isteklidir. Bu hemen hemen benim rolümle ilgili mevcut projemde otomasyon mühendisi.*
- Bana çerçevenizden bahsedin:

Sayfa 153

  - *Mevcut projemdeki çerçevem Maven tarafından oluşturulmuş, içinde DDT bulunan Selenium Java BDD çerçevesi. Çerçeve POM ve Singleton WebDriver tasarım modeli kullanılarak geliştirilmiştir. Geliştirmeye ve sürdürmeye aktif olarak katılıyorum OOPS'a dayanan yeni sayfa nesneleri, genel ve işlevsel yöntemler ekleyerek çerçeve.*
  - *Çerçeve, JUnit ile Hıyar kullanır ve Veri bütünlüğü Testini JDBC ve API Testini kullanarak yürütebilir. RestAssured kütüphaneleri. Çerçeve, otomasyon komut dosyalarını tüm testlerin yaklaşık% 80'ini yönetir. Uzaktan bağlanılan çerçeve Kaynak denetimi ve Jenkins tarafından Sürekli Entegrasyon için GitHub deposu. Salatalık Json ve html raporları üretir. başarısızlık ekran görüntüleri. İyi organize edilmiş ve anlaşılması ve uyarlanması kolaydır.*
- Toplantılar:
  - Sprint planlaması - 1. hafta 1. gün - Salı
  - Sprint demosu - 2. Pazartesi sabahı
  - Sprint retro - 2. Pazartesi öğleden sonra
  - Her gün saat 10'da günlük stand-up
- Bana uygulamanızdan ve şu anda hangi işlevler üzerinde çalıştığınızdan bahsedin
  - *Uygulamam web uygulaması, geniş bankacılık, yatırım ve aracılık hizmetleri veriyoruz. Şu anda biz ipotek taleplerinde Bireysel yaklaşım işlevselliği üzerinde çalışmak, bu nedenle müşteri ipotek için başvurduğunda, onay sonucu sadece kredi raporuna ve gelire bağlı değildir, aynı zamanda üyelik geçmişi ve varlıkları da önemlidir kararda rol. Varlıklarınızla ipoteğinizi sigorta ettirme veya onları dahil etme fırsatı verir. Bulunan Son Hata şartı, kredi raporundan sonra bu seçeneği seçebileceğinizi söylüyor. Sistem otomatik olarak size minimum gösterir Gerekirse kredinizi sigortalamak için önerilen miktar. Ayrıca manuel olarak girmenize olanak tanır ve minimum sınır değeri. Miktar kutusundan çıktıktan sonra daha az girdiğinizde kutunun yanında hata mesajı görüntülenmelidir. Ancak gerçek sonuç, yatırım türleriyle ilgili farklı bir mesajdı.*
- 2. Penceresiz bir odanız olduğunu, bir kapalı kapınız olduğunu, içinde 3 ampul ve dışarıda 3 anahtar olduğunu hayal edin. Sen kapıyı bir kez açmasına izin verilir. Hangi şalterin hangi ampule bağlı olduğunu nasıl tanımlayabilirsiniz.
- 3. Rest Api hakkında bir saatlik güzel sorular. Pojo dahil neredeyse her şeyi sordular.
- 4. Öğeler listeniz varsa, görünür olup olmadığını kontrol etmek için belirli bir öğeyi nasıl seçebilirsiniz?
- 5. Geliştiricilerle nasıl bütünleşirsiniz?
- 6. Test planı, test senaryosu oluşturma yaklaşımınız nedir?
- 7. Api'nin statik yöntemler kullandığını bildiğiniz gibi, test durumlarımızı yürütmek uzun zaman alıyor, bunları paralel çalıştıramıyoruz, Statik yöntemler kullandığı için, bu yürütme süresini nasıl kısaltabilir ve kısaltabiliriz?
- 8. Kitaplıktaki herhangi bir yöntemin işlevselliğini nasıl kontrol edebilirsiniz, düzgün çalışıp çalışmıyor mu?
- 9. Lütfen çerçevenizde POM'u nasıl kullandığınızı çizin
- 10. Stres testi, birim testi ve entegrasyon testini nasıl gerçekleştirirsiniz?
- 11. Paralel testi uygularken statik nesneler kullandığımızda bu çelişkiler verir, daha iyi önerileriniz var mı?
- 12. Ne tür API'leri biliyorsunuz, hangisini tercih ediyorsunuz, neden?
- 13. xpath ve css arasındaki fark nedir? hangisini tercih edersin? neden? zorluklar nelerdir?
- 14. Sunucuya istek gönderiyoruz ve java nesnesi olarak yanıt alıyoruz, Yanıtı nasıl test edebilirsiniz?
- 15. Junit'i neden Salatalık ile kullanıyorsunuz? TestNG değil mi?
- 16. Bana Salatalığın artılarını ve eksilerini söyleyebilir misiniz?
- 17. API kitaplığınızı nasıl yapılandırırsınız? Yöntemleri bu kitaptan nasıl çağırırsınız?
- 18. Bana kendinizden, çerçevenizden, rolünüzden bahsedin
- 19. Neden bu şirkete başvurdunuz, kültürümüz ve ne yaptığımız hakkında konuştunuz
- 20. Tek birimli test görevi, bir java görevi (kelimenin Palindrome olup olmadığını kontrol edin)
- 21. Şu anda yük dengelerimizle ilgili bir performans sorunu yaşıyoruz (tüm veri akışında grafik sorunu gösteriliyor) bu zorluğu nasıl çözebilirsiniz?
- 22. Arkadaşım şirketinizde Kalite Güvencesi direktörüdür, George'u tanıyor musunuz? ☺
- 23. Postacıdan bahsedin, Postacı kullanarak testi nasıl yaparsınız?
- 24. İsteğinize nasıl gövde ekleyeceksiniz (API)?
- 25. Test verilerinizi nasıl tanımlıyorsunuz? nereden buluyorsun?

• 26. Geliştirme ortamında test yapıyor musunuz?
• 27. Bir çerçeve oluştururken neleri hesaba katarsınız?
• 28. Zaman aşımı zorluklarını nasıl çözersiniz? Selenyumun sağladığı bekleme yöntemlerine güvenmiyorum? Nasıl buldun

zaman aşımı sorunu olduğunda uygun bekleme yöntemi?
• 29. API kitaplığımızda, bir parametreyi kabul eden, sunucuya istek gönderen ve geri dönen bir arama yöntemi vardır.

java nesnesi, bu yanıtı nasıl test edersiniz? (düzenlendi)
• 30. aaaabbcccdda ==> a4b2c3d2a için bir yöntem yazdınız, güzel.! Yönteminizin işe yarayıp yaramadığını kontrol etmek için hangi verileri kullanacaksınız?

doğru ya da değil? *(cevap bu soruların sonundadır)*
• 31. Boş zamanlarınızda ne yaparsınız?

Sayfa 154

• 32. İşyerinde boş kaldığınızda ne yaparsınız? sadece telefonunu alıp oyun mu oynuyorsun?
• 33. Hangi test senaryolarının otomatikleştirileceğine ve nasıl tanımlanacağına kim karar veriyor?
• 34. Ne sıklıkla uzaktan çalışıyorsunuz ve işin üretkenliğini azaltıyor mu?
• 35. Bir kalem aldı ve benzersiz bir figür çizdi ve içine nokta koydu, bu bir harita ve sen buradasın dedi, hangi yollar olacaksın

Bu haritada benzin istasyonlarını tanımlamayı mı seçmelisiniz? Telefonumu kurup google'layacağımı söyledim)

İdris Hamza

• Özgeçmişinizdeki tüm detaylarınıza bakma şansım olmadı, deneyiminizin üzerinden geçebilir misiniz?
• Esas olarak bir UA otomasyon geliştiricisi olduğunuzu söylemek güvenli mi?
• Açıölçer kullandığınızda fenerin hangi versiyonunu kullandınız?
• Selenium ana gücünüz mü?
• XPath ve CSS kullanmak arasında bana biraz fark söyleyebilir misiniz?
• Selenium Grid'i daha önce kullandınız mı?
• Veritabanları konusunda ne kadar rahatsınız?
• Oracle ile çalıştınız mı?
• Oracle Şemasında iki tablo vardır. İçlerinde bazı veriler var. İş, bu verilerin aynı olduğunu doğrulamaktır.

Buna nasıl yaklaşırsın?
• Komut Satırından memnun musunuz yoksa sürekli Windows üzerinde mi çalışıyorsunuz?
• Grep Komutunun ne için kullanıldığını bana söyleyebilir misiniz?
• Yani Postman kullanıyorsunuz. Söyle bana, RestAPI'ye aşina mısın?
• Öyleyse, AWS sertifikasına sahip olduğunuzu görüyorum, daha önce hangi hizmetleri kullandınız?
• Herhangi bir Performans Testi yaptınız mı?
• Java veya JavaScript kullanıyor musunuz?
• Java'da daha önce JSON verileriyle çalıştınız mı?
• Java'da bir String'i tam sayıya ve tamsayıyı String'e nasıl dönüştürebileceğimi söyleyebilir misiniz?

Zhansaya Zhangabatyrova

Fannie Mae ile teknik telefon gösterimleri (iki kez)
Telefonda iki teknik telefon tartışmasını geçtim, ancak WebEx toplantısında kodlar başarısız oldu.
• Java'da yapıcıyı aşırı yükleyebilir misiniz?
• Java'da ne tür tasarım kalıpları biliyorsunuz? (Sadece Singleton modelini açıklayabilirim)
• Java - değere göre geçer mi yoksa referansla geçer mi? (Doğru olduğundan emin değildim, ancak Java'nın geçtiğinden eminim.

Değer olarak, geçen parametrelerin bir kopyasını kullandığımız için, Akbar'ın bunu açıkladığını hatırlıyorum, o zaman unuttum)
• Java'da bellek sızıntısı nedir?
• Arayüz ve soyut sınıf arasındaki fark?
• Yapıcıda yöntemler nasıl elde edilir?
• HashMap ve HashTable arasındaki fark?
• Bir istisnanın ele alınıp alınmadığına ilişkin Son olarak blok yürütülecek mi?
• Dizide yinelenenleri bulmanın mantığını açıklayın?
• Dizi ve ArrayList arasındaki fark?
• Vektör nedir?
• Lambda ifadeleri? (Tanrıya şükür, daha önce okudum, bir şekilde açıklayabilirim)
• İşlevsel arayüz? (Bunu lambda'dan sonra takip etti, ama ben halledebilirim)
• 14. TestNG'de dinleyiciler nelerdir? (Ben kesinlikle bilmiyordum)
• TestNG'de test senaryoları nasıl gruplandırılır?
• Selenyum Bileşenleri?
• Selenium'da çapraz tarayıcı testi?
• ChromeDriver nasıl başlatılır?
• Selenium'da bekler mi?
• Birden çok sekme nasıl kullanılır?

Sayfa 155

Petrol ve gaz şirketi teknik telefon görüşmesi
• Selenyum türleri bekleme

• Mutlak eski yol ve göreli eski yol arasındaki fark
• Mevcut projenizde Jenkins'i nasıl kullanıyorsunuz?
• Maven'deki uygulayıcı eklentisinin işlevi nedir?
• Git zula?
• Projenizde Selenium grid kullanıyor musunuz? Neden?
• Soft assert ile hard assert arasındaki fark
• Selenyumdaki yer belirleyici türleri?
• Maven'de enstantane ile yayınlama arasındaki fark nedir?
• Maven'de bağımlılık kalıtımı nedir?
• Selenyum istisnaları
• Sayfa nesne modeli?
• DDF'nin hangi uygulamalarını kullanıyorsunuz?
• Test NG'sindeki ek açıklamaya bağlıdır
• Salatalıkta paralel test?
• İmleci Selenium'da herhangi bir konuma nasıl taşıyabilirsiniz?
• Testiniz başarısız olduğunda nasıl ekran görüntüsü alırsınız?
• Bir test senaryosunu otomatikleştirmeye nasıl karar verirsiniz?

Walmart laboratuarları F2F
1. F2F teknik röportajının ilk adımıydı (telefon taraması yok). 2 adamdı -> 1 geliştirici ve İK.
• bana projenden bahset
• API'yi nasıl test edersiniz
• henüz geliştirilmediyse API'yi nasıl test edersiniz?
• Henüz tasarlanmamışsa ve kabul kriteriniz yoksa API'yi nasıl test edersiniz?
• *** adında bir araç duydunuz mu (adını hatırlamıyorum) -> kullanılmadığında API'leri test etmek için kullanıldığını açıkladı
henüz tasarlandı mı?
• Jenkins kullanıyor musunuz
• nasıl bir iş kurduğunuzu bana söyleyebilir misiniz?
• yani geliştirici kodu GitHub'a gönderildiğinde testiniz çalışır mı? (gerçekten değil, nasıl çalıştığını açıkladım)
• yani, geliştiricilerle ayrı bir ortamınız mı var?
• git'i nasıl kullanıyorsunuz?
• kahvaltı yemeğinizin nasıl çalıştığını bana söyleyebilir misiniz?
• işlevselliği nasıl test edersiniz?
• bir hata bulduğunuzda ne yaparsınız?
• aws hakkında ne biliyorsun?
• herhangi bir .net aracı / hizmeti biliyor musunuz?
• gelecekte bir geliştirici olmak ister misiniz?
• projedeki tek test kullanıcısı olabilir misiniz?
• işyerinizde ne arıyorsunuz?
• sizin için harika bir çalışma ortamı nedir?
• boş zamanlarınızda ne yapmaktan hoşlanırsınız? -> ve seyahat hakkında çok konuştuk
• Walmart'ta alışveriş yapıyor musunuz?
• deneyiminiz nasıl?
• bununla ilgili en çok neyi seviyorsun?
• neyi değiştirirdiniz?

2. İlk adımı geçerseniz size küçük bir proje ve bunu yapmanız için 1 hafta süre verirler.

Sayfa 156

3. Son adım, verilen projenin mantığını ve uygulamasını açıkladığınız 3 saatlik F2F'dir. Lütfen görev dosyasına bakın
ekli

Ne yazık ki, aynı işe alım görevlisine sahip başka bir şirketten teklif aldığım için seçimimi yapmak zorunda kaldım ve
bir teklifi reddetmek ve Walmart ile devam etmek ya da bir teklif almak ve sonunda yaptığım Walmart ile son f2f'yi iptal etmek.
Görevin bir mantığını uyguladım; ancak çok fazla çalışma, optimizasyon, farklı durumlar için ek mantık, çok fazla
f2f'yi iptal ettikten sonra uygulamaya devam etmediğim tasarım çalışması. Bununla birlikte, herhangi birine yardımcı olabilirse, bir GitHub bağlantısı vardır:
https://github.com/Skarlet03/ProjectForWal.git

==========

not: F2F'de 3 kişi. Geliştirici üzerimde çok baskı yaptı ve hatta programın bir noktasında saygısızlık bile yaptı.
röportaj, ancak kapıdan çıktıktan 5 dakika sonra bir teklif aldım. Sadece kendinize güvenin.
• Çerçevenizi duymak istemiyorum Son 3 yıldır hangi işlevler üzerinde çalıştığınızı görmek istiyorum
• bana daha fazla bilgi ... daha fazla ... proje hakkında daha fazla bilgi (adam ona çok fazla içeri giremeyeceğimi söyleyene kadar durmadı
ayrıntılar)
• bana yakın zamanda bulduğun tüm hataları söyle
• bana 5 tane daha ver
• bu belirli işlevi nasıl test ettiniz?
• Tasarıma ihtiyacım var
• veritabanını nasıl test ettiniz?
• veritabanınızda 03-Ocak-2019 ve kullanıcı arayüzünüzde 01-03-2019 var, java kullanmadan tüm tarihleri böyle nasıl değiştirirsiniz

- Bir metin dosyası belge girişiniz var (özgeçmişiniz), java kelimesinin ("ja") ilk bölümünün kaç kez göründüğünü sayın
  belge (beyaz tahta)
- veritabanında üç rakamınız var, ancak kullanıcı arayüzünüzde çizgiler ve bazı başka karakterler var. Nasıl karşılaştırırsın
  (beyaz tahta)?
  - o 123456 -> A12-34-56
  - o 321654 -> M32-16-54
  - o 789456 -> K78-94-56
- Java regex nedir?
- Bir normal ifade için kullanabileceğiniz .matches () dışında başka bir yöntem var mı?
- SQL kısıtlamaları nelerdir
- uygulamanızın yapısı (beyaz tahta)
- çerçevenizin yapısı (beyaz tahta)
- test verilerini nerede saklıyorsunuz?
- 2 özel nesneyi nasıl karşılaştırırsınız?
- Paralel çalışmayı nasıl yaparsınız (tüm olası seçenekler istendi)
- Jenkins işi
- Son iki şirket hakkında pek çok soru: Örnek: Okul yanında nerede, orada çalışan insanların sizi yaptığını biliyorum
  bunu ve bu kişiyi bil
- Navy Federal'deki proje yöneticiniz nasıldı? Yönetmeniniz nasıldı?
- Bana şu anki projenizden bahseder misiniz?
- Şirketlerde vb. Çalıştığım yıllarda beni yakalamaya çalışıyorum.
- Temel ve basit Java soruları
- Temel ve basit Salatalık ve Selenyum sorusu
- Temel SQL soruları

Sayfa 157

string elde etmek için bir yöntem yazın ve aşağıdaki gibi geri dönün:
aaaabbcccdda ==> a4b2c3d2a

```
public class returnWord {
	public static void main (String [] args) {
		System.out.println (getword (“aaaabbcccdda”));
	}
	public static String getword (Dize kelimesi) {
		int count = 1;
		Dize ret = “”;
		for (int i = 0; i <word.length () - 1; i ++) {
			eğer (word.charAt (i) == word.charAt (i + 1)) {
				count ++;
				devam et;
			}Başka {
				ret = ret + word.charAt (i) + sayım;
			}
			sayım = 1;
		}
		ret = ret + kelime.charAt (kelime.length () - 1);
		dönüş ret;
	}
}
```