**Desktop Application Test**

Test otomasyonu, performans artışı, hataları minimum seviyeye indirme ve daha hızlı çözümler bulmak gibi kolaylaştırıcı çözümler bütünüdür. Masaüstü test uygulamaları, ayrıca uygulamanın güvenirliği, kullanılabilirliği, kararlılığı gibi kullanıcı odaklı servisleri de test eder. Kısaca;

Functional Testing, Backend(Database) Testing, GUI Feature Testing(Grafik), Memory Defect Testing, Load Testing, Compability Testing gibi çeşitlere ayrılır. Ayrıca test literatüründe Black Box Testing ve White Box Testing kavramları sıkça kullanılır. Black Box, yazılımın işlevsellik görevini yerine getirip getirmediğini kontrol ederken; White Box, yazılımın kod düzeyde test edilmesidir. Kaynak kod, veritabanı ilişkileri bu aşamada kontrol edilir. Yazılım testleri birim, entegrasyon ve sistem testi olmak üzere aşamalandırılabilir. Birim testinde yazılan fonksiyon veya prosedürlerin tutarlı ve işleve uygun olup olmadığı test edilir. Entegrasyon testinde ise sisteme entegre edilen alt sistemler sırasıyla farklı zamanlarda test edilir. Sistem testinde ise black box ve White box test aşamaları tamamlanmak üzere sistemin güvenliği de denenir.

Özel olarak Desktop Application Testleri ise;

* Performans Testi
  + Apache Jmeter
  + Citrix-Tru Client
  + Load Complete
  + Neoload
  + IBM
  + Rational Performance Test
* Backend Testi
  + DB(Veritabanı) Testing
  + API Testing
  + Server Testing

olarak sıralanabilir.

Bu testler için kullanılan Toollar ise şekilde gibidir.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| WinApp Driver | Winium | Test Architect | | Jubula | Tricantic Tosca |
| * Open Source * UWP * Windows Forms | * Windows Automation * WPF(Windows Foundation) | * Test Runner | * Java Test Application | | * UI Test * API Test * Test Data Managment |

Application Programming Interface, yazılım bileşenlerinin birbirleri ile iletişim kurmalarını kolaylaştıran bir araçtır. Bir kullanıcı programının, şirketin veri tabanındaki depolanmış verileri almak için kullandığı veri kümesi olarak da adlandırabiliriz. Örneğin blockchain sistemi kullanan bir sistem merkezi olarka API dağıtır. Programcı ise bu API’den verilerine veya kendi sitelerine entegre sağlarlar. Özetle sistemin sahip olduğu kaynakları, belirli koşullar altında dışarıdaki kullanıcılara servis edilmesidir.Rest ve Soap, başlıca API türlerindendir. API Testleri başlıca, UI, End to end, Integration ve Unit Testlerden oluşmaktadır.

Genel çerçeve de bir yazılım ürünü için testler şu şekilde özetlenebilir.

|  |  |
| --- | --- |
| ***Static Testing*** | White Box testler için geçerlidir. Programı run etmeden önce buglardan arındırmaya çalışmak olarak nitelendirilebilir. Kodu tekrar okumak bir Static Test örneğidir. |
| ***Dynamic Testing*** | Bu test türü genel de Black Box olarak değerlendirilir. Program, önceden belirlenen denemelere tabii tutulur, beklenen output ve elde edilen output kıyaslanır. Random yapılan testler veya gerçek zamanlı uygulanan testler Dynamic Test örneği olarak gösterilebilir. |
| ***Visual Testing*** | Programın yüzeysel olarak eksiklerini veya yapısal bozukluklarını kontrol eden test türüdür. |
| ***Component Interface Testing*** | Arayüz Testi, iki farklı yazılım sistemi arasındaki iletişimin doğru yapılıp yapılmadığını doğrulayan bir yazılım testi türü olarak tanımlanır.  İki bileşeni birleştiren bir bağlantıya arabirim denir. Bir bilgisayar dünyasındaki bu arayüz API'ler, web servisleri vb. gibi herhangi bir şey olabilir. |
| ***Mutating Testing*** | Mevcut testlerin kalitesini değerlendirmek için uygulanır. Daha kaliteli, daha güncel test yaratmak için kullanılır. |
| ***Fault Injection*** | Sistemin alışmadık, beklenmedik durumlarda vermiş olduğu reaksiyona dayalı bir test tekniğidir. Programa hataları, bugları tanıtarak uygulanabilir. |
| ***Code Coverage*** | Bir kodun ne kadar test edildiğini kontrol eder. |
| ***API Testing*** | API’nin isteklerinin, parametrelerinin, işlevselliğini doğrulamak, sınır değer analizi yapmak ve hataları ortaya çıkarmak için yapılmaktadır. Farklı uçlara dokunan uygulamalar ve sistemler arasında veri alışverişinin test edilmesi ve işlemlerin verimliliğinin gözlemlenmesi için yapılan testlerdir. |
| ***GUI Testing*** | Grafik kullanıcı arayüzü, son kullanıcıların bir yazılım ortamından faydalanmasına yardımcı olan bir yazılım aracıdır. |
| ***Integariton Test*** | Unit Test aşamasından sonraki aşamalarda uygulanması daha doğrudur. Birimlerin birbirleri ile olan ilişkisini test etmek için kullanılan bir tekniktir. Application Framework’ler ile birlikte çalışabilir. |
| ***Unit Test*** | Unit Test, bir yazılımın en küçük test edilebilir bölümlerinin, tek tek ve bağımsız olarak doğru çalışması için incelendiği bir yazılım geliştirme sürecidir. Buradaki amacımız yazılımın her biriminin tasarlandığı şekilde gerçekleştiğini doğrulamaktır. Geliştiricilerin kendileri yazar ve kontrol ederler. |