KAHOOT ----- SQL

1. SQL'in açılımı nedir?
2. "SELECT" SQL komutu ne işe yarar?
3. FROM SQL yan tümcesi aşağıdakiler için kullanılır...
4. SQL ifadelerinden hangisi doğrudur?
5. SQL'deki mantıksal işleçler nelerdir?
6. Aşağıdakilerden hangisi bir SQL toplama işlevidir?
7. Bir tablodan veri sorgulayan veya okuyan SQL ifadesi \_\_\_\_\_'dir
8. Aşağıdakilerden hangisi SQL SELECT deyimleri için doğru anahtar sözcük sıralamasıdır?
9. Tablolar da dahil olmak üzere veritabanı yapılarını yapılandırmak için kullanılan SQL komutlarının alt kümesi hangisidir?
10. SQL'de, bir tablonun depolama özelliklerini değiştirmek için hangi komut(lar) kullanılır?
11. SQL veri işleme dili (DML), veritabanı verilerini sorgulamak ve değiştirmek için kullanılır.
12. SQL'deki beş yerleşik toplama işlevi nelerdir?
13. SQL Join yan tümcesi, ikisi arasındaki ilgili sütuna dayalı olarak yalnızca iki tablodaki satırları birleştirmek için kullanılır.
14. SQL WHERE yan tümcesi:
15. Bunlardan hangisi SQL için geçerlidir?
16. Aşağıdaki SQL'de yanlış olan ne?
17. SQL'de, LIKE operatörü değerleri karşılaştırır
18. Tüm SQL kodları ne ile bitmeli?
19. Hangisi SQL'de birleştirme türü değildir?
20. Sonuç kümesini sıralamak için hangi SQL anahtar sözcüğü kullanılır?
21. Değeri olmayan bir özniteliği temsil etmek için hangi SQL anahtar sözcüğü kullanılır?

AGİLE

Bugün biraz daha bilgisayardan uzak SDLC ve devops dersleri gibi sözel olarak işleyeceğiz.

İlerde pedleri bitirdiğiniz zaman iş hayatına girdiğinizde o zaman karşılaşacağınız ortam yani nasıl bir iş yerinde olacaksınız, çalışma yönteminiz nasıl olacak, iş yerinde çünkü yazılım alanında çalışan birçok şirket agile yöntemini benimsemiş durumda, biz de bu derste bundan bahsedeceğiz. Yani teknik manada gördüğünüz kullanacağınız toollardan ziyade iş yapış yöntemi diyebiliriz.

Hangi konumda çalışıyor olmanız bir şey fark etmez, hangi işi yaptığınız fark etmez, hepsinin ortak noktası bir agile yöntemi uygulanıyor olacak şirketlerde.

O yüzden şanslı guruplardansınız diye düşünüyorum çünkü yazılım dünyasında çalışıp da agile yi bilmeyen scrum, jira nedir bilmeyen arkadaşlar bunu öğrenmekle uğraşanlar var. Sizler bunu daha önce öğrenip hazırlıklı gitmiş olacaksınız.

Şimdi bugünkü dersimizde

Agile nedir ona gireceğiz

Soru;

İş alış yöntemi olarak bir projede veya bir yerde görev aldınız, bir işi bitirip bir sonrakine mi başlarsınız ya da ben 3-5 işi alırım kafam karışmaz hepsini bir arada yaparım mı dersiniz.

Siz istemeseniz de bazen sistem içerisinde multitasking e sizi yönlendiren hususlar olabiliyor.

Bunu sormamın amacı bugünkü derste buna biraz giriş yapacağız, bu olayın artık bittiğini bu olayın yerine hem insanlara çalışma ortamında rahat bir atmosfer yaratmak hem de çalışan da bir insan onun zihin sağlığı da önemli, yani yapabileceği kapasitesi de belli yani gün içinde saat 6 oldu mesai bitti, bitmiş bir insan ne ailesine çocuğuna vakit ayıramaz ne de arkadaşıyla vakit geçirecektir bunlara zaman ayıramaz. Bitmiş bir kişi akşam tv karşısında uyuyakalır arkadaşlar.

O yüzden agile nin buna da bir yaklaşım ı var, ona da değineceğiz.

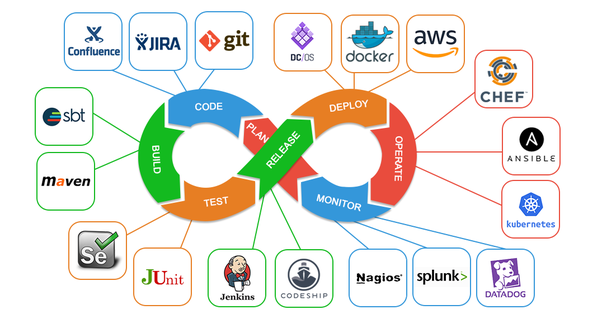
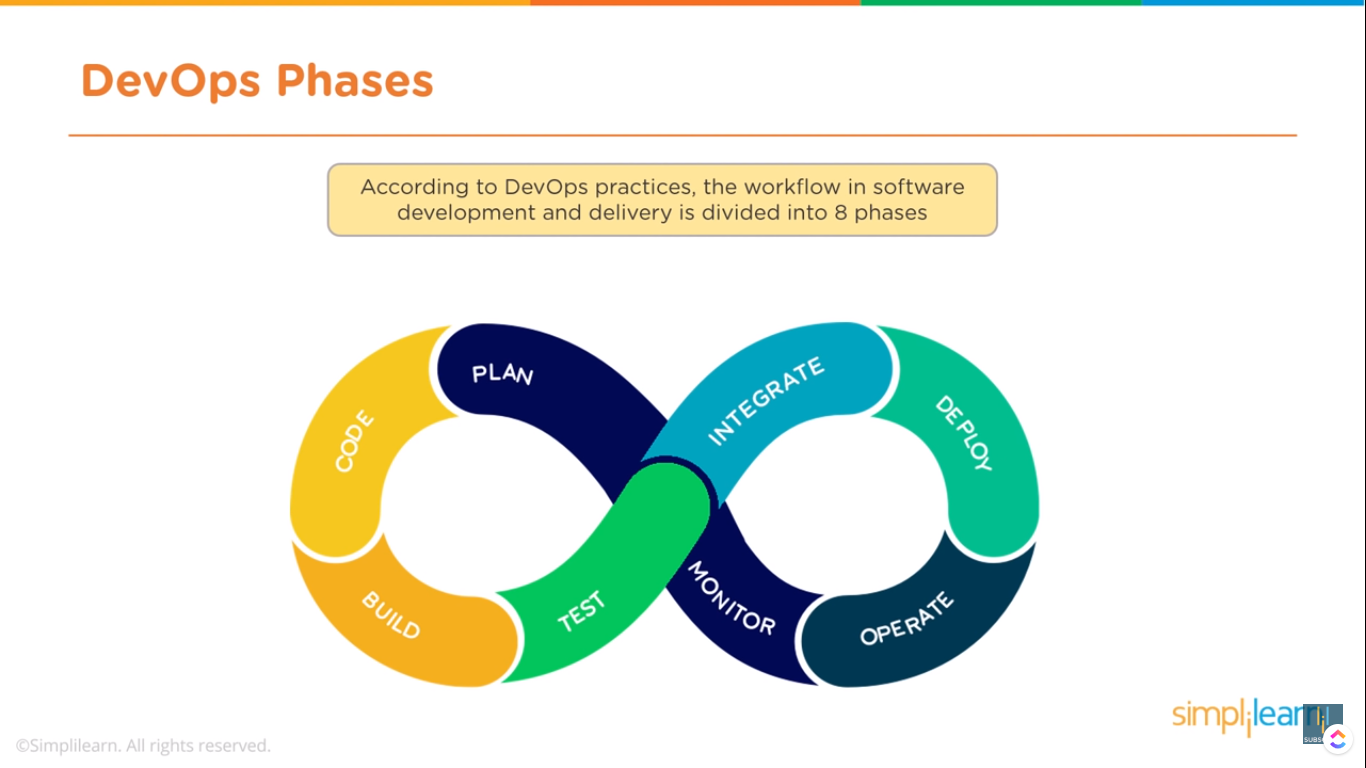
Biz SDLC de nelerden bahsetmiştik onları hatırlayalım.

SDLC bir Yazılım Geliştirme Yaşam Döngüsüydü Software Development Life Cycle

SRS ne anlama geliyor, Software Requirements Specification Document

Klasik ve modern modeller wodırfol modelini görmüştük

Devops u görmüştük neydi bu ????



Devops :

(SDLC) yani yazılım geliştirme yaşam döngüsünü kısaltmayı ve yüksek yazılım kalitesi sağlamayı amaçlar.

DevOps ----- en genel hali ile development ve operations kelimelerinin bir araya getirilmesinden oluşmuştur. (yazılım dünyasında developer ile bunları sisteme yükleyen bu kodları müşteri ile buluşturan ve bu kodların yedi 24 çalışmasını sağlayacak olan operasyon ekipleriyle daha iyi çalışabilmesini hedefleyen bir kültür.

Bunların bir araya getirilmesiyle bir ürün de ortaya çıkıyor, bir amaç var neden bunların bir araya getirilmesi lazım, bunların koordineli bir şekilde çalışıyor olması lazım bunlardan bahsetmiştik, neydi, bu yazılım geliştirme sürecini, daha kısa tutabilmek ve sürekli bir şekilde müşteriye , devamlı updete edilmiş, devamlı güncellenen ürün sağlayabilmeliyiz. Her an güncelleme geliyor şu an ki programlar biliyoruz. Bir program diğer programdaki güncellemeyi görüp bu bende de olsun deyip yeni bir güncelleme istiyor. Bunlar müşteriyi cezbetmek ve elinde tutmak için hızlı bir şekilde yazılımı her daim geliştirmek amaçlanıyor burada. Bir güncelleme olmadığında müşteri diğerine kayabilir diye bunu yapıyor. Kaliteyi arttırmak, devamlı sistemi güncel tutmak, müşteriyi kendinde tutmayı sağlar o nedenle bu continyus delivery, Continuous Integration kelimelerini IT dünyasında çok duyacaksınız.

Devops ta aslında bu takımları, dev ve ops dediğimiz ekibin arasındaki iş birliğini arttırmasının amacı da bu.

Dersimizde nelerden bahsedeceğiz;

* Öncelikle proje yönetiminde bir göz atacağız. --- modernleşmesi sürecine bakacağız.

Biz u derste de agile kısmını görmüş olacağız.

* Agile nin dünyaca ünlü manifestosu var buna bakacağız
* Bu manifestonun içerisindeki prensiplere bakacağız
* Bir özet gibi agile yaklaşımı nedir, bu iş yerinde nasıl uygulanıyor
* Bunun iki tane freamwork u var birincisi scrum, diğeri canban bunlar nedir nerelerde kullanılır bunların hepsini açacağız.

Şimdi öncelikle gelelim proje yönetim mekanizmaların, sistemlerinin modern hale geliş tarihine biraz göz atalım. Kısa bir özet geçelim.

\*\*

Agile yi daha önce duydunuz mu, Agile nin kelime manasını bilen var mı

Her ortama ayak uydurabilen, her gelişmeye karşı hal, tarz alabilen

Yazılım dünyasında (tam karşılığını veriyor aslında) gelişmelere, değişikliklere açık bir şekilde, geri bildirime açık bir şekilde ona karşı hemen bir hal almak yani uyum sağlayabilmek.

Yazılım geliştirme alanında bu geri bildirim çok önemli, daha önceden bu önemli olmasa bile bu çok önemli günümüzde. Bu müşteri memnuniyeti açısından da önemli, ürünün piyasada sürdürülebilir bir şekilde (tutunabilmesi için, kabul görebilmesi için) yer alabilmesi için de önemli. Birazdan onu da başlıklar altında göreceğiz.

Proje yönetimi aslında SDLC dersinde de görmüştük, proje yönetimi demek, bir hedef konur, yazılım açısından düşünürsek ve bu hedefi gerçekleştirebilmeniz için size bir zaman dilimi belirlenir( tahsis edilir) ve tabi sınırsız bir kaynağa sahip olmadığınız için bu hem para yönünden olabilir hem insan kaynakları olabilir, iş yerinin sermayesi olabilir bunlar belirlenir. Proje bu demek.

Projeyi yerine getirebilmek için de değişik yöntemler deniyoruz. İnsanlık tarihinde bu 2. Dünya savaşından sonra başlıyor bu proje yönetim sistemlerinin kuruluşu. Çünkü seri üretimler gibi şeyler işin içerisine girince 1940 dan 50 den sonra artık savaşlarında sona ermesiyle endüstri devriminin neticesinde fabrikaların açılması ve insanlar fabrikalar açıldıkça daha fazla nasıl kar edebilirim, seri üretimle az maliyetle daha fazla ürün nasıl üretebilirim düşünceleri ortaya çıkmış. Bu yazılımda da böyle arkadaşlar,

1970 lerde wotırfol modeli yazılımcıların, software developer ların tercih ettiği bir model olmuş. Biz bunu daha önceki dersimizde görmüştük zaten nasıl dı bu model, işin ta başlangıcında her şey belirleniyor, kalıpların değişmeyeceği, yüzde yüz uyacağınız bir planla başlıyorsunuz herhangi bir geri bildirim olmadan süresi ne kadar belirlenmiş ise o süre zarfında bu adımları gerçekleştirerek tamamlayarak istenilen ürüne ulaşıyorduk.

Gün geçtikçe müşterinin isterleri artınca ürünü ortaya koyan firmalar da bu isteklere göre cevap verebilmek için farklı yöntemler teknikler düşünmüşlerdir. Günümüze bir yansıması olmuştur agile modelde.

Standish Group un bir raporu var bu yansıda

Watırfol metodunda yazılım firmalarının başarıya ulaşma oranları gösteriliyor.

%11 lik dilime baktığınız aman planın başlangıcında müşterinin isteklerine göre çıkan ürün %11 oranında uyuşmuş, yani %11 oranındaki şirket bunu sağlayabilmiş. Yani aldığı isterlere göre hazırladığı proje %11 müşteriyi memnun etmiş ve ona teslim etmiş.

%29 u ne olmuş, teknolojinin hızla gelişmesiyle gerileyen programlar haline gelmiş, ya da anlaşmazlıklar olmuş olabilir ( kağıda yazılanla sahaya sürülen arasında bambaşka farklılıklar olabilir ) bu başarısızlıklara sebebiyet vermiş projelerin %29 u.

%60 ı ki bu en büyük dilimine bakacak olursak, bu belirlenen bütçe aşımı olabilir, zaman aşımı olabilir, ürünün yeterince randımanlı çalışamaması olabilir, her özelliğinin müşterinin en başta istediği gibi belirlendiği gibi tanımlanamaması ya da ürünün müşteriye uygun bir şekilde teslim edilememesi diyebiliriz. %60 oranında da başarısızlık yaşanmış, yarıdan fazla.

Tabloya bakınca insan bakın sadece müşterinin %11 i aldığı üründen memnun iken geri kalanı başarısızlıkla sonuçlanıyor.

Nokia buna örnek olabilir, çağa ayak uyduramadığı için piyasadan silinip gitmiştir. Bu da bir sebep aslında.

Peki Agile den sonra bu başarı ne olmuş ona bakalım

2011-2015 yılları arasındaki bir veri belki şu an daha da artmıştır bu rakamlar. %11 yerine %39 lara çıkmıştır.

Calınçlarda sıkıntı yaşanan kısımlar çok fazla azalmamış belki ama en azından tamamen başarısızlıkla sonuçlanan bu oran % 9 lara kadar düşmüş durumda.

Bu yüzden Agile metodolojisi günümüz yazılım geliştiricilere çok büyük imkanlar sağlıyor, müşterilerine uygun ürünü verebilmek için, müşteri memnuniyeti için gerçekten tercih edilen bir yöntem.

Şimdi biz bu yöntemi yavaş yavaş incelemeye başlayacağız.

Öncelikle manifestoyu inceleyeceğiz, bu yöntem biz ha deyince hadi bugün agile ye geçiyoruz demekle olmuyor.

Nasıl gelişiyor,

2001 yılında Utah, Snowbird'deki bir kayak merkezinde bir toplantıydı.

Bunların ortak noktaları hepsinin bir yazılım geliştirme alanından gelmeleri.

17 kişilik bir grup şöyle düşündü:

"Yazılım geliştirmek için hepimiz bu farklı yaklaşımları uyguluyoruz. Bir araya gelmeli ve düşündüğümüz şeylerde ortak noktalar olduğunu görmeliyiz.”

Piyasanın , içinde bulundukları sistemin sıkıntılarını bir şekilde dile getirip buna çözüm bulmak üzere böyle gayri resmi bir ortamda oturup konuşup agile manifestosunu ortaya çıkarıyorlar.

Ken şıvebır ın öncülüğünde toplanıyor gurup, orada ne konuşulmuş bundan bahseden kısa bir video. -----Alt yazı açalım---

<https://www.youtube.com/watch?v=mfQIjf2TPdk>

bu manifestoları belirleyip bunları dikte etmek değil, bunları uygulamamız sizin için daha değerli bunda diğerini kötülemek, onu çöpe atmak bundan uzaklaşın şuna gidin gibi bir yaklaşım içerisinde bulunmuyorlar.

Bu çok uzun bir listeden oluşmuyor. Kısa özet bir şekilde günümüz yazılım geliştiricilere ışık tutan bir manifesto değeri taşıyor.

Şuradaki 4 madde bu manifestonun özeti niteliğinde.

Mülakatlarda Agile nedir daha önce duydunuz mu dediğinde sizin onlara evet duydum, manifestosunun 4 adet maddesi var, değeri var, bunun altını dolduran 12 tane de prensipleri mevcuttur. Şunları önümüze koyuyor diye söyleyebilirsiniz.

Agile manifestomuz var, bunun altını dolduran prensiplerimiz var ve bir de Alliance dediğimiz bir kısım var, bu agile ile ilgili uzman kişilerin sertifika almasını sağlayan bir kuruluş

Peki biz bu agile manifestosunu nasıl anlamalıyız, nedir bu agile manifestosu şimdi ona bakacağız.

Evet gelelim şimdi o manifestonun 4 ana maddesini beraber bir incelemeye başlayalım.

1. Burada birini kötülemiyor, bireyler ve bireylerin etkileşimi proses ve toollara göre daha değerli diyor, yani buradaki amacımız bizim sağlıklı bireylerin ve o bireylerin birbirleriyle olan etkileşimi (iş yerindeki insanlardan bahsediyoruz ) üst seviyede olması demek, tabi diğer işler kötü seviyede olsun demiyoruz (program işleyişiş, toollar) bunlar önemsiz demiyoruz fakat bunlardan ziyade bireyler daha önemlidir öne çıkarılıyor.

Bunun getirilerini şöyle görebilirsiniz, bir çok arkadaş filimlerde veya reklamlarda Google ın ya da değişik yazılım firmalarının ofislerini görmüşsünüzdür işte, salıncaklar var, kaydırakla üst kattan alt kata inenler var kimi yazılımcılarda öğle arasında uyuyabilmeleri için ofislerin yan tarafına yataklar konulmuş veya artık bir çok firma yazılımcıların evden çalışabilmelerine olanak sağlıyor (bu sayede ofiste bir günde yapacağı işi evde birkaç saatte yapabiliyorlar arkadaşlar). Sosyallik vs gibi konularda belki biraz geri kalabilir evden çalışmak ama bazı günler en azından ofiste olmak geri kalan işleri evde rahat ortamda yapabilmek bir çok kolaylık sağlıyor. (yolda gidip gelmek zaman , yorgunluk bunlardan avantaj sağlamış oluyoruz)

1. Evet ikinci maddeye geçelim, ikinci madde de bizim önem verdiğimiz hususlardan bir tanesi çalışan yazılım. Ne kadar bireyler dedik, bireylerin öneminden bahsettik, fakat o değer verilen bireylerin de önem verdiği en büyük maddelerden bir tanesi şirketinin yada çalıştığı şirketin müşterilerine hatasız bir şekilde çalışan ürünü teslim edebilmeleri, teslimden sonraki dönemde, günümüzde artık cd ye yükledi kurdu benden çıktı değil, bir çok firma artık Cloud üzerinden müşterilerine siz Google dan kullanıcı adı ve şifre girer gibi karşı tarafa müşteriye kullanıcı adı ve şifre verdikten sonra sizin ürününüzü müşteri kullanmaya başlıyor.

Bu şu demek müşteri 7-24 çalışan bir ürün istiyor. O yüzden çalışan ürünün bu şekilde idame edilmesi bunun için çok önemli. Yani düşünebiliyor musunuz, mesela Trendyol diye bir firma var, muazzam bir Pazar alanı, oradan değişik markalar dükkanlarını açıp müşterileri oluşuyor. İşte düşünün akşam 8 – 9 alış-veriş yapıyorsunuz bir baktınız Trendyol kitlendi, kesildi, ne dersiniz? Hadi bir defa oldu neyse dersiniz 3-5 defa olunca ne yapacaklar herkes alacak bilgilerini çıkacak oradan ya amazon a gidecek ya n11 e gidecek başka bir yere gidecek değil mi.

Şirket açısından günümüzde özellikle şu kelimeyi çok duyacaksınız no down time boş zaman yok diye bir kelime var, sizde bunu çok duyacaksınız yazılım dünyasında çalışırken, bizim ürünümüzün bir dakikalık ya da bir saniyelik kesintiye uğraması kabul edilebilir bir şey değil artık. Çünkü teknoloji bunu gerektiriyor. Böyle bir şey varsa çok büyük hata var demektir, şirketin bunu gözden geçirmesi gerekir.

Tabi burada çalışan yazılımın neye göre çok daha önemli diyor, çok detaylı bir doküman yazmak tabi dökümanda olmalı ama her şeyi ayrıntılı yazıp anlatmak daha ürünü ortaya koyamamışken dökümanlarla uğraşmak yerine yazılımın çalışmasına odaklanmak gerektiğini vurguluyor.

1. Daha önceki yıllarda nasıl oluyormuş anlatıldığına göre, müşteri ile şirket çalışanları arasında bir duvar var, ürünü istedi müşteri daha sonra bekliyor şirket çalışanı ürünü ortaya çıkardığında müşteri ile çalışan arasında tartışmalar başlar, karşı taraftan biz bunu istememiştik şirket çalışanları siz bunu söylemiştiniz gibi böyle çıkan ürün üzerinde yada parada anlaşmazlıklar yaşanır. Bunları duymuşsunuzdur. Bu gibi olaylar her şirkette yaşanmıştır.

Tabi bu agile manifestosu sadece yazılım alanına hitap eden bir görüş değil diğer alanlarda da kullanılıyor belki diğer alanlarda çalışmış arkadaşlarda bunu görmüştür. Yani müşteriyle yaka paça olan bir çok şirket vardır. İsterler böyle değildi, ürün çıktı ürün istenen gibi değil gibi. Bunun sona ermesini istiyor agile manifestosu, müşteriyi yazılımın bir parçası olacak şekilde iyi bir iletişimin kurulması yani sağlıklı bir iletişimin olması ve o iletişim kanallarının da devamlı açık bir şekilde idame ettirilmesi gerekiyor.

Müşteriyi işin içerisine çekmek gerekiyor. Her geliştirme sonunda müşteri davet edilmeli, müşteriyle ilgilenen bir prodact owner veya proje yöneticisi oluyor, bu kişiler çalışan ürünün bir demosunu, bir ön gösterimini müşteriye sunuyorlar. Müşteri 5 ay sonra gelip ya biz böyle istememiştik gibi bir durumla karşılaşmamak için 2 haftada bir haftada bir olabilir sprint ne kadar büyüklükteyse her sprintin sonunda müşteriye kısmen çalışan veya şu ana kadar geliştirilen kısmı sunuluyor. Yapılan bir hatada ileride neden olabilecek büyük sıkıntılara sebebiyet vermeden buradan geri dönebilme fırsatı doğmuş oluyor.

1. Değişikliğe açığız; Ne demiştik SDLC de wotıdfol modelinde lineer bir düzlem üzerinde, düz bir çizgi üzerinde yapmış olduğumuz plana %100 bağlı kalarak ilerliyorduk ürün bitene kadar ve aşamaların sonunda bir ürün veriyorduk müşteriye.

Ama burada biz gelişimlere açığız diyoruz, nedir bu gelişim diyelim ki bir yazılım geliştirme firmasında yöneticisiniz, müşteri sizden twiter gibi bir program yazmanızı istedi. Ben böyle bir milli olan bir sosyal medya aracı geliştirmenizi istiyorum diyor. Ne kadar sürer bu olay 4 ay. Şimdi günümüzde bakıyorsunuz işte bir gün hikaye koyma özelliği ekleniyor, bir gün twiter de sohbet odası açılıyor orada konuşma özelliği ekleniyor, bir gün bakıyorsunuz video özelliği ekleniyor bir çok böyle yeni özellikler katılıyor yazılıma. Şimdi 4 ay öncesinden istediği isterleri haliyle süreç ilerledikçe 3 ay, 2 ay neyse belli aralıklarda size diyecek ki bakın bu da olsun, siz de tabi şunu diyeceksiniz bu ek bir maliyet gerektirir. Yaparız ama şu kadar bir süre daha lazım gibi siz de isterlerinizi söyleyeceksiniz bu konuda iletişim kanallarınız açık olacak bu sayede diğer isteklerine de cevap vermiş oluyorsunuz yani daha iyi bir şekilde müşteriyi memnun edecek şekilde ürününüzü müşteriye sunmuş oluyorsunuz.

özet geçiyoruz

Birinci, İndividuals and interactions (indivicuıls and intırakşıns) bireyler ve etkileşimler

Bireyler ve bireylerin birbiriyle iletişimi, birbiriyle çalışması uyumlu bir şekilde iletişim kanallarını açık tutup çalışmaları.

Proses ve tools deiğimiz, işleyiş, işlemlerin ele alınması, toplantıların ayarlanması, dökümanların hazırlanması, mesai saatlerini ayarlanması tabi bunlarda önemli ama bireyler şirket için daha önemli. Yani örnek şirkettesiniz bir iş çıkaracaksınız didindiniz başardınız, ikinci üçüncü işte insanların yapacak gücü kalmaz. Bu işte devamlılık gerekir….. Onun için en önemli olan şey, en büyük sermaye olan insandir. Bireyler agile metodolojisinde korunmuş oluyor, öne çıkarılmış oluyor.

İkincisi neydi, günümüz şirketlerinde düşünün artık bir ürünün kesintiye uğraması, ürünün müşteriye hizmet veremiyor olması ya da aldığınız bir işin sonunda belirlenen bütçe, süre her şey önünüze konuldu fakat çalışan bir ürün çıkaramadınız bu başarısızlığa yol açıyor. O yüzden böyle doküman yazmak çok muazzam çalıştık, şöyle oldu böyle oldu gibi söylemlerin bir manası kalmaz. Bunlarda yapılmalı tabi ama zamanında çalışan ve müşteriye teslim edilen ürün daha önemli.

Biz agile yi yazılımın dışında başka bir sektöre uygulanamaz mı?????

Üçüncüsü, Costumer collabration, Müşteri işbirliği (sözleşme ve müzakereler yerine)

Biz müşteriyi bir düşman cephesi olarak görmeyeceğiz, (onlar olmazsa zaten biz yokuzdur- çalışamayız) karşı taraf için de tabiki bu böyledir…

Müşteri de sermaye onda evet ama yapacakları işlerin kolaylaşması için dijital alanda teknoloji alanında bizden ürün bekleyen kişiler ikimizde birbirimizi tamamlayan bir parçayız. Bir taraf olmadığında, biz olmadığımız takdirde iş aksayacak sermaye dönmeyecek, para kazanamayacak, ürün üretilemeyecek, e bu taraf olmadığında ürüne ihtiyacı olan bir kişi, müşteri olmadığı zamanda yazılım geliştirme firmaları olmayacak. Bunun farkına varılması lazım. Kazan kazan metodolojisi ön planda olması lazım. İki tarafın birbiriyle didişmesi yerine ortak noktada buluşup işte her şeyi ortaya koyup açık açık ortaya koyup çalışan bir ürününün ortaya çıkması sağlanmış olacak.

Dördüncüde,

Wotırfoll --- geri dönüş yok, düz bir plan var çevik değilsiniz, müşteriyle iletişiminiz kopuk o yüzden başarılı olma şansınız çok düşük ve başarısız olduğunuzda da çok büyük maliyetlerle karşı karşıya kalıyorsunuz. (bir anda batmış olabilirsiniz)

Agile ----- sprintler sayesinde …

Agile öyle bir anda ortaya çıkmıyor, , kullandığımız araçlar, web sitesi hazırlayacaksınız, serverların artık çok hızlı ayarlanıyor olması, ve diğer gelişen otomasyon teknolojileriyle de bir ürünü belki birkaç gün içerisinde yeni bir sürümünü hemen piyasaya test edip deneyip sürme kabiliyetini de şirketler kazanmış durumda .agile nin ortaya çıkması kendiliğinden gelişen bir şey değil yeni gelişen teknolojilerin görülmesi ve gün geçtikçe daha yeni teknolojilerle beraber daha daha farklı yaklaşım tarzları da bizleri bekliyor olacak olması bu yaklaşımı ortaya çıkarıyor.

Agile Prensipleri

1. Birinci maddede çalışan ürünle alakalı bir madde, working software bizim için değerliydi,

Buradaki birinci maddede de aslında onu söylüyor. Bizim en önemli önceliğimiz müşterinin ihtiyaçlarını karşılamak, onu memnun edebilmek, onun isteğini en üst seviyeye çıkarabilmek, müşterinin talep ettiği bir ürünü ortaya koyabilmek (doğru ürün doğru çalışan program---asansör yazılımı gibi)

Müşteriyle karşı değiliz, müşteriyi karşı taraf görmek yerine biz de aslında müşterinin bir parçasıyız aynı işi beraber tutuyor gibiyiz.

1. Olmazsa olmazlardan biri, her daim değişime açık olacağız. Hangi siteye bakarsak bakalım, ya da sosyal medyaya her şey değişim halinde ve güncelleme gerektiriyor. Ve günümüz teknolojisinin bize sunduğu alt yapı buna müsait olduğu için, bu fronted de olabilir, bacend olabilir, hız bakımından yada hizmet bakımından her türlü bu müşteri talebini bu devoloper takımı değerlendirip, hızlı bir şekilde bu talebi yerine getiriyorlar. Agile de bu mümkün. Artık bir çok firma böyle yapıyor, çünkü siz sadece ürün teslim etmekle kalmıyorsunuz, artık Cloud la beraber ürünü idame ettirmekle de sorumlusunuz. Ne yapıyor müşteri şu an mevcut olan ürünü kullanıyor, aylık kullanım üzerinden para ödüyorlar. Ve bu ürün proje yöneticileri belirli aralıklarla müşteriye gidip taleplerini dinliyorlar. Onlardan geri dönen girdiye göre (yeni bir istekleri olabilir ya da bir sistem çalışmıyordur onun hakkında) biz bu isteği alıp birkaç hafta içerisinde yada isteğe göre birkaç gün içerisinde anında ürüne yansıtabiliyoruz. Bu çok önemli. Üçüncü maddede bununla bağlı aslında…
2. Çok kısa bir süre içerisinde, günümüz teknolojisi(otomeşın) sayesinde bu isteri dışarıdan gelen bu talebi ürüne yansıtabiliyoruz.

Şu aşamada bilmenize gerek yok ama Microservis yapısını duyacaksınız, özellikle devops da göreceksiniz koca bir internet sitesinin yada büyük bir alış-veriş sitesinin küçük küçük microservislerden oluştuğunu göreceksiniz. O yüzden tüm kodu anında indirip sisteme yeniden yüklemekle uğraşmıyorsunuz . Örneğin alış-veriş sepeti servisi ya da ürünün sergilenmesi servisini ayrı ayrı değiştireceksiniz. Yani o küçük bir kod parçacığını anında değiştirebilip update edebiliyorsunuz. Bunu çok duyacaksınız micro servisi, bize bu çevikliği bu hareket alanını sağlıyor.

1. İş hayatındaki kişilerle o ürünü kullanan yani piyasaya sürecek kişilerle gün içinde devamlı iletişim halinde olmamız ve onlarla beraber çalışabilmemiz gerekiyor. (korona zamanı bile bazı firmalar tırlarla arkasını toplantı yeri gibi ayarlayıp müşterileriyle burada görüşmeler yapmışlardır. ) Bu da müşteri memnuniyeti açısından önemli bir nokta. İnsanlar bunun önemini kavramış durumda.
2. Bu bireylere değen manifestoyu hatırlıyorsanız onunla ilişkili bir madde, burada kendi kendine motive olmuş ya da iyi bir çalışma ortamında birbirini destekleyen birbirine inanan (baktığında bu çalışan), üst alt durumunda bile bu kişi kaytaracak mı diye bakmayacak, böyle bir durum yok arkadaşlar. Şirkette şöyle bir ortam kurması gerekiyor insanlar birbirine güvenecek, elinden gelenin en iyisini yapacağına inanacak ve o kişi yardıma ihtiyaç duyduğunda uzaktan da olsa (zoom dan görüşüp sanki yanındaymış gibi beraber belki aynı kod parçacığının üzerinde çalışıyor, günün sonunda bakıldığında birkaç kişinin beraber yaptığı imece usulü kodlar yazılıp bitirilmiş oluyor. (gün içerisinde beraber tamamlayabiliyorlar.)
3. Yüz yüze deniliyor artık ama bu sanal ortamda devam edebiliyor. Bireylerin iş yerinde birbiriyle etkileşimi, iyi bir iletişim kurabilmeleri, bir takım halinda çalışıyor olabilmesi çok önemli, o yüzden gün içinde bunu çok kullanacaksınız, birebir gidip veya zoom dan görüşüp hata ayıklama olabilir yeni bir özellik ekleyip bunu gösterme, kodu inceleme gibi devamlı birebir etkileşim halinde olması gerektiğini belirtiyor.
4. Bu madde üzerinde durmuştuk, çalışan ürün bizim başarımızın en önemli ölçütlerinden biriydi.
5. Ve sekizinci madde de, Bir şirket kurdunuz, günde 10-15 saat çalıştınız, hafta sonu demediniz, bayram seyran demeyip çalıştınız. İki defa üç defa böyle çalıştınız. Sonra psikolojik nedenler ortaya çıktı, sağlık nedenleri ortaya çıktı, ailevi problemler gibi her türlü şey başınıza gelebilir. Bu tarz çalışmak yani sürdürülemez bir şekilde çalışmak daha sonra insanlarda farklı farklı sıkıntılara sebep olabiliyor. O yüzden sürdürülebilir bir çalışma hayatı önemlidir. Mesela bu kriterlere göre firma sahibi veya yönetici bu işte çalışacak şu kadar devops eng veya full stac elemanı bana lazım diyor (öngörürüyor bazı ölçeklere tabi tutarak) ama diğer taraftan bana 3 kişi yeter ben bu işi 3 kişiyle de götürürüm diyenler yolda kalabiliyor.

Ama agile nin bize getirdiği en büyük özelliklerden bir tanesi (özellikle çalışan tarafından baktığımızda) bizim elimizi rahatlatan bir şey çünkü bizim insan olarak belli bir kapasitemiz var, belli bir çalışma saatine dayanabiliriz, yıllık belli bir saat iznimiz olmalı, haftada yenilenmek için, en azından ailenize vakit ayırabilmek için iki günlük izin hakkımız olmalı.

1. (Nokia örneğinde olduğu gibi ) Ürün çıktıktan sonra biz ürünü bıraktık artık biz bu üründen iyi kazanırız 3 yıl beş yıl, böyle dememeli ya da şirketlerin böyle düşünmemesi lazım.

Ürününü bazen bazı şirketler denemek için veya piyasada sadece bu işi yapsın diye kişi tutuyorlar, yani sen bu ürünü kullan bize memnuniyet anketi ver. Ya da üründe hata var mı, veya bu kullanan kişiler arasında anket yapıyorlar, ürünün hangi tarafını beğeniyorsunuz, hangi taraftan rahatsızsınız, hangi kısımları değiştirmemiz lazım gibi dönütler alınıyor. Yani bu dizayn bakımından da, teknik bakımdan da kullanım bakımından da ürünün devamlı ölçülmesi, müşteriden bu geri bildirimin (bu bir firma olmayabilir- direk son kullanıcı insanlarda olabilir) bunlardan gelen bu geri bildirimi çok iyi tutmanız lazım. Hangi kesime hitap ediyorsanız ürününüzün size devamlı para kazandırmasını istiyorsanız ve ürününüzün devamlı aranan bir ürün olmasını istiyorsanız kısa aralıklarla bu istenen talebi karşılamanız lazım.

1. Ürünü basit tutmanız gerekiyor. (akıllı telefonlar yaşlılara ne adar zor geliyor). O yüzden hitap ettiğiniz kesime göre bunun dizayn edilmesi lazım. (lisedeki paket program yöneticileri)

Bazen işte uygulamalarda görüyorsunuz çok gereksiz yerde kapanmayacak reklamlar çıkabiliyor, kullanıcının moralini bozabiliyor bu. Ya da ürün reklamsız olsun, bir ürünün ayarlarına gitmeye çalışıyorsunuz öyle bir dizayn edilmiş ki onun başına oturan developer bunun çok kolay olduğunu düşünüyor, developer şuralardan girip ayarlara gidecek diyor ama son kullanıcı o çeviklikte olmayabilir ya da çok gözü açık birisi olmayabilir, kullanıcı saatlerce ayar butonunu arıyor yani buraya nereden gideceğim. Böyle olduğu zaman ürününüz maalesef batan firmaların listesine yazdırmaya aday haline geliyor.

Apple in neden bu kadar piyasada söz sahibi olduğunu anlarsınız. İşte şarj aletinin dizaynından telefonun dizaynına kadar basitliği ve sadeliği ön planda tutmaya çalışıyorlar ve bu da onları da başarıya götüren nedenlerden bir tanesi oluyor.

1. Kendi kendini organize eden bir takımdan bahsediyor, bu yazılım takımlarında çok önemli çünkü scrum a geçtiğimizde göreceğiz. Scrum da bir hiyerarşinin olmadığını görüyorsunuz, çalıştığınız yerde cio yu görmeyeceksiniz belki, takım liderinizle aranızda bir üst alt ilişkişi olmayacak ismiyle hitap edebiliyorsunuz, günlük işleriniz toplantılarınız beraber, ulaşılabilir bir kişi ve bu takım kendi kendine organize olmuş yani böyle iş kaynatmak, farklı şeylerden bahsedelim iş bir kenarda dursun değil de takımın en önemli önceliği elinde bulunan işi (serverları) artık işletilecek ne varsa en iyi şekilde işletmek ve dışardan gelecek taleplere en kısa sürede cevap vermek üzerine kurulu.

Ve kendi kendine organize olup birinin düştüğü yerde birine yardım lazımsa onu arkada bırakmayacak şekilde takım çevik davranıyor arkadaşlar. Gelen taleplere göre ya da yeni yapılacak dizayn a göre her türlü kolaylığı takım kendi içerisinde birbirine sağlıyor. Agile de tabi ki bu organizasyonu istiyor.

1. Bu madde çok uygulanan maddelerden bir tanesi, belirli aralıklarla, belirlenen düzenli aralıklarla takım, nasıl daha efektif olabiliriz, ya da bize ayrılan kaynakları daha efektif nasıl kullanabiliriz, daha hızlı nasıl cevap verebiliriz buna yönelik, geliştirilmeye yönelik olan toplantılar yapılıyor. Bunlarda hemen böyle kriz masaları gibi düşünmeyin öyle bir şey oldu hemen toplanın değil de, toplantının yeri saati ve süresi belli saat geciktirilmeden yapılır ve konuşulan bu konular aslında çeyrakte yapılacak konular oluyor. Yani yılın bu önümüzdeki çeyreğinde neler yapacağız veya bunu yapacağız. Diğer çeyreğinin bile konuşulduğu olabiliyor. Takım içinde konuşuluyor, işte şu uygulama güncellemeleri, serverların yenilenmesini şu zaman yapacağız gibi takım bu şekilde hem davranışlarını düzenliyor yani neleri yanlış yaptıysa ona göre önlem alıyor hem de önündeki uzun vadeli planları kumanda edebiliyor takım.

İlk dört maddeyi söyleyebilecek olan var mı

* Müşteri ihtiyaçlarını karşılamak
* Bireyler ve etkileşim
* Gelişim veya değişikliğe açık olmak
* Working software

Bunlar size soru olarak dönebilir.

Prensipleri de hemen şöyle özet geçelim

1. Müşterinin memnun edilmesi gerekiyor en üst seviyede
2. Değişikliğe devamlı açık olacağız
3. Gelişim çok sık olacak, ürününüze belki haftada bir iki defa yenilik kazandırıyor olacaksınız ve o ürününüzü belli aralıklarla müşteriyle buluşturuyor olacaksınız. Göstermeniz, sunmanız gerekiyor. (sık sık teslimat yapın)
4. Takımların birbiriyle çalışması
5. Birbirine güvenmesi ve desteklemesi, arkasını kollaması çok önemli
6. Yüz yüze görüşmeler, hem takım içerisinde de düşünülebilir hem de müşteriyle de o iletişim kanallarının açık tutulması olarak da düşünülebilir.
7. Çalışan üründen bahsettik
8. Sürdürülebilir bir çalışma alanı (Türkiye de bu muzdarip olunan bir alan), bireyin kendine vakit ayıramadığında bir yerden sonra dayanamıyor.
9. Burada ürününüzün piyasada istenir halde tutmak için devamlı piyasadaki ürün kullanıcılarının isteklerini göz ardı etmemeniz lazım, onları düşünüyor olmanız lazım ve (trendleri) aslında hitap ettiğiniz kesimin modası ne ise ona bakmanız gerekiyor. (facebook)
10. Basitlıkten bahsettik kullanıcının ürünü kullanabilmesi zorlanmadan, sıkılmadan, onu yormayacak şekilde olması gerek
11. Takımların kendi kendine organize olması önemli, burada scrum metodolojisinde göreceğiz, astlık üstlük ilişkisi yok, kimse birbirine sen şunu yap sen şunu al demiyor, zaten otomatik olarak herkes iyi olduğu alanda (herkesin iyi olduğu alanlar olabilir, kubernets dediğimiz bir tool var orada uzman durumda onunla ilgili iş olduğunda o oraya geçiyor, onu dökümente ediyor, arkasındaki bir elemanı alıp junyır gibi onu yetiştiriyor mesela. O yüzden takımın birbirine uyumu çok önemli. Hatta bazı firmalarda interviev lerde teknik interviev den önce veya beraber konuşuluyor bu konular takım uyumuna bakmak için yemek ayarlanıyormuş. Yani sizin takıma uyum sağlayıp sağlayamamanıza yönelik sorular yöneltiliyor, buna önem verecekler.
12. Takımın bir araya gelip neyi iyi yaptık neyi kötü yaptık, önümüzdeki dönemlerde planlarımız nasıl olmalı , işimizi kolaylaştıracak hangi adımları atıp daha rahat bir şekilde çalışabilirizin konuşmasını yapan bir takım oluşturmanız gerekiyor. Agile ye uyum sağlayabilmek için.

Bunlardan hangisi Agile prensiplerinden biri değildir.

1. patronu memnun et

Dersin başında bahsettiğimiz konuyu tekrar ele alalım

Her bir prosesi tek tek ele alıp çözmeye mi çalışırsınız, ya da birden fazla prosesi, işi aynı anda ele alıp çözüme götürmeye çalışırsınız.

Her şeyin belli bir sıraya göre işlemesi tabii ki gerekiyor, bunları prodac owner ler ayarlıyor.

Bir kişinin günde ne kadarlık bir işi yapacağı aşağı yukarı bilinen bir şey, o sınırı aşmayacak şekilde, üzerinden kalkacağı şekilde o işi alması, o kişinin kapasitesini aşmayacak şekilde o işi alıp bitiryor olması gerekir.

Agile bize ne getirdi

İnsanların kafasında klasik manada yanan o ampulü, klasik manada uyguladıkları o düşünceyi tamamen söküp yerine ışıl bir ampulü getirip oraya takıyorlar. Yani insanlar eski alışkanlıklarının bir çoğunu değiştirmiş oldu. İnsanların yazılım geliştirme alanında, çalışma alanında yaptığı şeyler bu.

Özellikle IT alanında yapılan araştırmalar şunu gösteriyor, ortalama çalışma süreleri iki yıl. Kendinizi kabul ettirdiğiniz sürece, takımla uyumlu bir şekilde çalıştığınız müddetçe sizi elinde tutmak için farklı seçenekler sunuyor size.

Agile nin bize getirdiği yeniliklerden bir tanesi de İT ekibiyle iş ekibinin birbiriyle arasında olan duvarları yıkmak.

Devops ta hatırlıyor musunuz bir duvar yıkmıştık o duvar kimlerin arasındaydı ?????

Agile de de bu bisinıs ekiple IT ekibi arasındaki iletişim kopukluğu giderilmiş oluyor.

Bununda scrum da göreceksiniz e, en önemli kazanımlarından bir tanesi prodact owner gibi bir rol. Yani bu iletişimin arttırılmasında büyük rol oynuyor. Ona da yeri geldiği zaman kim bu ne iş yapar ona bakacağız.

Agile nin yine bir diğer önemli maddelerinden bir tanesi de bizim değişikliğe her daim açık olmamızı, yeniliğe her daim açık olmamızı bir planı harfi harfine uygulamak yerine her saniyede her dakikada dışarıdan gelecek, müşteriden gelecek her tür değişikliğe devamlı açık oluyoruz ve bu değişiklik emin olun bizi olumlu yönde etkiliyor olacak, şirketin kar etmesi, yeni bir işin gelmesi

KAHOOOT

1. PMI neyin kısaltmasıdır
2. Agile bize geleneksel metodlara göre büyük bir avantaj sağlar
3. Aşağıdakilerden hangisi agile için yanlıştır
4. Agile manifestosu ile ilgili yanlıştır
5. Agile uygulamanın faydalarından biridir
6. Agile nin en önemli özelliği, en büyük önceliktir
7. Devoloper ekibi agile ye geçince neyi kaybeder
8. Agile manifes prensiplerinde bu agile manifestosu neyi ifade eder diyebiliriz
9. Değişikliğe açık olacağız, neyin yerine
10. Aşağıdaki manifesto maddelerinden hangisi yanlıştır.

AGGİLE METHODS

Agile yazılım geliştirmesinde bir yaklaşım tarzıydı. Şimdi bu iş yapış yönteminde iki tane freamwork, iki tane çerçeve çizeceğiz agile da. Tabi agile nin altındaki freamwork lere baktığınızda çok fazla terimler var, şu şemsiyenin altına baktığınızda FDD, TDD, Extreme, DSDM, Lean bunları duyan var mı bilmiyorum ama bunların en popüleri Scrum ve bir diğeri de Kanban geliyor. Birbirini tamamlayan noktalar var. Bunları da göreceğiz. Burada agileyi bir şemsiye olarak gördüğünüzde o agile ın uygulandığı yöntemler içerisinde scrum ve kanban öne çıkıyor iki tane freamwork olarak, biz de dersimizde bu iki freamwork ü sizlere anlatmaya çalışacağım.

Şimdi geçmişteki yine proje sistemlerine baktığımızda soldaki resimde koca bir yükü, koca bir projenin tamamını bir anda ele almaya, o işin tamamını yüklenme gibi bir işe girişiliyordu. Koca bir ürünü tek bir kod parçası olarak yazmaya çalışıyorlardı ve bir hata çıktığında o hatanın tespit edilmesi günlerce ayıklanması sorun nerde onun bulunması gibi bunlarla uğraşılıyormuş.

Ama günümüz teknolojisinin getirdiği şeylerden birisi bu , agile a geçildiğinde işlerin küçük parçalara ayırıp, üstesinden geline bilinir, tabi daha sonra bir araya getirilecek şekilde fakat bunlar parça parça ele alınıp belli bir zaman aralığında bunları yapılabilecek şekilde parçalara ayırabiliyoruz.

Önünüze koca bir doküman veriliyor, nereden başlayacağınızı bilmiyorsunuz. Bir plan olmadan dalamazsınız içine. O yüzden agile nin getirdiği bu parça parça çalışma iki gün üç gün daha rahat çalışma şeklinde ilerlemek gerekiyor.

Agile yi bir kelime ile anlatmak isterseniz ne dersiniz ?????

Scrum bir bakalım;

Scrum ı duyan var mı (scrum master ???)

Scrum, komplex ve büyük problemlerin altından en hızlı bir şekilde en güzel bir şekilde kalkabilmenin bir yöntemi. Burada problemi küçük parçalara ayırma, öncelik verme, kişilere atama, işlerin takibini yapma, takımı motive bir şekilde tutma bunları scrum ile sağlayabiliyoruz.

Basit bir yöntem, anlaması kolay, orada master yapmak uzmanlaşmak biraz zor, zaman zaman çalıştığınız yerlerde yıllarca çalışan arkadaşlar bile olsa bu konuda sık sık scrum ve kanbanla ilgili eğitimler alacaksınız.

Bu şekli bu dersimizde ve diğer seşında da göreceğimiz bir resim. Bu feedback in manasında olan ve itereşindan( yineleme) gelen bir şekil, itereşinla beraber üzerine koyarak ürünü geliştirmeyi bize anlatıyor.

Bu scrum ı agile nin altına koyduk ama aslında daha önce geliştirilmiş bu agilenin geliştirilmesine öncülük eden bu kişiler agile den önce bu metodolojiyi geliştirmişler. 90 yılında. Şu an % 50 den fazla yazılım firmaları dünyada bu scrum ı uyguluyor diyebiliriz.

Scrum Framework

Product owner ; bu kişi execut yetkisine sahip yani icra yetkisine sahip, müşterilerden isterleri alan, fedbacleri alan, bunları alıp sağ taraftaki takıma bunları rikuayrımıntlar şeklinde, iş emirleri şeklinde yansıtan kişi. Bu yapılacak listeyi takımın anlayacağı şekilde hazırlayan kişi bu arkadaşlar. (aslında takım içinden çıkan bir kişidir, iki tarafı da anlayan mühendis gibi düşünebiliriz.)

Bu proje bu kişi takvimi , ne kadar zamanda bitecek, hangi gün nereden ne alınacak, bütün öncelikleri product owner yönetiyor. (bu bir değil birkaç kişi de olabilir)

Bu kişi bizimle nasıl iletişim kuruyor, bizle iletişim kurduğu vasıta backlog (bunu da çok duyacaksınız). Biz hani koca koca dökümanlar demiştik ya aslında bunun dijitalleştirilmiş hali diyelim jira dediğimiz programlar sayesinde bu prodac backlog lar yazılıyor.

Nedir bu prodack backlog, iş listeleri, alışveriş sayfasının nasıl olacağı, tek tek adım adım anlatılır. Genelde bunlar PO yazmaz, bunun altını sinyor developer ya da sinyor enginier ler doldurur ve hazırlarlar. Sıfırdan yazılmıyor, ekleme çıkartma yapılabiliyor.

Biz burada bütün projeyi ayarladık 1000 tane kalem iş belirleniyor. Projenin başlangıcında hepsi doldurulmak zorunda değil. ( clarusway sprintleri de öyle )

Sprintler belirleniyor, spring miiting yapılıyor, iki haftada bir olabilir, ekipte kimler oluyor işte prodac owner geliyor, scrum master imiz var, development teamimiz var, bunlar bir araya geliyor, bu toplantıda amaçlarımız şu, önceliklerimiz şunlar biz projede bunları yapacağız, vakit kalırsa şu işide ilave edebiliriz diyor. Ardından takımdan yani işi bizzat icra edecek ekipten bir girdi varsa o da fikirlerini beyan edebilir eklenecek bir şey de böyle de olabilir diyebiliyor. Orada şu olacak bunu uygulayın diye dikte edilmiyor. Bu böyle takımın içerisinde tartışılıyor bu baclog lardan takımın kapasitesine göre iki haftada başarabileceği iş yüküne göre bagloglardan gerekli bilgiler alınıp siprinte atılıyor ve sprint başlatılıyor.

Ne yapılıyor iki hafta boyunca her sabah Dely stendap meeting oluyor, dün şununla uğraştım takıldım kaldım, yardıma ihtiyacım var, ya da her şey iyi gidiyor bugün şu işle uğraşacağım, şu dep biriyle topl olacak , akşam 1 saat erken çıkıp sonra sabah erken bu işi halledeceğim, bu genelde yazılı da yapılabiliyor, genelde jira gibi bir programı zoom dan takımın scrum master i ekrana yansıtıyor herkese söz verdikçe o kişinin üzerinde olduğu, ona ait olan işler gözüküyor ve anlatıyor.

Bu neyi sağlıyor, prodact owner altında belki beş altı kişi çalışıyor ama kimin ne iş yaptığını bilmiyor. Ama hergün bir 10 dk böyle anlattığında birincisi, durumdan haberdar olmuş oluyor. İkincisi devops alanına baktığınız zaman biri serverlar üzerinde bir şey yaptığında diğeri bundan haberdar oluyor ve problem çıkmıyor, birbirinden haberdar olmuş oluyorlar.

2-3 haftadan sonra, sprintten sonra bir ürün çıkıyor, web sitesi ana sayfası bitti. Müşteriyi çağırıyoruz ve anlatıyoruz. ( Spring Reviev de oluyor) kaç sprint yaparsanız hepsinde müşteriyle görüşüp üzerine ilave ede ede ürünü çıkarmış oluyoruz.

Sprint reprospektif miitingimiz oluyor, bu da müşteri gelmiyor buna müşteri dışında tüm ekibin toplandığı neleri iyi yaptık (önce övüyoruz) , olumluluğu her zaman önde tutuluyor, daha sonra uygun bir dille neleri yanlış yaptıysak onun düzeltilmesine yönelik, karşıyı da kırmadan şu şöyle olsa daha iyi diyebilir veya geliştirebileceğimiz alan varsa, şunu şu toolla yapsan daha hızlı olur gibi. Tabi bu tüm takımdan geliyor, bir kişi söylemiyor bunu. Bu şekilde haftalık yada iki haftanın değerlendirilmesi yapılarak takımın buradaki amacımız en iyiye en mükemmele ulaşmasını sağlamak. Kısa zamanda fazla faydayı sağlamaya çalışmak için bu işlemler yapılıyor.

Scrum frimwork e tepeden baktığımızda bu işin özeti bu şekilde, birazdan ayrıntılarına da bakmış olcağız.

Hiçbir şeyi sıfırdan yapmayacaksınız, kopyala yapıştır, ya da bakıp yapacağımız daha önceden yapılmış olanlar var onlara uyarlayarak yapacaksınız, çalışmış arkadaşlar bu şekilde yapıyor.

Soru:

Scrum projeleri nelere bölünür

Scrum practices (praktisis)

Rolleri, sermonileri ve artifectleri var.

Rolleri ne demiştik, prodact owner var, scrum master imiz var ve bir development teamimiz var. Buna ilave olabilir (data analistler olabilir)

Ardından seremonilerde, sprint planlama toplantısı var. Yığın halinde duran o işten iki haftalık sprint için yapacağımız işleri alıyoruz, sprintimize koyup başlıyorduk.

Daily scrum meet, ne yapıyorduk, günlük 10 dk yı geçmeyecek şekilde scrum bordun önünde ya da zoom dan yansıtarak takım ne üzerine çalıştı, bugün ne üzerinde çalışacak, sıkıştığı yerler var mı bundan bahsediyorlar.

Sprint Reivev topl vardı, bu müşteriyi de davet ettiğimiz, bize geri bildirimlerini sunduğu çok önemli toplantılardan bir tanesi, scrum ı agile yapan ya da agile yi agile yapan şeylerden bir tanesi. Müşteriden aldığımız geri bildirim toplantısı.

Sprint ritrospektif topl, bizim takım halinde neleri iyi yaptık, neleri kötü yaptık

Prodact baclog gruming---- siprint bittiğinde açıkta kalan işleri tekrar yeni siprinte döndürme, yapılamayan işlerin ele alındığı bir toplantı olabilir.

Artifecler neydi, elimizdeki dökümanlarımız diyebiliriz. Prodact baglog larımız---bunlar yığın halinde prodac owner de bulunan bacloglardı. Sprint baclog lar, sadece sprintte ele alınacak bacloglarımızdı.

Prodac inkremınt dediğimizde siprinti bitirip sağ tarafa attıklarımız yapıldı bitti tarafına çektiğimiz bacloglarımıza Prodac inkremınt diyoruz. En son bu şekilde her şey sağa atıldığında ürün müşteriye teslim edilecek hale geliyor. Her sprint sonunda çıkan, müşteriye sunduğumuz ürünümüz.

Scrum Roller

Product Owner,

Proje vizyonunu, gereksinimlerini ve önceliklerini tanımlayarak projenin başarısından sorumlu olan kişi.

Bizim projenin başarıya ulaşması için projenin tanımlanmasından, yapılacak işin önceliklendirilmesi, o işlerin yapılması esnasında bizim ihtiyaç duyacağımız kaynakların bize sağlanması gibi görevleri üstlenen ve bunu icra etmekle yükümlü kişi. Takımı yönetme yetkisi bu kişidedir.

Scrum Master

Ürün Sahibi'nin hedeflerine ulaşmasını engelleyecek engelleri kaldırmak için ekibe karşı sorumludur.

Bülent korkmaz gibi, takım kaptanı gibi düşünürsek, takımı motive eden, takımda en fazla çalışan PO karşı takımı temsil eden, takımın karşılaştığı sorunları dile getiren, takımın önündeki engelleri üst kademeye ileten kişi.

Takımın içinden biri, takımın içinde pişmiş, takımı ayakta tutan biri, bu prodact owner la ekip arasında iletişimi sağlayacak kişi de diyebiliriz. Developer da olabilir bu, mühendislikten gelmiş birisi de olabilir

Development Team

Ekip, çeşitli rollere sahip 3-9 kişiden oluşur ve Ürün Sahibinin hedeflerine en iyi şekilde nasıl ulaşılacağını belirlemek için kendi kendini organize eder.

Bizlerin aslında çalışacağı development team olacak, projenin isterlerine göre takım ayarlanabiliyor. Kimler olur bu ekipte…

Genelde 6-7 arası oluyor yoksa toplantılar çok uzun sürüyor, çünkü vakit çok önemli, 15 dk çok fazla oluyor.

Product Owner ile başlayalım.

PO ekipteki iş temsilcisidir.

Ürünün ve müşterinin paydaşlarının sesini temsil eder. Ürün Sahibi, Geliştirme Ekibi'nin çalışmaları sonucunda ortaya çıkan ürünün değerini en üst düzeye çıkarmaktan sorumludur. Bunun nasıl yapılacağı organizasyonlar, Scrum Takımları ve bireyler arasında farklılık gösterebilir.

Ürünün ve müşterinin **business representative (iş temsilcisi)** temsil eder.

Bir Product Owner ideal olarak aşağıdaki **özelliklere sahip olmalıdır**:

Available (eveylibıl) : Planlama ve açıklama için her zaman hazırdır.

**Business Savvy (bizinıs se vii) İş Bilgisi**: İş veya o alan hakkında derin bir anlayışa sahip olacak.

Communicative (kominikidiv): İyi bir dinleyicidir (iletişimsel), paydaşlara kolayca ulaşma ve mesajı karşı tarafa uyarlama becerisine sahip olacaktır.

Empowered (impavırd): Karar verme yetkisine sahip olacak ve onlardan sorumlu tutulabilecek.

Kararlı / belirleyici

Decisive ( disaysiv) Belirleyici: Ürünün yönüne karar verebilecektir..

PO ekibin her talebin ayrıntılarının açıkça anlaşılmasını ve ayrıca her talebin bir öncelik sırası olmasını sağlamak çalışır.

Ayrıca paydaşlarla günlük olarak etkileşim halindedirler. İnsanların bir ekiple etkileşime girmesi yeterli değildir.

Ayrıca iş bağlamında meydana gelen tüm değişikliklerle uyumlu olmalıdırlar.

Sonuç olarak, ürün vizyonunun koruyucusu, yapılacak işlerin birikmiş listesini tanımlar, yönetir ve bu iş kalemlerine öncelik verir.

Ayrıca, ekibi her kısa teslim süresinde mümkün olduğu kadar çok işi tamamlamaya zorluyor.

Scrum Master

Takımın bu talepleri nasıl karşılayabildiğini merak ediyorsanız, yalnız değilsiniz.

Scrum'ın kurucuları, "ürün sahibi" rolüne karşı bir denge oluşturma ihtiyacının farkındadır.

Böylece bir usta rolü yarattılar.

scrum ustası takımı korur ve süreci korur.

Scrum Ustaları, herkesin Scrum teorisini, uygulamalarını, kurallarını ve değerlerini anlamasına yardımcı olarak bu sorumlulukları yerine getirir.

İnsanların talepleri ile ekibin ihtiyaçları arasında denge kurarlar.

Bu rol, ekiplerin sürdürülebilir bir düzeyde performans göstermesini sağlamaktan sorumludur.

Dolayısıyla, Scrum Masters ideal olarak şunlara sahip olmalıdır:

Responsible (rispansıbıl) : Sorumlu: Ekibi harekete geçirmek ve ekip çıktısını en üst düzeye çıkarmak

Humble (hambıl) Alçakgönüllü: Takımın egoya izin vermeden zafere ulaşmasını sağlar.

Collaborative (kolabradiv) İşbirlikçi: Ekipte bir işbirliği kültürü oluşturur

Committed (kımedid): Bağlılık: Tam zamanlı Scrum Master'ları olmasalar bile takıma %100 bağlılık gösterin

Influential (influvençol) Etkileyici : Yetkisiz etki yaratabilmek

Knowledgeable (navlecıbıl): Bilgili: Scrum uzmanı ve teknik çevre bilgisi

Scrum Master, Ürün Sahibine aşağıdakiler dahil olmak üzere farklı şekillerde hizmet eder:

Scrum Takımındaki herkesin amaçları, kapsamı ve ürün alanını mümkün olduğunca anladığını garanti etmek.

Ürün İş Listesinin verimli bir şekilde ele alınması için stratejiler aramak.

Scrum ekibinin bir ürün biriktirme listesinin şeffaf ve açıklayıcı olması ihtiyacını takdir etmesine yardımcı olmak.

Analitik bir çerçevede ürün planlamayı bilmek.

Şirket Sahibinin, karı optimize etmek için ürün biriktirme listesinin nasıl yönetilebileceğini anlamasını sağlamak.

Çevikliğin anlaşılması ve uygulanması.

Gerektiğinde veya gerekli olduğunda scrum etkinlikleri sunmak

Geliştirme Ekibine kendi kendine organizasyon ve işlevler arası koçluk yapma

Geliştirme Ekibinin yüksek değerli ürünler yaratmasına yardımcı olmak

Geliştirme Ekibinin ilerlemesinin / başarısının önündeki engellerin kaldırılması

İstendiği veya ihtiyaç duyulduğu şekilde Scrum etkinliklerini sağlamak / kolaylaştırmak

Hangisi projeye müşterinin bakış açısıyla bakıyor?

Scrum Master'ın rolü nedir?

Development Team

Geliştirme Ekibi, her Sprint'in (yineleme) sonunda "Bitti" tanımını karşılayan, ürünün potansiyel olarak piyasaya sürülebilen bir parçasını teslim etmekten sorumlu profesyonellerden oluşur.

Ürün Parçasını yalnızca Geliştirme Ekibinin üyeleri geliştirir.

Geliştirme Ekipleri, kuruluş tarafından kendi çalışmalarını düzenlemek ve yönetmek için kurulan ve yetkilendirilen ekiplerdir. Ortaya çıkan sinerji, Geliştirme Ekibinin genel verimliliğini ve etkinliğini en üst düzeye çıkarır.

Geliştirme Ekibi üyeleri "programcı"dır.

Bunun anlamı, Geliştirme Ekibinin tüm üyelerinin karmaşık ortamlarda ürün veya hizmet geliştirmek için birlikte çalışmasıdır.

Geliştirme Takımları aşağıdaki özelliklere sahiptir:

Kendi kendilerini organize ediyorlar. Hiç kimse (Scrum Master bile) Geliştirme Takımına Ürün İş Listesini potansiyel olarak serbest bırakılabilir işlevselliğe sahip Ürün Parçalarına nasıl dönüştüreceğini söylemez;

Geliştirme Ekipleri, bir ürün Parçası oluşturmak için gerekli tüm becerilere sahip çapraz fonksiyoneldir(cross-functional);

Scrum, kişi tarafından gerçekleştirilen iş ne olursa olsun, Geliştirme Takımı üyeleri için hiçbir unvan tanımaz;

Scrum, test etme, mimari, operasyonlar veya iş analizi gibi ele alınması gereken alanlardan bağımsız olarak, Geliştirme Ekibinde hiçbir alt takımı tanımaz; Ve,

Bireysel Geliştirme Ekibi üyeleri özel becerilere ve odak alanlarına sahip olabilir, ancak sorumluluk bir bütün olarak Geliştirme Ekibine aittir.

Arkadaşınızın bugün derse gelmediğini varsayalım…

Dersi arkadaşınıza açıklamak zorunda kalsanız ne söyleyeceğinizi yazın.

Epic, User Story and Task

Scrum projelerinde kullanılan üç temel terim Epic, User Story ve Task'tır.

Bir scrum projesini tam olarak anlamak için bu kavramların ne olduğunu anlamak gerekir.

Bu kavramlar arasında hiyerarşi vardır. Bunlar hiyerarşik olarak birbirlerini oluşturan parçalar

Task;

Bir developer in günlük üzerinden kalkabileceği küçük küçük görevler, günde bir iki tane başarabileceği günlük işleri, hep atanan görevlerle halleder development ekibi.

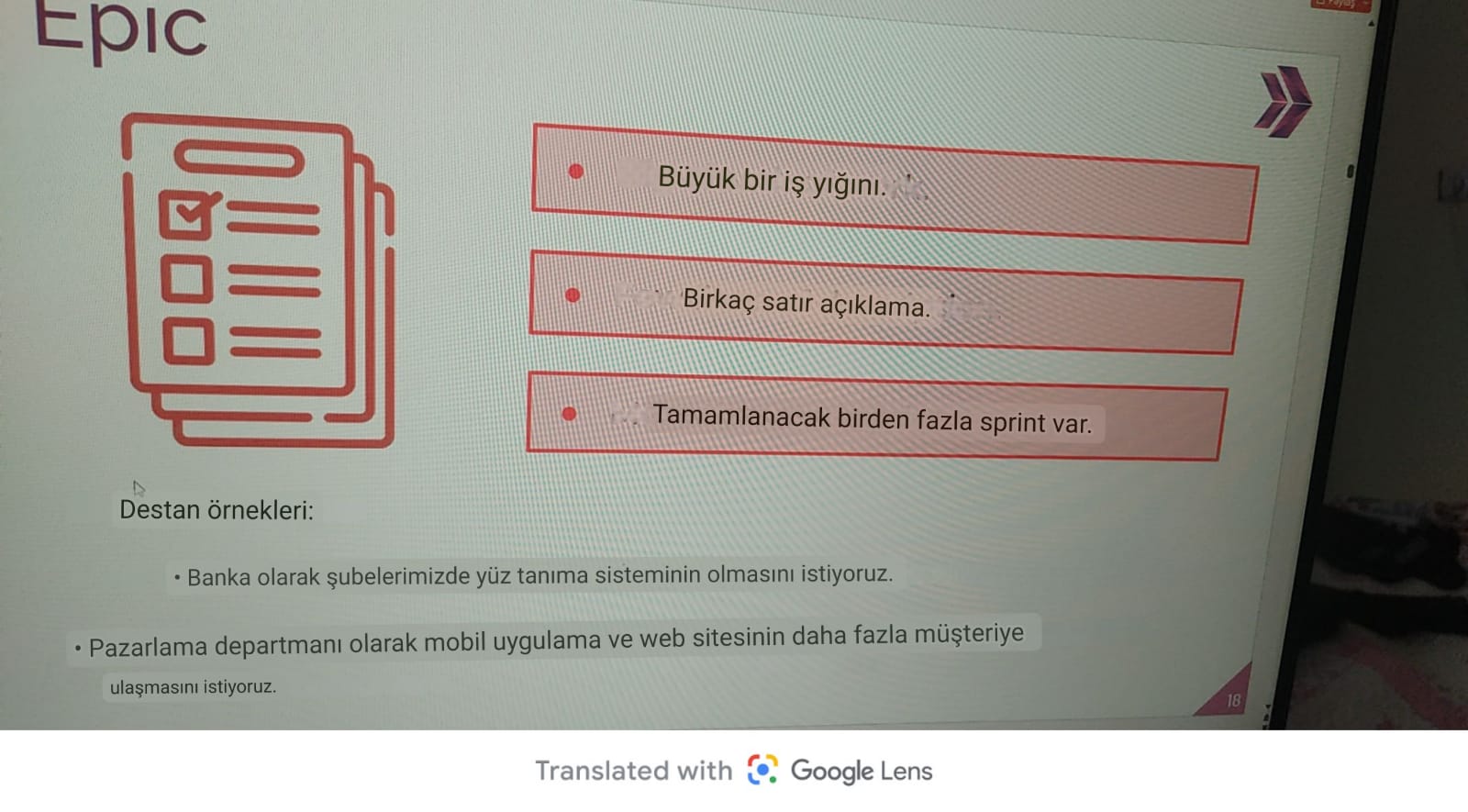
Projenin en küçük parçaları. Sub tasklarde olabilir, jira da bölüyorsunuz bunları da.

User Story;

Bir çok taskin bir araya gelmesiyle oluşan , bir web sitesinin bir sayfası gibi.

Epic,

user story nin birkaç tanesinin birleşmesi



Epik bir bankanın yüz tanıma sistemini epic olarak tanımlayabiliriz. Mobil web sitesinin yapılması epik olarak tanımlanabilir. Bunu ifade ederken standart tabirleri var;

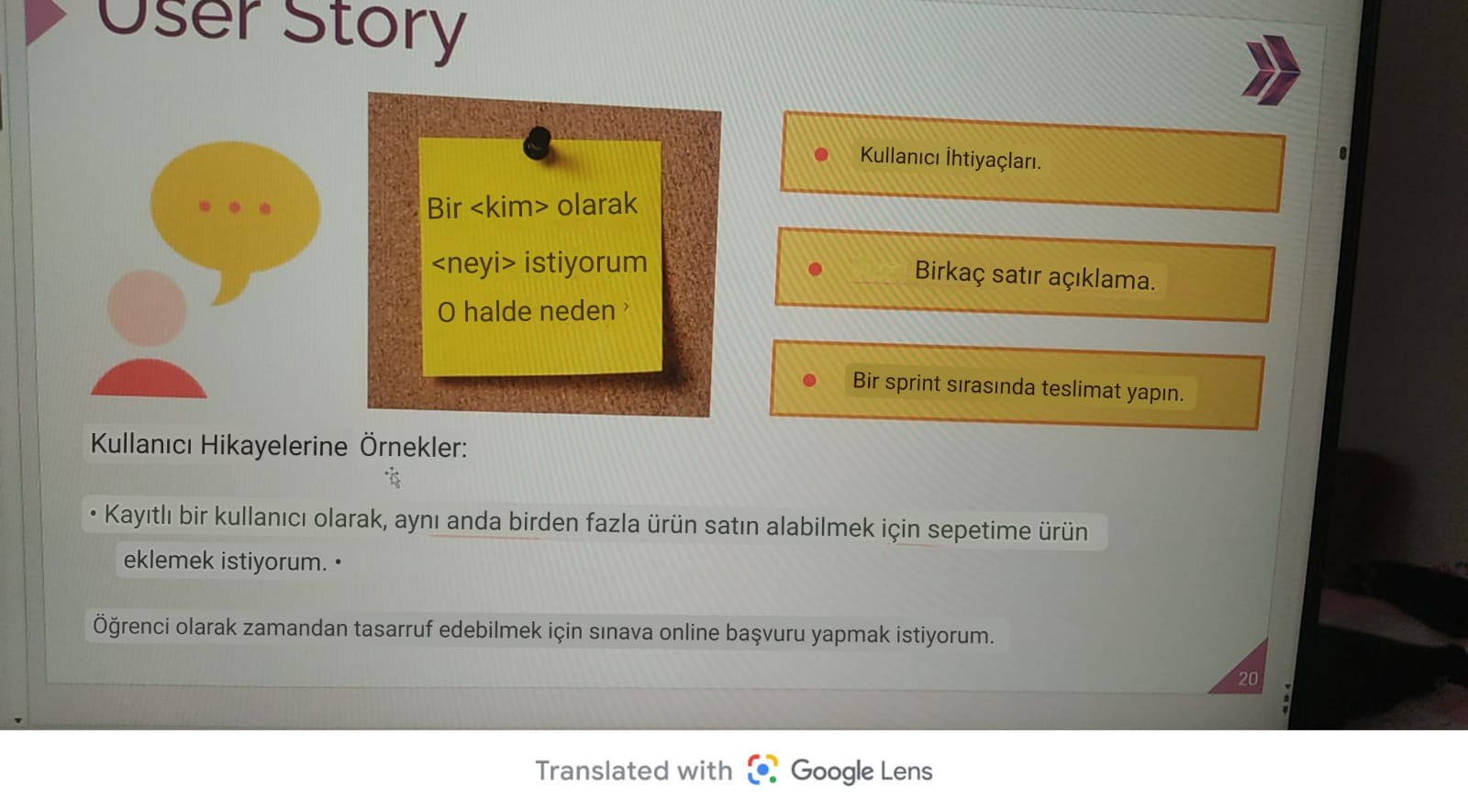
Ben bir banka olarak yüz tanımlama sisteminin servislerin altında yer almasını istiyorum. Müşteri bağlanırken sadece çift korumalı sistemin değil extra 3. Bir korunma sistemiyle bağlanmasını istiyorum diyebilir.

Ya da bir market olarak mobil uygulamamızın ve bir web sitemizin hayata geçirilmesini istiyoruz diyebilir.

epic i gördüğü zaman development ekibi için bir şey ifade etmez bu user story lerle ve task lerle bu işin nasılını anlatmamız gerekecek.

--

User story



*Ekibin müşteri için doğru olanı oluşturmasını sağlamanın en etkili yolu*

*geliştirme sürecinde bu ihtiyaçları akılda tutmak ve kullanıcı gözüyle kayıt altına almaktır.*

*Kullanıcı hikayesi, teknik olmayan bir dil kullanılarak birkaç kısa cümleyle ifade edilen bir yazılım sistemi gereksinimidir.*

*Aslında her user story, yazılımın bir işlevidir ve takımın bir veya iki haftalık bir sürat içinde bitirmeyi taahhüt edebileceği şeylerdir.*

Hangi usr olarak istiyor (KİM)

Ne istiyor (NE)

Niye istiyor ne faydası var (NİÇİN)

Amazon alış-veriş sitesini düşünün birkaç ürün baktınız 3 ünü birden almak istiyorsunuz.

Sisteme giren bir kişi şunu yapabilsin ve bu özellik sayesinde şunu yapabilsin ve birden fazla ürünü aynı anda alabilsin.

Veya bir öğrenci aöf okuyor, sınav için başvuru yapacağım… diyor neden gideyim başvurayım internetten başvurmak istiyorum diyor.

Sınava başvuru butonu o da başka bir servise açılıyor.

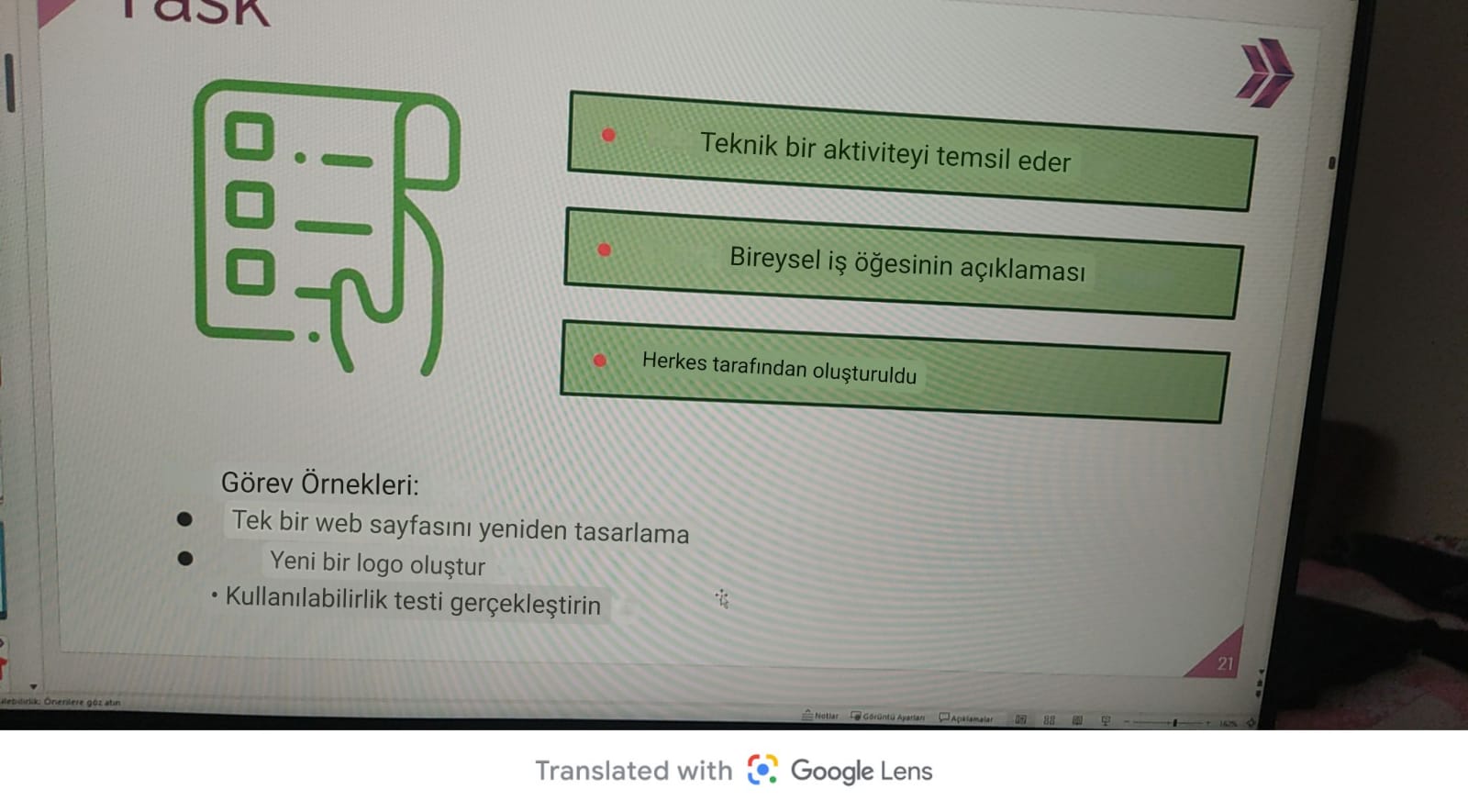
Task;

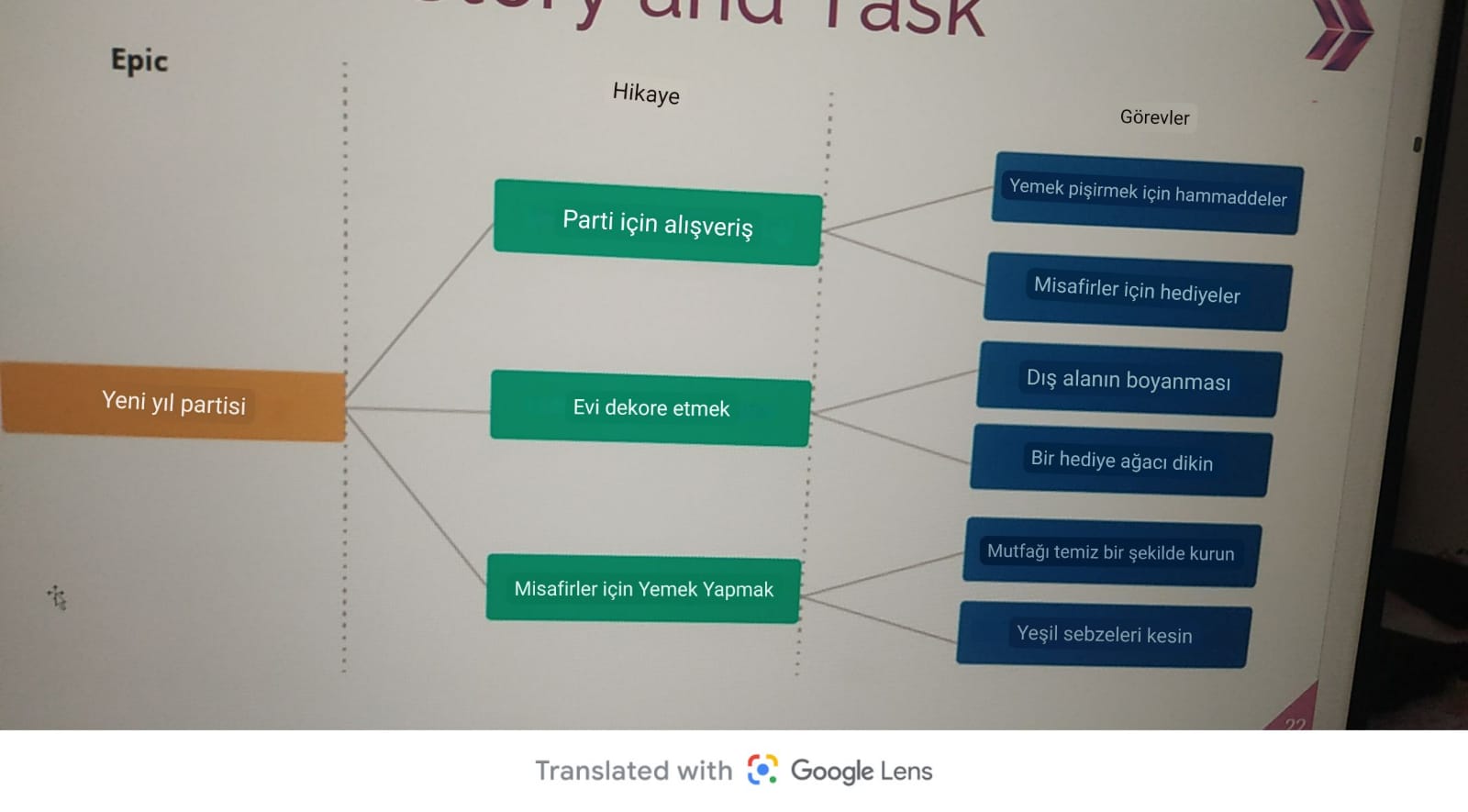
Tasklerde yeteri kadar zaman veriliyor

Siz gittiğinizde şirketlerde her şey hazır olacak iş yerlerinde siz birkaç değişiklik yapacaksınız, kopyala yapıştır yapacaksın.

Sizin yapmanızdan ziyade o kodu ya da işlemi anlatabiliyor musunuz, anlatamıyor musunuz ona bakacaklar. Sonra bu olmuş ya da olmamış şunları ekle geliştirme yap gibi şeylerle haşır neşir olacaksınız.

Developer için önemli olan task…



Bir örnek

Yeni yıl partisi vereceğim, bu konuyu hazırlamak için küçük parçalara ayırabilirim.

Bu örnekte 3 hikaye var.

Birincisi, partiye hazır olmak için alışverişe gitmeliyim.

Ve ilk kullanıcı hikayesi için görevler; Yemek pişirmek için markete gitme ve misafirlere hediye alacağım.

İkinci; Parti sahibi olarak evi dekore etmeliyim görevler şunlardır:

ve üçüncüsü:

Bir parti sahibi olarak yemek hazırlamalı ve yemek pişirmeliyim.

Bunları elle yapmıyoruz, bazı programlarla yapacağız bunları

(hangi proje yönetim aracına, tool, aşinasınız)

Estimation

Proje yönetimi söz konusu olduğunda, büyük soru her zaman “Bu ne kadar sürecek?”dir.

Doğru teslimat tahmini yapmak, müşteri memnuniyeti için en iyi yoldur.

--Her iş aynı büyüklükte değil, bu iş kaç adımlık, kaç saatlik bir iş---sağ üst köşede yazar.

--Her şirkette farklı olabilir bu numaralandırma

Tahmin etmek kolay değil aslında.. Yazılımcılar için işin en zor yanlarından biri.

Herkesi (geliştiriciler, tasarımcılar, testçiler, dağıtımcılar... herkes) takıma dahil etmek çok önemlidir.

Her ekip üyesi ürüne farklı bir bakış açısı getiriyor

Hikaye noktası, ekibin her bir kullanıcı hikayesinin boyutuna odaklanmasını sağlar.

Planlamanın amacı, hikayelerin kesin tamamlanma tarihlerini tahmin etmektir.

Ekip, ne kadar büyük olduğuna bağlı olarak her hikayeye bir puan değeri atar.

Takımların tahminde bulunmalarının bir yolu, hikayeleri tişört bedenlerine bölmek ve her bedene bir puan değeri atamak.

Örneğin, ekstra küçük özellikler için 1 puan, küçük özellikler için 2 puan ve orta özellikler için 3 puan, büyük özellikler için 4 puan ve ekstra büyük özellikler için 5 puan kullanmaya karar verebilirler.

Bir ölçeğe karar verdikten sonra, her hikayenin hangi kategoriye girdiğine karar vermelidirler.

--

Bu slaytta örnek bir hikaye noktası tahmini hile sayfası var

Örneğin, ekibiniz dağıtım için her zaman flask çerçevesini kullanır, ancak bu sefer Django çerçevesine ihtiyaç duyar.

Yeterli tecrübeniz yok ve bağımlılıkları bilmiyorsunuz, bu yüzden 8 veya 13 puan verebilirsiniz.

Scrum practis

Prodact baclog-----sprint baclog---- ürün bitince

Arada ilave istekler

Prodac baclog

Ürün biriktirme listesi, bir ürün oluşturmak için gereken kullanıcı hikayeleri/yeni özellikler, geliştirmeler, hata düzeltmeleri, görevler veya iş gereksinimlerinin bir listesidir.

Müşteri desteği, pazar talepleri ve genel iş analizi gibi girdi kaynaklarından derlenir.

Ürün biriktirme listesi, yeni bilgiler mevcut olduğunda talep üzerine güncellendiği için "canlı" bir yapıdır.

Ürün Sahibi, Ürün İş Listesinin içeriğinden, kullanılabilirliğinden ve sıralamasından sorumludur.

Ürün İş Listesi kalemlerinin bir açıklaması, sırası, (boyutu) tahmini ve değeri vardır.

Piroty (pa oridiy)-----öncelik ataması (pasta yapılacak)

Story point ----- daha kolay ulaşılabilen, önceliği olan üste yazılır

Bundan PO sorumlu

Üstteki sprintlerden kırmızı olan varsa daha zamanı gelmese bile alınabilir

Task; developer, devops bakabiliyorum, zaten slack den de görev gelecektir size

Prodact incremınt

Bir Sprint sırasında tamamlanan tüm Ürün İş Listesi kalemlerinin toplamı ve önceki tüm Sprint'lerin artışlarının değeri.

Bir Sprint'in sonunda, yeni Ürün Parçası "Bitti" olmalıdır.

Ürün Sahibinin onu serbest bırakmaya karar verip vermediğine bakılmaksızın, ürün parçası kullanılabilir durumda olmalıdır.

Product inctement, bir Sprint sırasında tamamlanan product backlog kalemlerinin ve geçmiş tüm Sprintlerin Ürün Parçalarının değerlerinin toplamıdır.

Sprint'in sonunda, yeni Ürün Parçası "Bitti" olmalıdır, yani mevcut olmalı ve Scrum Takımının "Bitti" tanımını karşılamalıdır.

Artış, vizyona veya hedefe doğru bir adımdır.

Ürün Sahibinin piyasaya sürmeye karar verip vermediği - kullanılabilir durumda olmalıdır.

Bu terimi anlamak için basit bir gerçek hayattan örnek alalım. Diyelim ki evinizi dekore etmek istiyorsunuz. Evinizin fayanslarını ve boyasını değiştirmek istiyorsunuz. Dekoratör bir odayı bir haftada, diğerini bir haftada, ardından mutfağı ve son olarak da oturma odasını bitirebilirdi. Haftanın sonunda bir oda bittiğinde sonucu görebilirsiniz. İlk hafta artışı bir oda ve ikinci hafta artışı küçük oda ve mutfaktır. Her hafta ürünün artan değerini alabilirsiniz. Ve sonunda, bütün ev artımdır Sprint'in sonunda yeni bir Ürün Parçası "Bitti" olmalıdır,

product backlog listesi öğelerine kim öncelik verir?

Seremoniler

Sprint planing--- PB lardan işleri alacağız ve onu parçalara böleceğiz

Daly stundap---- dün ne yaptım, bugün ne yapacağım, işime engel bloklar var mı

Sprint reivew---- müşteriye sunduğumuz yer, tanıtıyoruz, reklamını yapıyoruz

Retrospectif ---- hangi toool kullanalım, ne yapabilirim, ne iyi gitmedi, ne geliştirilebilir

Herkesin konuşmaya hakkı var

Sprinti bitiremeyince ne yapıyoruz---bir sonraki sprinte planlanıyor

PB Grooming -- Scrum üretim sürecinde, ürün backlog'u ürün sahibi, Scrum takımı ve ilgili paydaşlar arasında sürekli olarak güncellenen ve optimize edilen bir belgedir. Ürün backlog grooming, Scrum takımının ürün backlog'unu düzenlemesini ve iyileştirmesini sağlayan bir etkinliktir.

Tüm Sprint Öğeleri tamamlanamadığında ne olur?

Takımın tüm Sprint İş Listesi kalemlerini tamamlayamadığı bir durumda hiçbir şey olmaz.

Sprint, belirtilen tarihte tamamlanan öğelerle sona erer.

Geliştirme Takımı, Sprint İnceleme toplantısında tamamlanan öğeleri gösterir Tamamlanmayan kalemler product backlog a geri taşınır ve oradan tekrar önceliklendirilir.

Burndown Chart

İş bitim çizelgesi, zamana karşı yapılacak işin grafiksel bir gösterimidir. Grafikte, dikey eksen genellikle kalan işi, yatay eksen ise zamanı veya günleri temsil eder. Tüm işlerin ne zaman tamamlanacağını tahmin etmek için kullanışlıdır. Slaytta basit bir iş bitim grafiği örneği verilmiştir.

İş bitim grafiği, belirli bir sürat koşusunda veya sürümde kalan işin günlük ölçümünü sağlar, product backlog Sprint-by-Sprint'te kalan iş miktarını izler. Bu iş bitim çizelgesi her Sprintte güncellenir ve Scrum Takımı ve paydaşları tarafından görülebilir.

Üst kademenin ekibi ölçen kademenin eline veri almak için yapılan cartlar, sprintlerde işler ne kadar eritildi, hangi sürede bitirildi görmek için kullanılır.

Velcity cart

Hız Tablosu, her bir sprintte teslim edilen değer miktarını göstererek, ekibin gelecekteki sprintlerde ne kadar iş yapabileceğini tahmin etmenize olanak tanır.

Sprint planlama toplantılarınız sırasında, uygulanabilir olarak ne kadar iş yapabileceğinize karar vermenize yardımcı olması açısından yararlıdır.

Üst kademeye yardımcı olan veriler dedik, sprintte planlanan iş miktarı ne kadar, biz bu sprintte ne kadarını bitirebildik.

scrum master'ın sorumluluğundadır

kahot yapabiliriz

KANBAN

Kanban, Agile sistemi işletmede uygulamak için bir başka popüler metodolojidir.

Kanban, üretim süreçlerinde görsel bir yönetim sistemi olarak kullanılan ve Japon otomotiv şirketi Toyota tarafından geliştirilen bir yöntemdir. Kanban, Japonca'da "görsel kart" veya "görsel işaret" anlamına gelir.

Kanban yöntemi, üretim süreçlerinde verimliliği artırmak, envanteri yönetmek ve aşırı yüklenmeyi önlemek amacıyla ortaya çıkmıştır. İlk olarak 1940'lı yıllarda Toyota'nın üretim sistemi olan "Toyota Üretim Sistemi" (Toyota Production System - TPS) kapsamında kullanılmaya başlanmıştır.

\*\*\*

Toyota firmasında parçalar eksilince yeni üretim için stokta eksik parça olduğu için üretim gecikiyor. Buna bir çözüm bulmak için marketteki sistemi kendi iş yerlerine taşımak istiyorlar. Markette nasıl raflar hiç boş kalmıyor ona bakıp kendi iş yerine uyarlıyor.

Kanban'ın hikayesi 1940'lara kadar uzanıyor. Bu yıllarda Toyota, süpermarketlerin raflardaki stokları yönetmek için kullandığı modeli temel alarak üretim yöntemini güncelledi. Toyota, süpermarket tarafından kullanılan sistemi benimsedi. Kapasite seviyelerini gerçek zamanlı olarak izlemek için şirket, farklı üretim ekipleri arasında "kanban" adı verilen bir kart kullanmaya başladı.

Üretim hattında kullanılan bir kutu malzeme boşaltıldığında, depoya hangi malzemenin gerekli olduğunu açıklayan bir kanban gönderildi. Deponun sağladığı yeni malzemelerle üretime kesintisiz devam edildi ve süreç sürekli izlendi. Kanban yöntemi, üretimden son tüketiciye kadar tüm zinciri kontrol eder. Bu sayede üretim sürecinin çeşitli aşamalarında hem tedariğin aksamasını hem de malın fazla stoklanmasını engeller. Bu, sürecin sürekli izlenmesini gerektirir.

Hikayesi 1940'lara dayansa da kanban'ın bilgisayar teknolojisi dünyasındaki uygulaması nispeten yeni bir konudur. 2004'te David J. Anderson, yazılım geliştirme için kanban kavramını kullanma fikrini ortaya attı ve 2010'da adlı bir kitap yazdı."Kanban: Teknoloji İşiniz İçin Başarılı Evrimsel Değişim."Kanban'ın yazılım geliştirmede kullanımı David J. Anderson'ın denemelerinden sonra başlar.

Kanban sistemi, üretim sürecindeki malzeme akışını düzenlemek için kullanılır.

Kanban, JIT (Just-in-Time)(tam zamanında) üretim yaklaşımı ile ilişkilidir. JIT, müşteri taleplerine göre sadece ihtiyaç duyulan miktarda malzeme üretmeyi ve tedarik etmeyi hedefler. Kanban sistemi, JIT'in temel prensiplerinden biri olan envanteri azaltmayı ve aşırı yüklenmeyi önlemeyi destekler.

Kanban, neyin üretileceğini, ne zaman üretileceğini ve ne kadar üretileceğini gösteren görsel bir çerçevedir.

Kanban, iş akışı sürecini görselleştirmeye odaklanır. Kanbanın birincil amacı, süreçteki olası kusurları tespit etmek ve işlerin istenilen hızda ilerlemesini sağlamak için bunları düzeltmektir.

* Kanban'da belirli bir zaman sınırı yoktur
* Görevler için son teslim tarihlerinde esnek
* Belirli roller de yoktur

Kanban Boarad

Kanban panosu, kanban için her şey demektir.

Kanban panosu, iş akışı görselleştirme için bir araçtır. İş akışınızı ve görevlerinizi bir Kanban panosunda görselleştirmek, süreçlerinizi daha iyi anlamanıza ve iş yükünüz hakkında bir genel bakış elde etmenize yardımcı olur.

Bu yeni şeffaflık düzeyiyle, sorunlu çalışma aşamalarını hızla belirleyeceksiniz, Kanban panoları, iş öğelerini temsil etmek için kartları ve sürecin her aşamasını temsil etmek için sütunları kullanarak bir sürecin çeşitli aşamalarındaki işi görsel olarak tasvir eder. İlerlemeyi göstermek ve işi gerçekleştiren ekipleri koordine etmeye yardımcı olmak için kartlar soldan sağa hareket ettirilir.

Bu, slaytta gördüğünüz kanban panosu.

To do yapılması beklenen işler

Ongoin ---- yapmaya başladığınız, start verdiğiniz işler soldan bu tarafa kaydırıyorsunuz

Review--- merge edeceğiniz işlemleri buraya tanımlıyorsunuz bunu yetkili birisi alıp bir diğer aşamaya uygunmu diye bakıyor

Done--- daha sonra bu uygun görüldü, sistem çalışıyor, tekrardan ilave işlemlerden geçti, sıkıntı gözükmedi işleminiz bitmiş oluyor.

Kanban board a örnek jira da değil Trello board da hazırlanmış

Kanban boardlarda yer alan bilgiler kuruluştan kuruluşa farklılık gösterir. Başka bir deyişle, boardlara ne koyacağınız size kalmış.

Boardlar şunları içerebilir;

Kartın ne hakkında olduğuna dair bir isim veya çok kısa bir açıklama. Proje adı veya kimliği, özellikle devam eden birden fazla proje varsa. Başlangıç, bitiş, aşamalar vb. gibi iş akışı tarihleri.

Öncelik derecelendirmeleri olabiir, o işin hangi aşamaya geldiğiyle ilgili bize bilgi verir

Buradaki tıkıtlara tıkladığınızda o tikıtların detayları gözükecektir.

\*\*Proje yöneticisinin, normalde herkesin ne yaptığını tek tek not alması zor olabilir, bütün süreci yönetmek zor olabilir, board sayesinde atacağınız her adım tek tek kaydedilmiş oluyor.

Lead time

Tikıt--- tanımladığım yerdeki kısımda alıp done dediğimiz kısma kadar olan zamana diyoruz

1 tikıt ın ne kadar sürede tamamlandığını ifade eden terimdir

Ciycle time

Devam eden aşamaya girdiğinden itibaren başlar, yani siz kod yazmaya başladığınızda başlar bitireceğiniz ana kadar geçen süreyi ifade eder

Kanban prensipleri

Kanban, iş akışını yönetmek ve verimliliği artırmak için Dört Temel İlke ve Altı Temel Uygulama benimsemiştir.

Kanban'ın dört ilkesi:

1. Mevcut kurulum veya işlemde anında değişiklik yapmayın.

Yani o anda ekibiniz nasıl çalışıyorsa, nasıl bir yönteminiz varsa o çalışma yöntemini uygulamaya devam ettirmeliyiz kanbanda.

1. Artımlı, evrimsel değişimi sürdürmeyi kabul edin:

Değişime illaki gerek varsa bunu yavaş yavaş yapmalı, kanbana uyum sağlanacaksa bu değişiklikler yavaş yavaş olmalı.

Direnmeye yol açabilecek büyük değişiklikler yerine ekip içinde küçük artımlı değişiklikler yapın.

1. Scrum gibi size roller diktesi yok, roller oluşturulmuyor.
2. Hiyerarşi yok, PO veya SMaster gibi rol olmadığı için, takım içerisinde kimsenin kimseye üstümlüğü yok, kişilerin liderliği ele alıp kendi işlerini yapmaları gerekiyor, bu kaabiliyetini geliştirmesi gerek

işi aldın mı sorumlu sensin master falan yok

İlkeleri tek tek kontrol edelim

--Kanban Metodu sizden sürecinizi değiştirmenizi istemez.

Kanban, doğrudan mevcut iş akışına uygulanmalıdır. İş akışıyla ilgili.

Takımın rahat etmesi için gerekli değişiklikler yavaş ve kademeli olarak yapılmalıdır.

Kanban Yazılım Geliştirme Süreci veya Kanban Proje Yönetim Metodu diye bir şey yoktur.

Kanban'ın esnekliği ve basitliği, bir Kanban sisteminin halihazırda çalışmakta olanı kesintiye uğratmadan mevcut iş akışları üzerinde eşlenebilmesi gerçeğinde yatmaktadır.

--Artımlı Devam Etmeyi Kabul Edin

Büyük değişiklikler yerine küçük artımlı değişiklikler yapın

Büyük değişiklikler takım içinde dirence yol açar

-- İyi performans gösteren mevcut rollerinizi ve işlevlerinizi değiştirmeniz gerekmez

Ekip, roller ve unvanlarda gerekli değişiklikleri koordine edecek ve uygulayacaktır.

-- Organizasyonun tüm seviyelerinde sürekli iyileştirme teşvik edilir

Tüm ekip üyelerinin fikir üretmesi ve liderlik göstermesi arzu edilir.

Kanban, ekiplere ve yöneticilere değer akışını iyileştirmeye odaklanma ve kimin ne yaptığına odaklanmayı bırakma yeteneği sunar.

Kanban'ın amacı, insanları yönetmeyi bırakıp işi yönetmeye başlamaktır ve bunun için tüm ekip üyelerinin mevcut durumu anlamayı ve iyileştirme fırsatlarını görmeyi öğrenmesi için bu Kanban ilkelerini ve çok basit uygulamaları sunar.

Kanban, değer akışının daha fazla aşamasını veya organizasyonun daha fazla katmanını anlamaya genişledikçe, bu karar verme ve organizasyonu iyileştirme yeteneği tüm seviyelere uzanır.

Bu Kanban ilkesi bize liderliğin ön saflardaki insanların günlük eylemlerinden geldiğini hatırlatır.

Herkesin ekip / departman / şirket düzeyinde optimum performans elde etmek için sürekli iyileştirme zihniyetini teşvik etmesi önemlidir.

Sürekli iyileştirme herkesin sorumluluğundadır.

Kanban Prktis ----- uygulamaları

1. Visualizing the Workflow ( vijulazing tı workflow), İş Akışını Görselleştirme ; Kanban ile başlamanın ilk adımıdır Süreci görselleştirdiğinizde, sizin ve ekibinizin şu anda ne yapmakta olduğu görülebilir.

Yaptığımız işlerin görsel hale getirilmesi, daha iyi organize olmamızı sağlıyor. Yaptığımız organizasyon büyük ise takip edilesi kolay oluyor, arayıp bulmak yerine açıp bakınca anlıyoruz

1. Limit Work in Progress (limi work in pragras), Devam Eden Çalışmayı Sınırlandırın; (Devam Eden İşin Sınırlandırılması (WIP)), kanban'nın ikinci uygulamasıdır. Herhangi bir zamanda bir sütunda bulunabilecek maksimum kart sayısına Devam Eden Çalışma limitleri denir. Her sütuna bir sınır atadığınızda, ekibiniz kaldırabileceklerinden daha fazla çalışmayacaktır.

Bir sayı belirtiyorsunuz kapasite ekibinizin kapasitesinin üstüne çıkmaması için böyle limitler koyuyorsunuz

Wip nedir, sürdürülebilir iş için diyebiliriz. Takımın üzerinde fazla iş var diyelim burada kırmızı şekil çıkar o işi o kadar ekip üyesi yapamayacağı uyarısını verir bunları ayarlar, jira adminıster ayarlar

1. Managing Flow (menicing flav), Akışı Yönetme ; Görselleştirme ve her aşamanın sınırlandırılmasından sonra, üçüncü uygulama tamamen süreç akışını iyileştirmekle ilgilidir. Akışı yönetmek ve iyileştirmek, ilk iki uygulamayı uyguladıktan sonra kanban sisteminizin kilit noktasıdır. Çünkü bu noktada iki seçenek vardır. Work in Progress sınırları içerisinde sorunsuz bir akışla karşı karşıya kalacak veya işlerin birikmeye başladığını gözlemleyeceksiniz. Bu, ürünlerin teslim süresini etkiler.

Akışı siz yönetiyorsunuz, işlerimiizi biz planlıyoruz, haftalık….., odaklanmak daha kolay oluyor

1. Making Process Policies Explicit (meking proses palisis eksplsi), Süreç Politikalarını Açık Hale Getirmek ; Diğer 3 ilke pratikten çok teoriktir. Örneğin dördüncü uygulama hakkında Süreç politikalarının, temel ilkelerin ve çalışma yöntemlerinin netleştirilmesi kanbanın dördüncü uygulamasıdır. Bu konularda net olmak, ekibin süreci kolayca takip etmesini ve gerekli iyileştirmeler için önerilerde bulunmasını sağlar. Başka bir deyişle, mevcut süreç hakkında net olmayan ekip üyeleri sistemi iyileştiremez.

Şeffaf bir şekilde sürecin de nasıl işleyeceğini de biliyorsunuz. Politikamız, nasıl iş yapacağımız , hangi kurallara uyacağımız hepsinin açık bir şekilde ifade edilmesi lazım

1. Implementing Feedback Loops (implementin fidbek lups), Geri Bildirim Döngülerinin Uygulanması ; Burada beşinci uygulama devreye giriyor, müşterileriniz veya son kullanıcılar gibi sisteminizde yer almayan kişilerden geri bildirim almanız gerekiyor. Ayrıca, istenen işlevselliğin yüksek kalitede sunulmasını sağlamak için bir sistem içinde geri bildirim döngüleri gereklidir. Burada farklı türde testler gerçekleşir. Geri besleme döngülerini kısalttıkları için sürekli otomatik testler tercih edilir.

Toplantılar, geri bildirim, takımımıza yeni şeyler kazandırıyoruz.

1. Evolving Experimentally (ivalving expremendly), Deneysel Olarak Gelişmek ; Kanban evrimsel bir iyileştirme süreci olduğu için sisteminizi sürekli değerlendirmeli ve sürekli iyileştirmelisiniz. Bu, süreçte küçük değişiklikler yapmanıza ve iş akışını sabit bir hızda iyileştirmenize olanak tanır.

Agile ilkeleri uyguluyorsanız, asıl göreviniz sürecinizi sürekli olarak değerlendirmek ve gerektiğinde iyileştirmektir.

Yaptığımız işlemlerin doğru olup olmadığını bilimsel şeylerle ölçüyoruz, işleri yaptıkça takımı ölçmek lazım, iyi yaptıklarımızla, kötü yaptıklarımızla, scrum a benziyor, bu sayede takımın gelişmesini sağlıyoruz.

Scrum ve Kanban

Kanban ve scrum yöntemlerini birbirinden tamamen ayırmak kolay değildir.

Örneğin, çoğu insan scrum ekiplerinin scrum sürecinde kanban panoları kullandığına inanır.

Öte yandan, iki yöntem arasında bazı temel farklılıklar vardır.

* Scrum'daki sprintlerin başlangıç ve bitiş tarihleri varken, kanban'da belirli bir zaman sınırı yoktur.
* Scrum tahtasındaki görevler için son tarihler olsa da, kanban bu konuda daha esnektir,
* Ürün Sahibi, Scrum Master ve Geliştirme Takımı gibi bir Scrum projesinde belirlenen rollerin aksine, Kanban'da belirli roller yoktur.
* Takım her sprintten sonra Scrum panosunu yeniden düzenlerken, proje boyunca aynı kanban panosu kullanılır.

Daha spesifik bir sürece ihtiyacınız olduğunda Scrum daha iyi bir seçimdir.

Ancak tüm senaryoda çok fazla değişiklik yapmadan sürecin yürütülmesini iyileştirmek istiyorsanız Kanban kullanmalısınız.

Scrumban

Scrumban, adından da anlaşılacağı gibi scrum ve kanbanın birleşimidir. Scrumban, Scrum'dan Kanban'a geçmek isteyen takımlar için bir basamak taşı olarak da kullanılabilir.

Birçok yazılım geliştirme ekibi için hemen Kanban'a geçmek çok zor olacaktır. Scrumban, takımlara Scrum'ın tanıdık doğasından ayrılmadan Kanban'daki sürekli iyileştirme uygulamaları hakkında bilgi edinmenin bir yolunu sunar.

Scrum ve Kanban'ı Agile'ın yöntemleri olarak biliyoruz. Scrum, ürünler ve geliştirme projeleri için en uygun olanıdır. Kanban, üretim desteği için en iyisidir. Bakım projeleri için her ikisinin de en iyisini bir araya getiren Scrumban kullanıyoruz.

Scrumban, bugünlerde hem geliştirme hem de bakım projeleriyle hizmet endüstrilerinde çok popüler hale geldi. Scrumban, iş yükünü en aza indirmek ve çekme tabanlı bir sistem benimsemek isteyen ekiplerin ihtiyaçlarını karşılamak için ortaya çıktı. Scrum ve Kanban'ın bir karışımı, ekiplere proje metodolojileri tarafından aşırı yüklenmeden paydaşlara ve üretim ihtiyaçlarına uyum sağlama esnekliği sağlar. Scrumban, Kanban'ın esnekliğini ve görselleştirmesini Scrum'ın yapısına sağlayarak iş akışı yönetimine çok yönlü bir yaklaşım sunar.

1. Hangisi yanlış Scrum yaklaşımını kullanarak, ekibiniz kilometre taşlarını nasıl gerçekleştirmelidir?
2. Scrum master ne iş yapar
3. Her sprintin sonunda Scrum Takımlarının yaptığı bir şeyi seçin
4. Hangisi günlük Scrum toplantısında tartışılacak bir konu DEĞİLDİR?
5. Zamana karşı yapılması gereken işin grafik gösterimi.
6. Hangi toplantıda katılımcılar şunları tartışır: Ne iyi gitmedi?
7. Kanban'ın ilk prensibi .....
8. Kanban …………….. üzerine odaklanır.
9. Bir kolonda bulunabilecek maksimum kart sayısına …………. denir.
10. ......., iş akışınızda bir görev oluşturma ile kanban panosundan son ayrılışı arasındaki dönemdir.