**Software Testing (Yazılım testi) hiyararşisi**

**Functional Test**

|  |
| --- |
| **Whitebox**(beyaz kutu testi)(kodlar üzerinden yapılan test) teknik bilgi gerektirir.  Developer’lar tarafından yapılır.Kodların iç yapisini ele alır.   * Dahili güvenlik açıkları * Kodlama işlemlerinde bozuk veya kötü yapılandırılmış yollar * Kod üzerinden belirli girdilerin akışı * Beklenen çıktı * Koşullu döngülerin işlevselliği (conditional loops) * Her ifadenin, nesnenin ve fonksiyonun ayrı ayrı test edilmesi   (**unit test** whitebox’a girer) Unit Test (Birim testi) dinamik testin ilk düzeyidir ve önce Developerlarin ve ardından test mühendislerinin sorumluluğundadır.Birim testi, functionality hazır olunca bireysel olarak test edilir (Birbirinden ayrı bir şekilde) |
| **Blackbox**(karakutu testi) (kodlar gorunmez urun uzerinden test yapilir).illa teknik bir alt yapi gerektirmez. Kodlar uzerinden değil ürün üzerinde test yapilir |

**Blackbox Testleri**(karakutu testleri) (tester’lar yapar)

|  |  |  |
| --- | --- | --- |
| **Smoke test** ( tester’lar yapar, devepoler’larda bilir)  Unit Test, integration Testi, System/End To End Test ve UAT içerisinde  kritik olan en önemli functionality’lerin düzenli olarak çalışıp çalışmadığını gözden geçirmek için yapılır.  (sabah ilk iş olarak smoke test yapılır) | | |
| **integration (Entegrasyon)Testi**   * (kompanentleri birarada çalıştırma testi) * Birim testi tamamlandıktan sonra başlar * Amaç; Uygulamanın farklı bileşenlerinin müşteri gereksinimlerine göre çalışmasını sağlamaktır * Gerçek ve beklenen sonuçlar aynı olduğunda veya farklılıklar istemci girdisine göre açıklanabilir / kabul edilebilir olduğunda, entegrasyon testinin tamamlandığı kabul edilir. | **System/**  **End To End Test**  (baştan uca sistemin tamamının uyum içinde çalışması testi)   * Entegrasyon testinin tamamlanmasının ardından sistem testi başlatılır. | **UAT**  **(User Acceptance Test)**  (kullanıcı kabul testi)   * Son kullanıcının veya müşterinin spesifikasyonlarına veya son kullanıcılar/müşteriler tarafından sınırlı bir süre boyunca kullanıma dayalı nihai test. * Son Asama olarak her sey gözden gecirilir. |
| **Regression test** (Gerileme Testi) (testerler yapar)  Unit Test, integration Testi, System/End To End Test ve UAT içerisinde yapılan değişikliklerin yan etkilerinin izlenmesi ve mevcut olan hatalarında başka hatalar oluşturup-oluşturmadığının kontrol edilmesi için yapılan testlerdir. Regresyon Testi Gereksinimi; Yazılıma yeni özellik eklendiginde, Defect fix edildiginde, Performans sorunu düzeltmesi halinde, her splintten sonra veya 3 aylık dönemlerde yapilir  Minor(küçük) regression=> tüm functionalitylerin bir sprint sonunda test edilmesi  Major regression=> 3 aylik bir donemde tum reguirement (functinality)larin test edilmesi | | |
| **ReTesting=** hata anında yapılır. Hataların fix edilip edilmediğini görmek için yapılır | | |
| **Ad-hoc (monkey)** herhangi bir test yaptığımızda buna monkey test denir | | |
| **Sanity test**=kullanıcıya verilebilecek kadar olgun olan bir versiyon üzerine yeni eklenen bir özellik veya bug çözümü ardından yapılan ilk testtir. Aslında sadece bu özelliklerin ve bug'ların çözümü kontrol edilerek devamındaki testlere engel bir durum olup olmadığı test edilir.  Yani smoke test ile ürünün ana fonksiyonları, sanity test ile çözülen bug'ları ve eklenen özellikleri test edilir. | | |

**Non-Functional Testing**

|  |
| --- |
| **Performance testing**; uygulamanın hızı test edilir |
| **Stress testing** ; toplu bir şekilde birçok taraftan yük ve hız testi (alışılmadık derecede ağır yükler altındayken sistem fonksiyonel testi, belirli eylemlerin veya girdilerin ağır tekrarı, büyük sayısal değerlerin girilmesi, bir  veritabanı sistemine büyük karmaşık sorgular vs. aktarimi.) |
| **Load Testing**; (Sistemin yanıt süresinin hangi noktada düştüğünü veya başarısız olduğunu belirlemek için bir uygulamayı bir dizi yük altında test etmek gibi bir uygulamayı ağır yükler altında test etme.) |
| **Security test** Sistemin tüm potansiyel açık kapıları ve zayıflıkları araştırılır. |

**Kara Kutu Test Teknikleri**

Kara kutu test teknikleri ile projelerinizde daha efektif test senaryoları tasarlayabilirsiniz.

|  |
| --- |
| Equivalence partitioning =kategorilere bölerek test etme (eşit bölümlere ayırma) |
| Boundary value analysis = min ve max değerler üzerinden test etme (sınır değer Analizi) |
| Decision Table Testing= Karar Tablosu Testi |
| Use Case Testing= Kullanım Durum Senaryosu Testi |
| State Transition Testing= Durum Geçiş Testi |

**Kara Kutu Test Teknikleri Nelerdir ?**

Test tekniklerine ihtiyaç duyulmasını sebebi yazılan belli başlı test senaryoları arasında hedefe en kısa zamanda ve etkili şekilde ulaşılabilecek olanları seçmektir. Her teknik belli tarzdaki hataların bulunmasında kullanılır. Kara kutu test tekniklerinin amacı yapılacak testler için aynı etkiye sahip daha az kombinasyondan oluşan test senaryoları yazmaktır.

**Kara kutu test teknikleri aşağıdaki gibidir:**

1-Denklik Paylarına Ayırma --Equivalence Partitioning (EP) (kategorilere bölerek test etme)

2-Sınır Değer Analizi --Boundary Value Analysis (min ve max değerler üzerinden test etme)

3-Karar Tablosu Testi --Decision Table Testing

4-Kullanım Durum Senaryosu Testi --Use Case Testing

5-Durum Geçiş Testi --State Transition Testing

**1-Denklik Paylarına Ayırma --Equivalence Partitioning (EP) (kategorilere bölerek test etme)=** amacı benzer özelliklere sahip test senaryolarını gruplandırılarak daha az test senaryosu yazılmasıdır. Bu tekniğe göre aynı çıktıyı veren test girdi grubu için bir tane test senaryosu yazılması yeterli görülür. Bu test tekniği tüm test seviyelerinde uygulanabilir. Denklik sınıflarının doğru belirlenmesi test tasarımınızı ve dolayısıyla yapacağınız testlerin kalitesini olumlu şekilde etkileyecektir.

**2-Sınır Değer Analizi --Boundary Value Analysis (min ve max değerler üzerinden test etme)** Sınır değer analizi test tekniği tüm test seviyelerinde uygulanabilir. Uygulaması kolay hata bulma becerisi yüksektir. Bu teknik özellikle denklik paylarına ayırma test tekniği ile kullanılırsa daha etkili sonuçlar elde edilebilir.

**3-Karar Tablosu Testi --Decision Table Testing** Karmaşık iş kurallarına sahip sistemlerin test edilmesinde kullanılan test tekniğidir. Karar tablosu test tekniğinin en büyük avantajı test sırasında gözden kaçabilecek olasılıkların net bir şekilde listelenerek gözden kaçırma riskinin en düşük seviyelere indirilmesidir. karar tablosunu çıkartarak daha etkili test senaryoları yazabilir kritik senaryoları gözden kaçırma riskini en aza indirebilirsiniz.

**4- Kullanım Durum Senaryosu Testi**

Kullanım durum senaryoları sistem ile aktör arasındaki etkileşimi göstermek için yazılan senaryolar topluluğudur. Kullanım durum senaryosu testinin amacı sistemi baştan sona her bir senaryonun teker teker bütün sistemi kapsayacak şekilde test edilmesidir. Kullanım durum senaryolarında ana senaryo ve alternatif senaryolar bulunur. Ana senaryolarda sistemin yerine getirmesi gereken adımlar yer alır. Alternatif senaryolarda ise sistemin uç noktaları için test adımları yazılır.

Örneğin bir kullanıcı tanımlama ekranında ana senaryo ekranda yer alan tüm zorunlu alanların doldurularak kaydedilebilmesi iken alternatif senaryoda ekranda yer alan bir veya birden fazla zorunlu alanın boş bırakılarak sistemin verdiği tepkinin ölçülmesi olabilir.

**5- Durum Geçiş Testi**

Durum geçiş testinin amacı belli iş kurallarına bağlı olarak şartların oluşmasına ve bir durumdan diğerine geçilerek bir noktada sonlanması durumunu test etmektir. Bu tür sistemlerde bu durumlar durum geçiş diyagramı ile gösterilir.

Örneğin bir ATM örneğinde ilk olarak PIN kodu bekleniyor daha sonra girilen PIN kodu kontrol ediliyor eğer PIN kodu doğruysa kullanıcı hesabına erişebiliyor 3 hatalı PIN kodu girişinde ise ATM kartı yutuyor.

Durum geçiş test tekniği genellikle gömülü sistemlerin testlerinde kullanılmaktadır.