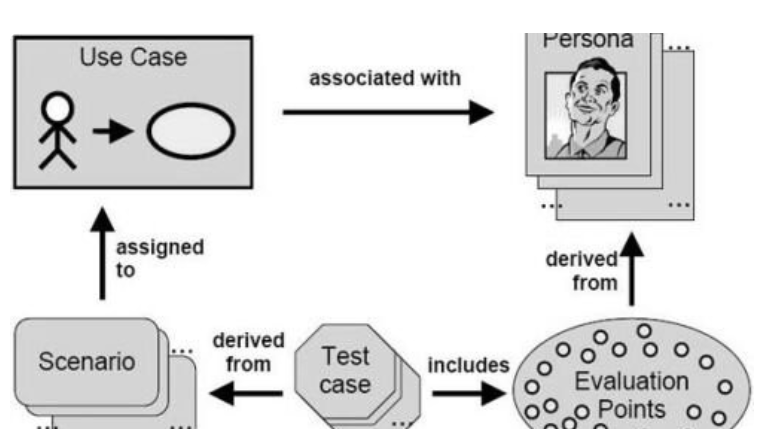
**Test Senaryoları, Durumları ve Adımları Hakkında Her Şey!**



Pek çok takımın, test dokümantasyonu üzerine sorunlar yaşadığını en azından kendi çevremden biliyorum. Buradaki genel hata ise; test üzerindeki kavramların yanlış anlaşılması. Konu üzerine aslında sadece bir doğru yaklaşım yok. Bugüne kadarki tecrübelerim, okuduklarım ve uluslararası test örgütlerinin yaklaşımlarını karşılaştırdığımda bu yazıyı yazmaya karar verdim. Yazıyı yazmama bir diğer etken ise; üst yönetimlerin, maalesef hala, test senaryolarının sayısının raporlanmasını istemesi. Daha önce de anlatmaya çabaladığım gibi, test sayısı hatasızlığı sağlamaz. Risklerinizi ne kadar azaltırsanız, kalitenizi o kadar yukarıya çekebilirsiniz.

Umarım istediğim gibi kısa, öz ve basit aktarabilirim. Zira bu da sektörün kanayan yaralarından biri. Bu yüzden, teknik açıklamalardan sonra, kısa bir örnekle aşağıdaki teknik açıklamaları somut hale getirmeye çabalayacağım.

**Test senaryosu nedir?**

Test senaryosu, test gerekliliklerinin, fonksiyonelitiye göre üst seviye bir sınıflandırılmasıdır. Bu dokümantasyonda test edilecek fonksiyonların senaryolarını dokümante etmeniz beklenir ve çok detaylı yazılmazlar.

Kafanızda daha iyi canlandırmanız adına biraz daha açmak gerekirse. Test edilecek bir modülünüz var, bu modülün fonksiyonlarını test etmek için belirlediğiniz kriterlerin dokümantasyonudur.

**Test durumu (Test Case) nedir?**

Test durumu, başarılı ve başarısız yöntemlerle koşulabilecek test senaryosu prosedürleridir. Aslında türkçe olarak anlatacak olursak, bir test senaryosu sonunda başarılı bir işlem gerçekleşmesini bekliyorsanız başarılı bir test durumunuz olur. Tam tersi olarak, test senaryosu sonunda başarısız bir işlem bekliyorsanız negatif bir test durumunuz olur. Tabii bu noktada, pozitif test durumları başarılı, negatif test durumları başarısızdır yargısı oluşmamalıdır. Negatif test durumlarınızın sonunda negatifi yaklamalısınız, pozitif test durumlarınızda ise pozitifi yaklamalasınız. Eğer test sonuçlarınızda elinizdeki sonuçlar eşleşiyorsa, testinizi başarılı ilan edebilirsiniz.

**Analiz dokümanındaki her gereksinime karşılık sadece bir senaryo mu olmalı mıdır?**

Aslında bu soruya hem evet, hem hayır denebilir. Buna sebep ise; testin içerik bağımlı olmasıdır. Eğer test ettiğiniz fonksiyon sadece bir butona tıklandığında içerik detayının getirilip getirilmediği ise burada sadece bir test senaryosu yeterli olabilir. Ancak, bir uçaktaki yere yaklaşma sistemini test ediyorsanız sanırım bir gereksinme karşılık 100 testiniz bile olabilir.

Anlatmak istediğim, aşağıdaki örneği de inceleyecek olursanız, bir ekrandaki testleri bir senaryo altında toplayıp, yüzlerce adıma sahip bir test senaryosu oluşturmayın. Senaryolarınızı mümkün olduğunca bölüp, test adımlarını her test senaryosu için ayrı yazmanız daha doğru olacaktır. Böylece, geliştirme ekibiniz de sistemde nelerin test edildiğini başlık halinde görme şansına sahip olacaktır.

**Test Adımı nedir?**

Test adımı, bir ekrandaki test edilecek şeylerin tek tek yazıldığı yer değildir. Test adımı, bir test senaryosunun gerçekleştirilebilmesi için gerekli adımlarının açıklandığı dokümantasyondur. Aşağıdaki örnekte de görebileceğiniz gibi, yazılan senaryoyu gerçekleyen işlemleri yaptıran açıklamalar ve bu atılan adımlar doğrultusunda, sistemin bize her adımda vermesi gerektiği cevapları içeren dokümantasyondur.

**Örnekleyecek olursak:**

**Fonksiyon:**Login Fonksiyonu

**Ön Koşul:**

* Kullanıcının login olabilmesi için, daha önceden başarılı bir üye kaydı olmalıdır.

**Gereksinimler:**

* Kullanıcının eposta adresini girmesi için bir text box olmalıdır.
* Kullanıcının şifresini girmesi için bir text box olmalıdır.
* Kullanıcının bu bilgileri server a göndermesi için bir buton olmalıdır.

**Test Senaryoları:**

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Test Adımları:**

Burada yukarıdaki tüm senaryolar için değil sadece bir senaryo için test adımı yazacağım.

|  |
| --- |
|  |

**Sorular:**

Analiz dokümantasyonundaki eksikler de aslında biz testçilerin gözünden geçer. Bu dokümantasyonların testlerine statik testler denir. Aslında bu soru şu an için geç kalınmış bir sorudur. Bir önceki bahsettiğim özgür testçinin değeri yazısında, testçinin işe ne zaman başlaması gerektiğine de değineceğim.

* Loginin başarılı bitmiş sayabilmek için; başarılı server gönderimi yeterli midir? Yoksa bu gönderimden sonra özellikle görmesi gereken bir sayfa var mıdır?

**Toparlayacak olursak:**

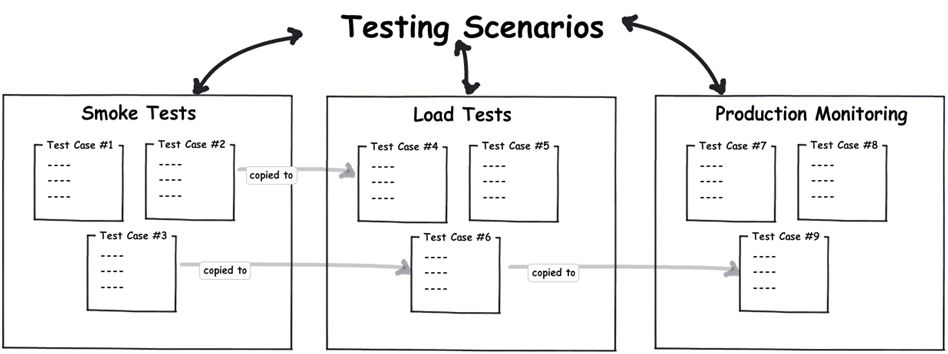
Her yiğidin yoğurt yiyişi tabiki farklıdır. Dokümantasyon konusunda takımdan takıma fark eden gruplama tarzları olabilir. Ekipten ekibe, kullanım farkları olabilir. Tüm bunların yanında global bırakılması gereken kısım ise; bir gereksinim için bir senaryo yazıp altına tüm ekranın testini bir senaryo üzerinde takip edilmemesi kısmıdır.

Testi daha takip edilebilir, geliştirme ekiplerine daha faydalı hale getirebilmek adına test senaryolarınızı ayırın. Ayrılmış test senaryoları ile, yazılımcı arkadaşlara başarılı ve başarısız testleri aktardığınızda, kodun neresine daha çok çaba harcamaları gerektiğini bile ayırabilirler. Bu durumda da bulgu kümelenmelerini(defect clustering) bile çözen bir yapıya gidebilirsiniz.

Sizlere tavsiyem, testin bu yapısını üst yönetimlerinize aktarmaya çabalayınız. Bu durumda sizlere test senaryo sayılarını değil, kodun kalitesini sormaya başlarlar. Bu durumda da, kalite için tek başınıza değil; bir ekip olarak çalışır hale gelirsiniz. Bu durumda müşterilerinize ve oradan da tekrar size mutluluk olarak yansır.

Bol bulgulu günler dilerim. Unutmayın bulunan her bulgu, mutlu bir müşteri demektir.

Test senaryoları örneklerle nelerdir?



Smoke Test (Duman Testi) Nedir?

Smoke Test (Duman Testi), geliştirilen yazılımın fonksiyonlarının doğru çalışıp çalışmadığını kontrol etmek amacıyla yapılan yazılım test sürecidir. Yazılım fonksiyonlarını test etmek için her yapı üzerinde yürütülen bir dizi testten oluşur. Amaç, hataları erkenden tespit edip önleyerek maliyet ve zaman kaybının önüne geçilmesidir. Duman testleri, sistem kararlılığını ve gereksinimlere uygunluğunu göstermek için tasarlanmıştır.

Neden Smoke Test Yapıyoruz?

Smoke Test, sistemin ilk aşamalarda doğruluğunu sağladığı için yazılım geliştirmede önemli bir rol oynar. Dolayısıyla, maliyetten ve zamandan tasarruf edebilir. Duman testi ile büyük hataların tespiti ve düzeltilmesi kolaylaştırılır. Kusurların çoğu yazılım geliştirmenin ilk aşamalarında belirlenir. Sonuç olarak Smoke Test, sistemi iyi bir duruma getirir. Duman testini tamamladıktan sonra ise fonksiyonel teste başlanır.

Smoke Test (Duman Testi) Nasıl Yapılır?

Duman Testi genellikle manuel olarak yapılır, ancak aynı şey otomasyon yoluyla da gerçekleştirilebilir. Organizasyondan organizasyona değişebilir.

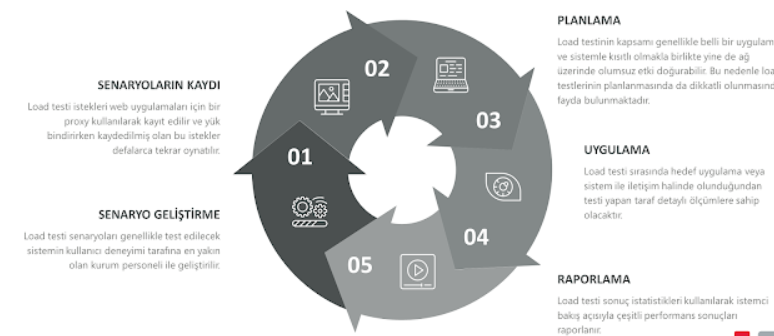
Manuel Duman testi

Genelde duman testi manuel olarak yapılır. Test geçerse, fonksiyonel teste devam edilir. Test başarısız olursa, yapı reddedilip düzeltme için geliştirme ekibine geri gönderilir ve yeni sürümle duman testi yeniden başlatılır. Duman testi yeni yapı üzerinde gerçekleştirilir ve sistemin doğruluğunu korumak için eski yapılarla entegre edilir.

Otomasyon ile Duman Testi

Yeni yazılım versiyonu dağıtıldığında manuel olarak testi tekrarlamak yerine, yapıya karşı kaydedilen duman testi durumları yürütülür. Ana işlevlerin hala düzgün çalışıp çalışmadığını doğrular. Test başarısız olursa, yapıyı düzeltebilir ve yapıyı hemen yeniden dağıtabilirler. Bu sayede zamandan tasarruf edebilir ve kaliteli bir yapı sağlanır. Test mühendisi, otomatik bir araç kullanarak, yazılım oluşturmada gerçekleştirilen tüm manuel adımları kaydeder.

LOAD TESTİ (YÜK TESTİ) METODOLOJİSİ NEDİR?



LOAD (YÜK) testi metodolojisi şu adımları içerir:

Load testi senaryoları genellikle test edilecek sistemin kullanıcı deneyimi tarafına en yakın olan kurum personeli ile geliştirilir. Bu personel bir sistem analisti veya iş birimi temsilcisi olabilir. Gerçekçi kullanım durumları göz önüne alınarak belirli uygulama kullanım senaryoları ortaya çıkarılır. Bu senaryolara toplam gerçekleştirilecek istekler içindeki oranları da verilebilir.

Load testi istekleri web uygulamaları için bir proxy kullanılarak kayıt edilir ve yük bindirirken kaydedilmiş olan bu istekler defalarca tekrar oynatılır.

Load testinin kapsamı genellikle belli bir uygulama ve sistemle kısıtlı olmakla birlikte yine de ağ üzerinde olumsuz etki doğurabilir. Load testinin canlı bir sistem üzerinde gerçekleştirilmesi halinde ise iş süreçlerinin olumsuz etkilenme riski daha da yüksek olacaktır. Bu nedenlerle load testlerinin planlanmasında da dikkatli olunmasında fayda bulunmaktadır.

Load testi sırasında hedef uygulama veya sistem ile iletişim halinde olunduğundan testi yapan taraf detaylı ölçümlere sahip olacaktır. Bu veriler kullanılarak istemci bakış açısıyla çeşitli performans istatistikleri raporlanır.

Load testi sırasında sistem yöneticilerinin de sunucu metriklerini takip etmesinde fayda bulunmaktadır. Bu şekilde sunucu üzerindeki kaynak kullanım istatistikleri de görülmüş olur ve bu istatistikler de load testi raporuna aktarılabilir. Test ekibi bu alanda sistem yöneticilerine destek olabilir.