T.C.
UFUK ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME ANABİLİM DALI
YÖNETİM BİLİŞİM SİSTEMLERİ PROGRAMI

ÖZ ÇERÇEVE (ESSENCE FRAMEWORK) TABANLI MOBİL YAZILIM GELİŞTİRME: ENDÜSTRİDE BİR UYGULAMA

YÜKSEK LİSANS TEZİ

MUSTAFA ERGAN

TEZ DANIŞMANI
DOÇ. DR. MURAT PAŞA UYSAL

ANKARA
2019

T.C.
UFUK ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME ANABİLİM DALI
YÖNETİM BİLİŞİM SİSTEMLERİ PROGRAMI

ÖZ ÇERÇEVE (ESSENCE FRAMEWORK) TABANLI MOBİL YAZILIM GELİŞTİRME: ENDÜSTRİDE BİR UYGULAMA

YÜKSEK LİSANS TEZİ

MUSTAFA ERGAN

TEZ DANIŞMANI
DOÇ. DR. MURAT PAŞA UYSAL

ANKARA
2019

KABUL VE ONAY

Mustafa ERGAN tarafından hazırlanan "Öz Çerçeve (Essence Framework) Tabanlı Mobil Yazılım Geliştirme: Endüstride Bir Uygulama" başlıklı bu çalışma, 10.06.2019 tarihinde yapılan savunma sınavı sonucunda başarılı bulunarak jürimiz tarafından Yüksek Lisans Tezi olarak kabul edilmiştir.

Doç. Dr. Alaattin PARLAKKILIÇ- Başkan

Doç. Dr. Murat Paşa UYSAL- Danışman

Dr. Öğr. Üyesi Mesut ÜNLÜ- Üye

Yukarıdaki imzaların adı geçen öğretim üyelerine ait olduğunu onaylarım.

Prof. Dr. Mehmet TOMANBAY

Enstitü Müdürü

UFUK ÜNİVERSİTESİ 1996 ANKARA SOSYAL BİLİMLER ENSTİTÜSÜ

BİLDİRİM

Hazırladığım tezin tamamen kendi çalışmam olduğunu ve her alıntıya kaynak gösterdiğimi taahhüt eder, tezimin kâğıt ve elektronik kopyalarının Ufuk Üniversitesi Sosyal Bilimler Enstitüsü arşivlerinde aşağıda belirttiğim koşullarda saklanmasına izin verdiğimi onaylarım:

† Tezimin/Raporumun tamamı her yerden erişime açılabilir.

10.06.2019

Mustafa ERGAN

TEŞEKKÜRLER

Bu çalışmanın yürütülmesinde başından sonuna destek olan, her adımda yardım ve katkılarını esirgemeyen tez danışmanım Sayın Doç. Dr. Murat Paşa UYSAL'a sonsuz saygı ve şükranlarımı sunarım.

Bana tez boyunca her türlü desteği sağlayan hayat arkadaşım, canım eşim Hilal'e, bugünlere gelmemi sağlayan ve yazılım üzerine yüksek lisans yapıyorsam onun sayesindedir dediğim canım annem Gülten'e çok teşekkür ederim.

Tez boyunca yardımlarını esirgemeyen tüm ERGAN, HAMAMCI ve ELMAS ailelerine çok teşekkür ederim.

Mustafa ERGAN

Ankara 2019

ÖZET

ERGAN, Mustafa. Öz Çerçeve (Essence Framework) Tabanlı Mobil Yazılım Geliştirme: Endüstride Bir Uygulama Yüksek Lisans Tezi, Ankara, 2019.

Teknolojik gelişmeler kapsamında mobil cihazlar ve uygulamalar her geçen gün baş döndürücü bir hızla gelişmekte, bu tür yazılımlardan işletme ve müşteri beklentileri de çok yönlü olarak artmaktadır. Bu doğrultuda karşılaşılan güçlükler ise çoğunlukla teknik düzeyde ele alınmakta ve alan bağımlı sınırlı çözümler üretilmektedir. Bu araştırmanın konusuyla ilgili bir diğer önemli problem ise mobil yazılım geliştirme süreçleridir. Yazılım mühendisliği araştırma alanı ve endüstrideki uygulamalar incelediğinde, faydacı yaklaşım çerçevesinde hali hazırdaki geleneksel veya çevik yazılım mühendisliği yöntem ve tekniklerinden faydalanıldığı gözlenmektedir. Ancak, teknolojik, yöntemsel ve bağlamsal açıdan farklı niteliklere sahip Mobil Yazılım Mühendisliği (MYM) uygulamalarının gittikçe karmaşıklaşacağı, bu alana özgü ve kuramsal temellere dayalı üst seviye mobil yazılım geliştirme yöntem ve tekniklerine ihtiyaç olduğu değerlendirilmektedir. Bu araştırmada söz konusu probleme çözüm üretebilmek amacıyla MYM'ne yönelik Öz Çerçeve'ye (Esence Famework) (ÖÇ) dayalı bir yazılım süreç modeli geliştirilmiş, siber güvenlik eğitimine yönelik bir mobil yazılım projesinde uygulama sonuçları gözlemlenerek tartışılmıştır. Çalışma Karma Araştırma Yöntemleri doğrultusunda, Eylem Araştırması (Action Research) ve Tasarım Bilimi Araştırma Yöntemleri (Design Science Research) (TBAY) kullanılarak bütünleşik yapıda yürütülmüştür. Elde edilen bulgu ve gözlemlere dayanarak ÖÇ'nin, MYM araştırma ve uygulama alanlarındaki eksikliklere belirli ölçüde çözüm getirebileceği, farklı bilgi alanları arasında da bir köprü vazifesini görebileceği düşünülmektedir.

Anahtar Sözcükler

Öz Çerçeve, Mobil Yazılım Mühendisliği, Scrum

ABSTRACT

ERGAN, Mustafa. Mobile Software Development Based on Essence Framework: An Application in Industry, Master's Thesis, Ankara, 2018.

Within the scope of technological developments, mobile devices and applications are evolving with each passing day at an incredible pace, and therefore, business and customer expectations from these types of software are also increasing in many ways. The difficulties encountered in this context are mostly handled at a technical level and domain-specific solutions are usually produced. Another important issue, which is also related to the topic of this research, is mobile software development processes. As the software engineering research area and applications in the industry are reviewed, it is observed that traditional or agile software engineering methods and techniques are mostly used. However, it is considered that Mobile Software Engineering (MSE) applications, which are technologically, methodically and contextually different, is expected to be more complicated in near future. Therefore, we think that theoretically high-level mobile software development methods and techniques are required. In order to produce solutions to this problem, a software process model, which is based on EF (Essence Framework) is developed, and its application is observed in a mobile software development project for cyber security education. The research study is carried out in an integrated way using Action Research and Design Science Research. Based on the research findings and results obtained, it is possible to state that EF may bring some solutions to the problems of MSE to some extent, and also it may be a bridge between different software engineering knowledge areas.

Keywords

Essence Framework, Mobile Software Engineering, Scrum

İÇİNDEKİLER

SİMGELER VE KISALTMALAR DİZİNİ

ÖÇ	: Öz Çerçeve
MYM	: Mobil Yazılım Mühendisliği
EA	: Eylem Araştırması
TBAY	: Tasarım Bilimi Araştırma Yöntemleri
MUG	: Mobil Uygulama Geliştirme
MUGM	: Mobil Uygulama Geliştirme Modeli
UML	: Birleşik Modelleme Dili
Bkz	: Bakınız
GIS	: Coğrafi Bilgi Sistemleri
ERP	: Kurumsal Yönetim Sistemi
SEMAT	: Yazılım Mühendisliği Metot ve Teorisi
F	: Fırsat
P	: Paydaş
İH	: İhtiyaç
YS	: Yazılım Sistemi
İYB	: İş Yapma Biçimi
AS	: Araştırma Sorusu
YT	: Yazılım Takımı
İKY	: İş Kırıım Yapısı

TABLOLAR DİZİNİ

ŞEKİLLER DİZİNİ

BÖLÜM 1

1. GİRİŞ

Mobil cihazlar ve özellikle cep telefonları gün geçtikçe gelişmekte, yenilenmekte, buna paralel olarak mobil hizmetlerle ilgili işletme ve müşteri beklentileri çok yönlü olarak artmaktadır (Wasserman T. , 2010). Mobil yazılım geliştiricileri bu kapsamdaki yazılım projeleriyle uğraşırken yazılım mühendisliğiyle ilgili başta yöntemsel ve teknik olmak üzere çeşitli güçlüklerle de karşılaşmaktadırlar (Wasserman A. , 2010). Mobil yazılım geliştirme çalışma alanındaki teknik düzeyde karşılaşılan belli başlı problem sahalarını aşağıdaki gibi sıralamak mümkündür:

- Farklı yapıdaki donanım ve yazılım platformları,
- Telefon operatörleriyle ilgili farklı özellikler ve sınırlılıklar,
- Mobil yazılım geliştirme yaşam döngüsünün çok kısa olma zorunluluğu,
- Bir mobil yazılımın birden fazla platformda çalışma zorunluluğu ve hızlı güncelleme gerektirmesi,
- Kullanıcı etkileşimleri, ara yüzleri, sensörler, kamere vb. donanım ve yazılım arasındaki çok boyutlu etkileşimler ve ilişkiler (Yamakami, 2016),
- Mobil yazılımlara özgü siber güvenlik model ve uygulamalar,
- Donanım, yazılım, güç yönetimi, disk kapasitesi, vb. konulardaki zorluklar,
- Mobil cihazlardaki uygulamaların birbirleri ve özellikle dış servisler arasındaki karşılıklı bağımlılıklar (Blum, 2010).
- Yazılım bileşenlerinin farklı platform ve yazılımlarda tekrar kullanılabilir olmasındaki güçlükler,
- Geleneksel yazılımlarda farklı olarak farkındalık kapsamında sürekli çevre ve ortamla iletişimde olma zorunluluğu,
- Mobil cihazlarındaki güç yönetimi sorunu ve mobil yazılımlarda bunu sağlayacak tekniklerin sınırlılığı, çeşitli uygulama güçlükleri,
- Doğal (native) ya da tüm ortamlarda çalışan (hibrid) web uygulamalarının birlikte çalışabilirliği,
- İşlevsel olmayan (non-functional) farklı nitelikteki yazılım kalite ihtiyaçları ve bunlardaki dinamik ihtiyaçların gerçek zamanlı güncellenme gereksinimidir.

Yukarıda kısaca bahsedilen ve çoğunlukla teknik sorunları farklılaştırmak ya da yenilerini eklemek mümkündür. Mobil yazılım geliştirme çalışma alanı ile bu konudaki araştırmalar incelendiğinde proje ve kurum düzeyinde çözümler getirilmeye çalışıldığı, yazılım mühendisliği bilgi alanın sağlamış olduğu yöntem, teknik ve araçların güncel proje ihtiyaçlarına uyarlandığı gözlenmektedir (Allen & Zamanian, 2010). Araştırmacıların bir bölümü hali hazırdaki yazılım mühendisliği çözümlerinin mobil yazılım süreçleri için yeterli olduğu, temel prensip ve teknikleri sağladığı görüşündedirler (Lab, 2010). Bunlara göre işlevsel ihtiyaçların daha basit ve sınırlı olması, teknolojik ihtiyaçlardaki farklılık, iş ihtiyaçları vb. durumlar çok farklı yazılım geliştirme yöntem ve süreçlerinin benimsenmesini, mobil yazılım mühendisliğine özgü yöntemlerin kullanılmasını zorunlu kılmaktadır. Örneğin Yazılım Ürün Hattı yönteminin MYM ilgili sorunlara ışık tutabileceği belirtilmektedir.

Öte yandan bu alanda çalışan uygulamacılar, Mobil Yazılım Mühendisliği (MYM) uygulamalarının gittikçe karmaşıklaştığı ve farklılaştığını, çoğu zaman diğer yazılım projelerini de kapsayabildiği, onların önemli bir parçası olarak projede önemli etkilerinin olabildiği ve bu eğilimin de öngörülemeyecek biçimde artacağı öne sürülmektedir. MYM'de belli başlı zorluk ve farklılıkları aşağıdaki gibi sıralamak mümkündür (Sedano, 2010):

- Geleneksel yazılım mühendisliği yöntem ve tekniklerinin uyarlanma güçlüğü,
- Yazılımın tekrar kullanılabilirliğinin sağlanması,
- Proje ihtiyaçlarındaki belirsizliklerin daha fazla olması,
- Çevik yazılım proje yönetimi uygulamalarındaki güçlükler,
- Yazılım proje takımının göreceli olarak daha küçük olması, yazılım mühendisliği disiplinindeki yaklaşımlar dışında hızlı ve kolay yazılım geliştirme sürecine yönelik sorunlar,
- Yazılım testlerine yönelik yöntem, teknik ve araçların uygulanmasında karşılaşılan sorunlardır.

Öte yandan yazılım çalışma alanı ve sektöründe önemiyle beraber popülerliği artan bir başka konu da siber güvenlik, bu konuyla ilgili ihtiyaçlar ve uygulamalardır (Francese, Risi, & Tortora, 2015). Bunlardan birisi de siber güvenlikle ilgili yetenek ve yeterliliklerin çevrimiçi eğitimlerle kazanılma gereksinimidir. Farklı yazılım çözümleri olmakla birlikte mobil uygulamaların sınırlı olduğu gözlenmektedir. Yazılım sektöründe

siber güvenlik alanındaki eğitimlerin sağlanacağı bir yazılım geliştirme modeli, bunun uygulanabileceği bir platforma da ihtiyaç bulunmaktadır (Griss, 2010). Kullanıcı, müşteri ve çevreyle ilgili kısıtlamalar bulunmaktadır. Ancak bu konuda en iyi uygulama haline gelmiş ve önerilen bir MYM yöntemi ise bulunmamaktadır.

İşletmelerin (enerji, yazılım, güvenlik vb. alanlarda çalışan) çeşitli gereksinimlerden dolayı (GIS, ERP, vb.) çeşitli platformlara mobil yazılımlar geliştirmesi gerekebilmektedir (Bhowmik, Alves, & Niu, 2013). Kullanılan yazılım geliştirme yöntemleri, teknikleri ve platformları çeşitli durumlara ve gereksinimlere göre değişebilmektedir. Bu durum önemli belirsizlikler ve problemlerin kaynağı durumundadır. Bulut hizmetleri ve telekomünikasyon gelişmesiyle birlikte mobil cihazlar ve cep telefonları da gelişmiştir. Zorluklar beraberinde birden fazla donanım ve yazılım platformu, birçok geliştirme çerçevesi ve programlama dili, farklı operatör kısıtlamaları, kısa iletişim döngüleri, bağlamın etkin kullanımı, hesaplama ve depolama da sınırlama gibi konuları gündeme getirmektedir (Kwon & Tilevich, 2015).

Yazılım Mühendisliği ve MYM araştırma ve çalışma alanındaki problem sahaları birlikte ele alındığında şu ihtiyaçların karşılanması gerektiği söylenebilir:

- Yazılım endüstrisinde çeşitli ihtiyaçlara yönelik ve farklı platformlar için mobil yazılım geliştirilmektedir. Yazılım süreç yönetimi, teknikleri ve araçları mobil yazılımının türü, yapısı ve teknolojisinden bağımsız, ortak bir çerçevede ele alınabilmelidir (Usman, Iqbal, & Kha, 2014).
- Uygulamacılar, mobil yazılım süreci boyunca her aşama ve durumu tanımlayabilmeli ve bunlara dayalı olarak proje sürecini etkili biçimde takip edebilmelidir (Yu & Yuan, 2011).
- Bir mobil yazılım geliştirme projesinde kullanılacak yazılım geliştirme yöntem, teknik ve araçları yazılım takımının ihtiyaçları, deneyimi ve kendi istekleri doğrultusunda esnek biçimde oluşturulabilmelidir.
- Çevik yazılım yöntem ve uygulamalar desteklenebilmeli, mobil yazılım proje süresince kullanılan teknik ve araçların gelişen durum ve yazılım takımının ihtiyaçlarını değiştirilmesi, güncellenmesi veya yeniden oluşturulmasına olanak verecek yöntem veya model olmalıdır (Mattsson, 2010).

- Kullanılan yazılım geliştirme yöntemi, değişik büyüklükte her türlü mobil yazılım sistemine uyarlanabilmeli, küçük çaplı projeden büyüğe doğru ölçeklenebilen ve evrilebilen özelliklere sahip olmalıdır (Nekoo & Vakili, 2009).

MYM endüstrisindeki uygulamalar, bilgi ve çalışma alanındaki karşılaşılan söz konusu problemlere çözüm getirebilmek amacıyla bu araştırmada, MYM yönelik bir yazılım geliştirme süreç modeli geliştirilmiştir. Ayrıca, bir yazılım firmasında bu modelin kullanıldığı deneysel ve aşamalı bir çalışmada araştırma sonuçları gözlenmiştir. Birinci aşamada MYM alanıyla ilgili literatür taranarak problem sahaları ve bunlara yönelik çözüm önerileri belirlenmiştir. İkinci aşamada bir yazılım firmasının ihtiyaçları dikkate alınarak Scrum'ın uygulandığı Öz Çerçeve tabanlı bir mobil yazılım geliştirme modeli önerilmiştir. Üçüncü aşama, söz konusu modelin içerdiği yöntem ve tekniklerin kullanıldığı ve siber güvenlikle ilgili eğitimlerin çevrimiçi verildiği, mobil bir uygulamanın geliştirildiği süreçleri içermiştir. Araştırma raporunun bundan sonraki bölümlerini, çalışmanın dayandığı kuramsal altyapı, kullanılan araştırma yöntemi, ÖÇ tabanlı yazılım süreci ve yazılım ürünün geliştirilmesi ile araştırmayla ilgili sonuç ve önerileri oluşturmaktadır.

Araştırma problemini belirlemek için aşağıda sorular sorulmuştur.

- Mobil yazılımın standart bir geliştirme yöntemi var mıdır?
- Var olan yöntemler her projede kullanılabiliyor mu?
- Standart bir platform bulunmakta mıdır?
- Mobil yazılımlar için gerekli olan ortak bir terim kütüphanesi var mı?

Bu sorular kapsamında aşağıdaki problemler orta çıkmaktadır.

- Mobil yazılımın standart bir geliştirme yöntemi bulunmamaktadır.
- Kişi veya takım özelinde mobil yazılım geliştirme olmaktadır.
- Mobil projelerde farklı platformlar oluşmaktadır.
- Her mobil projede farklı terimler kullanılmaktadır.

Bu araştırmanın amacı mobil yazılım süreç modeli geliştirmektir. Öz çerçeve kullanılarak ortak bir geliştirme yönetimi oluşturmak amaçlanmaktadır. Ayrıca dünyada mobil yazılım geliştirme için bir standart oluşturmak hedeflenmiştir. Mobil yazılım geliştiren ekipler sürekli olarak farklı platformlar kullanmakta ve bu çalışma ile standart platform oluşturulması amaçlanmıştır. MYM alanındaki çalışmaların ortak bir çerçevede

modellenmesine, tasarımına ve değerlendirilmesine olanak tanıyacak bir modelin geliştirilmesi ve söz konusu modelin, deneysel ve özellikle çevik yazılım geliştirmeyi içeren araştırmalara uygulanması sonucunda, gelecekteki MYM alanındaki bilimsel araştırmaların çok boyutlu çözümlemesini olanak tanıyacak bir girişimin başlatılması amaçlanmıştır.

Öz Çerçeve ile yazılım geliştirmede kullanılan yöntemler ve uygulamalar, kavramsal yapıda bir çekirdek içerisinde formal olarak ele alınmakta, kendine özgü alan bağımlı görsel ve metin tipi bir dil ve yapılarla ortak bir temelde bütünleştirilmektedir.

MYM çalışma alanındaki araştırmalar ve endüstri uygulamaları arasındaki uyumsuzluğun giderilmesi mümkündür.

Çevik yaklaşım doğrultusunda yazılım geliştirme yöntemlerinin ve uygulamalarının dinamik yapıda oluşturulabilmesi sağlanmaktadır. Yazılım süreçlerinin sağlıklı bir şekilde izlenmesi ile MYM yöntemleri ve uygulamalarının ortak bir temelde bütünleştirilmesi hedeflenmektedir.

1.1. Sınırlılıklar

Bu araştırmanın sonuçları mobil yazılım geliştiren firma ile MYM alanıyla sınırlıdır. Zaman ve kapsam kısıtlılıkları dikkate alınarak yazılım geliştirme süreci üç Sprint ile sınırlı tutulmuştur.

BÖLÜM 2

2. KURAMSAL ALTYAPI

2.1. Proje Nedir

Proje tanımı, insanların kafasında oluşturdukları bir düşünceyi, düşünceden sonra uygulama aşamasına geçirmeyi istemesiyle başlayıp, istek doğrultusunda hedefe ulaşılmasını sağlayan süreç olarak belirtilmiştir. Projenin hedeflerinin yalın olması, başlangıç bitiş zamanı olması projeyi operasyondan ayıran unsurdur. Operasyon kaynakları sınırlıdır, değişiklik yok denilecek kadar az ve tekrarlamaya müsaittir. Projede de kaynak sınırlaması vardır ancak tamamen yenidir, tekrarlamaz, gelişim gösterir (PMI, 2017).

Proje yönetimi ise, projenin hedeflerini ortaya çıkarmak için maliyet, hacim, zamanlama ve boyutunun kontrollü bir şekilde yönetilmesidir (PMI, 2017). Proje yönetiminin başlangıç, planlama, yürütme, izleme ve kontrol, kapanış süreçleri bulunur.

Proje yönetimi "entegrasyon, kapsam, maliyet, zaman, kalite, insan kaynakları, iletişim, risk ve tedarik yönetimi" unsurlarının bütünleşmesiyle oluşur (Community, 1994).

2.1.1. Entegrasyon Yönetimi

Proje entegrasyon yönetimi proje sürecindeki yönetimsel aktivitelerin tanımlanması, belirlenmesi ve birleştirilmesi gibi süreçleri kapsamaktadır. Projenin tamamını etkileyecek konuların tamamı entegrasyon yönetiminde ele alınır. Proje yönetimi ise süreç grupları içindeki adımları koordine etmek için gerekli unsurları içerir (PMI, 2017).

Entegrasyon Yönetiminin Süreçleri:

Başlangıç: Proje yöneticisinin görev ve yetkilerinin belirlenmesinden sonra, ilgili taraflarca projenin üst yönetim desteğinin sağlanmasıdır. Proje duyurusu olarak tanımlanabilir. Projenin başlaması resmi olarak gerçekleşir. Elde edilmesi planlanan ürünler veya hizmetler genel hatlarıyla belirlenmiştir. Tarafların projeden beklentileri ve hedefleri konusunda bilgi verir. Projenin maliyet planlaması için ön bilgi içerir (PMI, 2017).

Planlama: Projenin en önemli süreçlerinden biridir. Tarafların tamamının katıldığı iş planlamasının ve maliyet planlamasının daha gerçekçi hesaplandığı, projenin duyurulup yöneticisinin atandığı aşamadır. Planlama sürecinin devamlılığı proje yöneticisi ve firmanın uygunluğuyla sağlanır (PMI, 2017).

Yürütme: Proje yönetim planlamasının iş paketleri ve kaynaklar doğrultusunda belirlenmiş olan kapsam içerisinde yönetilmesi aşamasıdır. Kaynakların entegrasyonu bu süreçte önem kazanır.

İzleme ve Kontrol: Projenin yönetimi aşamasında planlandığı gibi hayata geçirilmesi, kalite, maliyet, süre ve amaçlarının değerlendirilmesidir. Projenin kontrolü önceden belirlenen kriterler doğrultusunda yapılır. İzleme aşamasında ise belirlenen hedeflerden elde edilen gerçeklikler araştırılır (Enver & Kovancı, 2004).

Kapanış: Projede tüm görevlerin tamamlanıp, kabulün geçici olarak yapılmasıyla projenin kapanış aşaması başlar. Hedeflerin karşılanma oranları ve kazanımların belirleneceği bir toplantı planlanır. Gerçekleştirilenlerin ihtiyacı karşılama oranına göre veri tabanı verileri güncellenir. Proje kapanış koşulları sağladığında güvence altına alınır ve sonlandırılır (Erdem & Younis, 2014).

2.1.2. Kapsam Yönetimi

Projenin başarılı olarak tamamlanması amacıyla, tüm çalışmaları projenin içermesini sağlamak için gerekli olan süreçtir. Proje kapsam yönetimi için öncelikle projeye nelerin dâhil edileceği ya da nelerin dâhil edilemeyeceği kontrol edilmelidir. Proje kapsamında iki önemli unsur vardır (Boz, 2015);

Ürün Kapsamı: Proje ihtiyaçları doğrultusunda belirlenen ürünü tanımlayan hizmetler ve özelliklerdir.

Proje Kapsamı: Hedefteki ürünün belirtilen özelliklere sahip şekilde teslim edilmesinin proje planı kapsamında belirlemesidir.

Proje kapsamının aşamaları vardır;

-Gereksinimlerin Toplanması: Paydaşların ihtiyaç tanımlaması ve belge düzenlenmesinin proje amacına yönelik yapılması sürecidir (Çift, 2015).

-Gereksinim Dokümantasyonu: İhtiyaçların her bir proje iş ihtiyacıyla bağlantısı açıklanır. Gereksinimler ilk aşamada yalın bir halde hazırlanır sonraki aşamalarda bilgi birikimiyle anlatılır (PMI, 2017).

-Gereksinim Yönetim Planı: Proje boyunca gereksinim analizinin nasıl yapılacağı ve belgeleneceği sürecidir. Fazlar arası ilişkilerin yönetilmişine katkı sağlar (PMI, 2017).

-Gereksinim İzlenebilirlik Matrisi: Projenin tüm aşamalarında izlenebilmesini sağlayan, gereksinimlerin ilk aşamayla bağlantısını gösteren araçtır. Gereksinimlerin iş ve proje ile bağlantılı hale gelmesini sağlar.

-Kapsamın Tanımlanması: Ürünün ayrıntılı şekilde açıklanması aşamasıdır.

-Proje Kapsam Bildirimi: Proje planlama sürecinde projenin teslimatları ve bu teslimatları yaratmak için gerekli çalışmalar konusunda daha fazla bilgi sahibi olunması için projenin kapsamı daha ayrıntılı bir şekilde tanımlanabilir. Bu süreç projenin yaşam döngüsü boyunca kendini sık sık tekrarlar (Kırmızı, 2017).

İş Kırılım Yapısının (İKY) Oluşturulması: Proje ürünlerinin ve proje çalışmalarının küçük ve verimli olarak yönetilebilir bileşenlere ayrılmasıdır.

İKY, proje ekibinin projenin hedeflerine ulaşmak için yürüttüğü çalışmaların hedefe yönelik belirlenip, ayrıştırılması aşamasıdır. Her seviye bir öncekine ayrıntılandırılarak devam eder. Çalışma paketleri için hesap kodları, çalışma paketlerinin kontrol hesapları belirteç ile sonuçlanır. Maliyet zaman çizelgesi ve kaynakların hiyerarşik olarak toplanmasına olanak sağlar (PMI, 2017).

Kapsamın Doğrulanması: Tamamlanan proje teslimatlarının kabulünü resmileştirme sürecidir.

Kapsamın Kontrolü: Üründe kapsam değişikliklerinin yönetilmesi sürecidir.

Tüm süreçler birbiriyle iletişim halindedir. Her süreç ihtiyatça bağlı olarak belirli sayıda kişinin çalışmasını gerektirir. Süreçler ayrı ayrı bileşen şeklinde detaylı tanımlanmış olmalıdır.

2.1.3. Zaman Yönetimi

Zaman yönetimi, projenin planlanan süre içinde tamamlanmasını sağlamak amacıyla gerçekleştirilen işlemlerin sürecidir (Kerzner, 2003).

Etkinlik Tanımlama: Planlanan projenin üretilmesi için iş dağılımının yapılmasıdır.

Etkinlik İlişkilendirme: Etkinlik ilişkilerinin yazılı halde sunulmasıdır.

Etkinlik Süre Tahminleri: Etkinliğin tamamlanmasına kadar gerekli tahmini süredir.

Program Geliştirme: Projede gerekli ilişki, süre ve kaynak ihtiyaçlarının planlanmasıdır.

Program Kontrolü: Proje programından aksaklıkların izlenip gerekli kontrollerin sağlandığı aşamadır.

Projedeki planlama takviminin hazırlanması ve yönetimi zaman yönetimi için gereklidir. Proje Yönetimi Bilgi Grubu rehberinde proje zaman yönetimini 7 aşamalı belirlemiştir (PMI, 2017). Proje zaman yönetimi süreçleri şunlardır:

Plan Zamanlama Yönetimi Süreci;

Proje yönetimi dünyasında, en değerli kaynağın zaman olduğu söylenebilir. Bir proje zamanlamanın gerisinde kaldığında, zaman farkını kapatmak ve işleri yeniden yoluna sokmak son derece zordur. Geciken bir aktivite veya olay, diğer işler üzerinde domino etkisi gösterir. Bu, projelerde zaman yönetiminin birinci öncelik olması gerektiği anlamına gelir (PMI, 2017).

Faaliyetleri Tanımla Süreci;

Bu süreç, projenin çıktılarını üretmek için yapılması gerekenleri tanımlar ve belgeler. Yani proje görevlerini tanımlar. Bununla çalışmanın getirisi proje görevlerinin tanımlanmış bir listesiyle sonuçlanacağıdır. Bir sonraki işleme ana girdi olduğu için bu süreç faydalıdır.

Sıralı Etkinlikler İşlemi;

Görev listesi kullanılarak, doğru şekilde sıraya koymak gerekir. Bu işlemin sonunda proje görevlerinin etkileşimleri görülür. Proje kaynaklarının verimli şekilde kullanılması ve proje çalışmalarının doğru ilerlemesine yardımcı olur.

Tahmini Faaliyet Kaynakları Süreci;

Hedefteki işin ne olduğunu anladıktan sonraki adım bunun için gerekli kaynakları yönetmektir. Tahmini faaliyet kaynakları süreci ihtiyaç duyulan insan kaynağı, ekipman ve malzemelerin miktarını da belirtir.

Tahmini Faaliyet Süreleri Süreci;

Bu süreçte belirlenen kaynakları kullanarak, her etkinliği yapmanın ne kadar süreceği hesaplanır.

Planlama Süreci Geliştirin;

Bütün süreçlerden toplanan bilgilerle planlama yapılır.

Kontrol Takvimi Süreci;

Proje zamanlamasını izlemek ve güncellemek, değişikliklerin uygun bir şekilde yönetildiğini ve projenin zamanlamalarını kontrol altında tutmak için gerekli araçları sağlar.

2.1.4. Maliyet Yönetimi

Maliyet yönetimi, belirlenen bütçe ağı içinde projenin tamamlanması kontrolünün yapıldığı süreçtir.

Maliyetlerin Tahmin Edilmesi: Proje süreçlerinin tamamlanması için gerekli bütçe planlamasının tahmini yapılmasıdır. Maliyetlerin tahmini her faaliyetin maliyetini kesinleştirmemizi sağlar. Zaman, malzeme ve diğer tüm maliyetler için çıkarılan maliyet tahminleri projenin tamamlanması için gerekli olası maliyetlerin sayısal değerlendirmeleridir. Maliyet tahminleri yalın veya detaylandırılmış olabilir. Faydalanılan tüm kaynaklar için maliyet tahmini yapılabilir. Bütçenin belirlenmesi aşamasında tüm tahminler toplanır ve temel çizgi oluşturulur. Gelecekte planlanan tüm harcamalar temel çizgi ile kıyaslanır. Projenin toplam maliyetini ölçmek, izlemek ve kontrol etmek için onaylanmış zaman fazlı bütçe, maliyet performans temel çizgisini oluşturur (PMI, 2017).

Proje mevcut durumun gözlemlenmesi ve maliyet çizgisindeki yönetim bütçenin güncellenmesi için gereklidir. Gerçekleştirilen maliyetler ayrılmış bütçe ile kıyaslanır ve iyileşmeyi gidilip gidilmeyeceği kararlaştırılır.

Proje boyunca süreçler birbirleriyle ve diğer bilgi alanlarındaki süreçlerle etkileşim halindedir. Projenin ihtiyaçları doğrultusunda süreçlerde bir ya da birden fazla kişinin çalışması gerekebilir.

2.1.5. Kalite Yönetimi

Proje kalite yönetimi projenin hedefini, gerekliliklerini ve sorumluluklarını içerir ve bunlarla ilişkili olarak organizasyon aşamalarını, aktivitelerini ve teslimatını kapsar. Proje kalite yönetimi projenin ana maddelerine bağımlı şekilde organizasyonun kalite yönetim sistemini uygular. Proje kalite yönetimi, kalite yönetiminin planlanması, kalite güvencesinin sağlanması ve kalite kontrolü gibi alt süreçler içerir (Community, 1994, s. 230).

Kalite Yönetim Planlanması, projenin hangi standartlara göre gerçekleştirileceği, hedeflenen standartlara uygun olması için ne tür işlemlerin yapılmasının gerektiği, yapılacak işlemler için gerekli olan çıktıların belgelendirme şeklinin planlandığı bir süreçtir. İyi planlanan ve sonucunda beklenen kalitede gerçekleştirilen projeler yüksek üretkenlik, düşük maliyet, artan paydaş memnuniyeti ve karlılığı da beraberinde getirir. Planlama yapılırken sebep-sonuç şemaları, akış şemaları, denetim çizelgeleri, pareto şemaları, histogramlar, kontrol grafikleri, dağılım şemaları gibi kriterler kullanılabilir (PMI, 2017).

Kalite Güvencesinin Sağlanması, gerekli standartlar çerçevesinde projenin tamamlanması için kalite kontrol ölçümlerinin ve kalite gereksinimlerinin denetlenmesi aşamasını kapsar. Kalite süreçlerinin geliştirilmesine katkıda bulunmak, devam eden projenin kusurlarını saptayıp önlem alınması açısından "Kalite Güvencesinin Sağlanması" önemli bir yere sahiptir. Kalite güvencesi doğrultusunda yapılan saptanımlar kanıtlanabilir olmalıdır (PMI, 2017).

Kalitenin Kontrolü süreci, planlar doğrultusunda yapılan uygulamaların sonuçlarına göre öngörülen değişiklikleri belirtmesi, performans değerlendirmelerin yapılması ve sonuçların kaydedilmesi süreçlerini kapsar. Önerilen değişiklikler kapsamında projenin tamamlanması açısından önem arz eden bir aşamadır (PMI, 2017).

Proje kalite yönetimi genellikle “Uluslararası Standardizasyon Teşkilatı “gerekliliklerini kapsar. Bu açıdan proje sonucunda ortaya çıkacak ürün verilerinin, modern kalite yönetim yaklaşımlarına dayandırılması ve ISO’ya uygun olması gerekmektedir.

2.1.6. İnsan Kaynakları Yönetimi

İnsan kaynakları kavramı kurumsal amaçlara erişebilmek için gerekli olan temel kaynaklardan biridir. Bu kavram, ekibin bünyesinde bulunan en üst yöneticiden en alt düzeydeki çalışana kadar bütün personeli kapsadığı gibi, örgütün dışında bulunan ve potansiyel olarak yararlanılabilecek işgücünü de içermektedir (Kaynak, 1998, s. 15).İşletmelerin kaynakları arasında en değerli olan insan kaynağıdır (Özgen, Öztürk, & Yalçın, 2002, s. 5).

İnsan kaynakları yönetimi iki temel düşünce üzerine kurulur (Palmer & Winners, 1993):

1.İnsan gücünün hedefler doğrultusunda kullanılması.

2. İş gören ihtiyaçların karşılanması ve tekâmül sağlanması.

Projelerde insan kaynakları yönetimi, ekibin organize edilmesi, yönetilmesi ve yönlendirilmesi süreçlerini kapsar. Proje ekibi, projenin sonuçlanması için rol ve sorumluluklarını almış kişilerden oluşur. Proje ilerledikçe proje ekibi üyelerinin türü ve sayısı değişkenlik gösterebilir. Proje ekip üyelerinin rol ve sorumlulukları tanımlanırken, tüm ekiptekilerin planlama ve karar alma aşamalarına katılmaları yararlıdır. Ekipteki bireylerin projeye başlangıç aşamasında dâhil olmaları planlama sürecinde daha fazla profesyonel bilgiye sahip olmalarını ve projeye daha fazla sorumlulukla yaklaşmalarını sağlayabilir (PMI, 2017).

İnsan Kaynakları Planlamasının Geliştirilmesi: Projeye ait görev, sorumluluk, gerekli beceriler ve belgelendirme etkileşiminin belirlenmesi, personel yönetim planının geliştirilmesi sürecidir (PMI, 2017).

İnsan Kaynakları Planı: Proje yönetiminin adımı olan insan kaynakları planı, projede insan kaynaklarının nasıl tanımlanacağı, görevlendirileceği, yönetileceği, kontrol edileceği ve sonuçta görevlerin nasıl sonlanacağı konularında kaynak göstericidir.

Proje Ekibinin Oluşturulması: Proje sorumluluklarını yerine getirecek insan kaynaklarının onaylanıp, ekibin kurulması sürecidir.

Kaynak Takvimleri: Proje için ekipteki herkesin çalışacağı zaman parçacıkları belirlenir. Ekip üyelerinin çizelge ile ilgili çatışmalarını kontrol edebilmek, zaman dilimlerini verimli kullanmalarını sağlamak açısından güvenli çizelge oluşturmak önemlidir.

Proje Ekibinin Geliştirilmesi: Proje başarımını artırmak için gerekli takım ve ortamların sağlanması sürecidir.

Proje Ekibinin Yönetilmesi: Proje performansını tespiti için ekip üyelerinin başarımının izlenmesi, buna yönelik dönüt sağlanması, sorunların çözülmesi ve farklılıkların yönetilmesi sürecidir (PMI, 2017).

2.1.7. İletişim Yönetimi

Projelerdeki iletişimin nasıl planlanacağını, yapılandırılacağını, izleneceğini ve kontrol edileceğini belirten proje yönetim planının bir nesnesidir. İletişim yönetim planının bileşenleri;

- Gereksinimler: hangi bilgiye kim neden erişmek istiyor belirlenir.
- İletilecek bilgiler: dil, içerik ve format olarak açıklanır.
- Zamanlama: bilginin paylaşım sıklığı ve şekli açıklanır.
- Gönderen ve alan: bilginin kimler arasında paylaşılacağı açıklanır.
- Gizlilik: bilgilerin erişim kısıtları açıklanır.
- Araçlar: bilgilerin iletim yöntemleri belirtilir.
- Kaynaklar: zaman ve bütçe alınarak gerekli kaynaklar edinilir.
- Eskalasyon: sorunların çözümlerinde yöneticiye ulaşma aşaması
- Güncelleme: yönetim planının güncelleme aşamasıdır.
- Terimler: kullanılan terimler anlaşılır olması için açıklanmalıdır.
- Akış: bilgi ile sorumluluklar akış olarak belirtilmelidir.
- Yasal: iletişimde yer alan kısıtlamalar alanıdır.

2.1.8. Risk Yönetimi

Risk yönetimi, kişi ve kurumların, mali durumların ne kadar risk barındırdığının belirlenmesi ve bu kıstasın uygun olan bir risk seviyesine çekilmesidir. Risk yönetiminin amacı, işletmenin istikrarlı şekilde çalışmasına devamı için gereken düzenlemeleri sunmak ve etkinlikteki mal ve kişilerin güvenliği ile işletmenin kâr payını korumaktır.

Böylece risk yönetimi, proje sürecinde oluşabilecek kayıpların minimum maliyetle kontrol edilmesi için gerekli doküman ve çalışmaların planlanması, yönetilmesi ve kontrol edilmesi olarak belirtilebilir (Çağırgan, 1997).

Projelerle ilgili olarak belirlenen riskler şunlar olabilir;

- Zamanda aksaklık olması
- Ürün gereksinimlerinin elde edilmemesi
- Bütçenin aşılması
- Hata oranı fazla olan ürün teslim edilmesi
- Ürün performansının istenenden düşük olması
- Tedariklerde yaşanan sorunlar
- Eleman temini/yetkinliği

Risk yönetim ilerlemesi birbirine bağımlı beş adımdan oluşur. Bu adımlar; riskin tanımı, riski değerlendirme ve hesaplama, risk düzeltme araçlarındaki seçeneklerden seçim yapma, seçilen değeri uygulama ve değerlendirip kontrolünü sağlama süreçlerinden oluşmaktadır (Daft, 1991).

*Riskin Tanımlanması: Riskin tanımlanmasıdır. Risk tanımlama aşamasında elde edilen veriler problemlerin çözümünü kapsar. Konuya yönelik yapılan araştırmalar doğrultusunda teknolojik, sosyal, politik belirsizliklerin azaltılması ve ihtimal kayıpların minimuma indirilmesi bu süreçtedir (Hertz & Thomas, 1983). Riskin tanımı ile belirsizliğin azaltılması ve sorunun çözümüne yol gösterecek bilginin görevi netlik kazanabilir. Kıstas olan parametreler ve onların belirsiz etkisini belirtmek için ileriye dönük planlamalar yapılabilir. Örneğin, ilerleyen zaman için işletme fırsatlarını tanımak amacıyla geleceğe yönelik en iyi, en kötü ve en olası senaryolar üretilir. Senaryoya karşıda çeşitli olaylar tanımı yapılır. Bu yapılanlar belirsizliği azaltma konusunda yol gösterici olabilir (Arman, 1997).

*Riskin Değerlendirilmesi ve Hesaplanması: Risk yöneticileri riskin tanımlandığı aşamada değerlendirmelidir. Olası kaybın ve gerçekleşmesinin önceliklerinin belirlenmesini gerektirir. Sıralama şöyle belirtilir (Vaughan & Vaughan, 1995, s. 32).

Çok önemli riskler: İflas ile sonuçlanabilecek riskler.

Önemli riskler: İflasla sonuçlanmayacak lakin maliyeti sarsacak riskler.

Önemsiz riskler: Performansı etkileyen riskler.

Kaynaklarda Heinrich yasası şeklinde bahsedilen ve örgütsel risk yönetiminde kullanılabilecek kaza nedenleri sıralamasında, her büyük kazaya karşılık 29 küçük kaza ayrıca 300 civarında başarısızlık vardır (Michael & Greenfield, 1959). Bahsedilen yasadan yola çıkarak: Büyük olmayan kazalar ya da başarısızlıklar nitekim büyük bir kaza veya riskli durumun habercisi olabilir.

*Risk Düzeltme Araçları Arasında Alternatifi Seçmek: Risk belirlenip değerlendirildikten bir sonraki adım, riske karşı kullanılacak yaklaşım şeklinin ve tekniğin belirlenmesidir. Risk yönetim teknikleri, uzak durmak, göze almak, azaltmak ve kaçınmaktır (Vaughan & Vaughan, 1995, s. 32).

*Seçilen Alternatifin Uygulanması: Konu ya da problem anlaşılır şekilde ortaya konduktan sonra düşünülen alternatifler değerlendirilir. Çok fazla alternatif çıkacaktır. Bunlar çeşitli zamanlarda, derecelerde ve her derecede farklı ihtimaller bulunacak şekilde görülebilmektedir (Newhman, 1979, s. 133-134).

*Değerlendirme ve Kontrol: Seçilen alternatif uygulanırken belirlenen çözüm şeklinin beklenen sonucu verip vermediği etkinlik seviyesinde izlenir. Çözüm üreticisinin seçtiği çözüm şeklinde gerçekleşen ile beklenen arasındaki fark fazla ise çözümü düzeltmek ya da değiştirmek gerekecektir (PMI, 2017). Riskler yeni riskleri oluşturabilir ve çözümü zor ciddi sorunlar oluşabilir, bunun önüne geçmek için risk yönetimi önem taşır.

2.1.9. Tedarik Yönetimi

Proje Tedarik Yönetimi, proje için gerekli olan ürün, hizmet ya da sonuçları proje ekibi dışından temin etmek için gereken kademeleri, sözleşmeleri ve yetkili proje ekibi üyeleri tarafından belirlenen emirleri hazırlamak aynı zamanda yönetmek için gerekli sözleşme yönetimi ve değişiklik kontrol süreçlerini kapsar. Harici bir etkinliğin hazırladığı sözleşmenin ve proje ekibine yüklediği yükümlülüklerin idare edilmesini kapsar (PMI, 2017).

Proje Tedarik Yönetimi Süreçleri, proje ekibi ile ürünü satın alan arasında düzenlenmiş hukuki belge olan sözleşmelerle ilgilidir. Proje tedarik aşamaları;

Tedariklerin Planlanması: Bu düzeydeki kararlar, projedeki hangi ürünün, ne miktarda üretileceği ve hammaddesinin tedarik edileceği yerler gibi konuları kapsamaktadır (Yıldızöz, 2006).

Tedariklerin Yürütülmesi: Bir satıcı seçme ve sözleşme imzalama sürecidir. Tedariklerin yürütülmesi sürecinde, proje ekibi fiyat ya da teknik teklifleri alır ve önceden tanımlanmış seçim kriterlerini kullanarak süreci takip ederek uygun satıcıyı seçer. Sürecin tamamlanması sonucunda satıcı seçilir ve sözleşme tamamlanır (PMI, 2017).

Tedarik İşlerinin İdaresi: Tedariklerin iş yönetimi, satıcı performanslarının gözlemlenmesi ve gerekli ise değişikliklere gidilmesi sürecidir. Tedarik işlerinin idaresi ile birlikte ürün satıcısına yapılacak ödemelerin ödeme standartlarına uygun olması sağlanır. Bu kısımda yapılan işler ile gerçekleştirilen ödemeler karşılaştırılmaktadır. Aynı zamanda tedarikçinin sözleşmeye uygun çalışma durumu, gerekli olan durumda gidilecek çözümler belirlenir (Tosun, 2014) .

Tedarik İlişkilerinin Yönetilmesi: Tedarik zincirinin yönetimi aşamasında tedarikçi ile iletişimi geliştirmeye yöneliktir. Bu müşteri ilişkilerinin yönetilmesi amacıyla ortaya çıkmış ve müşteri ile olan ilişkiler de yönetim sayesinde oldukça başarılı sonuç ortaya çıkarmıştır. Esas olarak tedarikçiler boyutunda da uygulanması amaçlanmaktadır. Dolayısıyla her geçen gün önem kazanmaktadır (Karakış, 2007).

Tedarik Kapanışı: Birbiriyle veya diğer süreçlerle ilişki halindeki tüm süreçlerin kapanması aşamasıdır.

2.2. Yazılım Projeleri

Yazılım geliştirme yaşam döngüsü, yazılımın hem üretim hem de kullanım süresi boyunca devam eden evrelerin tümünü kapsar. Yazılım aşamalarının planlanarak sürdürülmesini sağlar. Yazılımın gereksinimleri sürekli değişir ve genişler, söz konusu aşamalar süre gelen döngü biçiminde ele alınır. Döngü içindeki bir aşamada ileri gitmek veya geri dönmek söz konusudur (Demirci, 2015).

2.2.1. Yazılım Geliştirme Süreçleri

2.2.1.1. Gereksinim ve Analiz

Yazılım projesinin ilk aşamasında proje için gerekli ana modüller saptanmalı, proje amaç ve hedefleri ayrıntılı olarak belirlenmelidir. Proje varsayımları göz önüne alınarak kullanıcı açısından faydaları değerlendirilmelidir. Projede zaman kaybı oluşturacak çok önemli olmayan veya etki olmayan nitelikler bir sonraki faza aktarılabilmelidir. Böylelikle proje süreci çok uzamadan tamamlanmış olacak ve özelliklerin gerekliliği tekrar değerlendirilebilecektir. Kullanılacak en uygun yazılım dili, yazılım mimarisi, sunucu ihtiyaçları belirlenmelidir.

2.2.1.2. Tasarım

Sistem tasarım kısmı gereksinimlerin tamamlanması ile başlar. Tasarım, müşterinin gereksinim ve isteklerini karşılamak için yazılım ürününün özellikleri, yetenekleri ve ara yüzlerinin saptanması aşamasıdır. Bu aşamada UML diyagramları, Veri tabanı Varlık İlişkileri diyagramları (ER Diyagram), kullanıcı ara yüzleri ve sistem mimarisi tasarlanır (Erol, 2015).

2.2.1.3. Kodlama

Tasarım aşamasının belirli bir seviyeye gelmesiyle birlikte kodlamaya geçilir. Müşterinin teslimini istediği ürünün programlama kısmıdır. Yazılım uzmanı, kod geliştirme standartlarına bağlı kalarak geliştirme sürecini kaliteli kodlama için sürdürür. İyi kodu tanımlayacak olursak, okunabilirliği ve bakımı kolay olan yalın koddur. Basitlik ve sadeliği prensip alan yazılıma göre yeni mezun olan birisi kodu 1-2 günlük bir süreçte anlayabiliyor, yorumlayabiliyor ve yeni durumlara göre şekillendirebiliyorsa kodunuz iyi bir koddur.

2.2.1.4. Test

Sistemin artık gerçek hayata geçtiği adımdır. Şirketin kullanacağı yazılım söz konusu olduğunda ilgili birimlerde yazılımla alakalı eğitimlerin alınması, donanımsal eksiklerin tamamlanması, bağlantılı birimlerle ve yazılımlarla uyum sağlanması bu aşamadadır.

Yazılım artık aktif bir projedir. Kullanımı aşmasında bazı problemlerle karşılaşılır. Problem çözümleri için önerilen fikirler buna bağlı da ihtiyaçlar ortaya çıkar. İhtiyaçların değerlendirilmesi sonrasında bakım kısmına geçilir.

2.2.1.5. Bakım

Test aşamasının tüm adımları geçildikten sonra yazılım ürünü müşteriye teslim edilir hale getirilir. Müşteriye teslim edilmeden önceki versiyonla farkını da belirten kullanım kılavuzu hazırlanır. Teslim aşaması ile bakımından başlar. Yazılımın iyileştirilmesi, hataların giderilmesi ve yeni fonksiyonların eklenme kısmıdır.

2.2.2. Yazılım Geliştirme Süreçleri Türleri

2.2.2.1. Gelişigüzel Geliştirme

Gelişigüzel geliştirmede belirli bir model ya da yöntem bulunmaz. Yazılım geliştirme süreçlerinin herhangi birisi yer almaz. Yazılım tamamıyla geliştiriciye bağlıdır. İlerleyen zamanlarda yazılımı hazırlayan kişi bile anlamakta zorlanabilir. Bu yüzden kontrolü ve bakımı çok zordur. Çoğunlukla 1960'lı yıllarda kullanmıştır. Yazılımı tek başına yapanlar tarafından tercih edilmiştir. Yazılım diğer süreçleri barındırmadığından geliştirilmesi kolaydır (Duran, 2009).

2.2.2.2. Barok Modeli

Yazılım yaşam döngülerinin esas adımları bu döngüde kontrol edilir. Çoğunlukla 1970'li yıllarda tercih edilmiştir. Adımlar arasındaki ilişkiler belirtilmemiştir." Belgeleme" adımı bu modelde ayrı bir aşama gibi ele alınarak, yazılım geliştirilmesi ve test edilmesinin ardından yapıldığı için günümüzde çok tercih edilmez. Diğer modellere aykırı bir durumdur (Duran, 2009).

2.2.2.3. Şelale Modeli

Şelale yönteminde yazılım geliştirme süreci analizin yapılması, tasarım tamamlanması, kodlama, test edilmesi, sürüm ve bakım gibi adımları barındırır. Alışılmış yazılım

metotlarında bu aşamalar lineer olarak devam eder. Her aşama, başlangıç noktasında bir önceki adımda ürettiklerini bulur. Barındırdığı değişiklikleri bir sonraki adımda kullanılacak şekilde değiştirir (Şeker, 2008).

2.2.2.4. Helezonik Model

Barry Boehm tarafından 1986 yılında açıklanmıştır. Bu modelde risk analizi ön plandadır. Yinelenen artımsal bir yaklaşım vardır. Doğrudan adım tanımlama gibi bir faz bulunmaz. Model yaklaşımı vardır, süreç 4 gruba ayrılır. Planlama, projenin amacının belirlenmesi için gereklidir. Risk analizi, risklerin belirlenmesi ve doğuracağı hataları belirlenmesini sağlar. Üretim, ara ürünün elde edilmesini sağlar. Kullanıcı değerlendirmesi ise kullanıcının ürün hakkındaki yorumlarını belirtir. Üretim süresince kullanıcı tarafından değerlendirilmesi temeline dayandığı için avantajlı bir modeldir. Aynı zamanda kullanıcı süreç boyunca proje geliştiricileriyle etkileşim halinde olduğu için kodlanması ve sınanması daha erken başlar (Ak, 2012).

2.2.2.5. Artırımsal Model

Artırımsal model, yazılımın küçük parçacıklar halinde döngü içerisinde geliştirilmesini esas alır. Proje küçük zaman dilimlerine ayrılır. Her zaman diliminde bir döngü tamamlanır. Döngüler sonucunda planlanmış çıktılar yeni fonksiyonlar olarak eklenir. Bu sayede artırımsal olarak geliştirme yapılır. Bir döngüde henüz çıktı alınmamış olsa da diğer döngünün tasarımına başlanabilir. Her döngüde yeni bilgiler elde edilir ve bunlar projenin gelişimi için katkı sağlar. İlk döngüde elde edilen bilgi projenin ilerleyen döngüsünde değişikliğe uğrasa bile maliyetsiz şekilde güncellenebilir. Artırımsal model döngülerle geliştirilmeye yönelik projelerde kullanım için uygundur. Yazılım fonksiyon parçalarına ayrılabiliyor olmalı, parçalar kendi içinde ele alınmalıdır (Lekesiz, Yazılım Projelerinde Proje Yönetim Süreçlerinin İncelenmesi, 2013).

2.2.2.6. Döngüsel Model

Döngüsel modelin artırımsal model ile çok fazla benzer noktası vardır. Döngüsel yazılım geliştirme modeli, planlama aşamasından başlayarak proje yaşam döngüsünün tüm süreçlerini barındırır. En büyük avantajı artırımsal modelde de avantaj öngörülen, her

adımda gerek duyulan değişikliklerin yapılmasının az maliyetli olmasıdır. Her iki modelin avantajlarının birleştirilmesi sonucu oluşan "Artırımsal Döngüsel Model" her aşamada ilk adıma geri bildirim yapar.

Döngüsel model ile artırımsal modelin birleşimi sonucu oluşan "Artırımsal Döngüsel Model" günümüzdeki en etkin modeldir. Adımlarda geliştirilen fonksiyonları sonraki aşamaya aktarabilmek ve son aşamadan ilk aşamaya geri bildirim yapabilmek bu modelde mümkündür (Somyürek, 2018).

2.2.2.7. Çevik Yazılım Geliştirme Modelleri

Çevik, yazılım süreçlerinde esneklik ve güç kazanımı için kullanılan, zamanlamayı kısaltan kavramsal bir yöntemdir (Cockburn, 2001).Proje boyutu önemli olmaksızın küçük parçalara ayrılır, her parça kendi içinde bir proje gibi değerlendirilerek geliştirilir. Her proje bitimi sonunda müşteriye biten proje oranı bilgisi verilir." Çevik" ile projelerin hedeflenen planlaması iki ila dört hafta arasındadır. Müşteri memnuniyeti açısından projelerin kendi içinde çalışır durumda olması önemlidir. Ekibin proje esnasında iletişim halinde olması çevik yazılımın hızının artmasını sağlar. Parçacık halinde ilerleyen projede geriye dönük hata çözümü daha hızlı ve pratiktir. Çevik metotların avantajları verimliliğin yüksek, esnek, hata yüzdesi az, hızlı ve maliyetsiz çözüm sunuyor olmasıdır (Çamoğlu, Akbayır, Yücalar, & Bayraklı, 2010).

Çevik modellemenin temel özelliği modelleme yöntemlerinin neler olduğunu ve bunların ayrıntılarını söylemek yerine veri modelleme ve ara yüz modellemenin nasıl uygulanması gerektiğini söylemesidir (Pekel, 2008).

2.2.2.7.1. Çevik Yazılım Geliştirme Manifestosu

Yazılım endüstrisinin, her gün değişen bilişim dünyasına uyum sağlaması için yazılım geliştirme yöntemi olan çevik yazılım yöntemi, 2001 yılında 17 yazılımcı tarafından bir bildiri halinde ilkesel olarak bildirilmiştir. Bu bildiride süreçler ve araçların yanı sıra bireyler ve etkileşimlere, yazılımın çalışmasının kapsamdan önde olduğu, müşteri iş birliğinin sözleşmelerden daha öncelikli olduğu, sabitlikten çok değişime açık olmaya değer verildiği belirtilmiştir (Beck, Cockburn, Jeffries, & Highsmith, 2001).

Çevik Yazılımın Prensipleri:

- Yazılım teslimatı erken ve periyodik olarak yapılıp müşteriyi memnun etmelidir.
- Yazılım son aşamasında bile olsa değişime açık olmadır.
- Çalışır haldeki yazılım müşteriye teslim edilerek sürekli test edilir olmalıdır.
- Müşteri ve proje ekibi birlikte çalışmalıdır.
- İhtiyaç olan koşullar sağlanıp, müşteri motivasyonu artırılmalıdır.
- Bilgi paylaşımı yüz yüze olmalıdır.
- Yalınlık ve iş dışına çıkılmaması Agile parçasıdır.

Çevik Metodolojilerin Getirileri:

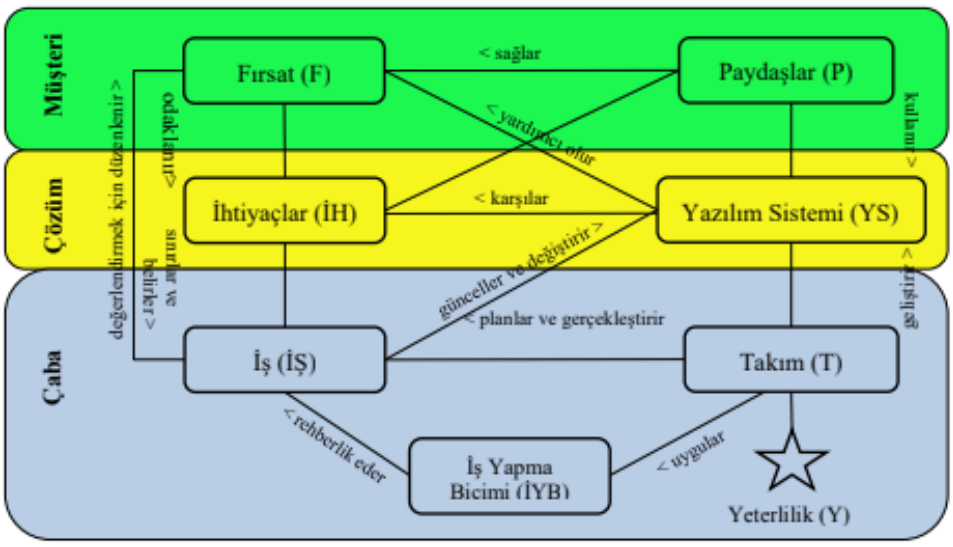
- Düşük Risk: Tekrarlanan yazılım geliştirme yöntemleri, proje risklerini azaltarak başarı oranını arttırır ve hata oranlarını minimize ederek verimliliği yükseltir. Projenin en başında oluşturulan parçacıklar önceden test edilerek eksiklikler gözlenir ve proje hızlı olarak şekillenir.
- Değişimin Teşvik Edilmesi: Yazılım projelerinde değişim vazgeçilmezdir. Orta çaplı projeler başlangıca göre %30 değişime uğrar. Çevik metodoloji değişimi engellemek yerine müşterinin istekleri doğrultusunda değişimi avantaja dönüştürür. Değişimi parçacıklar sayesinde projeye entegre eder (Highsmith, 2002).
- Karmaşıklığın Yönetimi: Projelerde büyüme ilerledikçe karmaşıklık artar. Çevik yöntem metodolojisi parçacıklar sayesinde karmaşıklığın önüne geçer.
- Sürekli Yazılım Teslimi: Çevik yazılımda sürekli olarak müşteriye kod parçacıkları teslim edilir. Çalışmayan bitmiş proje yerine, çalışan kod parçacıkları müşteri memnuniyetini artırır.
- Yüksek Kalite: Projenin başlangıç aşamasından itibaren tüm süreçler test edilerek devam edilir. Böylelikle hatalar proje çok büyümeden fark edilebilir (Schwaber, 2003).
- Müşteri ihtiyaçlarına daha iyi cevap veren çözümler: Çevik yöntemin değişikliğe açık olması ve müşteri isteklerine yönelik yaklaşımı sayesinde projeler müşteri ile birlikte değiştirilebilir. Bu sayede müşteri isteklerine en iyi derece çözüm üretilir. Projenin başlangıç aşamasında müşterilerin istekleri hakkında net bir şekillenme yoktur, proje ilerledikçe istekler biçimlenmeye başlar. Çevik yöntem parçacıklar sayesinde istekleri gerçekleştirebilir. Müşteri memnuniyeti ön plandadır (Schwaber, 2003).

Çevik yazılım yöntemleri kendi içerisinde temelde aynı ancak süreçlerde değişiklikler olan alt dallara ayrılmaktadır. Bunlardan popüler bir tanesi de Scrum yöntemidir. Bu konuyla ilgili olarak ilerleyen bölümlerde detaylı bilgilendirme yapılacaktır.

2.3. Öz Çerçeve (Essence Framework)

Diğer bölümlerde anlatıldığı gibi yazılım projelerini desteklemek amacıyla birçok süreç ve yöntem vardır. Araştırmalara göre yeni ve özgün yazılım ürününün yaygın olarak kullanımı için kavramsal tasarım sonrasında temel araştırma ve ürün geliştirme gibi birçok aşamadan geçilmesi gerekmektedir (Uysal M. , A Formal Method for Mapping Software Engineering Practices to Essence, 2018). Ve proje süreci uzun yıllar devam edebilmektedir. Öz Çerçeve (Essence Framework) çerçevesi süreç, yöntem, teknik ve uygulamalar gibi süreçleri ortak zeminde birleştirmeyi amaçlamaktadır. Yazılım geliştirme süreçlerini esnek ve genişletilebilir bir yaklaşımla modellemektedir (Uysal M. , Using Concept Algebra For Mapping Software Practices To Essence Framework, 2018).

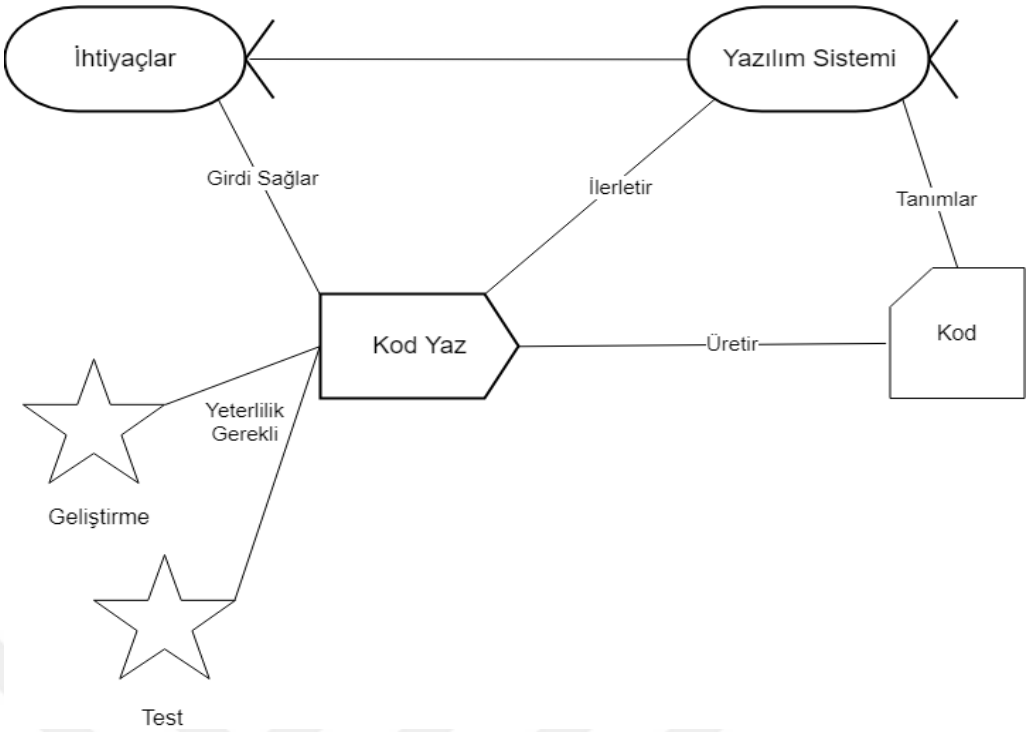
Software Engineering Method and Theory (SEMAT) tarafından geliştirilen öz (Essence), benzer belirtimlerden farklı olarak dil ve hazır bir çekirdekten (kernel) oluşan çerçeve ile karşımıza çıkmaktadır. Çekirdek, öz (Essence) model dili kullanılarak ortak kullanım için üç ilgi alanında (concern area) konumlandırılan yedi temel odak (Alpha) ve bu odaklar aralarındaki ilişkiyi tanımlayan şekilde gösterilmektedir. Bu model, Şekil 1'de verilmiştir (Uysal & Giray, 2017).



Şekil 1: Öç Çerçeve (Essence Framework) Alpha Bileşenleri

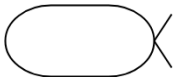
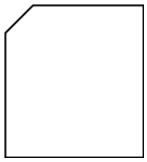
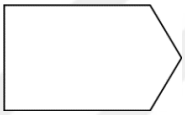
Yazılım geliştirme sürecinde yer alan etkinlikler ve varlıklar toplanarak Müşteri (Customer), Çözüm (Solution) ve Çaba (Endeavor) adıyla üç ilgi alanı tanımlanmıştır (Uysal & Giray, 2017). Müşteri ilgi alanında, bir yazılım sistemini geliştirmeye veya değiştirmeye uygun kılan şartlar Fırsat (Opportunity), bu sistemini etkileyen veya etkilenen insanlar, gruplar veya kuruluşlar ise Paydaşlar (Stakeholders), olarak tanımlanmaktadır. Çözüm ilgi alanında, Fırsatı değerlendirmek ve Paydaşların ihtiyacını karşılamak için yazılım sisteminin barındıracağı özellikler Gereksinimler (Requirements), yazılım, donanım ve verilerden oluşan ana üretim değeri ise Yazılım Sistemi (Software System), olarak tanımlanmaktadır. Çaba ilgi alanında, sonuç elde etmek için yapılan zihinsel veya fiziksel çabanın yer aldığı faaliyetlerdir. İş (Work), Yazılım Sisteminin geliştirilmesi, bakımı, teslim edilmesi veya desteklenmesi için aktif olarak çalışanlar grubu Takım (Team), Takım tarafından yapılan işlere rehberlik eden özelleştirilmiş süreç ve araç setidir. İş Yapma Biçimi (Way of Working) olarak tanımlamaktadır (Uysal & Giray, 2017).

Öz (Essence) çekirdeğini oluşturan her bir odak, farklı yaşam döngüsü basamağı ve basamaklar arasındaki geçiş denetimleri ile tüm sürecin sağlığı hakkında soyut seviyede bilgi sağlamaktadır. Öz (Essence) çerçevesi, sağlamış olduğu dil ve hazır çekirdek ile yazılım geliştirme çabalarında kullanılan farklı pratik ve yöntemlerin tek bir ortak zemin üzerinde birleştirme yeteneği sağladığı gibi kendine özgü alan bağımlı dili kullanarak çekirdeğin genişletilebilmesi olanağını da sağlamaktadır. Bu genişletme olanağı ile farklı yazılım geliştirme pratikleri ve yöntemleri aynı ortak zemin altında tanımlanabilmektedir. Tanımlanan yöntemlerin, proje veya ürünler için yürütülmesini gerçekleştirecek araç ise öz (Essence) çerçeve tarafından sağlanmaktadır (Uysal & Giray, 2017).



Şekil 2: Öz (Essence) Dili Kullanılarak Yazılım Süreçlerindeki Bileşenlerin Gösterilmesi

Tablo 1. Öz (Essence) Diline Göre Yazılım Süreçlerinin Açıklaması

Eleman Tipi	Gösterim Şekli	Açıklama
Alfa		Yazılım sisteminin çalışmasını, ilerlemesini ve sağlığının değerlendirilmesini sağlar.
Çalışma ürünü		Geliştiricilerin yazılım sisteminde faaliyetleri yürütürken ürettiği somut öğelerdir.
Aktivite		Geliştiricilerin yaptığı aktiviteler.
Yeterlilik		Belirli bir iş için gerekli olan yetenekleri, kazanımları, bilgileri ve becerileri içerir.

BÖLÜM 3

3. ARAŞTIRMA YÖNTEMİ

Bu araştırmanın temel amacı Mobil Yazılım Mühendisliği (MYM) alanına yönelik bir uygulama geliştirme süreç modeli geliştirmek ve bu modelin uygulama sonuçlarını gözlemlemektir. Çalışma, Karma Araştırma Yöntemleri (Mixed Research) çerçevesinde Eylem Araştırması (Action Research) (EA) ile Tasarım Bilimi Araştırma Yöntemleri (Design Science Research) (TBAY) doğrultusunda bütünleşik bir yapıda yürütülmüştür (Tablo 2). EA kapsamında araştırmanın iki döngüsü ve üç aşaması bulunmaktadır. Her bir aşamasının çıktısı bir sonraki aşamanın girdisini oluşturmuştur. EA yaşam döngülerinde TBAY'nin tasarım, geliştirme, test ve değerlendirme etkinlikleri gerçekleştirilmiştir. EA birinci döngüdeki araştırma etkinlikleri ile birinci araştırma sorusu, EA ikinci döngüsü ile de ikinci araştırma sorusu cevaplanmaya çalışılmıştır. Cevaplanmaya çalışılan araştırma problemleri aşağıdaki gibidir:

AS-1. MYM ve Mobil Uygulama Geliştirme (MUG) alanlarındaki problem sahaları ve çözüm önerileri nelerdir?

AS-2. MYM alanına yönelik bir mobil uygulama geliştirme modeli nasıl geliştirilebilir?

Birinci aşamada MYM alanıyla ilgili literatür taraması gerçekleştirilerek problem sahaları ve bunlara yönelik çözüm önerileri belirlenmiştir. Daha sonra, EA kapsamında mobil uygulama geliştiren bir firma belirlenmiş, kullandıkları yazılım geliştirme süreçleri ve literatürden elde edilen bilgiler incelenmiştir. Bunun sonucunda gerek MYM çalışma alanındaki yazılım süreçleriyle ilgili problemlere ve gerekse ilgili firmanın ihtiyaçlarına cevap verebilecek bir Mobil Uygulama Geliştirme Modelinin (MUGM) olmadığı gözlenmiştir (Uysal M. , Towards a software engineering research framework: Extending Design Science Research, 2016). Söz konusu bu durum aynı zamanda araştırmanın problemin tespiti ve formüle edilmesi aşamasını da oluşturmuştur.

İkinci aşama ÖÇ Tabanlı Mobil Yazılım Geliştirme Modelinin geliştirilmesi ve değerlendirilmesiyle ilgili TBAY etkinliklerini (tasarımı, geliştirilme, test ve değerlendirilme) içermiştir. Bu aşamanın çıktısı MUGM'dir (Uysal M. , Extending the Design Science Research Methodology for software engineering, 2015). EA kapsamındaki birinci döngü olan bu aşamada araştırmacı ve ilgili firma temsilcileri

birlikte çalışarak MUGM'nin ihtiyaçları ne ölçüde karşıladığı, uygulanabilirliği belirlenmiş, edinilen tecrübeler aktarılmış ve modelle ilgili gerekli olan düzenlemeler ve güncellemeler gerçekleştirilmiştir.

Üçüncü aşama EA'nin ikinci döngüsü ile TBAY kapsamında mobil yazılım ürününün test ve değerlendirilmesi, araştırmanın çıktısı olan MUGM'nin değerlendirilmesi etkinliklerini içermiştir.

Tablo 2. Araştırma Yöntemi

<table>
<tr><td colspan="2">Eylem Araştırması (EA) Birinci Döngü</td><td>Eylem Araştırması (EA) İkinci Döngü</td></tr>
<tr><td>Aşama-1</td><td>Aşama-2 (ÖÇ Tabanlı Mobil Yazılım Geliştirme Modelinin Geliştirilmesi ve Değerlendirilmesi)</td><td>Aşama-3 (Mobil Yazılımın Geliştirilmesi, Test ve Değerlendirilmesi)</td></tr>
<tr><td rowspan="2">Problemin Tespiti ve Formüle Edilmesi</td><td><u>TBAY Etkinlikleri (Tasarım)</u>

1. Scrum'ın ve Kullanıcı Hikayesinin, ÖÇ yazılım geliştirme bileşenleriyle eşleştirilmesi ve ÖÇ bilgi alanında gösterimi
2. ÖÇ tabanlı mobil yazılım geliştirme modelinin tasarım,

TBAY Çıktısı: ÖÇ Tabanlı Mobil Yazılım Geliştirme Modeli</td><td><u>TBAY Etkinlikleri (Geliştirme ve Test</u>

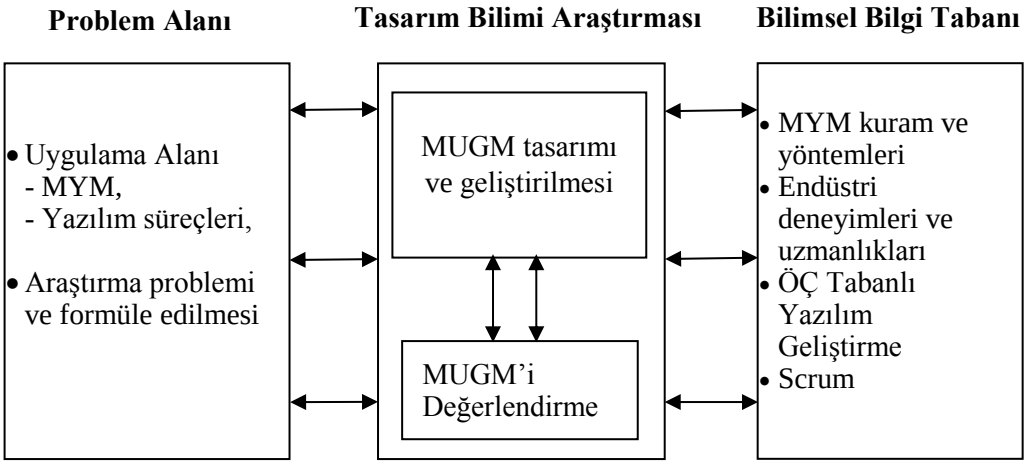
1. Mobil yazılımın tasarımı,
2. Mobil yazılımın geliştirilmesi,
3. Mobil yazılımın test edilmesi

TBAY Çıktısı: Mobil Yazılım</td></tr>
<tr><td>Mobil Yazılım Geliştirme Modelinin Değerlendirilmesi</td><td>Mobil Yazılımın Değerlendirilmesi</td></tr>
<tr><td colspan="3">Araştırmanın Kuramsal Temelleri</td></tr>
<tr><td colspan="3">Mobil Yazılım Mühendisliği, Endüstrisindeki En İyi Uygulamalar, Öz Çerçeve Yaklaşımı, Scrum, Kullanıcı Hikayesi</td></tr>
</table>

Son yıllarda yapılan araştırmalar incelendiğinde sosyo-teknik bir yapıya sahip olan bilişim sistemlerine yönelik Eylem Araştırması (EA) ve Tasarım Bilimi Araştırma Yöntemlerinin (TBAY) kullanımının arttığı ve giderek benimsendiği gözlenmektedir (Baskerville & Michael , 2004). Bu iki yöntemin birlikte kullanımını destekleyen görüş ve çalışmalar da artmaktadır (Livari & Venable, 2009). Bu çalışmada, MYM uygulama alanı ve firmanın ihtiyaçlarından hareket edilerek karma araştırma yaklaşımı benimsenerek bu iki araştırma yöntemi bütünleşik yapıda kullanılmıştır.

3.1. Tasarım Bilimi Araştırma Yöntemi

Yazılım ve özellikle son yıllarda bilişim sistemleriyle ilgili araştırmalarda yaygın olarak kullanılmaya başlanan araştırma yöntemlerinden diğeri de TBAY'dir. Fen ve Sosyal Bilimlerde benimsenen araştırma yaklaşımından kısmen farklı olan TBAY'de, çevre ve gerçek hayat problemlerinden hareket edilerek çeşitli işlev ve özelliklere sahip sistem, araç, yöntem, teknik veya modellerin geliştirilmesi, iyileştirilmesi hedeflenmektedir (Uysal M. , Extending the Design Science Research Methodology for software engineering, 2015). TBAY'da, bir amaca yönelik ürün geliştirmenin yanı sıra bilimsel araştırma alanına yönelik katkıda bulunmak da hedeflenmektedir. Şekil 3'de (Uysal & Giray, 2017) gösterildiği gibi (a) endüstri problem alanı, (b) tasarım bilimi araştırması ile (c) bilimsel bilgi tabanı TBAY'nin üç ana bileşenini oluşturmaktadır. TBAY yinelemeli bir uygulamayı gerektirmektedir. Her bir yinelemenin çıktısı geliştirilecek ürüne veya modele ait bir bileşeni içermektedir. Bu aşamalarda ürüne ait bileşen değerlendirilmekte ve aynı zamanda problem alanı ile bilimsel bilgi tabanına olan katkılar ortaya konulmaktadır.



Şekil 3:TBAY Temel Bileşenleri

3.2. Veri Toplama, Evren ve Örneklem

Çalışmanın evrenini mobil yazılım geliştirme yöntemleri oluşturmuştur. EA çerçevesinde amaçlı örneklem kullanılarak mobil yazılım geliştiren bir firma belirlenmiştir. Literatür taraması, EA ve TBAY kapsamında Scrum da gerçekleştirilen planlama, değerlendirme ve retrospektifi aktivitelerinde toplantı tutanakları, günlük Scrum toplantı görüşmeleri, kullanıcı hikayesi kartları ve MUGM'nin uygulama neticesinde elde edilen veriler araştırma verisini oluşturmaktadır. Verilerin kaydı için Google Drive uygulamasına ait bir dizin oluşturulmuştur.

3.3. Veri Analizi

Sprint'in planlama, değerlendirme ve retrospektifi aktivitelerinde toplantı tutanakları tutulmuştur ayrıca kullanıcı hikayesi uygulamasında da kullanıcı kartları oluşturulmuştur. Bu toplantı tutanakları ve kullanıcı hikayesi kartları EK-3, EK-4 ve EK-5'da verilmiştir. Ayrıca Scrum günlük toplantılarında görüşülen konular toplantı görüşmeleri olarak kayıt edilmiştir. Scrum'da yapılan üç Sprint'de değerlendirme aktivitesinde doküman analizi yapılmıştır. Bir sonra ki Sprint'e bu analiz aktarılmış ve Sprint buna göre planlanmıştır. Ayrıca uzman görüşleri alınarak veriler analiz edilmiştir.

3.4. Geçerlilik ve Güvenirlik

Çalışmanın görünüş ve kapsam geçerliliğini arttırmak amacıyla MYM ve MUG alanında çalışan yazılım ve alan uzmanlarının görüşlerine başvurulmuştur. Kapsam geçerliliği için yine aynı alanda çalışan akademisyenlerin görüşleri alınmıştır. Scrum değerlendirme ve retrospektifi aktivitelerin de uygun ortam sağlanarak paydaşlar, yazılım ekibi ve personeller rahat biçimde ürün ve süreçleri değerlendirmeleri istenmiştir. Dış güvenirliği artırmak için araştırma süreci ayrıntılı biçimde planlanmış, veri toplama süreci detaylandırılarak kontrol edilmiş, kayıtlar ayrıntılı ve diğer araştırmacılara açık biçimde dizinde tutulmuştur.

BÖLÜM 4

4. ÖÇ TABANLI YAZILIM GELİŞTİRME SÜREÇ MODELİ

4.1. ÖÇ Tabanlı Mobil Yazılım Geliştirme Sürecinin Tanımlanması

Öz (Essence) dili, öz (Essence) çekirdeğinin en üstündeki uygulamaları, bir sonraki hedef duruma veya durumlara nasıl ulaşılacağına veya mevcut durumun nasıl korunacağına dair açık rehberlik ederek yazılım süreçlerinin tanımlanmasına olanak tanımıştır. Essence unsurlarını ve dili uygulamaları açıklamak için bir kelime olarak ya da başka bir deyişle uygulamaları özümsemek için özü (Essence) kullanmak anlamına gelmektedir. Yazılım mühendisliği çalışma alanında yazılım takımları genellikle değişik uygulamalara ihtiyaç duymakta, bunlar arasındaki çakışmaları ve çatışmaları ortadan kaldırmak için birleştirilmeleri gerekebilmektedir (Uysal & Giray, 2017). Böylece, öz (Essence) dili, takımların kendi seçtikleri birçok uygulamayı içeren bir yöntem oluşturmasına izin veren bir kompozisyon mekanizması içerir. Bu bölümde, siber güvenlik alanında ihtiyaçtan hareket edilerek öz (Essence) doğrultusunda mobil bir uygulamanın nasıl geliştirildiğiyle ilgili konular ele alınmıştır. Öz (Essence) ile tanımlanan uygulamaların yazılım takımlarının karşılaştığı ortak güçlüklere nasıl çözüm olabileceği tartışılmıştır (Uysal M. , Using Concept Algebra For Mapping Software Practices To Essence Framework, 2018). Öz (Essence) ve Scrum çerçevesinde aşağıdaki dört temel yazılım geliştirme aşamaları izlenmiştir:

- Projede kullanılacak ÖÇ bileşenlerinin belirlenmesi,
- ÖÇ yaklaşımıyla durumun çözümlenmesi,
- Kapsam ve proje kontrol noktalarının belirlenmesi,
- Projede kullanılacak yazılım teknik, araç ve uygulamalarının belirlenmesi

4.1.1. Projede Kullanılacak ÖÇ Bileşenlerinin Belirlenmesi

Bir yazılım geliştirme çalışmasında dikkat edilmesi gereken önemli şeyler alfalardır. Çekirdek, bazı evrensel alfaları yani paydaşları, fırsatı, gereksinimleri, yazılım sistemini, işi, ekibi ve çalışma biçimini tanımlar. Bunlar, yazılım geliştirmek için hangi yolu seçerseniz seçin gerekli olan durumlardır. Buna ek olarak, çekirdek ekiplerin sağlıklı

ilerlemesini ve dolayısıyla yazılım geliştirme çabalarını değerlendirmek için bir hatırlatma olarak durumların tanımlarını ve kontrol listelerini sağlar.

Ancak, çekirdeğin alfaları, izlenecek tek şey değildir. Yazılım ekiplerinin, özellikle de gereksinim öğeleri olmak üzere, izlemek için başka durumlarda bulunmaktadır.

Başvurulacak uygulamalara karar verdikten sonra, yazılım ekibinin, sağlıklı ilerlemeyi sağlamak için izlenmesi gereken önemli şeyler (alfalar) konusunda girdileri bulunmaktadır. (Bkz. Tablo 3).

Tablo 3. Uygulanan Siber Güvenlik Eğitim Portali Uygulamaları

Pratik	Tanım	İzlenecek Alfalar
Scrum	Bir birikmiş işler çalışan yazılım sistemlerinin yinelenen gelişimi için bir uygulama	Sprint İstekler Maddeler
Kullanıcı Hikâyeleri	Bir yazılım sisteminin kullanıcısına değer sağlayacak işlevselliği yakalamanın bir yolu	Kullanıcı hikâyesi
Mobil Yazılım	Yazılım sisteminin kullanıcıya sunulacak olan ara yüzü.	Mobil Yazılım

- Scrum, Sprint olarak adlandırıldığı her yenilemeyi geliştirme için bir uygulamadır. Sprint alfayı izlemek için ihtiyacımız olan bir yapıdır. Scrum ekipleri iş listelerini bir iş listesinde tamamlamak için yönlendirir. Scrum terminolojisini kullanan ürün iş listesi olarak bilinen bu birikmiş işler de alfa olarak kabul edilebilir.
- Kullanıcı hikâyeleri, ihtiyaç maddelerini daha kısa ve öz bir şekilde ifade etmeye yönelik bir uygulamadır ve değerlere odaklanır. Kullanıcı hikâyeleri alfadır.

- Mobil yazılım ihtiyaç öğelerinin kullanıcı hikâyelerine göre yazılımını sağlayan bir uygulamadır. Mobil yazılım öze (Essence) göre alfadır.

Doğru alfaları tanımlamak önemlidir, çünkü alfanın durumunu açık bir şekilde değerlendirme ve takip etme ihtiyacı nedeniyle alfa bir şey yapmanın bir maliyeti olmuştur. Örnek olarak, birlikte çalışmayı deneyimleyen yalın bütçeli küçük ekipler risklerini sadece gizli bilgi ile takip etmeye karar verebilirken, daha karmaşık çabalarda çalışan büyük ekiplerin alfalar gibi temel riskleri açık bir şekilde takip etme ihtiyacını ve geri ödeme oranını görmemektedirler. Bir ekibin ilerleme ve sağlık durumlarını değerlendirirken kullandığı doğru durumları ve her alfa için doğru kontrol listelerini belirlemek de önemli olmuştur. (Uysal & Halici, Representing essence of software engineering in enterprise architecture knowledge domain, 2018).

4.1.2. ÖÇ Yaklaşımıyla Durumun Çözümlenmesi

Bu bölümde, öz (Essence) doğrultusunda siber güvenlik alanında çalışan bir firmanın güvenlik eğitimine yönelik bir mobil yazılım geliştirilmekte; buna ait geliştirme süreçleri, yazılım etkinlikleri, kullanılan öz (Essence) teknikleri, araçları ve edinilen tecrübelere değinilmiştir. Mobil uygulama güvenlik açıklarını sürekli nasıl giderilmesi gerektiğini personele öğretilmesini hedefleyen bir mobil öğrenme sistemidir.

Yazılım geliştirme birçok çaba gerektiren bir problem çözme etkinliğidir. Yazılım geliştirme, bazı problemlerin ya da bir fırsatları tanımlamanın bir sonucudur. Birçok sorun kaynağı olabilir. Bazıları inşa edilen veya inşa edilmiş olan yazılım sisteminin kendisi ile ilgili olabilir. Örneğin, yazılım sistemi zaten mevcutsa, orijinal gereksinimleri ve/veya mevcut yazılım sistemine yol açan çözümü tam olarak anlaşılamayabilir. Paydaşların katılımıyla ilgili problem de olabilir; problemleri daha iyi anlamak için yazılım sisteminin kullanıcıları ile mülakatlar gerekebilir, zaman ve kaynak kısıtlılıkları olabilir. Yazılım ekibi ve personeli arasındaki iletişim problemleri karşılaşılan diğer sorunlar arasındadır. Daha az deneyimli bir geliştirici, çok meşgul ve deneyimli bir geliştiriciden gereken desteği alamıyor olabilmektedir.

Tüm problem çözme etkinliklerinde olduğu gibi yazılım mühendisliğinin farklı boyutlarında bulunan birden çok ilgili problemi içerebilen problemi anlamaya başlar. Yazılım mühendisliğinin boyutlarına örnek olarak paydaşlar, gereksinimler, yazılım sistemi ve ekip dâhildir. Paydaşların ihtiyaçları da anlaşılmak zorundadır. Öz (Essence)

yapısı itibariyle, yazılım mühendisliğinin her boyutunda nerede olduğunu anlamanıza yardımcı olan bir yaklaşım sunmaktadır. Örneğin, alfalar, geliştirme çabaları hakkında sorular sormamıza yardımcı olarak yazılım süreçleri ve bileşenleri hakkında bilgiler toplanılmasına yardımcı olmaktadır.

Şekil 4: Öz (Essence) ile Siber Güvenlik Eğitim Portali

Yazılım takım lideri ve ekibi bir araya gelerek yapıştırma notları ve öz (Essence) alfalarını kullanarak çaba hakkında bildiklerini yakalamıştır. Şekilde görülen metinler, her alfa ile ilgili bilgi türüne ilişkin bir fikir edinmenize yardımcı olmak amacıyla, alfa için ilgili bilgilerin yalnızca bir örneğidir. Bu bilgiler şu şekildedir:

Müşteri bakış açısından:

Paydaşlar: Bu çalışmadaki paydaşlar, güvenlik birimi ve dijital dönüşüm ofisidir. Dijital dönüşüm ekibine şirketin işini genişletmek için dijital projelerin iş modelindeki değişiklerle beraber personel deneyimindeki iyileştirmeler projenin iyiye gitmesini sağlamıştır. Güvenlik birimi hangi eğitimlerin hangi birimlere veya hangi eğitimlerin önemli olması gerektiği gibi konularda geri bildirimlerini sunmuştur.

Fırsat: Şirkette birçok güvenlik açığı oluşuyordu ve bunların şirket personeline eğitim olarak sunulması sağlanmıştır. Bu nedenle, siber güvenlik eğitim portali personelin bu konuda eğitimini sağlamak için çok uygun bir platform olmuştur. Siber Güvenlik Eğitim Portali, yeni katılan personele eğitimlerin yeniden düzenlenip tek tek yapılmasının yerini alarak tek bir uygulama ara yüzünden tüm personelin erişebileceği bir yapı sunmuştur.

İlgili çözüm alanının perspektifinden:

Gereksinimler: Güvenlikle ilgili verilmesi gereken eğitimlerin hızlı bir şekilde verilmesi sağlanmıştır. Hızlı bir şekilde verilmeyen eğitimler şirketi maddi ve manevi zararlara uğratmıştır. Uygulama ile personele eğitim verilmesi sağlanmıştır.

Yazılım sistemi: Siber güvenlik eğitim portali bulut tabanlı bir mobil uygulamaya sahiptir. Buda personellerin erişmek için kendi dijital cihazlarına herhangi bir yazılımı indirip yüklemelerine imkânı sağlanmıştır.

Çaba bakış açısından:

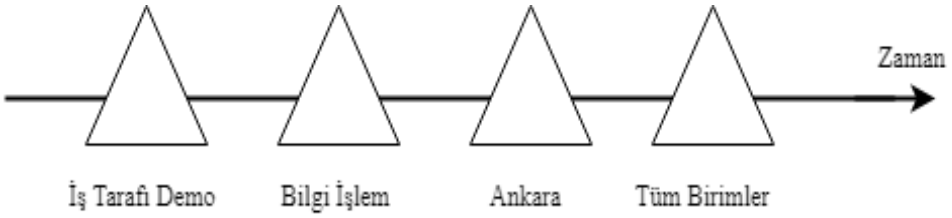
İş, yönetim tarafından bir ay içinde yazılım ekibinin yeni ürünü sunmaları istenmiştir ve 3 Sprint içeresinde projenin tamamlanması sağlanmıştır.

Ekip, yazılım takım lideri ve yazılım ekibidir.

Çalışmanın Yolu, yazılım takım lideri ve ekibi, öz (Essence) imkânlarını sağlıklı ilerleme sağlamak için kullanmıştır. Sorunların nerede olduğunu ve projenin gelişme aşamasında nereye odaklanmaları gerektiğini anlamak için alfa durumlarını ve kontrol listelerini kullanmıştır. Aynı zamanda sorunların çözümünde gereken faaliyetlere karar verme aşamasında da kullanılmıştır.

4.1.3. Proje Kapsamı ve Kontrol Noktalarının Belirlenmesi

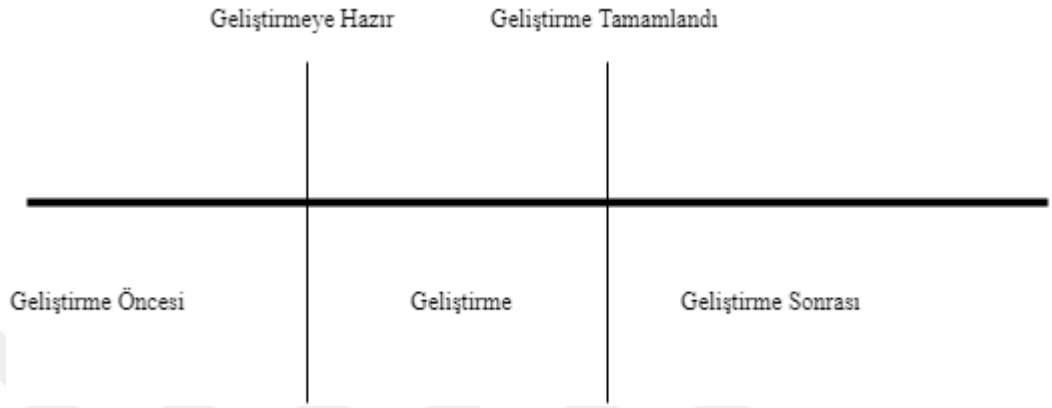
Dijital dönüşüm ekibi ve siber güvenlik ekibi, siber güvenlik eğitim portalinde eğitimlerinin personellere nasıl yaygınlaştırılacağını tartışmıştır. Az sayıda dâhili kullanıcıyla başlanmıştır ve şekildeki gibi aşamalı olarak tüm birimlere, personellere giden aşamalı bir yaklaşım üzerinde anlaşılmıştır.



Şekil 5: Uygulamayı Yayınlama Yol Haritası

Siber güvenlik eğitim portali deneme kullanıcıları dışındaki ilk gerçek kullanıcılar olan bilgi işlem birimi ile yaygınlaştırılmıştır. Bilgi işlemin seçilmesi siber güvenlik saldırılarının çoğunlukla bilgi işlem birimini hedef almasından kaynaklıdır. Bilgi işlemden sonra şirketin bir bölgesi olan Ankara ele alınmıştır. Siber güvenlik birimi bu kullanıcı gruplarını ara ara ödüllendirerek bu konuda insanların eğitime istekleri artırılmıştır.

Öz (Essence) alfaların durumları, her bir durumun kontrol listeleriyle birlikte, ekibin geliştirmeye başlamak için ön koşullar ve gelişimin tamamlanma kriterleri hakkında anlaşmaya varması için ileriye dönük bir yol sağlamıştır. Yazılım takım lideri öz (Essence) kartları için "Geliştirmeye Hazır" ve "Geliştirme Tamamlandı" şeklinde iki ana kontrol noktası belirlemiştir. (Bkz. Şekil 6)



Şekil 6: Geliştirilmesi İçin Kontrol Noktaları ve Aşamaları

Yazılım ekibi iki toplantı yaparak birincisinde "Geliştirmeye Hazır" kontrol noktasında eklenmesi gereken alfaları seçmiştir, ikinci toplantıda alfaların ihtiyaçları konusunu ele almıştır. Yazılım ekibi öncelikle birinci turda tüm alfaları belirleyerek bir yol haritası çıkartmıştır. Yazılım takım lideri ikinci toplantıda alfaların ihtiyaçları konusunda ekip ile bir çalışma yapar.

Birinci turda yazılım takım üyeleri tüm alfaları not almışlardır ve kendilerine not aldıkları alfaları göz önünde bulundurarak bir görüşme sağlamıştır. Ekip içinde oylama ile alfa devam etsin mi yoksa etmesin mi kararlaştırılır. Gerekli olan alfalarla projeye devam edilmiştir. Herkesin hem fikir olması durumunda "Geliştirmeye Hazır" alfası kararlaştırılmıştır.

İkinci turda alfaların gereklilikleri tartışılarak gerçekten alfa ihtiyacı olup olmadığı noktasında bir birlik sağlanmıştır. Burada çeşitli durumlar ortaya çıkabilir örneğin gerekli alfa için finansman onayı gerekli olması ya da başka bir birimden onay alınması gerekmiştir. Şirket bünyesinde finansman onayı olmadan işe başlanması mümkün olmamıştır. Burada yazılım takım lideri resmi fon onayı gelinceye kadar geliştirmeye başlamak için ilgili birimlerden onay olarak geliştirmeyi başlatmıştır.

Yazılım ekibi paydaşların katılımına ihtiyaç duyduklarını göstermek zorundadır. Ayrıca gereksinimlerin sınırlı olduğunu ve yazılım sisteminin seçilmiş mimariye uygunluğunu

kabul etmişlerdir. Bu konuda önemli teknik riskler ele alınmalıdır. Ekip her durumla ilgili kontrol listeleri üzerinde anlaşmıştır.

Şekil 7: Kabul Edilen Alfalar

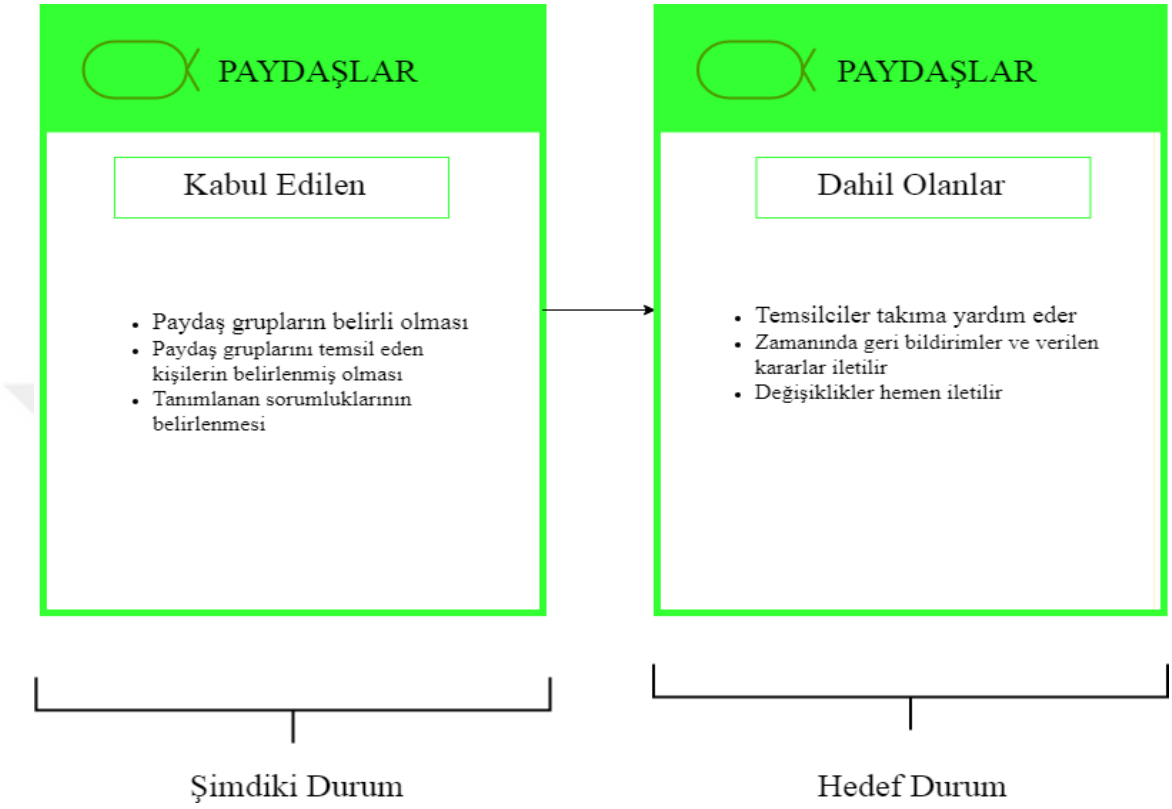
Yazılım takımı bir sonraki iterasyonda başarmaları gereken durumları değerlendirmiştir. Yazılım ekibi ihtiyaç duydukları durumları dile getirerek alfa kartlarını oluşturmuştur. Bir sonra ki hedeflenen iterasyon, bir önceki yenilemeyi destekler şekilde olmuş ve tutarlı olması gerekmiştir. Bu iterasyon için şekildeki alfa kartları oluşturulmuştur (Bkz. Şekil 8).

Şekil 8: Uygulanması Gereken Alfalar

Yazılım takımı mevcut durumu değerlendirerek bir sonra ki hedefe ulaşmak için alfa kartlarını şimdiki durumu ve hedeflenen durum olarak bir sonraki sürümde hedeflenen durumları belirlemiştir. Bazı alfa kartlarında değişimler fazla olurken bazı alfa kartları olduğu gibi kalmıştır. Burada ilgili alfanın iterasyonlardan ne kadar değişerek etkilendiğine bağlı bir durum oluşmuştur.

· Müşteri alanı:

Paydaşlar: Kabul Edilen => Dâhil Olanlar (Bkz. Şekil 9)

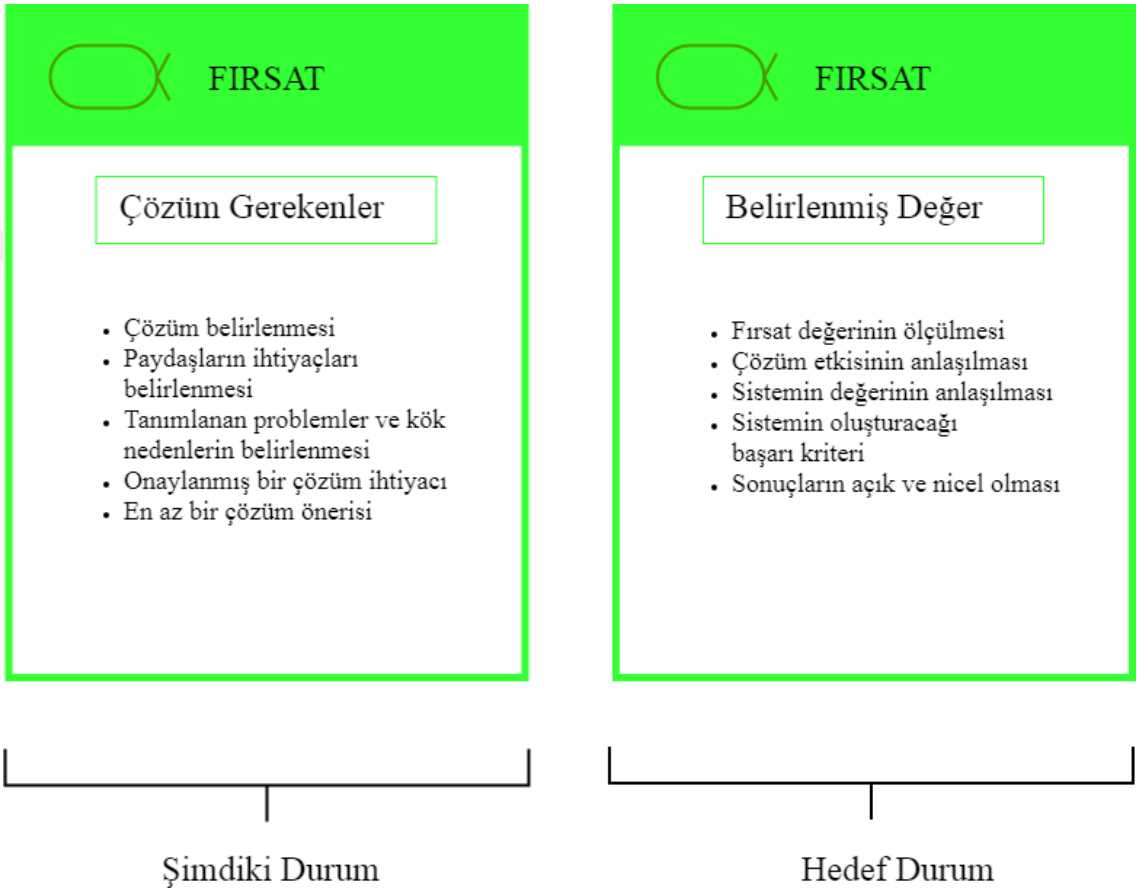


Şekil 9: Paydaşlar- Şimdiki ve Hedef Durum

Ekip için kritik iki paydaş bulunmaktadır; dijital dönüşüm ofisi ve siber güvenlik ekibidir. Alfa kartlarındaki kartların bağımlılıklarını iletip paydaşlarla birlikte ilerlemek gerekmektedir. Burada önemli bir görev ortaya çıkmıştır, paydaşlarla toplantı yapılıp alfaların hedef duruma ulaşması için gerekli olan durumların anlatılması sağlanmıştır.

· Fırsat:

Çözüm Gerekenler => Belirlenmiş Değer (Bkz. Şekil 10)

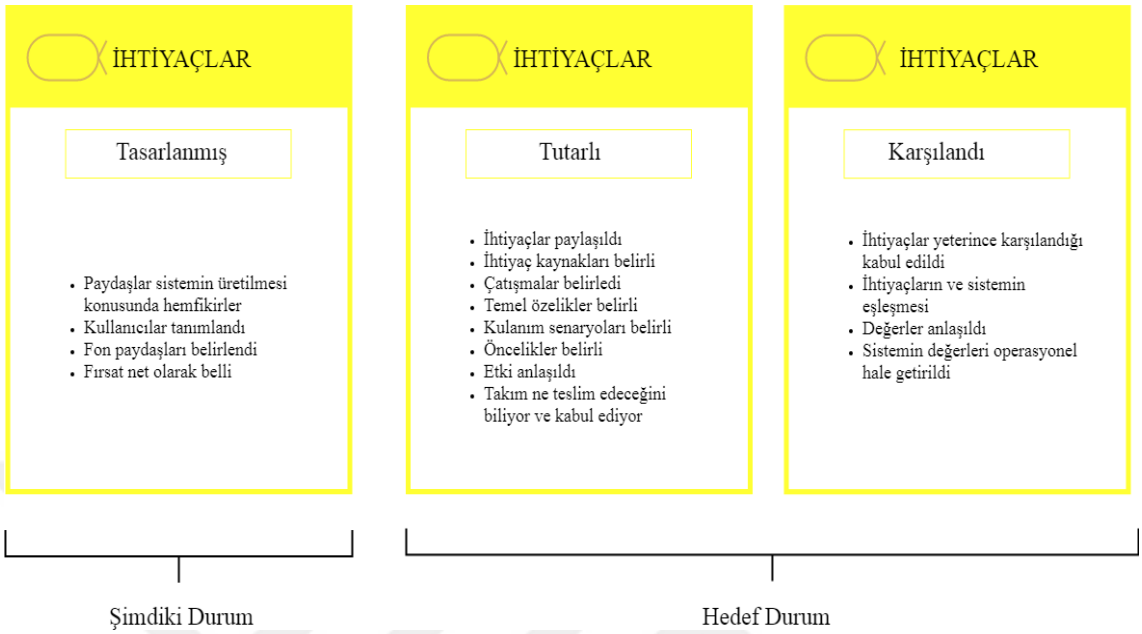


Şekil 10: Fırsat- Şimdiki ve Hedef Durum

Gerekli eğitim verilerini oluşturmak için bir çözüme ihtiyaç bulunmaktaydı. Burada paydaşların beklentilerinin göz önüne alınıp bir çözüm oluşturulmuştur. Gerekli eğitimler personel tarafından takip edilip eğitimleri mobil uygulama üzerinde tanımlanmıştır. Burada personelin daha kolay kullanacağı bir platform oluşturmak için test ortamında eğitimler ile kısıtlı bir personel kitlesinin görüşleri alınarak iyileştirme yapılmıştır.

· Çözüm Alanı:

Gereksinimler: Tasarlanmış => Uyumlu, Adresli (Bkz. Şekil 11)



Şekil 11: İhtiyaçlar- Şimdiki ve Hedef Durum

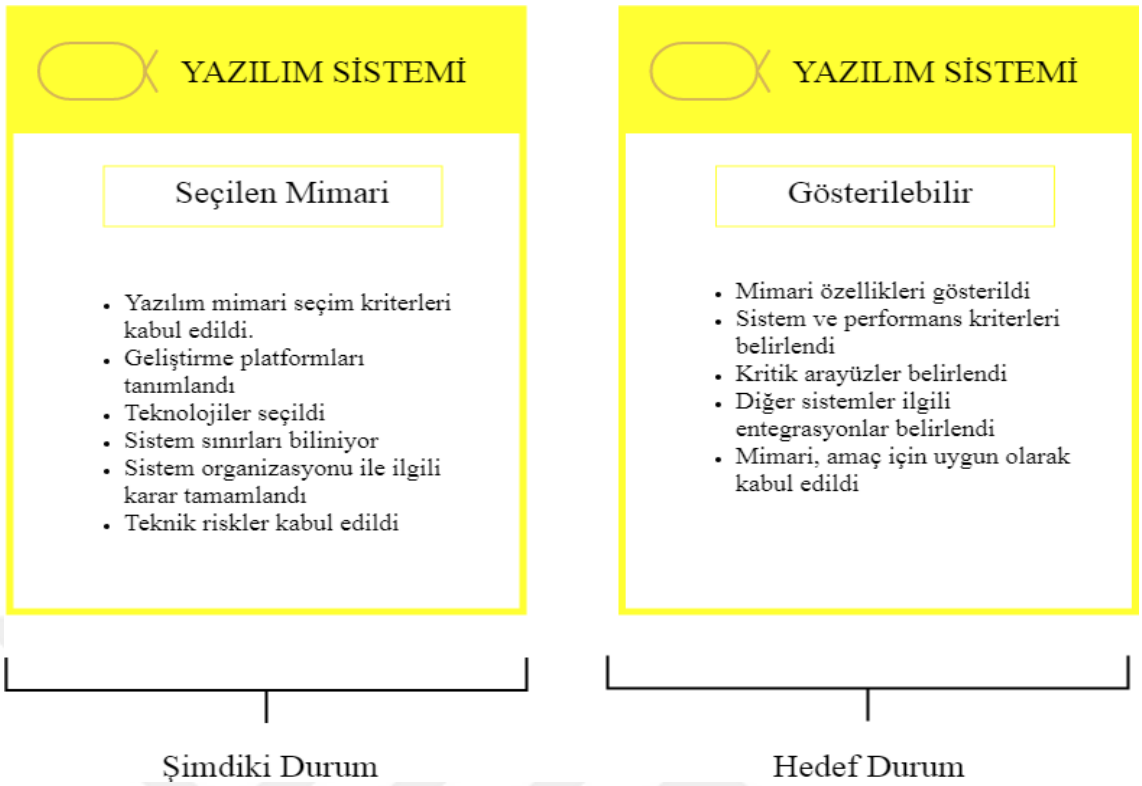
İhtiyaçlar örnek olarak bir ihtiyaç durum söz konusudur. Durumlar karşısında ekibin ihtiyaçları ortaya çıkartılmıştr. Örnek ihtiyaç öğesi üzerinden konuyu özetlersek.

· İhtiyaç Öğesi 1: Sistem personeller için gerekli eğtimleri sunar.

Eğitimlerin belirlenmesi ve bunun personele sunulması için hangi birim için hangi eğitimin gerekli olduğuna karar verilmiştir. Ekip durumların gerekliliklerini yerine getirmek için hızlıca çalışma yapmıştır. Yazılım ekibi hangi ekibin hangi eğitimi alacağı konusunda güvenlik birimleriyle anlaşmıştır, test verileri ile kontrolü sağlanmıştır.

Yazılım Sistemi:

Seçilen Mimari => Gösterilebilir (Bkz. Şekil 12)

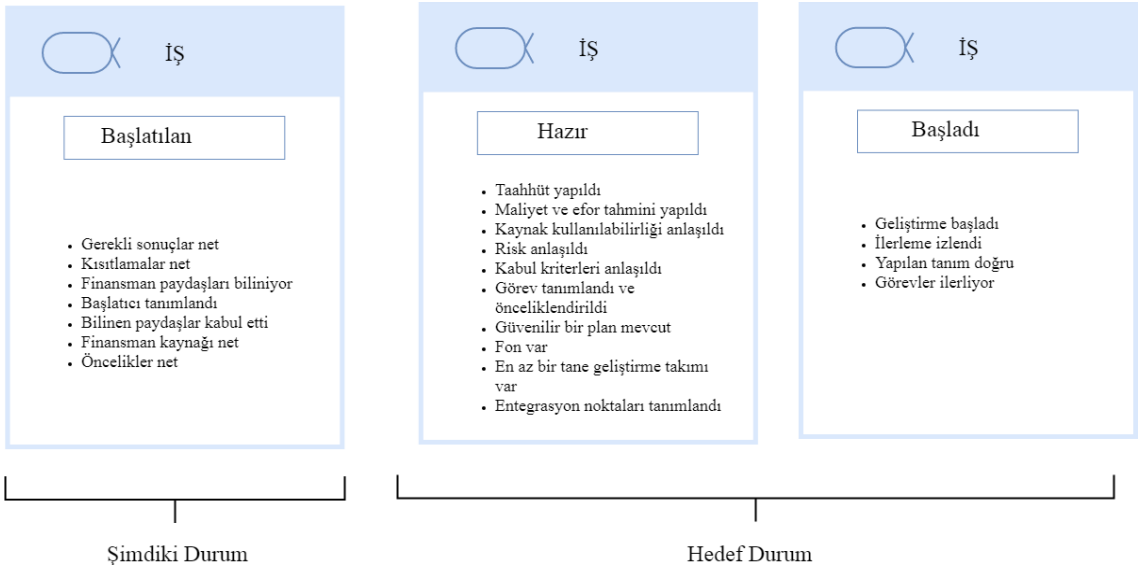


Şekil 12: Yazılım Sistemi- Şimdiki ve Hedef Durum

Gösterilebilir duruma gelmesi için yazılım sisteminin mimarisinin oluşturulmuş, kritik bölümlerinin kodlanmış, yazılıma entegre edilmiş ve testler yapılmıştır.

Çaba alanı:

İş: Başlatılan => Hazır, Başladı (Bkz. Şekil 13)

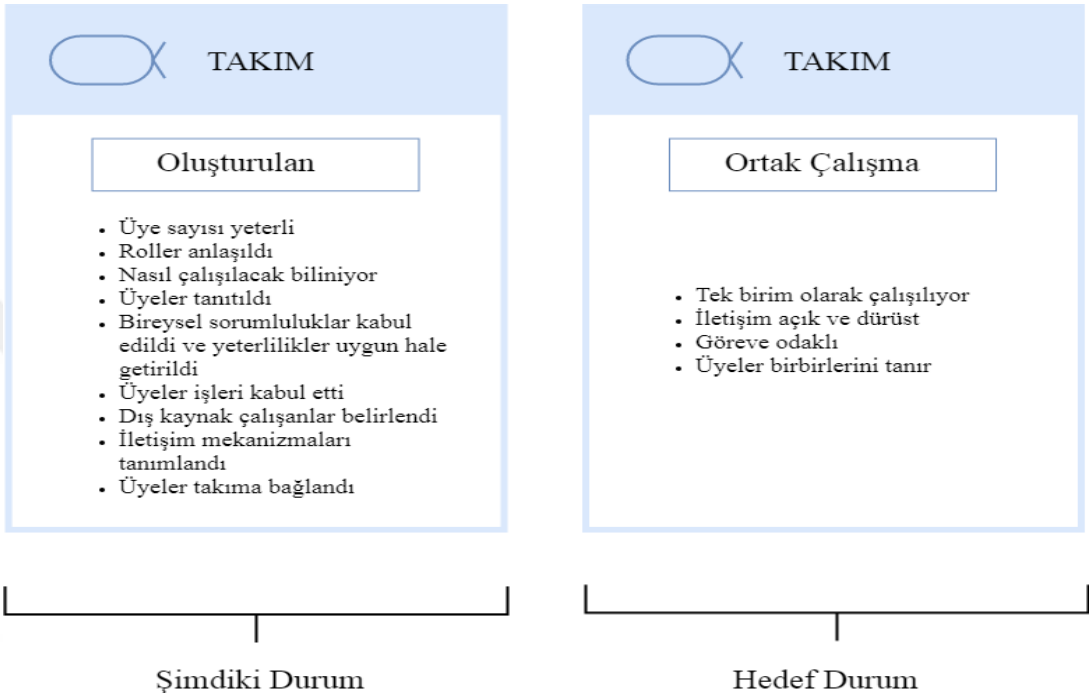


Şekil 13: İş- Şimdiki ve Hedef Durum

Hazır durumuna gelmek için ekip görevleri iterasyona uyacak şekilde küçük parçalara bölünmüştür. Görevler küçük parçalara ayrıldıktan sonra riskler anlanarak tehlikelerin önüne geçilmiştir.

Takım:

Oluşturulan => Ortak Çalışma (Bkz. Şekil 14)

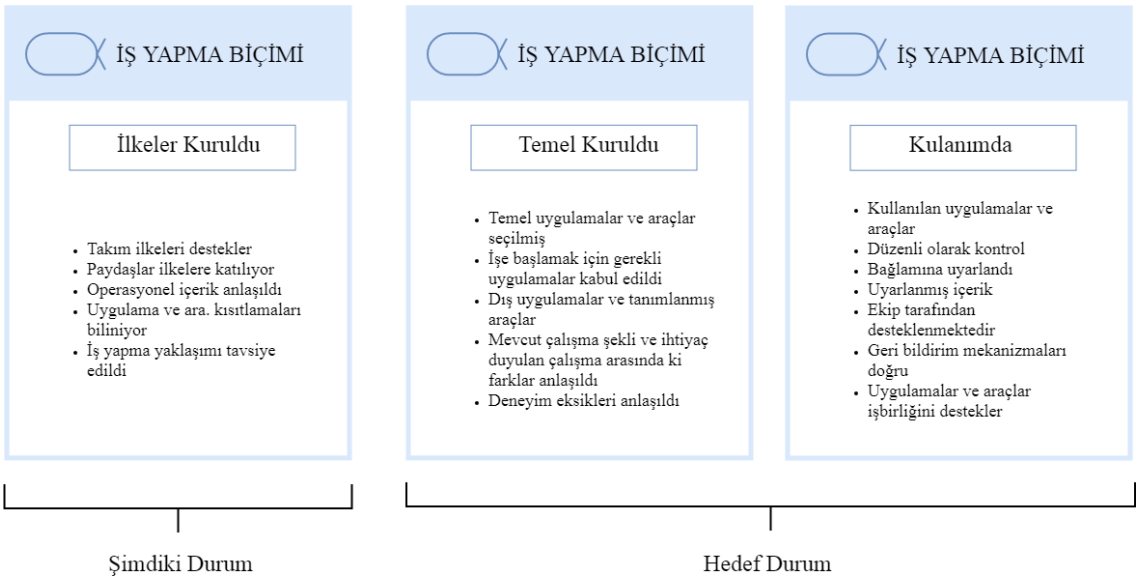


Şekil 14: Takım- Şimdiki ve Hedef Durum

Ekip çalışması birçok sorun getirmiştir. Ekibin bu sorunları aşması gerekmiştir. İletişimin açık ve dürüst olması, işlerin göreve uygun olarak ilerlemesi, üyelerin birbirlerini tanıması için çeşitli etkinlikler yapılıp ekip içi iletişim kuvvetlendirmesi gerekmiştir. Örneğin ihtiyaç öğesi 1 için ekibin iş paylaşımı sırasında görev dağılımın iyi ve net olması, iletişimin kuvvetli olması gerekmiştir. Yazılım ekibi burada ekip ruhunu oluşturup görev ve yetkileri tam olarak uygulamıştır.

Çalışma Şekli:

İlkeler Kuruldu => Temel Kuruldu, Kullanımda (Bkz. Şekil 15)



Şekil 15: İş Yapma Biçimi- Şimdiki ve Hedef Durum

Ekip üyeleri arasında görev dağılımı yapılarak geliştirme ve test ortamı oluşturulmuştur. Öncelikle bir depo oluşturuldu ardından test ortamı ve senaryoları üretilmiştir. Mimarinin şekillenmesinden sonra yazılım ekibi ortamı ve mimariyi kurdu.

4.1.4. Projede Kullanılacak Yazılım Teknik, Araç ve Uygulamaların Belirlenmesi

Ekip ihtiyaçlarının izlenmesi için dünya üzerinde birçok iş takip sistemi bulunmaktadır. Projenin hızlı ilerlemesi, iş takip ortamına entegre edilmesi eğitimlerinin verilmesi gibi birçok durum ortaya çıkacağı için elektronik tabloda ihtiyaçlar tutulmuştur.

Ekip toplantı sonunda paydaşların ihtiyaçları, ihtiyaçların sınırları ve hangi paydaşın hangi ihtiyacı finanse ettiğini biliniyordu ve ekip gereksinimlerini bunları göz önüne alarak tasarlanmıştır.

Yazılım ekibi tasarlamak, kodlamak ve test için doğru ihtiyaçları ortaya koymuştur. Bunun için çeşitli yöntem ve uygulamalara ihtiyaç duyulmuştur. Sürecin bütün olarak ve iteratif yapıda ilerlemesi için Scrum ekibe yararlı bir yöntem olmuştur.

Uygulama olarak kullanıcı hikâyeleri kullanılmış ve ihtiyaçlara yönelik rehberlik sağlamıştır. Birçok Scrum ekibi de ihtiyaçların giderilmesi için kullanıcı hikâyelerini kullanmıştır. Bir yazılım sistemi kullanıcısına değer olarak işlevselliği açıklayan hikâyenin yazılı açıklaması, kullanıcı hikâyesidir. Önce tüm detaylar olmadan yüzeysel

yazılmıştır. Sık sık toplantılar yapılarak siber güvenlik ekibinin ihtiyaç detayları paydaşlar arasındaki tartışmalarla ele alınmıştır. Gereksinimlerin siber güvenlik ekibinin gerçek ihtiyaçlarını karşılamasına yönelik uygulanmıştır.

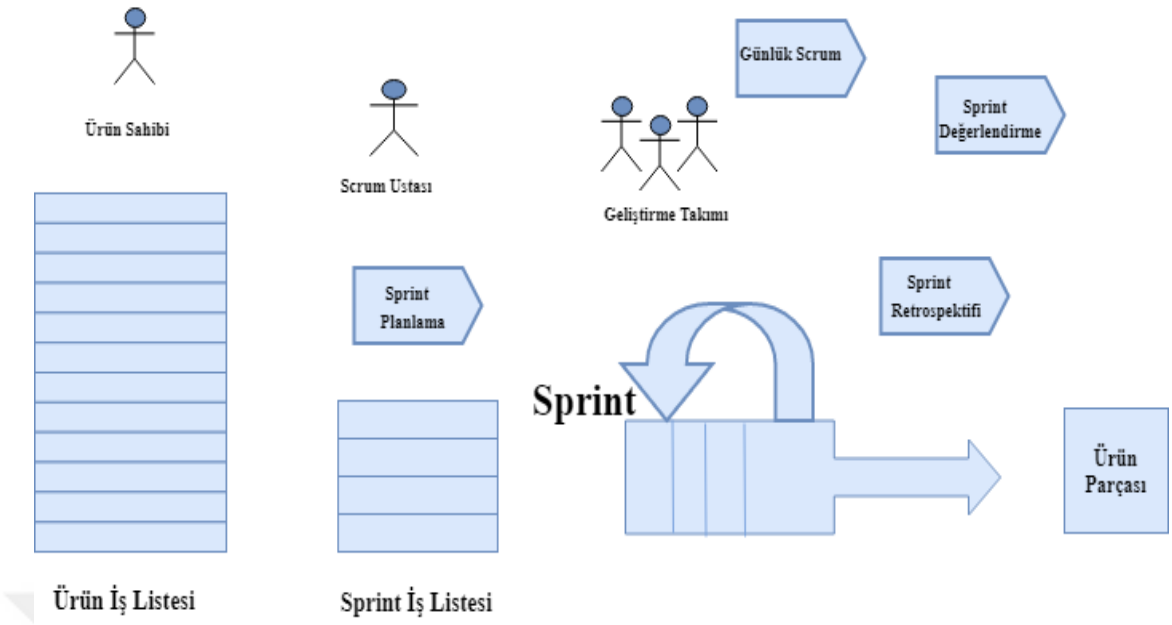
İhtiyaçlar göz önüne alındığında eğitimlerin her yerden ve her an erişilebilir olması için teknoloji olarak mobil yazılım uygundur. Ayrıca mobil yazılımın dünya pazarında en büyük oyuncu ve kullanıcı dostu olması da bu duruma katkı sağlamıştır. Yazılım ekibi bu ihtiyaçları göz önünde bulundurduğunda mobil yazılımı kullanmaya karar vermiştir.

Öz (Essence) ve Scrum, kullanıcı hikâyesi ve mobil yazılımın birlikte kullanılması iş birliği, mühendislik ve teknik uygulamaların bir karışımıdır. Scrum takım iş birlikleri hakkındadır. Kullanıcı hikâyeleri mühendislik uygulamalarıdır. Mobil yazılım ise son derece teknik bir uygulamadır.

Firmada bir dizi başarılı projenin ardından, tüm geliştirme ekiplerinin Scrum ve kullanıcı hikâyelerini kullanmaları yönetim tarafından zorunlu kılınmıştır. Örgütlerin belirli uygulama ve araçları zorunlu kılmasının sebebi genellikle eğitimi ve iletişimi basitleştirmektir. Belirli amaç doğrultusunda bir şeyi yapmaya yönelik tekrarlanabilir yaklaşım uygulamayı daha anlaşılır yapmaktadır.

4.2. Öz Çerçeve ve Scrum

Şekil 16 Scrum'un büyük resmine genel bakışı göstermektedir. Paydaşlar ve yazılım ekibinin birlikte nasıl çalıştığıyla ilgili açık bir rehberlik sağlar. Takımın yapması gereken tüm işler önce sıralı bir listeye (ürün iş listesi) yerleştirilmiştir (Scharff & Verma, 2010). Ürün iş listesi, listenin en üstünde en önemli görevleri bulundurur. Ürün birikimindekilere iş listesi ürünü denir (Redwine & Riddle , May 1985). Bir iş listesi ürünü, takımın kendilerini düzeltmek için yapabileceği bir şey veya düzeltmek zorunda oldukları kusurların bir parçası olabilir.



Şekil 16: Scrum'ın Büyük Resmi

Scrum'ın kalbi Sprint olup, genellikle bir ile dört hafta arasında sabit bir uzunluğa sahiptir ve bu süre boyunca takım, geliştirilecek ürünün potansiyel olarak taşınabilen bir artışının üretilmesini içeren belirli bir hedefe ulaştırır. Bir Sprint'te yapılacak iş listesi ürünleri, takımın yaklaşmakta olan Sprint'te çalışılacak en yüksek öncelikli iş listesi ürünleri kabul ettiği Sprint planlama faaliyeti ile seçilir (Flora & Chande, 2013). Bu iş listesi ürünleri, ürün iş listesinden Sprint planına taşınır. Bu aktivite, her bir sprintin ilk gününde, neyin teslim edilebileceğini ve Sprint zaman periyodunda kabul edilen şekilde nasıl teslim edilebileceğini belirlemek için birlikte çalışan tüm geliştirme ekibi tarafından yapılır. Sprint planlama faaliyetinin iki bölümü vardır. İlk kısımda ürün sahibi, Sprint'in ve Sprint hedefine ulaşması halinde ürün istek listesinin hedeflerini ekibine açıklar. Toplantının ikinci bölümünde ekip, elde edeceği iş listesi ürünlerini tahmin eder ve Sprint hedefine karar vermesini sağlar. Bu iş listesi ürünleri daha sonra Sprint iş listesine taşınır (Balzer , ve diğerleri, 2004).

Sprint sırasında her gün, ekip çalışmalarını senkronize etmek ve önümüzdeki 24 saat için bir plan oluşturmak üzere toplanır. Buna günlük Scrum denir ve 15 dakika ile sınırlıdır (Rahimian & Ramsin, 2008). Günlük Scrum'da, her takım üyesi son toplantıdan bu yana neler yaptığını, bugün ne yapmayı planladığını ve Sprint hedefini yerine getirmesini engelleyen durumların neler olduğunu açıklar. Günlük Scrum'da sorunlara çözüm üretilmez. Gerektiğinde sorunları daha detaylı anlamak için ayrı bir toplantı düzenlenir.

Sprint'in sonunda, ekip ürettikleri ürünü gözden geçirmek için bir Sprint değerlendirme etkinliği gerçekleştirmiştir. Bu gözden geçirmede paydaşlar, ürün istek listesinde yer alacak ürün iyileştirmelerini de belirlerler. Her sprintin sonunda, takım Sprint retrospektifi aktivitesine sahiptir (Stol & Fitzgerald , 2015). Sprint retrospektifi, geliştirme takımının bir sonraki sprintte uygulanacak çalışma tarzlarını iyileştirme konusunda hemfikir olduğu bir fırsattır.

Scrum'da ürün sahibi, Scrum ustası ve geliştirme takımı olmak üzere üç ana rol vardır (Elibol & Erol, 2014). Ürün sahibi, müşteri ve kullanıcılarla olan etkileşimine dayanarak ürün iş listesini beslemekten sorumludur. Ayrıca, iş listesi ürünlerinin önceliklendirilmesini de yönetir. Geliştirme takımı, her iş listesi ürünlerinin uygulanmasına yönelik çabanın tahmin edilmesinden sorumludur.

Scrum ustası rolü Scrum'a özgüdür. Scrum ustası, Scrum etkinliklerini kolaylaştıran ve ekip üyelerini Scrum etkinliklerini takip etme konusunda motive eden bir hizmetçi liderdir (Abrahamsson, ve diğerleri, 2017).

Az önce tanımladığımız Scrum aktiviteleri oldukça basitken, ekipler genellikle onları kendi durumlarına göre düzenlemiştir. Bu takımın uygulamanın çalışma şeklini kabul etmesini, hangi aktivitelerin ekip üyelerinden beklendiğini, faaliyetleri gerçekleştirirken sahip oldukları seçenekleri ve ne kadar ayrıntıya sahip olduğunu anlamalarına yardımcı olmuştur.

Scrum, ekiplerin iteratif gelişmeyi oldukça iş birlikçi bir şekilde yürütmelerine yardımcı olacak faaliyetlerden oluşan bir uygulama olarak temsil edilmiştir. Yani bir ürün sahipliği uygulaması, bir ürün iş listesi uygulaması, bir iteratif geliştirme uygulaması ve bir retrospektif uygulamadır (Sutherland, Viktorov, Blount, & Puntikov, 2007).

Farklı uygulama yazarlarının anahtar kavramlarını ifade etmenin farklı yolları olacaktır. Aynı şey için farklı terimler kullanıyor ve aynı terimin uygulamalarında farklı anlamları olabilir. Bazı yazarlar bu problemi asıl yazarın neyi geliştirdiğini yeniden tanımlayarak ele alırlar, ancak öz (Essence) gibi bir standart kullanmazlar, ancak kendi terimlerini oluştururlar (Vlaanderen, Jansen, Brinkkemper, & Jaspe, 2011). Bu, fikirlerini dünyaya katkıda bulunan insanlar arasında iş birliğine dayalı bir ortam oluşturmaz. Her yazarın aynı standart dili kullanıyor olması özün (Essence) ele aldığı konulardan biridir. Geliştiricilerin birçok uygulamayı öğrenmesi çok daha kolay hale getirmektedir.

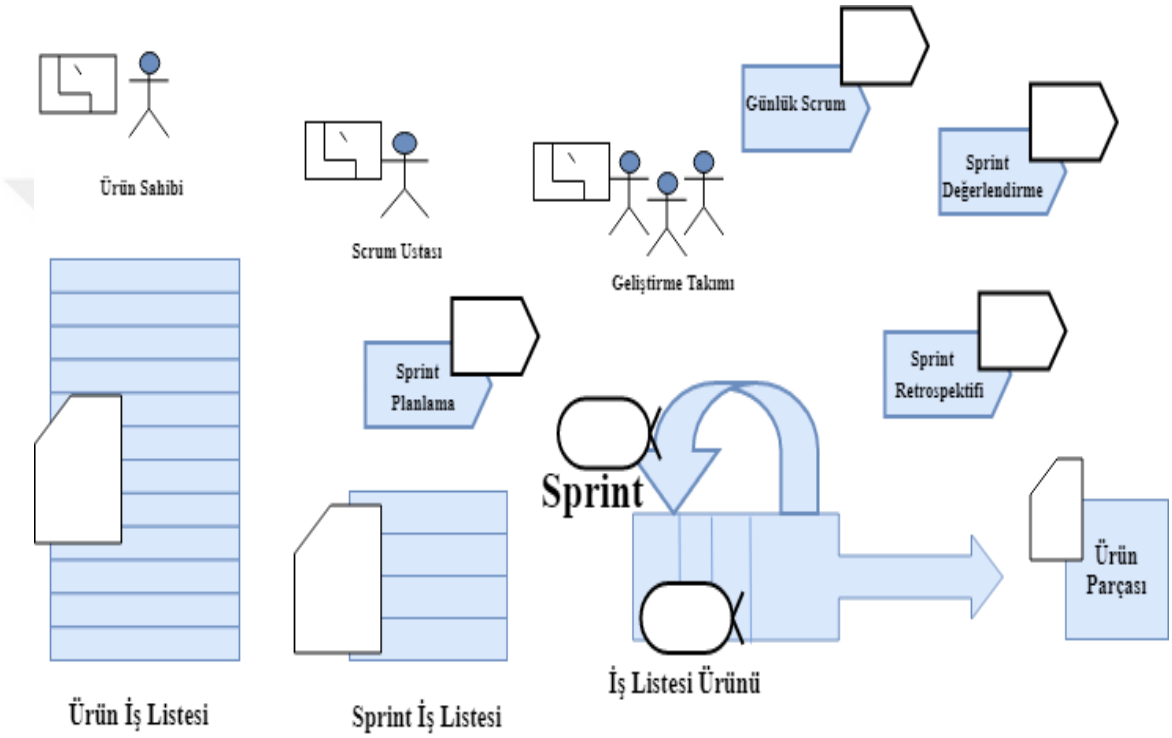
Şekil 17'de gördüğünüz şekil, öz (Essence) dili sembollerini kullanarak temsil edilen Scrum büyük resminin öğeleridir (Sjøberg, Dybâ, & Hannay, 2007).

Rol: Ürün sahibi, Scrum ustası, geliştirme takımı

Aktivite: Sprint planlama, günlük Scrum, Sprint değerlendirme, Sprint retrospektifi

Çalışma Ürünü: Ürün İş Listesi, Sprint İş Listesi, Ürün Parçası

Alfa: Sprint, İş Listesi Ürünü



Şekil 17: Scrum'ın Büyük Resminin Öz (Essence) ile Gösterilmesi

4.2.1. Scrum Bileşenleri

Tablo ***4***, Scrum Öz (Essence) uygulamasındaki öğelerin bir özetini sunar.

Tablo 4. Scrum Öz (Essence) Öğeleri

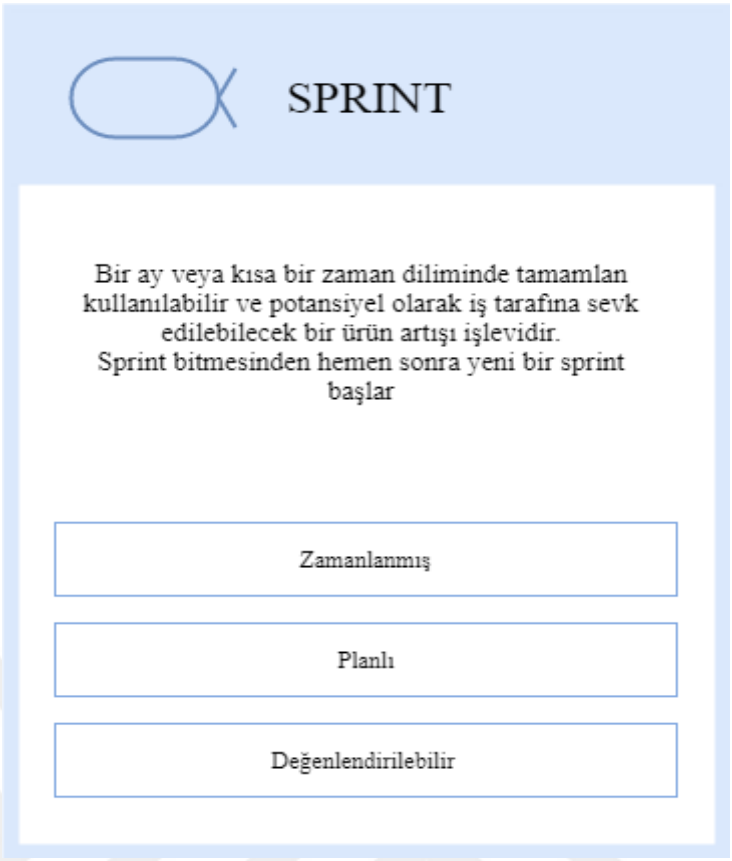
Eleman	Tip	Açıklama
Sprint	Alfa	Genellikle bir ile dört hafta sabit bir uzunluğa sahiptir ve bu süre boyunca takım, geliştirilecek ürünün potansiyel olarak taşınabilen bir artışının üretilmesini içeren belirli bir hedefe ulaşır. Önceki Sprint'in tamamlanmasından hemen sonra yeni bir Sprint başlar.
İş Listesi Ürünü	Alfa	Gelecekteki bir sürümde üründe yapılacak bir değişiklik
Sprint İş Listesi	Çalışma Ürünü	Sprint için seçilen ürün iş listesi maddeleri yanı sıra, arttırmanın gerçekleştirilmesi ve Sprint hedefinin gerçekleştirilmesi için bir plandır. Sprint iş listesi, geliştirme takımının Sprint hedefine ulaşmak için gerekli olduğunu belirttiği tüm çalışmaları görünür kılar.
Ürün İş Listesi	Çalışma Ürünü	Üründe ihtiyaç duyulabilecek her şeyin öncelikli bir listesi olmalıdır. Üründe yapılacak herhangi bir değişiklik ihtiyacın kaynağıdır.
Ürün Parçası	Çalışma Ürünü	Bir Sprint sırasında tamamlanan tüm ürün iş listesi maddelerinin ve önceki tüm Sprint'ler sırasında tamamlanan öğelerin toplamıdır. Artış "Tamamlandı" olmalı, bu da tarif ettiği yazılım sisteminin kullanılabilir olması ve tamamlanmış tanımını karşılaması gerektiği anlamına gelir. Bir ürün iş listesi öğesi veya bir artırma "Tamamlandı" olarak tanımlandığında, herkes "Tamam" ne anlama geldiğini anlamalıdır. Bu, Scrum takımına göre önemli ölçüde değişmekle birlikte, üyelerin şeffaflığının sağlanması için çalışmanın tamamlanmasının ne demek olduğu konusunda ortak bir anlayışa sahip olmaları gerekir.

Günlük Scrum	Aktivite	Ekip, ilerlemeyi değerlendirmek, faaliyeti senkronize etmek ve yoluna giren ve çözülmesi gereken eylemleri gündeme getirmek için her gün, aynı saat ve yerde toplanır. Toplantı genellikle 15 dakikadır.
Sprint Planlama	Aktivite	Sprint artışında nelerin teslim edilebileceğine ve anlaşmanın yapılması için gereken çalışmanın nasıl sağlanacağına karar verilir.
Sprint Değerlendirme	Aktivite	Sprint'in geri bildirimleri toplamak ve daha sonra ne yapılması gerektiğini tartışmak için çıktılarının zaman çizelgesiyle incelenmesi.
Sprint Retrospektifi	Aktivite	Tüm ekip çalışma şeklini yansıtmak için Sprint sonunda toplanır. İyileştirmeler belirlenir ve önceliklindirler ve eylemler kabul edilir. Bir sonraki retrospektifte, sonuçlar değerlendirilir.
Ürün Sahibi	Rol	Ürünün değerini ve geliştirme ekibinin çalışmalarını maksimize etmekten ürün sahibi sorumludur. Bunun nasıl yapılacağı organizasyonlar, Scrum takımları ve bireyler arasında büyük farklılıklar gösterebilir. Ürün sahibi, ürün iş listesi yönetmekle sorumlu tek kişidir. Ürün iş listesi yönetimi şunları içerir: • Amaç ve görevleri en iyi şekilde gerçekleştirmek için ürün iş listesinde yer alan öğeleri istemektedir. • Geliştirme Ekibinin gerçekleştirdiği işin değerini optimize etmektedir. • Ürün İş Listesi'nin herkes tarafından görülebilir, şeffaf ve net görünmesini sağlamak ve geliştirme takımının ne üzerinde çalışacağını göstermektedir. • Geliştirme ekibinin ürün iş listesindeki öğeleri gerekli seviyeye kadar anlamasını sağlamak.

Scrum Ustası	Rol	Scrum ustası, Scrum'ın anlaşılmasını ve kabul edilmesini sağlamaktan sorumludur. Scrum ustası, Scrum takımı için hizmetçi lideridir. Scrum ustasının yardım ettiği diğer şeyler arasında: • Scrum faaliyetleri • Engelleri kaldırın • Herkesin Scrum'ı anladığından emin olun • Scrum takımı, net ve özlü ürün iş listesindeki ihtiyacını anlıyor olması gerekmektedir
Geliştirme Takımı	Rol	Scrum takımları, tekrarlı ve artımlı ürünler sunarak, nasıl yaptıkları ve kendini geliştirme konusundaki geri bildirim fırsatlarını en üst düzeye çıkarır. En iyi geliştirme takımının büyüklüğü çevik kalabilecek kadar küçüktür ve bir Sprint içinde önemli çalışmaları tamamlamak için yeterince büyük olmalıdır.

4.2.1.1. Sprint

Bir Sprint belirtildiği gibi, bazı yararlı çalışmaların tamamlandığı bir zaman kutusudur. Scrum rehberinde tanımlanmış açık alfa durumları yoktur. Bununla birlikte, alfa durumları ayarlayan yeni takımlar için çok faydalı olduğu bilinir (Shaw, 2002). Takımların Sprint'lerine ve Sprint içerisindeki her bir aktiviteye hazırlanmak için ne yapmaları gerektiğini anlamalarına yardımcı olacaktır. Sprint alfa durumları aşağıdaki gibidir:



Şekil 18: Sprint Öz (Essence) Alfa Kartı

- Zamanlanmış: Sprintin başlangıç ve bitiş tarihleri zamanlanır ve tüm ekip üyeleri tarihlerin farkındadır.
- Planlı: Sprint kapsamında elde edilecek hedefler, Sprint kapsamındaki birikmiş işler öğeleri de dâhil olmak üzere kabul edilir.
- Değerlendirilebilir: Sprintin sonucu değerlendirilir. Bu, ürünün ve ekibin çalışma şeklinin değerlendirmesini sağlar.

Kontrol listeleri sağlandığında, daha az deneyimli bir geliştiriciye gereksiz hatalar yapılmaması için rehberlik etmiştir. Bu basit kontrol listeleri öğrenme için iyi bir kaynaktır. Bununla birlikte, fazla kuralcı olmak, uygulayıcının kendi başına düşünmesini ve kontrol listelerini kesin olarak görmesini engellemiştir. Her şeyde olduğu gibi nelerin gizli olacağı dengelemiştir.

4.2.1.2. İş Listesi Ürünü

İş listesi ürünü, gelecekteki bir sürümde yapılacak bir değişikliktir (Abrahamsson, Salo, Ronkainen, & Warsta, 2002).

Şekil 19: İş Listesi Ürünü Öz (Essence) Alfa Kartı

Alfa, Şekil 19'te tanımlanan aşağıdaki durumlara sahiptir:

- Yapılacaklar: İş listesi ürününün bir sonraki Sprint içinde tamamlanması gerektiğine karar verilir.
- Hazır: Ekip, iş listesi tamamlama konusunda nasıl bir karar vermeleri gerektiği konusunda anlaşmak için ürün sahibi ile çalışır.
- Yapıyor: Bu durumda, takım öğe üzerinde çalışır ve tamamlanmasını sağlar.
- Tamamlandı: İş listesi ürünü tamamlandı.

4.2.2. Scrum İş Ürünleri

Scrum uygulaması aşağıdaki iş ürünlerinden oluşur:

- Ürün İş Listesi
- Sprint İş Listesi
- Artış

4.2.2.1. Ürün İş Listesi

Ürün iş listesi, bir üründe ihtiyaç duyulabilecek her şeyin öncelikli olarak sıralanan bir listesidir. Üründe yapılacak herhangi bir değişiklik için tek gereksinim kaynağıdır (Kniberg, 2007).

Şekil 20: Ürün İş Listesi Öz (Essence) Alfa Kartı

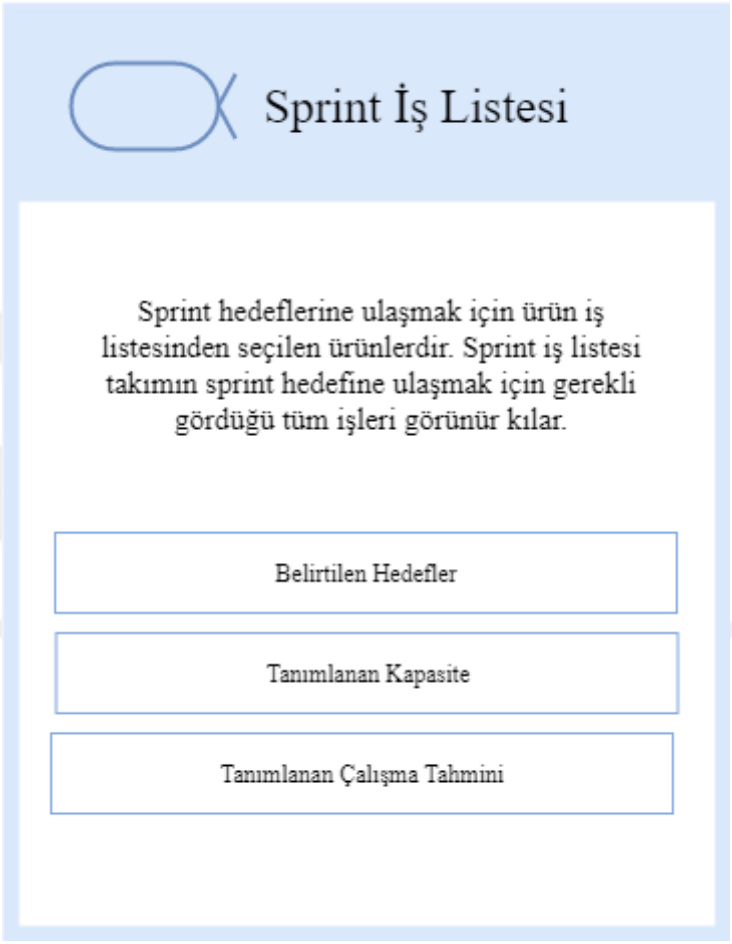
Ürün biriktirme listesindeki ayrıntı seviyeleri yalnızca bir tanedir:

• Sipariş Edilen Ögeler: İş listesi ürünleri, ürün iş listesinde toplanır. Bu, bir elektronik tablo şeklinde veya bazı biriktirme yönetimi aracı içinde olabilir. Önceliklerine göre sıralanırlar, böylece bir sonraki Sprint iş listesi için yüksek öncelikli olanlar seçilebilir (Jacobson , Meyer , & Soley , 2009).

Daha karmaşık çabalar için, daha fazla ayrıntı seviyesi olabilir. Örneğin, ekip ürün ihtiyaç listesinde yer almayı öncelik için gerekçeleri tanımlamak istenmiştir, böylece ekip üyeleri gereksiz tartışmalardan kaçınmıştır (Kitchenham , ve diğerleri).

4.2.2.2. Sprint İş Listesi

Bir Sprint koşu planı çalışması ürünüdür. Sprint için seçilen ürün iş listesi ürünün kümesidir. Ayrıca üzerinde anlaşmaya varılan Sprint hedefini gerçekleştiren bir artış sağlamak için bir plan içerir. Bir Sprint iş listesi, geliştirme ekibinin Sprint hedefine ulaşmak için gerekli olduğunu belirttiği tüm çalışmaları görünür kılmaktadır.



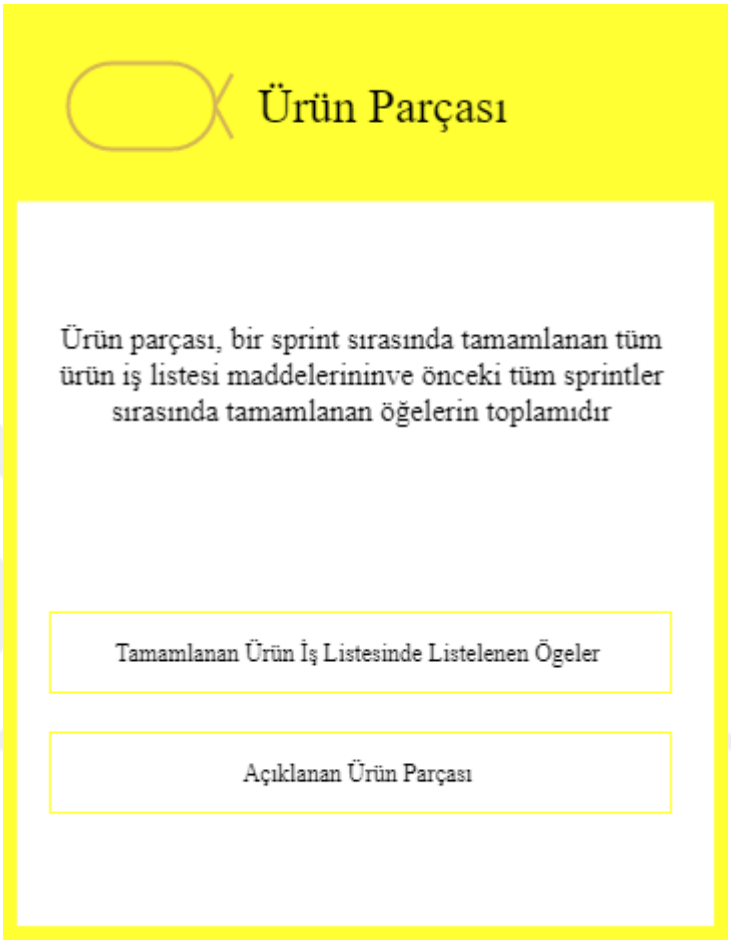
Şekil 21: Sprint İş Listesi Öz (Essence) Alfa Kartı

Sprint İş Listesi, aşağıdaki ayrıntı seviyelerini içerir (Malhotra , 2006):

- Belirtilen Hedefler; Sprint hedefi açıkça belirtilir ve ekip üyeleri için hedefi belirlemiştir.
- Tanımlanan Kapasite; Takımın yapabileceği iş miktarı tahmin edilir. Bu şekilde, ekip Sprint'te çok az veya çok çalışması olduğunu belirlemiştir.
- Tanımlanan Çalışma Tahmini; Ekip, Sprint içerisinde hangi iş listesi ürünlerinin tamamlanabileceğini ve Sprint içinde tamamlamayı umdukları hedef tarihlerini kabul etmişlerdir.

4.2.2.3. Ürün Parçası

Ürün parçası, bir Sprint sırasında tamamlanan tüm ürün iş listesi maddelerinin ve önceki tüm Sprint'ler sırasında tamamlanan öğelerin toplamıdır.



Şekil 22: Ürün Parçası Öz (Essence) Alfa Kartı

Artış iş ürünü aşağıdaki detay seviyelerine sahiptir:

- Tamamlanan Ürün İş Listesinde Listelenen Öğeler; Ürün parçasını oluşturan iş listesi ürünlerini açıkça listelenmiştir.
- Açıklanan Ürün Parçası; Ürün parçasını çalışabileceği ortamlar, bilinen sorunlar vb. gibi ürün parçası hakkında daha fazla bilgi verilmiştir. Ortama göre hangi tarayıcı sürümünü, hangi işletim sistemini kullanılacağı bu özel içerik ekip tarafından kararlaştırılmıştır (Klein & Myers , 1999).

4.2.3. Scrum'da Öz Çerçeve Rolleri

Scrum, ürün sahibi, Scrum ustası, geliştirme ekibi olmak üzere üç rol tanımlıdır. Rol, bir veya daha fazla kişinin kabul ettiği sorumlulukların bir listesidir. Öz (Essence), rolleri ve ekip organizasyonunu kalıplar olarak modellemenizi sağlamıştır (Schwaber, 2004).

Ürün Sahibinin sorumlulukları hakkında en önemli bilgileri içeren ürün sahibinin kartını göstermektedir. Kart, ürün sahibinin ürün iş listesi yönetiminden ve her bir ürünün ekip üyelerine açık olmasını sağlamak ve ürün listesini ekibinin görünür olmasını sağlamaktan sorumlu olduğunu göstermiştir. Kart ayrıca, ürün sahibinin Scrum ekibi tarafından üretilen değerin optimize edilmesini sağlamaktan sorumlu olduğunu göstermektedir, bu da Scrum ekibinin çalışması, Sprint hedefine ulaşmak için değer sağlamıştır (Beedle, Devos, Sharon, Schwaber, & Sutherland, 1999).

Scrum ustası olarak görev yapan kişi, Scrum faaliyetlerini yürütürken takıma koçluk eder. Ekip üyeleri, belirsiz iş listesi ürünleri gibi engellerle karşı karşıya kaldıklarında engellemeyi ortadan kaldırmak için çalışmıştır. Örnek olarak, yazılım takım liderinin ekibi lisans ücreti gerektiren bir kütüphane kullanma konusunda bir engelle karşı karşıya kaldı. Lisanslama çözülmezse, yazılım ekibinin yazılımın bir bölümünü yeniden yazması gerekmiştir. Scrum ustası sorunu şirketin yasal temsilcisi ve finans sorumlusu ile görüştü. Tartışma sonrasında, engeli giderilen lisans ücreti gerektirmeyen alternatif bir kütüphane kullanmayı kabul ettiler.

Geliştirme takımı, ürün sahibi ve Scrum ustası kendi kendini düzenliyorsa, bu takımın dışında kimsenin takıma her bir Sprint hedefine nasıl ulaşacağını söylemediği anlamına gelmiştir. Ekip, gerekli tüm işi başarmak için gereken uzmanlığa sahip kişileri içermiştir.

Bu kartlar, Scrum ekibi üyeleri tarafından, bir tahtaya yerleştirilerek veya kolaylıkla kabul edilebilecekleri, kendilerine sorumluluklarını kabul ettiklerini hatırlatan bir şekilde sürekli hatırlatmayı sağlayarak kullanılmıştır.

4.2.4. Scrum'ın Uygulanması

Yazılım takımının Scrum nasıl uygulanacağını tartıştığı pek çok soru vardır. Scrum etkinliklerinin net ve açık bir şekilde yapılmıştır. Net ve açık tanımlamalar ekip üyelerine hatırlatma görevi görür. Ayrıca uygulamaları tanımlamak için öz (Essence) kullanarak, uygulamalarımızda düzeltmemiz gereken boşluklar görüldü. Bunun nedeni, öz (Essence)

dilini kullanarak uygulamalar ifade edildiğinde hangi çekirdek alfanın ilerlemekte olduğunu ve hangilerinin etkilenmediğini görülmüştür.

Tablo 5. Scrum Uygulanışı

Scrum Elemanı	Siber Güvenlik Eğitim Portali ekibi bunu nasıl yaptı?
Ürün sahibi	Dijital dönüşüm ofisi ürün sahibi olmuştur.
Scrum Ustası	Yazılım takım lideri takım için Scrum Ustası olmuştur.
Sprint	Yazılım ekibi iki haftada bir iterasyon yapmıştır.
Sprint Planlama	Haftalık (Her pazartesi sabahı)
Günlük Scrum	Her sabah (salı, çarşamba, perşembe), pazartesi ve cuma günleri Sprint planlama, değerlendirme ve retrospektifi yapılmıştır.
Sprint Değerlendirme	Haftalık (Her cuma sabahı)
Sprint Retrospektif	Haftalık (Sprint değerlendirmesinden sonra cuma sabahı)

Yazılım takım liderinin, Scrum ustası rolünde, Scrum aktivitelerini gözlemleyerek kontrol etmiştir.

Scrum şu aktiviteleri içerir:

• Sprint Planlama,

• Günlük Scrum,

• Sprint Değerlendirme

• Sprint Retrospektifi.

4.2.4.1. Sprint Planlama

Yazılım ekibi pazartesi sabahı Sprint planlama ile Scrum'u öz (Essence) üstünde kullanmaya başlamıştır. Ürün iş listesindeki hangi öncelikli işlerin geçerli Sprint iş listesine girmesi gerektiğine karar verilmiştir.

Ancak bu sadece ürün iş listesinden bir ürün seçmek ve onları Sprint iş listesine taşımaktan ibaret değildir. Aynı zamanda, aşağıdaki sorular çözümlenmiştir:

- "Bu Sprint için seçilen işler uygun şekilde hazırlandı mı?" Düzgün bir şekilde hazırlanmış olması işlerin bir sonraki Sprint içerisinde mevcut olan sürede takım tarafından tamamlanabilecek kadar küçük işlere bölündüğü anlamına gelmiştir. Bu, ekibin işin tamamlanmasını tahmin etmesi için seçilen her bir birikmiş işler öğesi hakkında bilgi sahibi olması gerektiği anlamına gelmiştir.
- İlgili diğer bir soru şudur: "Geliştiriciler, her iş listesi ürününü tamamlamanın ne kadar çaba harcanacağını tahmin edebilirler mi?" Olmazsa, bu ekibin ürün sahibine geri dönmesi ve daha fazla bilgi edinmesi veya sprintte çalışılmaya hazır olmayan bu ürünü reddedilmesidir.
- Diğer bir soru şudur: "Ekip, bu sprintte tamamlanması için önerilen işlere onay verebileceklerine karar verirken kapasitelerini değerlendirdi mi?" Bu soru özellikle Scrum uygulaması içindeki Sprint planlama faaliyetinden bulunmuştur. Her ekip üyesinin çaba için ne kadar zaman harcayacaklarını düşünmeleri gerektiği ve ekibin Sprint'te tamamlamayı taahhüt ettiği ürünleri bulmanın yeterli zamanı olduğuna inanmaları gerektiği anlamına gelmiştir.

Yukarıdaki sorular, ekiplerin neden uygulamaya ihtiyaç duyduklarına ya da duymadıklarına dair bir mantık sunmuştur. Takımlar aşağıdaki gibi diğer soruları da sormuştur:

- "Ekip üyelerimiz önceki soruları soracak kadar deneyimli mi?" Ekip yeterli deneyim sahibi değilse, diğer ekip üyelerinin yardım edip edemeyeceğine veya ekip dışından yardım isteyip istemediğine karar vermeleri gerekir. Ekip üyelerinin çoğu veya tamamı Scrum'da yeniyse, ilk birkaç sprintinde onlara rehberlik etmesi için bir koça da ihtiyaç duyulabilir.

4.2.4.2. Günlük Scrum

Günlük Scrum, Scrum ekiplerinin her gün yürüttüğü basit bir etkinliktir. Toplantıyı 15 dakika tutmak gibi birkaç rehber ilkesi vardır, sadece geliştiriciler konuşur ve üç ana soruyu cevaplamaya odaklanır (Sutherland, Viktorov, Blount, & Puntikov, 2007):

- Son günlük hatalardan beri ne yaptım
- Daha sonra ne yapmayı planlıyorum
- Hangi engellerle karşılaşıyorum

Günlük işler yaparken yazılım ekibi her gün aynı saatte ve aynı yerde buluşmuştur. Toplantıyı sadece 15 dakika tutmuşlardır, yazılım takım lideri, ekip üyelerini olabildiğince kısa tutmaya, sadece üç soruyu cevaplamaya odaklamıştır. Her ekip üyesi soruları cevaplarken, diğerleri dinlemiştir ve dâhil olmaları gereken bir durum oluşursa toplantı sorasında sadece ilgili kişilerle mini bir görüşme ayarlanmıştır. Bu şekilde tüm ekip üyelerinin toplantıya katılmaları önlenip harcanan zaman en aza indirilmiştir. Birilerinin işlerini yapmalarını engelleyen bir engel tespit ettiğinde, Scrum ustası sorunu çözmeye çalışmıştır, ancak bazen diğer ekip üyeleri de sorunu nasıl çözeceklerini bildiklerinde öne çıkıp yardım etmişlerdir. Çözümü günlük görüşmelerde tartışmamışlardır, sorunu çözmek için yer alması gerekmeyen diğer ekip üyelerinin zamanını almamak için bu yöntem seçilmiştir. Günlük toplantılar, ekibin çalışmayı kontrol altında tutmasına yardımcı olmuştur.

4.2.4.3. Sprint Değerlendirme

Sprint değerlendirme, ürünün paydaşlar tarafından gözden geçirilmesi sağlanmıştır. Bu değerlendirmenin odağı, bir önceki Sprint planlama oturumunda ne ürettiklerini temel alarak ekibin ne ürettiğini göstermişlerdir. Ürün toplayıcısı olarak Scrum ustası bu toplantıya önderlik etmiştir. Toplantının odaklanmasını sağlamak için iş tarafı, her bir değerlendirmeye önceki Sprint boyunca ekibin üzerinde çalışmayı kabul ettiği Sprint iş listesi öğelerini gözden geçirerek başlamıştır. Ardından her bir öğe gösterilmiştir ve Scrum ustası daha sonra katılımcılara, taahhüt edilen her bir Sprint birikim öğesinin gerçekleştirildiğini kabul edip etmediklerini sormuştur. Sadece Sprint boyunca tamamlanan Sprint planı ürünleri gösterilmiştir. Eğer bir şey kısmen tamamlanmışsa, gösterimi tamamen gösterilebilecek bir sonraki Sprint'e kadar ertelenmiştir. Taahhüt

edilen Sprint planı maddeleri tamamlanmadığında, ürün sahibi Sprint değerlendirmesi sırasında bunu açıklamıştır ve eksik öğeyi ele alma planını açıklamıştır. Sprint değerlendirme ayrıca ekip üyelerinin paydaşlardan değerli geri bildirim almaları için bir fırsattır sunmuştur. Sprint değerlendirmesi sonunda ürün sahibi, değerlendirmenin sonucunda ve ekibin değerlendirmeden sonra gelecek Sprint değerlendirmesini ele almak için yaptığı eylemleri özetlemiştir.

Sprint değerlendirme aşağıda ki maddeleri içermektedir:

• Katılımcılar, ürün sahibi tarafından davet edilen kilit paydaşlar ve Scrum takımıdır.

• Ürün Sahibi, ürün iş listesi kalemlerinden hangilerinin "Bitti" olup olmadığını açıklar.

• Geliştirme Takımı, Sprint boyunca neyin iyi gittiğini, hangi sorunlarla karşılaştığını ve bu sorunları nasıl çözdüğünü tartışır.

4.2.4.4. Sprint Retrospektifi

Sprint retrospektifinin amacı, ekibin çabalarını nasıl yaptıklarını gözden geçirme yöntemine göre incelemelerini ve bir sonraki Sprint yapılacak yöntemlerde iyileştirmeler yapmayı kabul etmelerini sağlamışlardı. Bu iyileştirmelerin sonuçları net olmuştur.

Sprint retrospektifi aşağıdaki maddeleri içermiştir:

• Son sprintin insanlar, ilişkiler, süreç ve araçlar bakımından nasıl geçtiğini gözlemlenmiştir.

• İyi giden noktaları ve muhtemel iyileştirme alanlarını tespit edip sıralanmıştır.

• Scrum takımının iş yapış tarzını iyileştirecek bir plan oluşturulmuştur.

Sprint retrospektifin açık bir uygulama haline getirilmesindeki değer, geliştirme ekibinizden neyin iyi çalıştığını ve neyin işe yaramadığını öğrenmek ve ekibin bir sonraki Sprint sırasında yöntemlerini geliştirmek için farklı şekilde neler yapabilecekleri konusunda anlaşma sağlanmıştır. Sprint retrospektif ayrıca, takımın önerdiği iyileştirmelerle nasıl ilerleyeceğine dair kararlar veren takımlara rehberlik etmede yardımcı olmuştur.

Sprint retrospektifi, öz (Essence) dilinde, Scrum gibi daha büyük bir uygulama içindeki bir faaliyet veya bir uygulamanın kendisi olarak temsil etmiştir. Örneğin, birçok kuruluş

retrospektiflerini ayrı bir uygulama olarak ortaya koyuyor ve ekiplerin bir sonraki Sprint içinde uygulanabilecek pratik iyileştirmeleri seçmelerine yardımcı olacak uygulama kriterlerine yer veriyor. Bu kriterler: küçük, ölçülebilir, ulaşılabilir, alakalı ve test edilebilir olmuştur. Bu niteliklerin ekipler tarafından, iyileştirmeler için bir sonraki Sprint içerisinde uygulanabilecekleri değerlendirmelerine yardımcı olmak amacıyla kullanılması sağlanmıştır. Örneğin, bir ekip üyesi önerilen bir iyileştirme ile ilgili olarak aşağıdaki soruları sorabilir:

• Bu geliştirme, ekibin bir sonraki sprinti uygulayabilmesi için küçük mü?

• Geliştirmeyi başardığımızda objektif olarak değerlendirmek için neyi ölçeceğimizi biliyor muyuz?

• Bu iyileşmeye ulaşılabilir mi? Tüm ekip üyeleri iyileştirmeyi uygulamak için gerekli becerilere sahip değilse, o zaman elde edilemeyebilir.

• Geliştirme alakalı mı? Ekiplerin hedeflerine ulaşmalarına yardımcı olmayan, ekiplerin çıkarması gerekli birçok geliştirme olabilir.

• Geliştirme test edilebilir mi? İşi başarıp tamamlandı diyebilmek iyileştirmeyi nasıl test edeceğimizi bilmemiz gerekir.

Öz (Essence) kendi başına deneyimli ekiplere ya da uygulamaların olduğu basit bir iş üzerinde çalışmakta yardımcı olmuştur. Ancak işlere daha karmaşık veya daha önce hiç birlikte çalışmayan daha fazla insan dâhil olduğunda, o zaman ekip üyelerinin yapması beklenen faaliyetler ve üretilen iş ürünlerinde beklenen detayın derecesi konusunda yanlış iletişim riski artmıştır. Ayrıca farklı ekip üyelerinin ilerlemelerini değerlendirme riski ve paydaşlarla iletişim riskleri oluşturmuştur.

Scrum projede iki önemli şekilde odaklanmanıza yardımcı olur:

1. Uygulamanın en önemli kısımlarını çağırmak.

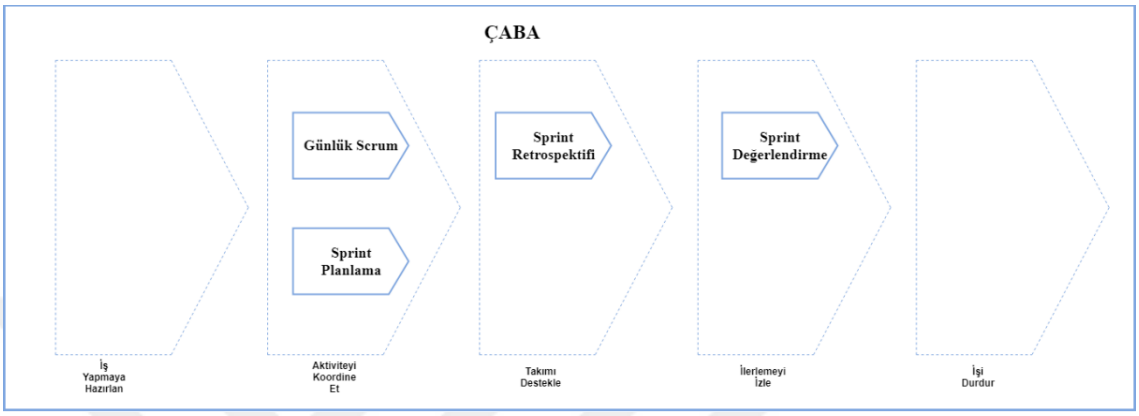
2. Bu önemli kısımların ne olduğunu açıkça ortaya koymak.

4.2.5. Scrum Uygulamasının Görselleştirilmesi

Ekipler gerekli uygulamaları kullanmayı kabul ettiklerinde, kendi uygulamalarında boşlukların nerede olduğunu daha net görebilmişlerdir. Bunun nedeni, referans olarak öz (Essence) çekirdeğine sahip olmalarıdır. Bu yeni belirli uygulamaların nerede gerekli

olabileceğini ya da mevcut uygulamalarında iyileştirmeler yapmaları gerektiğini görmelerini sağlamıştır.

Ne demek istediğimizi göstermek için, Şekil 23'de öz (Essence) çekirdeğinin ilgilendiği ilgi alanındaki tüm faaliyet alanlarını göstermektedir. Ayrıca Scrum etkinlikleri tarafından doldurulan faaliyet alanlarını gösterir.



Şekil 23: Çaba Bakış Açısından Scrum

Görüldüğü gibi Scrum çalışma alanında dört faaliyet alanından üçünde etkili olmuştur. Özellikle "çalışmaya hazırlan" ve "işi durdur" faaliyetlerinde etkinliği yoktur. Ekibi kurma, işi hazırlama, birlikte çalışma konularında anlaşmak ve işi bitirmek önemlidir. Diğerleri Scrum uygulama kapsamı dışındadır (Paasivaara, Durasiewicz, & Lassenius, 2009).

4.3. Kullanıcı Hikâyesi

Yazılım ekibinin çalışmalarında kullanıcı hikâyeleri uygulamaya nasıl başladığını açıklamıştır. Kullanıcı hikâyeleri, ekibin kullanıcılarının bakış açısından ne yaptıklarının değerini düşünmesini, araştırmasını ve anlamasını sağlama avantajına sahip olmuşlardır. Kullanıcı hikâyesi uygulaması, özellikle küçük ekipler için popüler bir uygulamadır (Cohn, 2003).

Bir kullanıcı hikâyesi, inşa ettiğimiz sistemdeki bir sistemin kullanıcısı için değerli olan işlevselliği tanımlamıştır. Kullanıcı hikâyeleri, 1990'lı ve önceki yıllarda başarılı olduğu kanıtlanmış bir yaklaşıma dayanmakta ve sistem kullanıcısı ile geliştirici arasında gayrı resmi tartışmalar yapılmıştır. Bir kullanıcı hikâyesi, hikâyeyi tartışmak için kullanılan ve hikâyeyi tamamlamak için neyin gerekli olduğunu bildirmeye yardımcı olacak testlerle

birlikte kullanılan yazılı bir açıklama içermiştir. Tamamen, kullanıcının ihtiyacını karşılayacağı kabul edilen her şeyi kastedilmiştir (Azar, Smith, & Cordes, January/February 2007). Kullanıcı hikâyeleri fikri, bir işlevsellik parçasının ne olduğunu ve bu rolün nasıl bir fayda sağladığını açıklığa kavuşturmak için tartışmayı kolaylaştırmanın bir yolunu sağlamıştır. Bir kullanıcı hikâyesi genellikle 5 x 3 indeks kartlarda çok özlü bir formatta veya şablonda aşağıdaki gibi yakalanmıştır:

Bir <rol veya kullanıcı türü> olarak <sistemin burada olmasını istediğiniz işlevi burada listelemek> istiyorum, böylece <burada ulaşmak istediğiniz hedefi listeleyin>.

Bir örnek olabilir: "Bir banka müşterisi olarak, doğrudan işverenimin maaş çekimi elektronik olarak bana gönderebilmesi için doğrudan para yatırma kabiliyetine sahip olmak istiyorum." Bu şablon, "Kim", "Ne" ve "Neden" sorularını içerir ve aşağıda ki soruları cevaplar:

- Kim değeri alacak?
- Neyi başarmamız gerekiyor?
- Biz neden bunu yapıyoruz?

Kullanıcı hikâyeleri genellikle "3 x 5" kartlarda yakalanır. Kullanıcı hikâye kartları, elbette bir kullanıcının ihtiyaç duyduğu her şeyi sağlamamıştır. Ekibimiz, kullanıcılarla sohbet etme ihtiyacını hatırlatmak için kullanılan yer tutuculardır. Konuşmanın amacı, detayları çözmektir. Bu ek ayrıntılar karta eklenebilir veya ek hikâyeler aracılığıyla yakalanabilir. Kullanıcı hikâyelerinin birincil değeri, geliştirme ekibi ile kullanıcı arasında bir konuşma gerçekleştirmişlerdir (Beck & Fowler, Planning Extreme Programming , 2000).

İyi bir kullanıcı hikâyesi yazmak için aşağıdaki ölçütleri kullanmak faydalı olmuştur. İyi bir kullanıcı hikâyesi:

- Bağımsız
- Tartışılabilir
- Değerli
- Tahmin edilebilir
- Küçük
- Test edilebilir olmalıdır.

İlk olarak, kullanıcı hikâyeleri birbirinden bağımsız olmuştur, böylece her biri ayrı olarak geliştirilmiştir. İkincisi, kullanıcı hikâyelerini kullanırken konuşmayı teşvik etmenin nedeninin en azından bir kısmı, kullanıcı ile geliştiriciler arasındaki görüşmeleri desteklemiştir. Tartışılabilir durumu teşvik etmek için kullanıcı hikâyelerinin müzakere edilmelerini sağlayacak şekilde yazılmıştır. Tartışılabilir anlayışı, bağlılığı teşvik etmiştir. Üçüncüsü, bir kullanıcı hikâyesi kullanıcı için değerli olmuştur. Konuşma, ekip üyelerinin bir gereksinimin gerçek amacını ve her hikâyenin kullanıcıya getirdiği değeri anlamalarına yardımcı olmuştur. Her hikâyenin kullanıcıya değer vermesini sağlamanın bir yolu, kullanıcının hikâyeyi yazmasına gerçekten ilgi göstermiştir. Dördüncü olarak, bir kullanıcı hikâyesi tahmin edilebilir olmuştur. Ekip üyeleri ve kullanıcılar, kullanıcı hikâyeleri üzerinde birlikte çalıştıkları için amaç geliştiricinin hikâyeyi tamamlamak için gereken iş çabasını tahmin etmesini sağlamak için ortaya çıkacak yeterli ayrıntılara sahip olmuştur. Beşinci olarak, kullanıcı hikâyeleri küçük olmalıdır. Genellikle hikâyeler ilk yazıldığında, belirli bir yinelemeye sığmayacak kadar büyüktür ve bu nedenle daha küçük hikâyelere bölünmüştür. Bir yinelemeye sığmayacak kadar büyük olan bu büyük hikâyelere genellikle epik denir (Beck, Extreme programming Explained Embraced Change , 2000). Geliştiriciler ve kullanıcılar arasında yapılan konuşmalarda, ihtiyaç duyulan daha küçük hikâyeler ortaya çıkmıştır. Altıncı, iyi bir hikâye için akılda tutulması gereken önemli bir kriter, tamamlandığı zaman test edilebilir olmasıdır. İlk olarak testleri yazmak, hikâyenin test edilebilir olmasını sağlamaya yardımcı olur ve hem kullanıcının hem de geliştiricinin hikâyeyi tamamlamanın ne demek olduğu konusunda hem fikir olmasını sağlanmıştır.

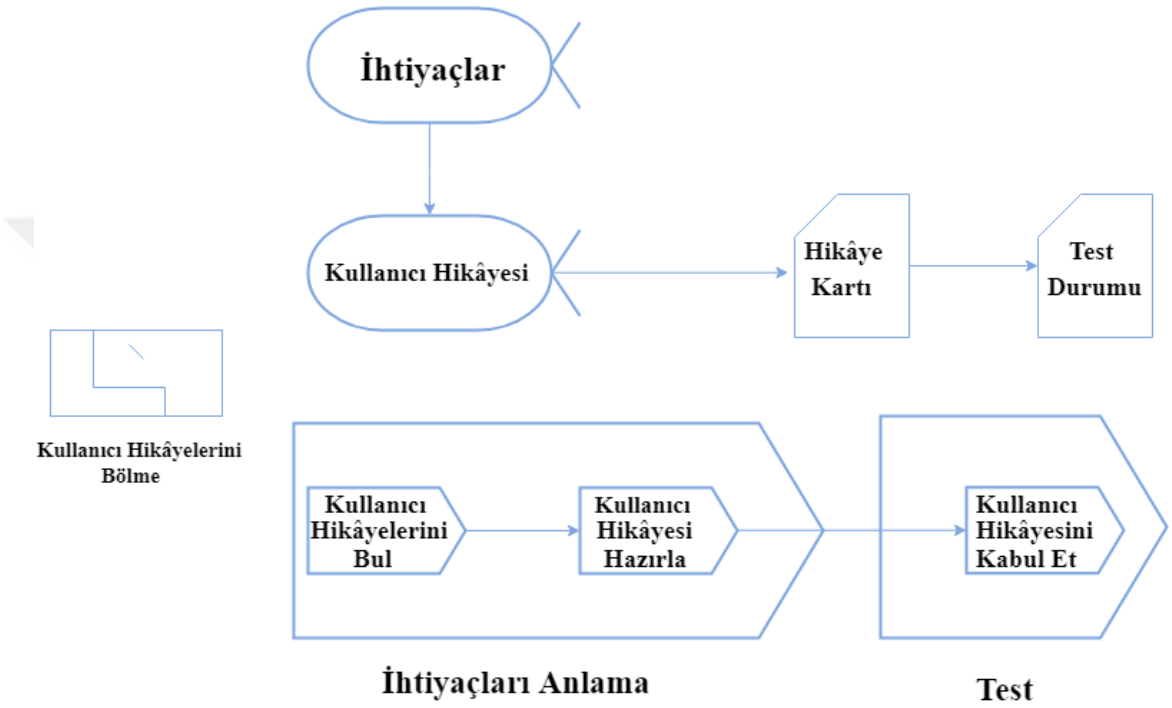
Kullanıcı hikâyeleri kullanılırken sıklıkla yeni başlayanlar için ortaya çıkan soru şudur:

Fakat neden bir kullanıcı hikâyesindeki “öyleyse” maddesine ihtiyacımız var?

“Öyle” yan tümcesinin bir kullanıcı hikâyesini ifade etme biçimine eklenmesinin sebeplerinden biri, geliştiricilerin kullanıcının nihai amacını anlamalarıdır. Bu evrimsel ihtiyaçların gelişimini desteklemeye yardımcı olmuştur. Evrimsel ihtiyaçların geliştirilmesi ile kullanıcının mevcut seçenekleri ve ihtiyaçları hakkında daha fazla şey öğrendikçe gereksinimler gelişebilmişlerdir. Bu aynı zamanda geliştiricinin alternatif çözümler sağlamada ki seçeneklerini açık tutmuştur.

4.3.1. Kullanıcı Hikâyelerinin Öz (Essence) ile Tanımlanması

Tıpkı Scrum'daki yaptığımız gibi, kullanıcı hikâyesi uygulamasının, kullanıcı hikâyelerini ve kullanıcı hikâyelerini çevreleyen çeşitli öğelerin birbirleriyle nasıl ilişkili olduğunu anlayarak takıma nasıl rehberlik ettiği konusunda çok açık olunması sağlanmıştır. Şekil 24, öz (Essence) dilini kullanan kullanıcı hikâyesi uygulamasını ifade etmiştir.



Şekil 24: Öz (Essence) Göre Kullanıcı Hikâyesi

Bu uygulamanın karmaşık gereksinimleri alt alfalara (kullanıcı hikâyesi alfası) ayrıştırmanın bir yolu olduğu açıktır. Her kullanıcı hikâyesi, bir hikâye kartı ile tanımlanmıştır ve bir test durumu ile doğrulanmıştır. Kullanıcı hikâyesi uygulamasının çeşitli aktiviteleri vardır:

I. Kullanıcı hikâyelerini bul,
II. Kullanıcı hikâyesi hazırla,
III. Kullanıcı hikâyesini kabul et.

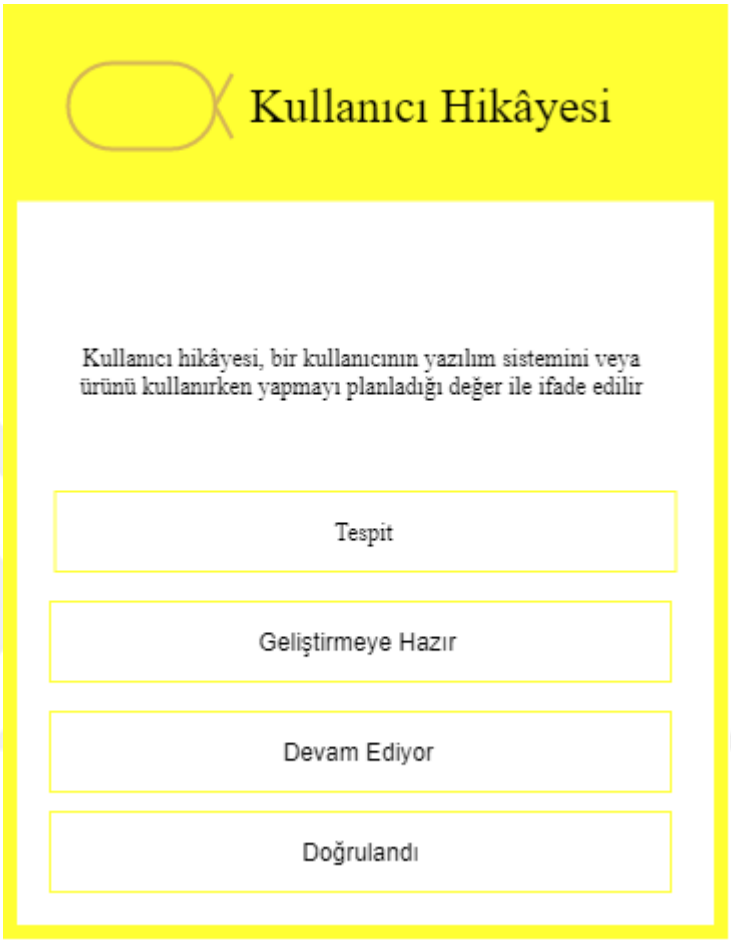
Kullanıcı hikâyesinin öz (Essence) göre Tablo 6 tablosuna detaylandırılmıştır.

Tablo 6. Kullanıcı Hikâyesinin Öz (Essence) Öğeleri

Elaman	Tip	Açıklama
Kullanıcı Hikâyesi	Alfa	Bir yazılım sisteminin yapabileceği, sistemin bir kullanıcısına sağlayacağı değer ile ifade edilebilecek bir şey.
Hikâye Kartı	Çalışma Ürünü	Bir kullanıcı hikâyesinin temel ayrıntılarını yakalayan bir dizin kart veya eşdeğeri.
Test Durumu	Çalışma Ürünü	Bir kullanıcı hikâyesinin tam ve doğru bir şekilde uygulanıp uygulanmadığını değerlendirmek için test girişlerini ve beklenen sonuçları tanımlar.
Kullanıcı Hikâyelerini Bul	Aktivite	Bir yazılım sisteminin yapabileceği değerli şeyleri tanımlanmıştır. Bunları hikâye kartlarında basit ve kısa başlık açıklamaları olarak yakalanmıştır.
Kullanıcı Hikâyesi Hazırla	Aktivite	Kabul kriterleri ve test durumlarının anlaşılmasını ve iyileştirilmesini sağlamak amacıyla kullanıcılarla tartışılarak geliştirilmek üzere bir kullanıcı hikâyesi hazırlanmıştır.
Kullanıcı Hikâyesini Kabul Et	Aktivite	Kullanıcı hikâyesi uygulaması, müşteri/kullanıcı temsilcisi tarafından kabul edilip kabul edilinceye kadar müşteri / kullanıcı ile yakın iş birliği içinde geliştirilmiştir.
Kullanıcı Hikâyelerini Bölme	Model	Küçük şeyler daha hızlı yapılmıştır. Çevik gelişimde, daha büyük hikâyeleri daha küçük hikâyelere bölerek kullanıcı hikâyelerinin boyutunu azaltmak için sürekli bir çaba olmuştur.

4.3.2. Kullanıcı Hikâyesi Alfası

Kullanıcı hikâyesi, bir kullanıcının yazılım sistemini veya ürünü kullanırken yapmayı planladığı değer ile ifade edilmiştir.



Şekil 25: Kullanıcı Hikâyesi Öz (Essence) Alfa Kartı

Bir kullanıcı hikâyesi genellikle aşağıdaki durumlarda ilerler:

• Tespit: Kullanıcı hikâyesi, net bir şekilde ifade edilen değer ile tanımlanır. Takımın ürün birikiminde yer almaktadır.

• Geliştirmeye Hazır: Ekip, kullanıcı hikâyesinin gerekliliklerini yerine getirmede rol oynayan üyelerin net olduğunu belirten kullanıcı hikâyesinin ayrıntılarını tartışmıştır. Bu, kullanıcı ara yüzleri, uygulama detayları ve benzeri hakkında detaylar içermiştir.

• Devam Ediyor: Bu durumda, takım kullanıcı hikâyesini yerine getirmek için çalışmıştır.

• Doğrulandı: Kullanıcı hikâyesi, ürün sahibi olan üye gibi nitelikli bir kullanıcı temsilcisi yani iş birimi tarafından doğrulanmıştır.

4.3.3. Hikâye Kartı

Hikâye kartı, bir kullanıcı hikâyesinin istekleri yakalayan bir indeks kartı veya eş değeridir. Normalde “Bir <rol> olarak, <işlev> 'i istiyorum, böylece <hedef>” biçiminde ifade edilir.

Şekil 26: Hikâye Öz (Essence) Alfa Kartı

Bir kullanıcı hikâyesi farklı detay seviyelerinde ifade edilebilir:

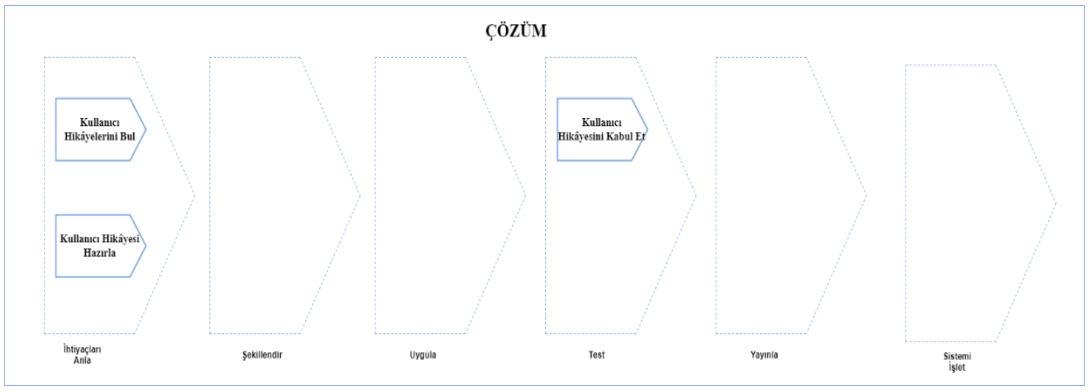
- Değer İfade Edildi: Kullanıcı hikâyesinin değeri yukarıda açıklanan ortak formatı kullanmak gibi açıkça ifade edilir.
- Listelenen Kabul Kriterleri: Kullanıcı hikâyesinin yerine getirilmesi için kabul kriterleri açıkça ifade edilmiştir.
- Yakalanan Konuşma: Takımın kullanıcı hikâyesi hakkındaki tartışmaları, takımın kullanıcı hikâyesinin ayrıntılarının arkasındaki mantığı daha net anlayabilmesi için yakalanmıştır. Bu genellikle bir çeşit tartışma yönetimi aracı gerektirmiştir.

Geliştirme ekibimizin, projeleri üzerinde kullanıcı hikâyesini denemeye karar vermelerini sağlayan iki temel zorluk olmuştur. İlk olarak, yazılım ekibinin üyeleri bazen kendileri geliştirdikleri sistemin amacını merak ederken bulmuşlardır. Bu genellikle Scrum ustası ile canlandırılmış tartışmalarla sonuçlanmıştır. Bu nedenle sadece ürün listesindeki işler maddelerini sıralamak yerine, iş tarafı bir kullanıcı hikâyesinde ürün listesinde işleri geliştirmeye biraz zaman harcayarak ortaya çıkan gereksinimlerin ekibin geliştirdikleri sistemin amacını daha iyi anlamalarına yardımcı olacağını kabul etmiştir. Bu aynı zamanda, dijital dönüşüm ekibi gibi sistemi paydaşlarıyla tartışırken iş tarafıyla yardımcı olmuştur.

4.3.4. Kullanıcı Hikâyesinin Görselleştirilmesi

Kullanıcı hikâyesindeki üç etkinlik projede yalnızca iki etkinlik alanını kapsamıştır. Özellikle, "Sistemi Şekillendir" etkinlik alanını kapsayan hiçbir aktivite yoktur. Gereksinimlerin yapısı ve yazılım sisteminin yapısı da dâhil olmak üzere çözüm alanının yapısıyla ilgilenen faaliyet alanıdır.

Ne demek istediğimizi göstermek için, Şekil 27: Çözüm Bakış Açısından Kullanıcı *Hikâyesi* öz (Essence) çekirdeğinin ilgilendiği ilgi alanındaki tüm faaliyet alanlarını göstermektedir. Ayrıca kullanıcı hikâyesinin etkinlikleri tarafından doldurulan faaliyet alanlarını gösterir.



Şekil 27: Çözüm Bakış Açısından Kullanıcı Hikâyesi

Ekip, öz (Essence) aracılığıyla uygulamalarına bakarak, mevcut uygulamalarının güçlü yönlerini ve zayıf yönlerini görebilmektedir. Örneğin, takım küçük olduğunda ve sadece birkaç gereksinimi olduğunda, kullanıcı hikâyesi formatı ekip için iyi sonuç vermiştir.

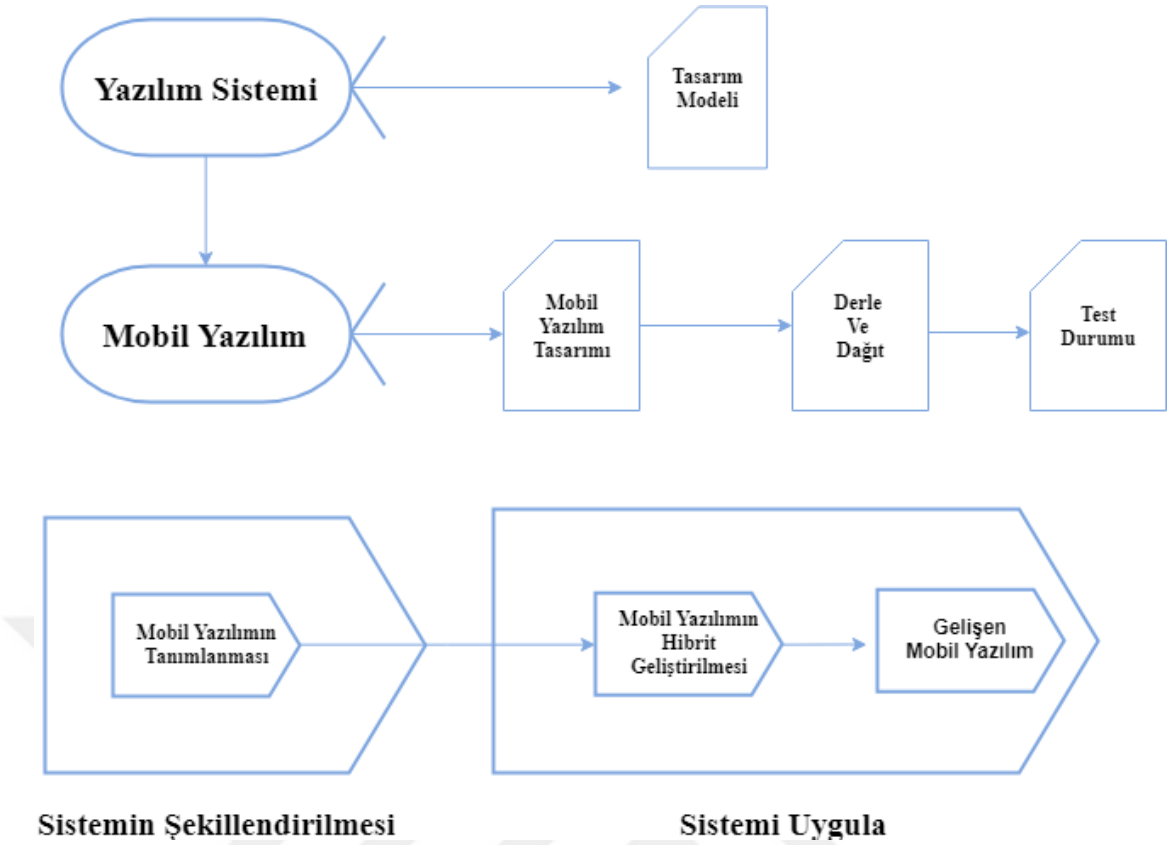
BÖLÜM 5

5. ÖÇ TABANLI MOBİL YAZILIMIN GELİŞTİRİLMESİ

Mobil platformlarda çalışan ve iyi tanımlanmış ara yüzlere sahip olan kullanıcıların mobil platform ile erişebileceği yazılımdır. Bu bölümde, yazılım geliştirme ekibinin geliştirilmesinde mobil yazılımı nasıl kullandıklarından bahsedilmiştir. Mobil yazılım geliştirmek, yazılım sistemini mobil platform aracılığı ile kullanıcıya sunmaktadır. Mobil cihazlar popülerliği ve insanların mobil yazılıma ilgisi her geçen gün artmaktadır. Kullanıcı ara yüz tarafını mobil yazılım ile gerçekleştirerek daha kullanışlı ve daha erişilebilir bir platform sunmak istenmektedir.

Uygulamamızı tüm ortamlara uyumlu (hibrit) geliştirerek IOS, ANDROİD ve diğer ortamlar tarafından çalışacak şekilde gerçekleştirmek planlanmaktadır. Hedef bir kez geliştirdikten sonra bütün ortamlara entegre edilmesini sağlamaktır.

Essence göre mobil yazılımı Şekil 28 görsel olarak tanımlayabiliriz. Mobil yazılım çalışma ürünlerimiz tasarım, derleme ve dağıtma ve test durumudur. Mobil yazılımın tanımlanması, mobil yazılımın tüm ortamlara uyumlu (hibrit) geliştirilmesi ve gelişen bir mobil yazılımın oluşturulması öz (Essence) göre aktivitelerimizdir.



Şekil 28: Mobil Yazılımın Öz (Essence) ile Tanımlanması

Mobil yazılım, ekibin tanımlanmasından dağıtımına kadar ilerleyeceği bir alfadır. Bir alfa ilerlemeyi ve sağlığı ölçmek için kullanılabilecek durumlara sahip bir elementtir. Bir yazılım sistemi mobil yazılımın alt fazını oluşturduğundan yazılım sisteminin ilerlemesi ve sağlığı her bir mobil yazılıma bağlıdır. Tasarım modeli, yazılım sistemini koddan daha yüksek bir soyutlama seviyesinde tanımlayan bir iş ürünüdür. Bir tasarım modeli, kodun anlaşılmasını basitleştirmiştir. Bu, normal olarak tasarım modelindeki öğelerin koddaki öğelere izlenebilir olduğu anlamına gelmiştir. Mobil yazılım tasarımını, ara yüzlerinden davranışına ve iç tasarımına kadar tanımlayan bir çalışma ürünüdür. Mobil yazılım test durumları, mobil yazılım için test durumlarını yakalayan bir iş ürünüdür.

Yukarıdaki alfaların ve iş ürünlerinin nasıl ilerleyeceği konusunda açık rehberlik sağlayan bir dizi faaliyet vardır.

Öz (Essence) göre alfalarımız, çalışma ürünlerin ve aktivitelerin açıklaması Tablo 7'de açıklanmıştır.

Tablo 7. Mobil Yazılımın Öz (Essence) Öğeleri

Eleman	Tip	Açıklama
Mobil Yazılım	Alfa	Mobil platformlarda çalışan ve iyi tanımlanmış ara yüzlere sahip olan kullanıcıların mobil platform ile erişebileceği yazılım.
Mobil Yazılım Tasarımı	Çalışma ürünü	Mobil Yazılım sorumluluklarının ve bu sorumlulukları nasıl yerine getirdiğinin bir açıklaması.
Derle ve Dağıt	Çalışma ürünü	Mobil yazılım ilgili mobil platforma göre derlenmesi ve ilgili platforma dağıtılması(yayınlanması).
Test Durumu	Çalışma ürünü	Mobil Yazılım davranışını doğrulamak için yapılan testler.
Mobil Yazılım Tanımlanması	Aktivite	Gereksinimleri uygulamak ve sorumluluklarını belirlemek için yazılım sisteminde gereken uygun bir mobil yazılımın yapılması.
Mobil Yazılımın Tüm Ortamlara Uyumlu (Hybrid) Geliştirilmesi	Aktivite	Tüm mobil ortamlarda kullanılacak mobil yazılımın geliştirilmesi.
Gelişen Mobil Yazılım	Aktivite	Mobil yazılımın işlevselliğini aşamalı olarak geliştirilmesi, hızlı kodlanması ve test edilmesi.

Mobil yazılımın belirlenmesi, bu etkinlik, mobil yazılımın amacını ve kapsamını tanımlar ve açıklar. Hikâyemizde hatırladığın gibi, yazılım takım lideri ve ekibi gerekli eğitimler motoru üzerinde çalışmıştır

Mobil yazılımı geliştir, yazılımın yeni işlevler eklemek için devam eden bir faaliyetidir. Bir yazılımın geliştirilmesi, her bir bölümün amacını yerine getirmesini sağlamak için kodun değiştirilmesini ve test edilmesini içerir.

Şekil 29: Mobil Yazılım Öz (Essence) Alfa Kartı

Mobil yazılımın oluşturma ve sunma aşağıdaki durumları çözer:

- Tanımlandı; İlk önce, mobil yazılımın kapsamı açık olmuştur. Ekip üyeleri, mobil yazılımın hangi işlevlerden sorumlu olduğunu anlamıştır.
- Kolay Erişim; Mobil yazılımın kullanmanın asıl amacı, personelin her yerde rahatça ve kolayca erişmesini sağmaktadır. Bu şekilde ekip üyeleri, üretim ortamında ki bütün mobil ortamları rahatça değiştirmiştir. Ve bulut tabanlı bir çalışma yapısı olduğu için içeriği hızlıca değiştirerek ihtiyaç duyulan eğitime göre şekillendirmiştir.
- Tüm Mobil Ortamlar; Mobil yazılım hızlı bir şekilde güncellenip tüm mobil ortamlara dağıtılabilir. Bu, diğer mobil ortamlarla ile iş birliği içinde entegre edilip test edilebilmesi için minimum bir dizi gereksinimin karşılanması ile başlar.

- Tamamlandı; Ekip daha sonra, mobil yazılımın gerekli tüm ara yüzlerini yerine getirmek için mikro hizmet geliştirecektir. Aynı zamanda, ekip mobil yazılımın yapısını geliştirmiştir, böylece genişletilebilir, yani yazılım sisteminde yeni işlevsellik, tüm mobil ortamlara yansır.

Geliştirmesi yapılacak olan ürünle ilgili olarak yapılan toplantılar sonrasında ürün listesi Tablo 8'de verilmiştir. Tez kısıtlılıkları arasında 3 Sprint içinde yapılacak ve birinci fazda yapılacak bitmesi planlanan Sprint iş listesi oluşturulmuştur. Sistemin temel gereksinimleri karışılması beklenen iş ürünleri oluşturularak Sprint'ler başladığında gerekli olan maddeler listelenmiştir.

Tablo 8. Ürün İş Listesi ve Sprint İş Listesine Uygunluk

Madde	Ürün İş Listesi Maddeleri	Sprint İş Listesine Uygunluk
1	Dâhili roller grubunu ayarlanması	✅
2	Dâhili kullanıcı grubunu ayarlanması	✅
3	Rol ekranlarının yazılım ihtiyacının tanımlanması	✅
4	Kullanıcı ekranlarının yazılım ihtiyacının tanımlanması	✅
5	Roller ve kullanıcıların güncellenmesi ve silinmesi	✅
6	Kullanıcıların sisteme girişi	✅
7	Kullanıcının kendi bilgilerini güncellendiği ekranların geliştirilmesi	❌
8	Şirket sistemi üzerinden kullanıcı bilgilerin direk alınması	❌
9	Çoklu dil desteğinin eklenmesi	❌

10	Dâhili kullanıcı grubunu ayarlanması	✓
11	Gerekli eğitim listelerin hazırlanması	✓
12	Gerekli eğitim içeriklerinin hazırlanması	✓
13	Eğitimlerin listelenmesi	✓
14	Eğitimlerin video olarak gösterilmesi	✓
15	Eğitimlerin doküman olarak gösterilmesi	✓
16	Eğitimlerin yönetici tarafından girilmesi	✗
17	Yeni eğitim eklendiğinde kullanıcılara bildirim gitmesi	✗
18	Çevrimdışı videolu eğitimlerin izlenmesi	✗
19	Tanıtım ekranların yapılması	✗
20	Test sorularının ayarlanması	✓
21	Test başarı oranlarının ayarlanması	✓
22	Eğitimlerin testlerinin listelenmesi	✓
23	Test soruların sorulması	✓
24	Test sorularının bitmesi sonrası sonucun hesaplanması	✓
25	Testlerin direk yönetici tarafından girilmesi	✗
26	Testlerin başarı oranlarının ara yüz tarafından değiştirilmesi	✗
27	Yöneticiler için raporlama ekranları	✗

28	Personeller için raporlama ekranları	✗
29	Personel memnuniyet anket sorularının ayarlanması	✗
30	Personel memnuniyet anket ekranlarının yapılması	✗

Sistemin çalışması ve birinci fazda sonuç olarak beklenen üç temel hedef üzerinde durularak bunlarda Sprint hedefleri olarak planlanmıştır.

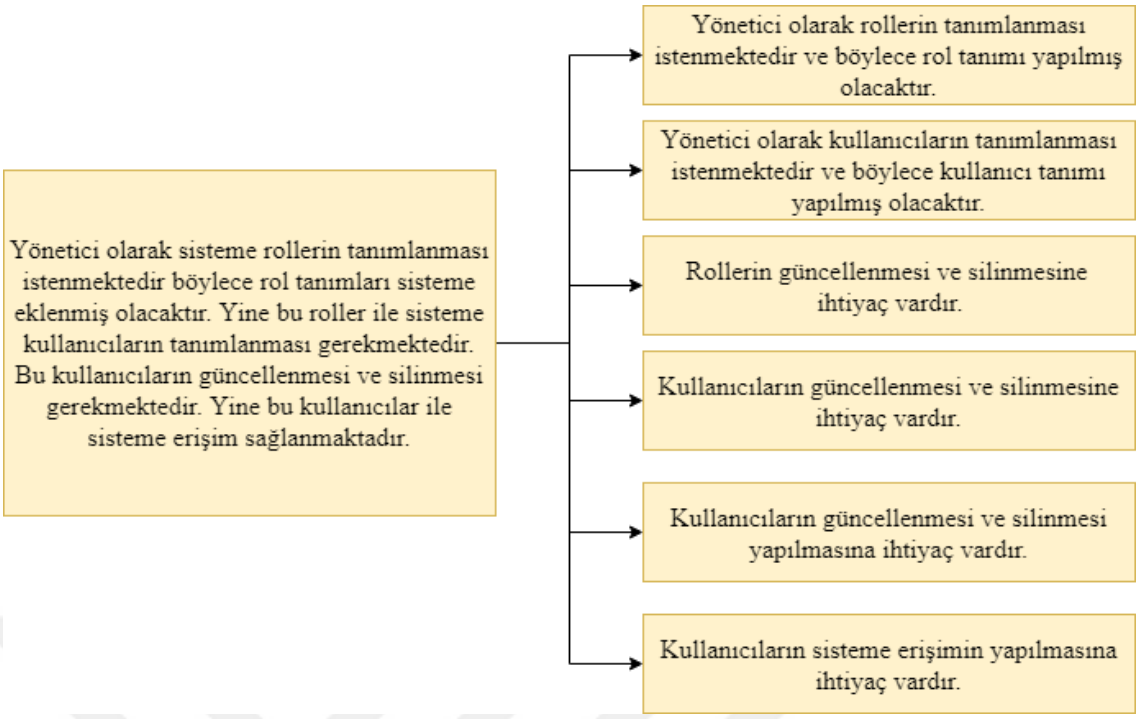
- Hedef 1: Personellerin sisteme girişi için kullanıcı tanımlanması ve rollerin eklenmesi gerekmektedir. Bu bilgiler ile sisteme giriş işleminin sağlanmıştır
- Hedef 2: Sistem personeller için gerekli eğitimleri sunmuştur.
- Hedef 3: Personeller aldıkları eğitimler sonrası değerlendirme testine girer. Bu testler sonuçlarında başarılı veya başarısız olurlar.

5.1. Sprint 1

Personellerin sisteme girişi için kullanıcı tanımlanması ve rollerin eklenmesi gerekmiştir. Bu bilgiler ile sisteme giriş işleminin sağlanması gerekmektedir. Bu bizim için sprint bitiminde sprint hedefine ulaştırmak için uğraş vereceğimiz konu olmuştur. Bu hedef için Sprint planlaması yapılıp kullanıcı hikayelerinin tanımlanması yapılmıştır.

5.1.1. Sprint Planlama

Sprintin ihtiyaç öğesinin iş tarafına göre yapılması için kullanıcı hikâyelerinin çıkarılması gerekmiştir. Şekil 30 üzerinde epik üzerinden 6 tane ihtiyaç öğesi tanımlanmıştır.



Şekil 30: Sprint 1 Hedefinin Kullanıcı Hikâyesi

Bu kullanıcı hikâyelerine göre Sprint planlama etkinliğinde Scrum ustasının yapmış olduğu toplantıda ihtiyaçları maddelemek, önceliklendirmek ve tahmini zaman verilmesi gerçekleştirilmiştir.

Tablo ***9***'da bu durum özetlenmiştir.

Tablo 9. Sprint 1 İş Listesi Ürünleri ve Tahmini Zaman Planlaması

Madde	İş Listesi Ürünü	Rol	Öncelik	Tahmini Zaman
1	Dâhili roller grubunu ayarlanması	İş Tarafı	Yüksek	2 hafta
2	Dâhili kullanıcı grubunu ayarlanması	İş Tarafı	Yüksek	2 hafta
3	Rol ekranlarının yazılım ihtiyacının tanımlanması	Geliştirme Takımı	Yüksek	1 hafta
4	Kullanıcı ekranlarının yazılım ihtiyacının tanımlanması	Geliştirme Takımı	Yüksek	1 hafta
5	Roller ve kullanıcıların güncellenmesi ve silinmesi	Geliştirme Takımı	Orta	1 hafta
6	Kullanıcıların sisteme girişi	Geliştirme Takımı	Yüksek	1 hafta

Yapılması gereken maddelerin çıkarılmasından sonra Scrum tarafından çalışan ekipler konular üzerinde çalışmaya başlamıştır. İş tarafı ihtiyaç olacak rollerin ve kullanıcıların çıkarılması için siber güvenlik birimi ile çalışmayayapılmıştır.

Yazılım ekibi ihtiyaçları öncelik durumlarına göre mobil yazılımın geliştirme etikliğini gerçekleştirmesini yapmıştır.

5.1.2. Sprint Süreci

Sprint uygulamaya artık ekip başlamıştır ve Sprint için önemli olan konulardan birisi günlük Scrum'ların yapılmasıdır. Geliştirme ekibinin dört tane gündem maddesi vardır. Ekip bu günlük Scrum toplantılarında 3 soruya cevap aramıştır.

Ekibin Sprint'i tamamlamasına yardım etmek için dün ne yaptın?

Ekibin Sprint'i tamamlamasına yardım etmek için bugün ne yapacaksın?

Ekibin önünde hangi engeller mevcut?

Ekip toplantılarının on beş dakikadan fazla tutulmaması gerekmektedir. Scrum ustası bu konuda bu işin denetleyicisi ve yürütücüsü olarak görev almıştır.

Geliştirme ekibinde geliştiriciler hangi işleri yapacaklarını ortaya koyarak Sprint tamamlanmasını için toplamda ki dört gündem maddesi ile başlamıştır. Önünde ki engeller neler olduğu konusunda bir iletişim geçmiştir. Burada örneğin dâhili kullanıcı ekibi için yazılım geliştirme ekibinin bu listelere ihtiyacı vardı burada Scrum ustası devreye girerek şimdilik örnek veriler ile çalışması gerektiğini yeni veriler geldiğinde veri tabanında bu veriler güncellenerek bu sorunun gidereceğini ileterek işe engel olan durumun önüne geçmiş oldu.

Sprint yürütmek için diğer önemli adım her görev için ne kadar çaba kaldığının analizin çıkartılması yapılmıştır. Dört gündem maddesi için dört haftalık Sprint planın örneği

Tablo ***10*** gösterilmiştir.

Tablo 10. Sprint 1 İş Listesinin Günlere Göre Kalan Eforları

İşler	1. Gün	2. Gün	3. Gün	4. Gün	…..	28. Gün
Rol ekranlarının yazılım ihtiyacının tanımlanması	7	6	6	5		0
Kullanıcı ekranlarının yazılım ihtiyacının tanımlanması	7	5	5	5		0
Roller ve kullanıcıların güncellenmesi ve silinmesi	7	7	6	6		0
Kullanıcıların sisteme girişi	7	6	6	6		0

Dört iş listesi ürünlerinin geliştirme aşamaları şu şekilde gelişmiştir.

1.Rol ekranlarının yazılım ihtiyacının tanımlanması: Öncelikle rollerin olacağı tablolar oluşturulmuştur. Bu ilgili tablo mobil yazılım ile gerekli olan bağlantısı sağlanmıştır. Ve mobil ara yüz tanımlanarak bu veri tabanı tablosu ile ilişkilendirilmiştir. Ara yüzden girişi sağlanan rol bilgileri ilgili rol veri tabanı tablolarına kayıt edilmiştir.

Burada ki temel ihtiyaç sistemi giriş sağlayacak personellerin yetkilerini ayarlamaktır. Örneğin yönetici yetkisine sahip bir personelin yeni bir rol ve personel ekleyebilmesine imkân sağlamaktır. Ayrıca personellerin birimlerine bağlı olarak eğitimlerin değiştirilmesine olanak sağlanmıştır. Rollere göre personellerin mobil ara yüzde karışılacağı ya da yapacağı işlemler değişecektir.

2. Kullanıcı ekranlarının yazılım ihtiyacının tanımlanması: Kullanıcıların veri tabanı tabloları oluşturulmuştur. Bu ilgili tablo mobil yazılım ile gerekli olan bağlantısı sağlanmıştır. Ve mobil ara yüz tanımlanarak bu veri tabanı tablosu ile ilişkilendirilmiştir.

Ara yüzden girişi yapılan kullanıcı bilgileri ilgili kullanıcılar veri tabanı tablolarına kayıt edilmiştir.

Bir yazılım sistemi için olmazsa olmazlar arasında kullanıcılar bulunmaktadır. Personellerin sistemde birer kullanıcı olarak tanımlanıp sisteme giriş sağlamasına olanak sağlayacaktır.

3. Roller ve kullanıcıların güncellenmesi ve silinmesi: Veri tabanına kayıt edilen rol ve kullanıcı bilgilerin güncellenmesi ve silinmesi gerekmektedir. Güncelleme ve silinmesi ilgili mobil ekranların geliştirilmesi yapılmıştır. Ardından bu ara yüz güncellenen veya silinen verilerin ilgili bağlantılar ile veri tabanından silinmesi işlemi gerçekleştirilmiştir.

Şirket içeresinde personeller sürekli olarak değişmekte veya işten ayrılmaktadırlar. Bu gibi durumlarda mobil yazılımında buna uyum sağlayarak rol ve personel bilgilerin güncel tutulması gerekmiştir. Mevcut rol tanımın silinmesi durumunda sistemden bu kayıtın silinmesi gerekmiştir. İlgili rol tanımı değişime uğrayabilir o zaman yine rol bilgisinin değiştirilmesi gerekmektedir. Personel tarafında ise personel işten ayrıldığında mobil yazılım üzerinden silinmesi gerekmektedir. Personelin birimi, telefon gibi bir güncelleme yapması durumunda sistem üzerinden kullanıcı bilgisi değiştirilmesi gerekmektedir.

4. Kullanıcıların sisteme girişi: Eklenen rol ve kullanıcı bilgileri ile sisteme giriş kontrolü sağlanmaktadır. Mobil ekranların geliştirmesinden sonra kullanıcı ve şifre bilgisi yazılım tarafından veri tabanı katmanı olup olmadığı kontrol edildikten sonra sisteme giriş başarılı veya başarısız olarak sonuçlandırılmaktadır.

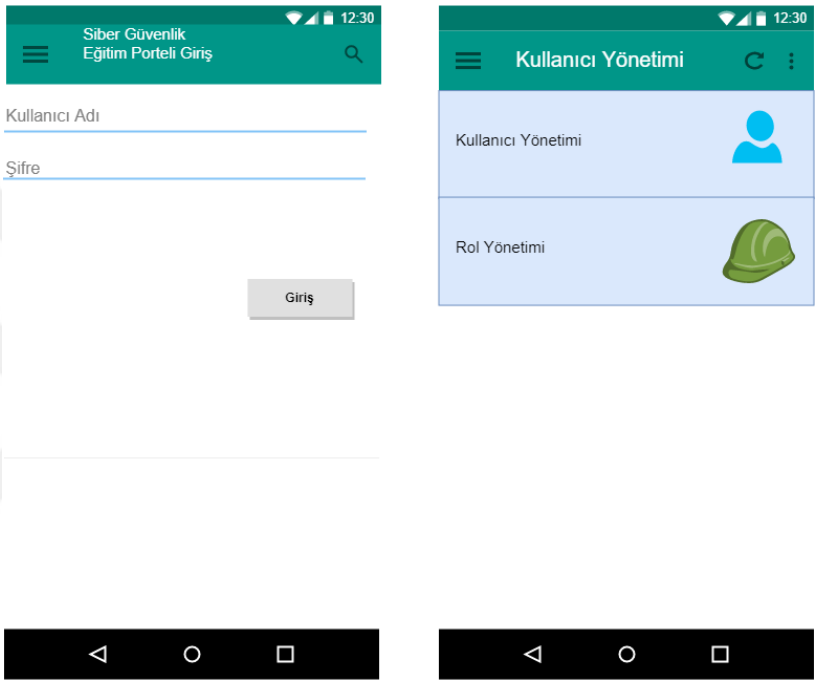
Mobil yazılım bulut tabanlı çalışan bir uygulama olmasından dolayı sistemin güvenliği ve yetki kontrolü için sisteme giriş sayfası ihtiyacı bulunmaktadır. Kullanıcıya verilen bir kullanıcı adı ve şifre ile kullanıcı sisteme giriş sağlayarak eğitim listelerini görmektedir.

5.1.3. Sprint Değerlendirme

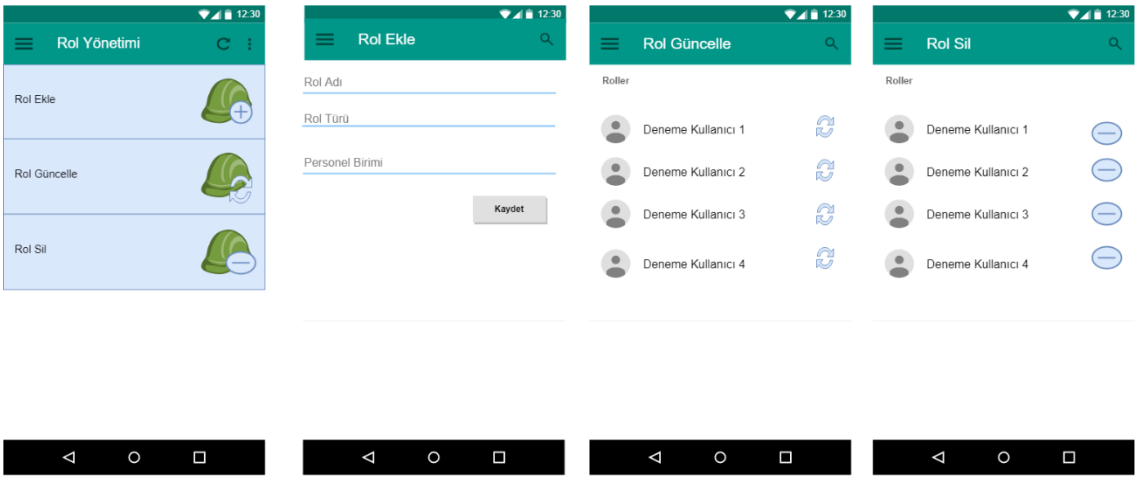
Sprint süresinin bitmesinin ardından, ürün paydaşları ile Sprint değerlendirme toplantısı yapılarak Sprint hedefine ulaştığı değerlendirilecektir. İş tarafı rollerin ve kullanıcıların çıkarılması konusunda siber güvenlik ekibi ile çalışmış ve rollerin tanımı tamamlanmıştır. Fakat dâhili kullanıcıların oluşturulması konusunda siber güvenlik birimi ile bazı kullanıcılar ve yetkiler konusunda tam bir netlik oluşmamıştır. Bundan dolayı rol maddesi

tamamlandı olarak bildirilse de dâhili kullanıcıların tamamlanması bir sonraki Sprint ele alınarak değerlendirilecektir.

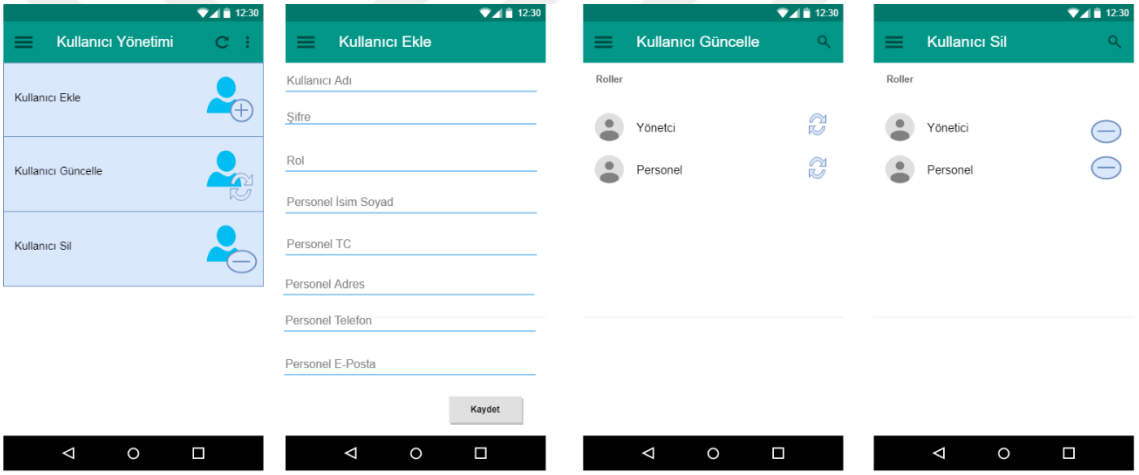
Geliştirme ekibi kendi maddelerini tanımlayarak mobil yazılım ihtiyaçlarını ortaya koymuştur. Geliştirme ekibi Şekil 31, Şekil 32 ve Şekil 33 ara yüzleri gösterilen geliştirmeleri paydaşlara gösterilerek işin tamamlandı olarak bitmesini talep etmiştir. Paydaşlar tarafından bu işler test edilerek bir sonra ki Sprint işler tamamladıysa çekilmesinin gerçekleşmesi planlanmıştır.



Şekil 31: Giriş ve Kullanıcı Ekranları



Şekil 32: Rol Ekranları



Şekil 33: Kullanıcı Ekranları

5.1.4. Sprint Retrospektifi

Sprint bitmesinden sonra çabanın doğru harcandığını ve Sprint iyileştirme için Scrum ustası eşliğinde bir toplantı yapılarak sprintin eksik veya iyi olan durumlarının ortaya çıkarılması hedeflenmiştir.

İyi özellikler olarak deneme rolleri ve kullanıcı ile geliştirme ekibi hızlı bir şekilde geliştirmeleri tamamlamıştır.

Geliştirme ekibi bu Sprint için çok fazla çaba harcayarak ve fazla mesai yaparak Sprint hedefine ulaşmıştır. Tahmini zamanlama konusunda biraz daha esnek olunması gerektiğine karar vermiştir. Yazılım geliştirirken istemeyen durumlar

gerçekleşebilmekte, bu durumlar geliştirmeyi yavaşlamakta ve hedefe ulaşmak için daha fazla çaba harcanması gerekmektedir. Örnek olarak şirket planlı bir internet kesintisinin olması yazılım ekibinin işlerini yavaşlamasına sebep olmuştur.

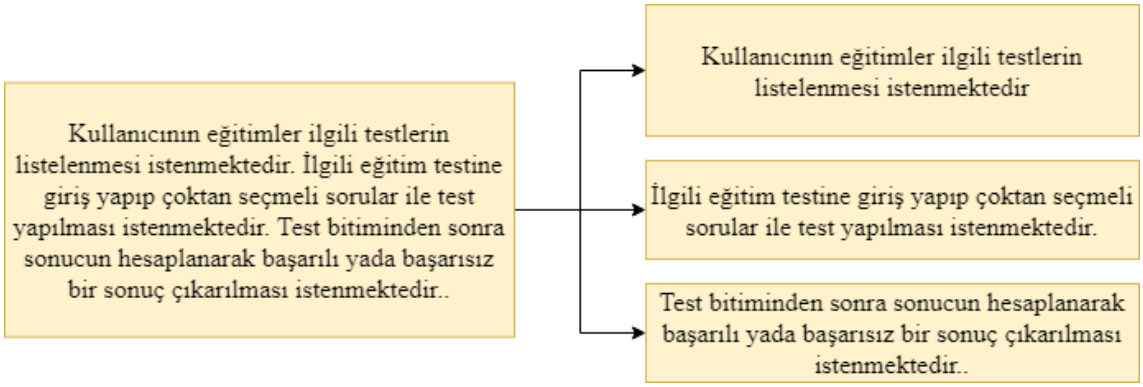
İş tarafının kullanıcıların belirlenmesi eksik kalması konusunda başka birimlerle iletişim konusunda hızlı davranılması gerektiği ortaya çıkmıştır. Sprint planlama toplantısı ardından başka birimlerle ilgili maddeler çıkması durumunda diğer birimlerle hızlı bir şekilde toplantı yapılıp konunun ilerletilmesi gerekmiştir.

5.2. Sprint 2

Sistem personeller için gerekli eğitimleri sunar. Personellerin sisteme girişi tamamlandıktan sonra gerekli eğitimlerin listelenmesi gerekmektedir. Bizim sprint sonunda ulaşmamız gereken hedef bu işlevin yerine getirilmesidir. Bu hedef için sprint planlaması yapılıp kullanıcı hikayelerinin tanımlanması gerekmiştir.

5.2.1. Sprint Planlama

Sprintin planlama yapılması için kullanıcı hikâyelerinin çıkarılması gerekmektedir. Şekil 34 üzerinde epik üzerinden 4 tane ihtiyaç öğesi tanımlanmıştır. İhtiyaç öğeleri biraz kısa tutularak geçen Sprint için yapılan Sprint retrospektifinde yazılım ekibine daha fazla zaman tanımlanması yapılmıştır.



Şekil 34: Sprint 2 Hedefinin Kullanıcı Hikâyesi

Bu kullanıcı hikâyelerine göre Sprint planlama etkinliğinde Scrum ustasının yapmış olduğu toplantıda ihtiyaçları maddelemek, öncelendirmek ve tahmini zaman verilmesi gerçekleştirilecektir. İş tarafının diğer birimlerle yapacağı görüşmelerin hemen planlama

toplantısının ardından yapılması talep edilmiştir ve böylelikle geçen Sprint oluşan iş maddelerin bitmesinin önüne geçilmesi planlanmaktadır. Sprint iş listesi ürünlerini ve tahmini zaman planlaması Tablo 11 ‘de gösterilmiştir.

Tablo 11. Sprint 2 İş Listesi Ürünleri ve Tahmini Zaman Planlaması

Madde	İş Listesi Ürünü	Rol	Öncelik	Tahmin Zaman
1	Dâhili kullanıcı grubunu ayarlanması	İş Tarafı	Yüksek – Biten Sprint Öğesi	0,5 hafta
2	Gerekli eğitim listelerin hazırlanması	İş Tarafı	Yüksek	2 hafta
3	Gerekli eğitim içeriklerinin hazırlanması	İş Tarafı	Yüksek	1,5 hafta
4	Eğitimlerin listelenmesi	Geliştirme Takımı	Yüksek	2 hafta
5	Eğitimlerin video olarak gösterilmesi	Geliştirme Takımı	Orta	1 hafta
6	Eğitimlerin doküman olarak gösterilmesi	Geliştirme Takımı	Yüksek	1 hafta

Yapılması gereken maddelerin çıkarılmasından sonra iş tarafı siber güvenlik birimi ile toplantı yaparak geçen haftadan eksik kalan kullanıcı listesi ve yeni gündem maddeleri konusunda ilerlemeye başlamıştır.

Geliştirme ekibinin yapması gereken üç tane ekran bulunmaktadır. Ekibin projede artık daha hızlı yol olarak daha rahat ve kolay olarak işleri yetiştirmesi planlanmıştır.

5.2.2. Sprint Süreci

Ekip günlük Scrum toplantıları bu Sprint boyunca sürdürmeye devam etmiştir. Bu Sprint ekip yine 3 sorunu cevabı vererek dün ne yaptım, bugün ne yapacağım ve önümde engeller nelerdir şeklinde günlük iletişim sağlamıştır. Ve önlerine gelen engelleri gidermiştir. Örneğin eğitim içerikleri olmaması konusunda Scrum ustası örnek veriler sağlayarak iş tarafın sağlayıcı veri gelene kadar ki engellerin önüne geçilmiştir.

Her görev için ne kadar çaba kaldığının analizin çıkartılması gerekmektedir. Üç gündem maddesi için dört haftalık Sprint planın örneği Tablo 12 gösterilmiştir.

Tablo 12. Sprint 2 İş Listesinin Günlere Göre Kalan Eforları

İşler	1. Gün	2. Gün	3. Gün	4. Gün	…..	28. Gün
Eğitimlerin listelenmesi	14	13	12	12		0
Eğitimlerin video olarak gösterilmesi	7	6	5	5		0
Eğitimlerin doküman olarak gösterilmesi	7	7	6	6		0

Üç iş listesi ürünlerinin geliştirme aşamaları şu şekilde gelişmiştir.

1. Eğitimlerin listelenmesi: Eğitimlerin tutulacağı veri tabanı tablosu oluşturulmuştur. Mobil yazılım ara yüz ekranı yapılmıştır. Eğitimler veri tabanına test olarak direk eklenmiştir. Bu eklenen eğitimler tablodan gerekli yazılım yazılarak mobil ara yüz ekranında gösterilmesi sağlanmıştır.

Personeller sisteme giriş sağladıktan sonra sistemin en önemli amacı olan eğitimlere ulaşacaktır. Burada hangi eğitimleri olduğunu, önem derecesini, eğitimlerin türünü ve hangi eğitimleri tamamladığını görecektir.

2. Eğitimlerin video olarak gösterilmesi: Video olarak sunulan eğitimler eğitim başlıkları veri tabanından tutulurken videolar veri tabanından saklanması çok uygun olmayacağı için ilgili sunucu dosya yollarında veri tabanı ile ilişkilendirilerek tutulması sağlanmıştır. Yine mobil ara yüz yapılarak bir video oynatma sistemi eklenmiştir.

Sistemde iki tip eğitim bulunmaktadır; doküman ve video. Video olan eğitimler, eğitim listesinde yer alan tipe göre içeriği sunulmuştur Eğitim listesinden video eğitimi seçilmiş olan personeller mobil uygulama içerisinde bu ilgili videoları izleyerek eğitimi almış olacaklardır.

3. Eğitimlerin doküman olarak gösterilmesi: Doküman olarak sunulan eğitimler dokümanların yazılı metin içermesinden dolayı çok fazla boyut kaplamayacağı için veri tabanında tutulmuştur. Mobil ara yüz sağlanarak dokümanların gösterilmesi sağlanmıştır.

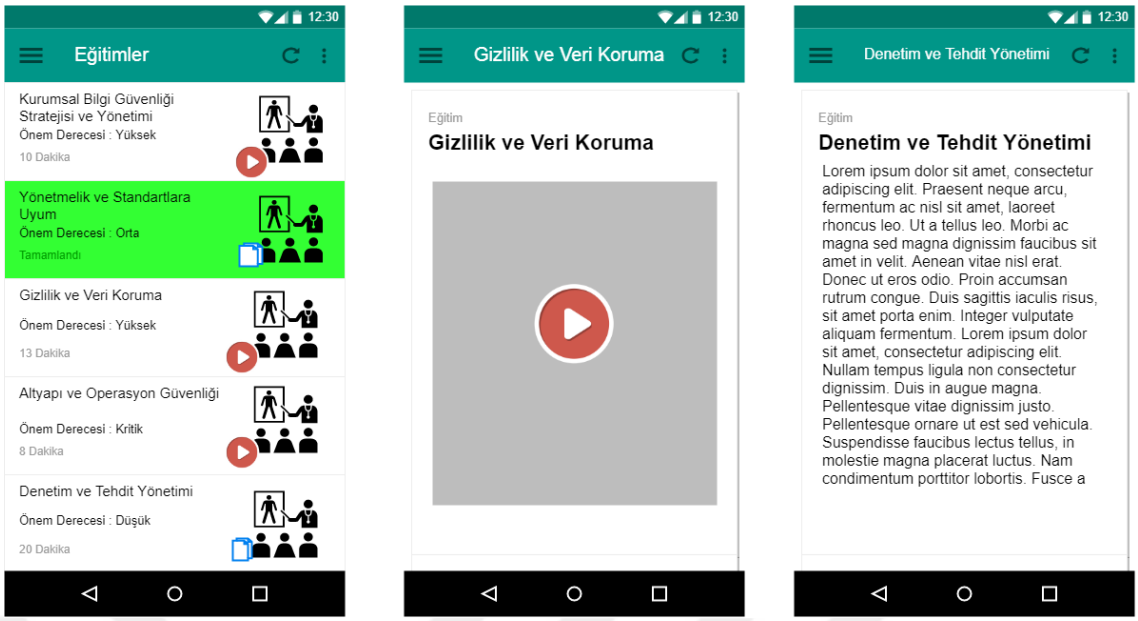
Güvenlik eğitimlerin büyük çoğunluğu doküman eğitimlerinden oluşmaktadır. Doküman olarak eğitim listesinden seçilmiş olan eğitimlerin mobil ortam ile okunması sağlanmıştır.

5.2.3. Sprint Değerlendirme

Sprint süresinin bitmesinin ardından ekip paydaşlar ile tekrar ikinci Sprint değerlendirmek için bir araya gelmiştir. Öncelik gündem maddesi geçen hafta teslim edilen işlerin test edilmiş olmasıydı. Paydaşlar tarafından olumlu test sonuçları bildirilmiş ve bu ekip için ciddi bir isteklendirme kaynağı olmuştur.

Bu Sprint için iş tarafı bir önceki Sprint retrospektifinden aldığı geri dönüşler ile işlerin yetiştirilmesi konusunda diğer birimlerle daha sıkı çalışarak Sprint sonunda geçen Sprint kalan öğelerin tamamlamış ve bu Sprint için gerekli olan eğitim listeleri ve içerikleri konusunu tamamlamıştır.

Geliştirme ekibi mobil yazılım için gerekli olan eğitimler listelenmesi, video eğitimlerinin gösterilmesi ve doküman görüntülenmesini ekranların ihtiyaçlarını geliştirerek paydaşlara sunmuştur (Bkz. Şekil 35). Gelecek sprinte kadar paydaşların test etmesi beklenmiştir.



Şekil 35: Eğitim Ekranları

5.2.4. Sprint Retrospektifi

Ekip Sprint bitmesinden sonra çaba değerlendirmesi yapmak için bir araya gelmişti. Ekip artık Sprint süreçlerine alışıp daha başarılı işler ortaya koymaktadır. Sprint ekip gittikçe alışmaktadır.

Bu hafta bütün işlerin paydaşlara teslim edilmesi ve paydaşların test sonucunda başarılı sonuçlar dönmesi ekibin isteklendirme artırmıştır.

Ekibin bu hafta eksik kaldığı, tahmini zamanla ve öncelik vermeleri konularında biraz aceleci davrandığı ve işlerin istediği sürelerde bitmemesidir. Bir işin erken biterken bir işin geç bitmesidir. Buda ilerleyen zamanda iki tane tahmini zaman konusunda uzun iş verilmesi durumunda işlerin yetişmemesine sebep olur. Ya da az süren işler verilmesi durumunda ekibin sprintin büyük kısmını boş geçirmesine sebep olur. Bu konuda Scrum ustası biraz daha özenli davranıp durumları daha çok inceleyecek tahmini zamanlamayı daha gerçek zamana yaklaştırmaya çalışacaktır.

Sprint bitmesinden sonra çabanın doğru harcandığını ve Sprint iyileştirme için Scrum ustası eşliğinde bir toplantı yapılarak sprintin eksik veya iyi olan durumlarının ortaya çıkarılması hedeflenmiştir.

İyi özellikler olarak deneme rolleri ve kullanıcı ile geliştirme ekibi hızlı bir şekilde geliştirmeleri tamamlamıştır.

Geliştirme ekibi bu Sprint için çok fazla çaba ve fazla mesai yaparak Sprint hedefine ulaşmıştır. Tahmini zamanlama konusunda biraz daha esnek olunması gerektiğine karar vermiştir. Yazılım geliştirirken istenmeyen durumlar gerçekleşebilmekte ve böyle durumlar geliştirmeyi yavaşlatmakta ve hedefe ulaşmak için daha fazla çaba harcanması gerekmektedir. Örnek olarak şirket planlı bir internet kesintisinin olması yazılım ekibinin işlerinin yavaşlamasına sebep olmuştur.

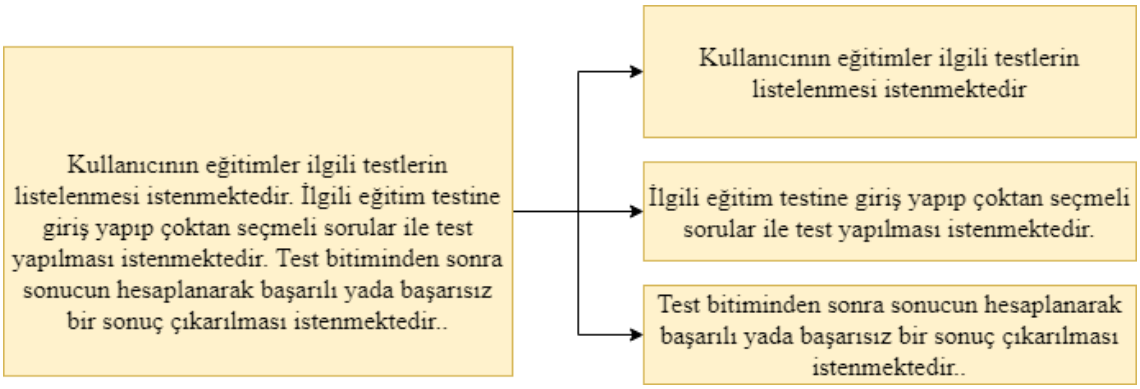
İş tarafının kullanıcıların belirlenmesi eksik kalması konusunda başka birimlerle iletişim konusunda hızlı davranılması gerektiği ortaya çıkmıştır. Sprint planlama toplantısı ardından başka birimlerle ilgili maddeler çıkması durumunda diğer birimlerle hızlı bir şekilde toplantı yapılıp konunun ilerletilmesi yapılmıştır.

5.3. Sprint 3

Personeller aldıkları eğitimler sonrası değerlendirme testine girer. Bu testler sonuçlarında başarılı veya başarısız olurlar. Buradaki amaç gerçekten verimli dinleyip ya da okuduğunu ölçmektir. Başarılı geçen iki Sprint sonunda ekip üçüncü ve son Sprint için çalışmaya başlamıştır. İki başarılı Sprint sonunda ekibin motivasyonu çok yüksek bir şekilde bu Sprint başlamış ve bu hedefi tamamlamak için planlama toplantısı yapıp gerekli olan hedefleri tamamlaması yapılmıştır.

5.3.1. Sprint Planlama

Ekip için üçüncü Sprint planlanması olması sebebiyle artık ekip bu konuda ciddi bir performans kazanmıştır. Şekil 36 üzerinde epik üzerinden 4 tane ihtiyaç öğesi tanımlanmıştır. İhtiyaç öğeleri biraz kısa tutularak geçen Sprint için yapılan Sprint retrospektifinde yazılım ekibine daha fazla zaman tanımlanması yapılmıştır. Bir önceki retrospektif değerlendirmesinde tahmini zamanların daha iyi ayarlanmasının ihtiyacı vardır. Scrum ustası kullanıcı hikâyelerinde tahmini zaman ayarlaması konusunda daha fazla çalışacaktır. Kullanıcı hikâyeleri şekilde özetlenmiştir.



Şekil 36: Sprint 3 Hedefinin Kullanıcı Hikâyesi

Scrum ustası kullanıcı hikâyeleri üzerinden tahmini zaman ve öncelik için eski iki Sprint göz önüne alınarak daha gerçeğe yakın bir yaklaşım ile tahmin ve önceliklendirme planlamıştır. Scrum ustası Tablo 13 işleri maddelenmiştir ve tahmini zaman ayarlamalarını gerçekleştirmiştir.

Tablo 13. Sprint 3 İş Listesi Ürünleri ve Tahmini Zaman Planlaması

Madde	İş Listesi Ürünü	Rol	Öncelik	Tahmin Zaman
1	Test sorularının ayarlanması	İş Tarafı	Yüksek	3 hafta
2	Test başarı oranlarının ayarlanması	İş Tarafı	Orta	1 hafta
3	Eğitimlerin testlerinin listelenmesi	Geliştirme Takımı	Yüksek	1,5 hafta
4	Test soruların sorulması	Geliştirme Takımı	Yüksek	2 hafta
5	Test sorularının bitmesi sonrası sonucun hesaplanması	Geliştirme Takımı	Orta	1,5 hafta

Ekibin iki başarılı Sprint sonrası yüksek bir tempo ile yeni Sprint başlamıştır.

5.3.2. Sprint Süreci

Son Sprint'e gelen ekibin artık günlük Scrum konusunda tecrübeleri çok fazla artmıştır. Artık eskiden tam oturmayan 15 dakikalık toplantılar çok daha dakik ve verimli olmaya başlamıştır. Scrum ustası sorunları gidermek için, iş tarafı ve paydaşlar iletişimi artırmış ve durumu engelleyen konulara daha hızlı ve başarılı çözümler sunmaktadır. Ekip bu Sprint boyunca düzenli olarak ne yaptık ne yapacağız ve önümüzde ki engeller nedir sorularına cevap vererek günlük Scrum'ları tamamlamıştır.

Her görev için ne kadar çaba kaldığının analizin çıkartılması yapılmıştır. Üç gündem maddesi için dört haftalık Sprint planın örneği Tablo 14'de gösterilmiştir.

Tablo 14. Sprint 3 İş Listesinin Günlere Göre Kalan Eforları

İşler	1. Gün	2. Gün	3. Gün	4. Gün	…..	28. Gün
Eğitimlerin testlerinin listelenmesi	10	9	8	12		0
Test soruların sorulması	7	6	5	5		0
Test sorularının bitmesi sonrası sonucun hesaplanması	11	11	11	11		0

Üç iş listesi ürünlerinin geliştirme aşamaları şu şekilde gelişmiştir.

1. Eğitimlerin testlerinin listelenmesi: Testlerin tutulacağı veri tabanı tablosu oluşturulmuştur. Mobil yazılım ara yüz ekranı yapılmıştır. Testler veri tabanına test olarak direk eklenmiştir. Bu eklenen eğitimler tablodan gerekli yazılım yazılarak mobil ara yüz ekranında gösterilmesi sağlanmıştır.

Personeller eğitimleri tamamladıktan sonra bu eğitimleri gerçekten başarı ile tamamladığını ölçmek için sistem tarafından teste tabi tutulur. Burada hangi testlerin olduğu, kaç soru olduğu ve hangi eğitimlerin tamamlandığı bilgisini içerir.

2.Test soruların sorulması: Veri tabanına kayıt edilmiş olan testler kullanıcın seçmiş olduğu test içerikleri ilgili olarak mobil ara yüz ile personele sorulmaktadır. Burada ilgili test soruları belirli bir sıralamaya göre gelmektedir. Mobil ara yüz ile dört cevap hakkı sunulmakta ve bu cevaplardan birisinin doğru olduğunun algoritması geliştirilmiştir. Personelin her işaretlediği soru sonrasında bu cevaplardan hangisinin doğru veya yanlış olduğu veri tabanına kayıt edilmiştir.

Personellerin eğitim testlerini listeledikten sonra bu testlere girerek başarılı olmaları gerekmektedir. Siber güvenlik birimi bu testlerin başarı oranlarını sürekli takip ederek ilgili personele bilgilendirme yapacaktır. Personel ilgili teste girdiğinde ilgili test kadar çoktan seçmeli soru sorulacaktır. Çoktan seçmeli sorulardan yanlızca bir tanesinin doğru cevabı olacaktır.

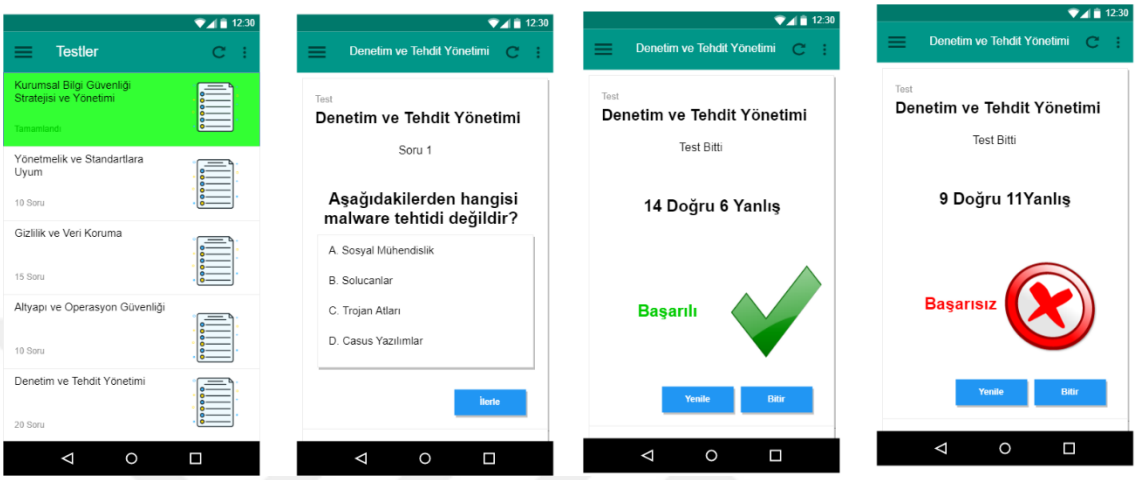
3.Test sorularının bitmesi sonrası sonucun hesaplanması: Testlerin bitmesinin ardından veri tabanına kayıt edilmiş testler ilgili olarak iş birimin vermiş olduğu hesaplamalara göre bir başarı oranı çıkartılacaktır. Sistemde başarı oranı geçen durumlar için ilgili test tamamlandı durumuna dönüştürülmüştür.

Personeller eğitimi tamamladıktan sonra siber güvenlik biriminin vermiş olduğu oran üzerinden başarıya ulaşıp ulaşmamalarına göre eğitim tamamlandı durumuna gelecektir. Siber güvenlik birimi bu sonuçları sürekli takip edecektir. Personel test listesine geri dönüş sağladığında eğitimi tamamlandı olarak görmesi sağlanmıştır.

5.3.3. Sprint Değerlendirme

Son Sprint olması sebebiyle paydaşlardan eski test sonuçlarında bir hata olup olmadığı öğrenildi ve yeni geliştirme ekranların test konusunda hızlı bir geri dönüş beklenildiği iletildi. Paydaşlar ikinci Sprint sonuçlarından oldukça memnun olduklarını ilettiler. Son

Sprint değerlendirme toplantısında iş tarafı gerekli testlerin ayarlanması ve içeriklerin ayarlanmasını siber güvenlik birimi ile başarılı bir şekilde sonuçlandırmıştır. Geliştirme takımı da gerekli olan mobil yazılımı gerçekleştirerek paydaşların test etmesine sunmuştur. (Bkz. Şekil 37). Paydaşlar projenin son sprinti olması sebebiyle gerekli olan başarıyı sağladıklarını ve şirkete büyük bir katma değer kattıklarını ifade etmişlerdir.



Şekil 37: Eğitim Test Ekranları

5.3.4. Sprint Retrospektifi

Sprintin artı ve eksi durumlarının başarıyla sonuçlanması ve işin sonucuna erişilmesi herkes için büyük bir başarı olmuştur. Ekip öz (Essence) kullanarak daha hızlı ve etkili bir yol aldığını ifade etmiştir. Üç sprintin başarılı sonuçlanması sonucunda ekip kutlamayı hak etmiştir. Bir kutlama partisi planlanarak bir sonraki projeler için ekip isteğinin artırılması planlanmıştır.

BÖLÜM 6

6. SONUÇ VE ÖNERİLER

Raporun bu bölümünde gerçekleştirilen çalışmanın sonuçları ile gözlemlenen bulgular dikkate alınarak ana hatlarıyla özetlenmiş, ileriye dönük çalışmalarla ilgili çeşitli önerilere yer verilmiştir. Araştırmanın amacı doğrultusunda gözlenen önemli sonuçlar şunlardır:

- ÖÇ ve MYM ile ilgili yöntem, teknik ve araçların kavramsal, görsel ve metin biçiminde tanımlanabildiği bir yaklaşım sunulduğu,
- ÖÇ'nin MYM için olduğu kadar geleneksel yazılım mühendisliğini oluşturan ana bileşenler arasındaki ilişkileri basit ve anlaşılır biçimde modelleyebildiği,
- ÖÇ'nin yazılım takımı içerisinde gerek iletişimde gerekse iş yapma biçimlerinde kavramsal ve görsel olarak ortak bir lisanı oluşturduğu,
- ÖÇ'nin, çevik yazılım geliştirme süreçlerinin modellenmesi, izlenmesi ve değerlendirilmesi yönünde kavramsal düzeyde önemli katkılarda bulunduğu,
- ÖÇ'nin MYM alanında nelerin yapılacağı ve bunların nasıl yapılacağını tanımlı biçimde ayırarak ilgi alanları oluşturduğu, sorumlulukların dağıtımını, yazılımın izlenmesini kolaylaştırdığı,
- ÖÇ'nin farklı ölçekteki MYM projelerine ait yöntem, teknik ve araçlarının kullanımında çeşitli esneklikler sağladığı,
- ÖÇ'nin firma bünyesine katılan yeni ya da tecrübesiz yazılım mühendislerinin eğitimi ve oryantasyonunda olumlu katkı sağlar,
- MYM takımlarının farklı projelere, kendi kültür ve kapasitelilerine göre kendi öznel yazılım geliştirme süreçlerini oluşturabilmelerine olanak sağlayabileceği gözlenmiştir.

Olumlu sonuçlarına ek olarak ÖÇ ile ilgili gözlenen bazı sınırlılıkların aşağıdaki gibidir:

- ÖÇ ile ilgili kapsamlı bir eğitimin alınması gerekir, uygulamalarda bu konuda uzman kişilerin desteğine mutlaka ihtiyaç vardır,
- Geleneksel yazılım mühendisliği uygulamalarına alışkın ve tecrübeli personelin uyum sağlamada ve ÖÇ'nin katkılarını özümsemede güçlükler yaşanmaktadır,

- Firma bünyesinde ÖÇ ile geliştirilen projelerde ve özellikle başlangıç aşamasında beklenenin aksine önemli oranda dokümantasyonu arttırır, bu ise takım içerinde motivasyonu olumsuz yönde etkilemektedir,
- Özellikle çok küçük ölçekli yazılım firmaları için gerek ÖÇ'nin uygulamasında ve gerekse sürece kaynak sağlanmasında ciddi zorluklar içermektedir,
- ÖÇ'nin teslim edilen yazılımların bakımı, güncellenmesi ve idamesiyle ilgili rehber ve ilkeler sunamadığı, bu tür projeler için kullanılan ÖÇ araçlarında sınırların olduğu gözlenmiştir.

Bu çalışmada elde edilen ve gözlenen sonuçlar doğrultusunda MYM uygulamaları ve araştırmalarına yönelik öneriler ise şunlardır:

- Araştırmanın amaçları dışında gözlenen önemli bir bulgunun da ÖÇ'nin yazılım mühendisliği araştırmalarını bütünleşik yapıda ele alabilecek, modelleyebilecek ve farklı araştırmalara ait sonuçların karşılaştırılabilmesine olanak sağlayabilecek yapıda olduğudur. Bu kapsamda yazılım mühendisliği araştırma yöntemlerinin tasarımında ÖÇ'nin kullanıldığı yeni çalışmalara ihtiyaç bulunduğu düşünülmektedir.
- Yazılım bakım ve onarımını, özellikle yazılım yeniden yapılamayı kapsayacak biçimde ÖÇ yaklaşımının güncellendiği araştırmalara ihtiyaç olduğu değerlendirilmektedir.

Sonuç olarak bu çalışma doğrultusundaki bulgu ve gözlemlerin ÖÇ'nin, MYM araştırma ve uygulama alanındaki önemli eksikliklere çözüm getireceği, farklı bilgi alanları arasında köprü vazifesini göreceği, yazılım mühendisliği kuram, araştırma ve uygulamalarındaki sorunlara da ışık tutacağı yönündedir. Araştırma raporu, ortaya konulan olumlu ve olumsuz bulgular, araştırma sınırlılıkları ile önerilerinin dikkate alındığı yeni çalışmaların yapılması çağrısıyla son bulmaktadır.

KAYNAKÇA

Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jaalinoja, J., Korkala, M., & Salo, P. (2017, Sep 20). Mobile-D:an agile approach for mobile application development. *In Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, 174-175.

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). *Agile Software Development Methods.* Espoo, Finland: VTT Publications 478.

Ak, M. (2012). Cobit'in Yazılım Geliştirme Sürecinin İyileştirilmesine Uyarlanması. *Yüksek Lisans Tezi,Gazi Üniversitesi*.

Allen, S., & Zamanian, K. (2010, October 28). Key challenges and emerging trends in mobile software engineering. *Workshop on Mobile Software Engineering*, 11-17.

Arman, T. (1997). *Risk Analizine Giriş.* İstanbul: Alfa Yayıncılık.

Azar, J., Smith, R., & Cordes, R. (January/February 2007). Value-Oriented Requirements Prioritization in a Small Development organization. *IEEE Software*.

Balzer , B., Litoiu , M., Müller , H., Smith , D., Storey , M., Tilley , S., & Wong , K. (2004). 4th International Workshop on Adoption-Centric Software Engineering. *in Proceedings of 26th International Conference Software Engineering*, 748-749.

Baskerville , R., & Michael , D. (2004). Special Issue on Action Research in Information Systems. *Making IS Research Relevant to Practice*, 329-335.

Beck, K. (2000). *Extreme programming Explained Embraced Change .* Addison-Wesley .

Beck, K., & Fowler, M. (2000). *Planning Extreme Programming .* Addison-Wesley Press .

Beck, K., Cockburn, A., Jeffries, R., & Highsmith, J. (2001). Agile Manifesto. *Agile Manifesto*.

Beedle, M., Devos, M., Sharon, Y., Schwaber, K., & Sutherland, J. (1999). SCRUM: An extension pattern language for hyperproductive software development. *Pattern Languages of Program*, 637-651.

Bhowmik, T., Alves, V., & Niu, N. (2013, August 14-16). Porting mobile games in an aspect-oriented way: an industrial case study. *14th International Conference on Information Reuse and Integration*, 49-53.

Blum, A. (2010, October 28). Zero-latency Development. *Workshop on Mobile Software Engineering*, 32.

Boz, C. (2015). *Bilişim Projesi Yönetimi.* https://slideplayer.biz.tr/slide/2602886/ adresinden alındı

Cohn, M. (2003). *User stories applied for Agile Software Development .* Addison-Wesley .

Community, T. P. (1994). *A Guide to the Project Management Body of Knowledge.* Boston.

Çağırgan, M. (1997). Risk Yönetimi. *İstanbul Üniversitesi Sosyal Bilimler Enstitüsü Yüksek Lisans Tezi*, 23-26.

Çamoğlu, K., Akbayır, D., Yücalar, F., & Bayraklı, S. (2010, Ocak). Bir Çevik Yazılım Geliştirme Sürecinin Uyarlanması ve Uygulanması. *Havacılık ve Uzay Teknolojileri Dergisi*, 57-67.

Çift, Z. A. (2015). Başarılı Liderlik. *Yönetim,İletişim ve Psikoloji*. Samsun, Türkiye.

Daft, R. (1991). *Management.* USA: Dryden Press.

Demirci, A. (2015, Mart 3). 2019 tarihinde https://herturbilgi.com/yazilim-gelistirme-yasam-dongusu/ adresinden alındı

Duran, M. (2009, Mart 23). 2019 tarihinde http://www.mehmetduran.com/Blog/Makale.html/Yazilim-Gelistirme-Surec-Modelleri/299 adresinden alındı

Elibol, M., & Erol, Ç. (2014). *Mobile Application Development With Agile Methodology ,Economics and Business Communication.* Katowice, Poland: Dyduch, W. And Pankowska, M. , House of the.

Enver, E., & Kovancı, A. (2004). Havacılık ve Uzay Teknolojileri Dergisi. *Proje Yönetimi ve İnsan Kaynakları İlişkisi*, s. 76.

Erdem, O. A., & Younis, A. E. (2014, OCAK 10). Bilişim Teknolojileri Dergisi. *Yazılım Projelerinin Geliştirme Sürecinde Yönetim*, s. 5.

Erol, A. (2015, Şubat 1). 2019 tarihinde http://yazilimgelistirmeyontemleri.blogspot.com/2015/02/yazlm-gelistirme-yasam-dongusu-sdlc.html adresinden alındı

Flora, H., & Chande, S. (2013). A Review and Anaysis on Mobile Application Developmnet Processes Using Agile Methodolojies. *International Journal of Research in Computer Science*, 8-18.

Francese, R., Risi, M., & Tortora, G. (2015, May 16-17). Management, sharing and reuse of service-based mobile applications. *2nd ACM International Conference on Mobile Software Engineering and Systems*, 43.

Griss, M. (2010). New patterns, components and approaches for mobile SE. *Workshop on Mobile Software Engineering*.

Hertz, D. B., & Thomas, H. (1983). *Risk Analysis and its Applications.* New York, US: Wiley.

Highsmith, J. (2002). “What is Agile Software Development?” CrossTalk The Journal of Defense Software Engineering. 4-9.

Jacobson , I., Meyer , B., & Soley , R. (2009). The SEMAT initiative: A Call for Action. *Dr Dobb's Journal*, 9.

Karakış, İ. (2007). Tedarikçi İlişkileri Yönetimi. *İstanbul Teknik Üniversitesi,Yüksek Lisans Tezi*, 140.

Kaynak, T. (1998). *İnsan Kaynakları Yönetimi İ.Ü. İşletme Fakültesi Yayınları.* İstanbul.

Kerzner, H. (2003). *Project Management A Systems Approach to Planning.* 8th Ed.: John Wiley and Sons Inc.

Kırmızı, M. (2017). Proje Yönetim Metodolojileri ve Örnek Bir ERP Proje Yönetimi Metodolojisinin Değerlendirilmesi. *Yüksek Lisans Programı*, 22.

Kitchenham , B., Pfleeger, S., Pickard , L., Jones , P., Hoaglin , D., Emam , K., & Rosenberg , J. (tarih yok). Preliminary guidelines for empirical research in software engineering. *IEEE Trans. Softw. Eng., vol. 28*, s. 721-734.

Klein , H., & Myers , M. (1999). A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS Quarterly* , 67-94.

Kniberg, H. (2007). *Scrum and XP from the Trenches.* United States of America: C4 Media Inc.

Kwon , Y., & Tilevich, E. (2015, May 16-17). Facilitating the implementation of adaptive cloud offloading to improve the energy efficiency of mobile applications. *2nd ACM International Conference on Mobile Software Engineering and Systems*, 87-91.

Lab, A. R. (2010). MDE, DSL and tooling for effective context management in ubiquitous computing. Santa Clara: USA.

Lekesiz, F. S. (2013, July). Yazılım Projelerinde Proje Yönetim Süreçlerinin İncelenmesi. *TOBB Ekonomi ve Teknoloji Üniversitesi*, 28,29.

Livari, J., & Venable, J. (2009). Action research and design science research Seemingly similar but decisively dissimilar. *European Conference on Information Systems*, 1642-1653.

Malhotra , R. (2006). *Empirical research in software engineering: concepts, analysis, and applications.* NW: CRC Press,Taylor & Francis Group.

Mattsson, M. (2010). Towards a Mobile Software Engineering Education. *International Conference on Information Society*, 528-535.

Michael, & Greenfield. (1959). Normal Accident Theory,. *Heinrich,.H.W.Industrial Accident Prevention*, 25.

Nekoo, A., & Vakili, K. (2009). A Practical Course on Mobile-Software Engineering, Fourth International Conference on Software Engineering Advances. 389-393.

Newhman, W. (1979). *Karar Vermenin Temel Evreleri.* Ankara: TODAİE Yayınları Çev.: Kenan Sürgit.

Özgen, H., Öztürk, A., & Yalçın, A. (2002). *İnsan Kaynakları Yönetimi, Nobel Kitapevi.* Adana.

Paasivaara, M., Durasiewicz, S., & Lassenius, C. (2009). Using Scrum in Distributed Agile Development: A Multiple Case Study. *2009 Fourth IEEE International Conference on Global Software Engineering*, 195-204.

Palmer, M., & Winners, K. (1993). *İnsan Kaynakları.* İstanbul: Rota Yayınları.

Pekel, I. Ö. (2008, Ekim 19). Kasım 2018 tarihinde https://e-bergi.com/y/cevik-modelleme-ve-cevik-yazilim-gelistirme/ adresinden alındı

PMI. (2017). *"A guide to the project management body of knowledge," (PMBOK GUIDE), Project Management Institute* (6th ed. b.). Pennysylvania, Newtown Square, USA: Global Standard.

Rahimian, V., & Ramsin, R. (2008). *Designing an Agile Methodology for Mobile Software Development: A Hybrid Method Engineering Approach.* Marrakech, Morocco: 2nd International Conference on Research Challenges in Informations Science.

Redwine , S., & Riddle , W. (May 1985). Software technology maturation. *Proceedings of the Eighth International Conference on Software Engineering*, 189-200.

Samovar. (2010). Academic Research Team of Samovar Lab. *MDE, DSL and tooling for effective context management in ubiquitous computing, Workshop on Mobile Software Engineering, October 28, 2010. Santa Clara, CA, USA.*

Scharff, C., & Verma, R. (2010). Scrum to support mobile application development projectsin a just-in-time learning context. *Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering*, 25-31.

Schwaber, K. (2004). *Agile Project Management with Scrum.* Redmond, United States: Microsoft Press.

Sedano, T. (2010, October 28). Mobile software engineering: it's just a domain. *Workshop on Mobile Software Engineering*, 94-97.

Shaw, M. (2002). What makes good research in software engineering? *International Journal of Software Tools for Technology Transfer*, 1-7.

Sjøberg, D., Dybâ, T., & Hannay, J. (2007). Systematic review of theory use in software engineering experiments. *IEEE Transactions on Software Engineering vol. 33(2)*, s. 87-107.

Somyürek, S. (2018, Mart 8). Şubat 2019 tarihinde https://ders.im/dokuman/yazilim-gelistirme-surec-modelleri-sibel-somyurek adresinden alındı

Stol, K., & Fitzgerald , B. (2015). A holistic overview of software engineering research strategies. *Proceedings of the 3rd International Workshop on Conducting Empirical Studies in Industry, co-located with ICSE* , 123.

Sutherland, J., Viktorov, A., Blount, J., & Puntikov, N. (2007). Distributed Scrum: Agile Project Management with Outsourced Development Teams. *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*, 274.

Şeker, Ş. E. (2008, 29 Kasım). Eylül 2018 tarihinde http://bilgisayarkavramlari.sadievrenseker.com/2008/11/29/selale-modeli-waterfall-model/ adresinden alındı

Tosun, S. (2014). Proje Yönetiminde Tedarikçi Seçimi İçin Bir Model Önerisi. *Yıldız Teknik Üniversitesi Doktora Tezi*, 48-49.

Usman, M., Iqbal, Z., & Kha, M. (2014). *A model-driven approach to generate mobile applications for multiple platforms.* 21st Asia-Pacific Software Engineering Conference.

Uysal, M. (2015). Extending the Design Science Research Methodology for software engineering. *8th Engineering and Technology Symposium.* Ankara.

Uysal, M. (2016). Towards a software engineering research framework: Extending Design Science Research. *International Research Journal of Engineering and Technology*, s. 21-27.

Uysal, M. (2018). A Formal Method for Mapping Software Engineering Practices to Essence. *International Journal of Software Engineering & Applications*, 9(6):1-10.

Uysal, M. (2018, 09 29-30). Using Concept Algebra For Mapping Software Practices To Essence Framework. *4th International Conference on Software Engineering*. Copenhagen, Denmark.

Uysal, M. P., & Giray, G. (2017). Yazılım Mühendisliği Araştırmalarına Öz Çerçeve Yaklaşımı. *UYMS*, 7.

Uysal, M., & Halici, A. (2018). Representing essence of software engineering in enterprise architecture knowledge domain. *International Conference on Advances in Business Management and Information Technology (ICABMIT)*, (s. 12-15). Tokyo, Japonya.

Vaughan, E., & Vaughan, T. (1995). *Essential of Insurance: A Risk.* New York.

Vlaanderen, K., Jansen, S., Brinkkemper, S., & Jaspe, E. (2011). The agile requirements refinery:Appling SCRUM principles to software product management. *Information and Software Technology*, 58-70.

Wasserman, A. (2010, November 7). Software engineering issues for mobile application development. *Workshop on the Future of Software Engineering Research*, 94.

Wasserman, T. (2010, October 28). Bringing software engineering practices to mobile application development. *Workshop on Mobile Software Engineering*, 36.

Yamakami, T. (2016, April 13). A three-dimensional view model of open source-aware software development for large-scale mobile software platforms. *4th IEEE International Conference on Digital Ecosystems and Technologies*, 130-136.

Yıldızöz, H. (2006). Tedarik Zinciri Yönetimi ve Bir Uygulama. *Yıldız Teknik Üniversitesi,Yüksek Lisans Tezi*, 67.

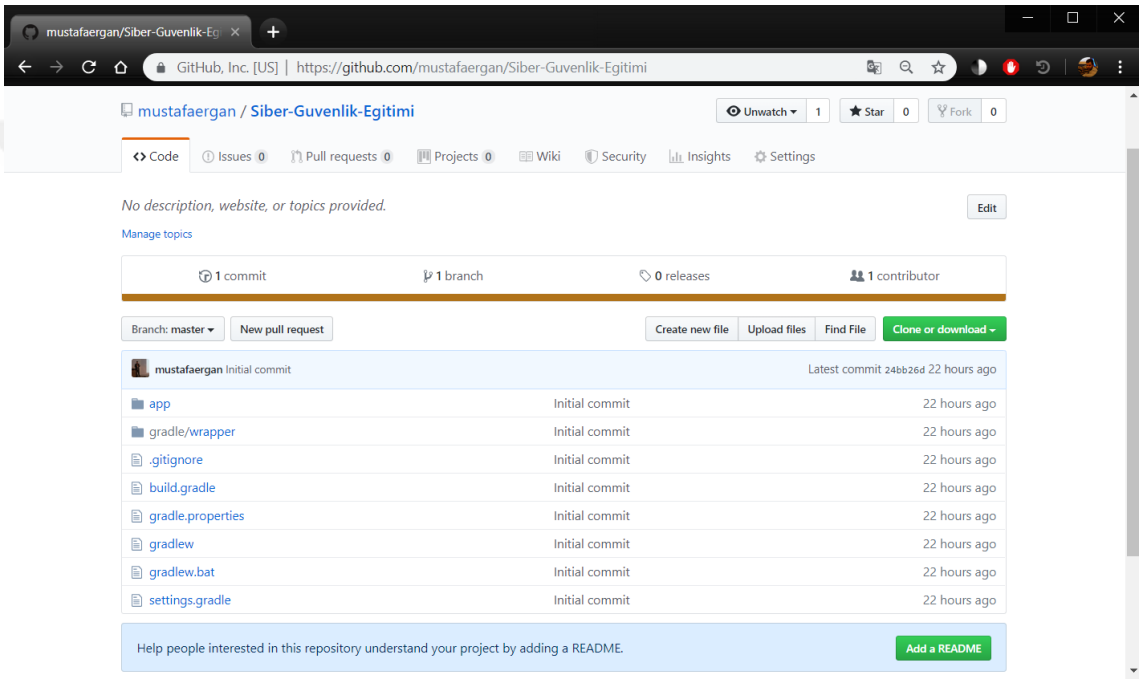
Yu , W., & Yuan, H. (2011). An approach to explore mobile software engineering advances in cloud computing environment. *35th IEEE Annual Computer Software and Applications Conference Workshop*, 110-115.

EKLER

EK – 1. Mobil Yazılım Kodları

Dünya genelinde tüm yazılımcıların kullandığı kod paylaşım web sitesinde **açık erişim** olarak paylaşılmıştır.

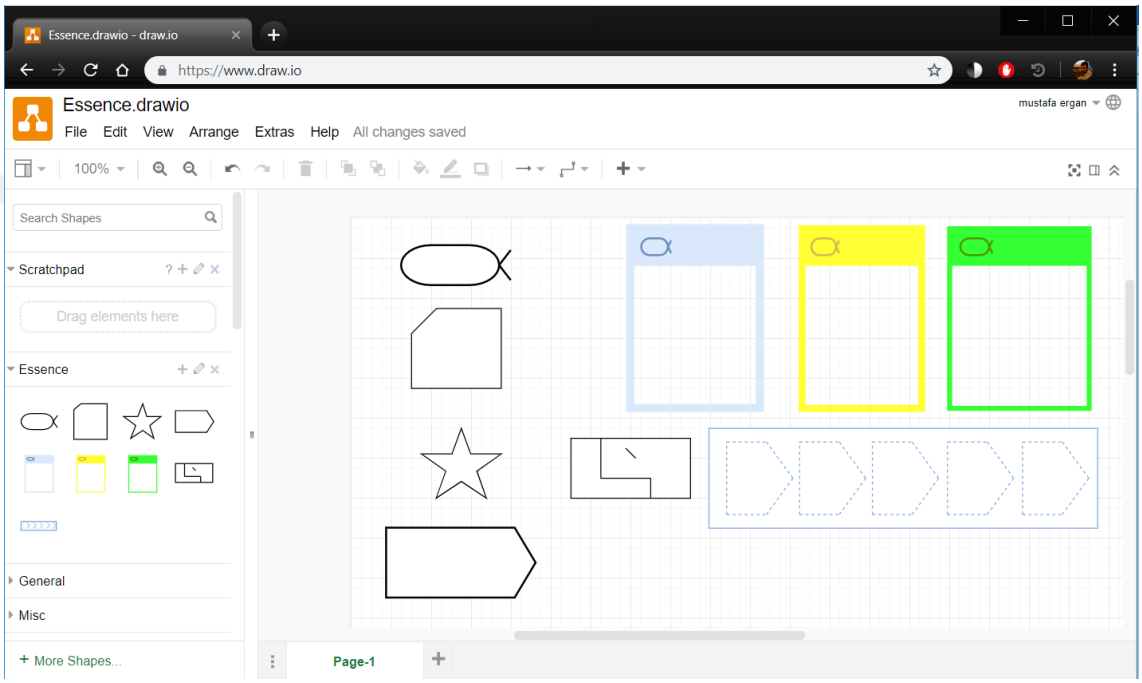
https://github.com/mustafaergan/Siber-Guvenlik-Egitimi



EK – 2. Öz Çerçeve (Essence Framework) Grafik Kütüphanesi

Ücretsiz olan https://www.draw.io/ uygulaması üzerinde Öz Çerçevenin grafik çizimlerinin yapılması için oluşturulmuş çizim kütüphanesinin XML kodları github üzerinden **açık erişim** olarak paylaşılmıştır.

https://github.com/mustafaergan/essence-draw.io



EK – 3. Sprint Toplantı Planları

Tablo 1. Sprint Planlama Toplantı Planı

Zaman	Süre	Aktivite	Kim Yönetiyor
09.00	10 Dakika	Sprint Hedefi	Ürün Sahibi
09.15	1,5 Saat	Görev Dağılımı ve Zamanları	Scrum Ustası ve Yazılım Ekibi
10.45	15 Dakika	Sprint Kapasitesini Hesaplama	Scrum Ustası ve Yazılım Ekibi
11.00	10 Dakika	Kabul Etme	Scrum Ustası ve Yazılım Ekibi
Sprint Sonrası			
15.15	45 Dakika	Tahtanın Hazırlanması	Scrum Ustası

Tablo 2. Sprint Değerlendirme Toplantı Planı

Zaman	Süre	Aktivite	Kim Yönetiyor
09.10	10 Dakika	Sprint Hedefi	Ürün Sahibi
09.20	5 Dakika	Gereksinimlerin İncelenmesi	Ürün Sahibi
09.25	15 Dakika	Sprint Durumu	Scrum Ustası
09.40	15 Dakika	Gösterim	Yazılım Ekibi
09.55	10 Dakika	Geri Bildirim	Scrum Ustası
10.05	5 Dakika	Kapanış	Scrum Ustası

Tablo 3. Sprint Retrospektifi Toplantı Planı

Zaman	Süre	Aktivite	Kim Yönetiyor
09.00	5 Dakika	Başlıkların Hazırlanması	Scrum Ustası
09.05	10 Dakika	Öncelikli Tekliflerin Değerlendirilmesi	Yazılım Ekibi
09.15	20 Dakika	Veri Toplama	Yazılım Ekibi
09.35	20 Dakika	Anlam Oluşturma	Yazılım Ekibi
09.55	20 Dakika	Ne Yapacağına Karar Ver	Yazılım Ekibi
10.15	15 Dakika	Kapanış	Yazılım Ekibi

EK – 4. Sprint Planlama Toplantı Tutanakları

Tablo 1. Sprint* Planlama *Tutanağı

İş	Sprint	İş İsmi	Geliştirme Türü	Başlangıç Zamanı	Bitiş Zamanı	Süre	Durum	Yayınlanma Tarihi	Hedef
1									
2									
3									
4									
5									
….									

Tablo 2. Sprint* Değerlendirme *Tutanağı

İş	Sprint	İş İsmi	Teslim Edildi mi?	Hedeflenen Amaca Uygun mu?	Harcanan Süre	Yayınlanma Tarihi
1						
2						
3						
4						
5						
….						

Tablo 3. Sprint* Retrospektifi *Tutanağı

İş	Sprint	İş İsmi	Yanlış Yapıldı mı?	Daha İyi Yapılabilir mi?	İyileştirme Ne Eklenebilir?
1					
2					
3					
4					
5					
….					