Extreme Programming Proje Planlaması

(Extreme Programming Serisi)

KurumsalJava.com
KurumsalJavaAkademisi.com

Özcan Acar
Bilgisayar Mühendisi
http://www.ozcanacar.com

Extreme Programming (XP) Serisi Hakkında

Extreme Programming birçok çevik metodun kullanıldığı en popüler çevik süreç türlerinden birisidir. Bu makale serisinde Extreme Programming ve uygulaması, gerçek projelerden edinilen tecrübeler doğrultusunda incelenmektedir. Bu seride yer alan makaleleri http://www.kurumsaljava.com/category/xp/ adresinden edinebilirsiniz.

XP Serisinden Diğer Makaleler

- Çevik Süreç Nedir?
  http://www.kurumsaljava.com/2008/12/02/cevik-surec-agile-process-nedir/
- Sürekli Entegrasyon (Continuous Integration)
  http://www.kurumsaljava.com/2008/11/26/surekli-entegrasyon-continuous-integration/
- Test Güdümlü Yazılım – Test Driven Development (TDD)
  http://www.kurumsaljava.com/2008/11/26/test-gudumlu-yazilim-test-driven-development-tdd/
- Extreme Programming Nedir?
  http://www.kurumsaljava.com/2008/11/21/extreme-programming-nedir/

Giriş

Bir geminin rotası sefer öncesi kaptanı tarafından planlanır. Bu planda geminin demir atacağı limanlar ve seyahatin son noktası olan hedef liman yer alır. Böyle bir planın oluşturulması görevlerin dağıtımı ve hedefin tayini açısından önem taşımaktadır. Yolculuk esnasında dış etkilerden dolayı rotada değişiklikler meydana gelebilir. Kaptan ulaşmak istediği hedefi bildiği için gerekli değişiklikleri yaparak rotayı hedefe uyumlu hale getirir. Bir projenin gidişatını bir geminin rotasıyla kıyaslayabiliriz. Projeninde bir rotası olmasi gerekir, aksi taktirde hedefe ulaşmak için gerekli çalışmaların nasıl ve hangi zaman biriminde yapılması gerektiği konuşunda yanlış anlaşılmalar oluşur. Projenin rotası proje planında yer alır. Bu makalede

- proje planının ne olduğunu,
- sürüm ve iterasyon planlamasının nasıl yapıldığını,
- sürüm ve iterasyon planlarının online ve offline ortamlarda nasıl tutulduğunu

yakından inceleyeceğiz.

Proje Planı

Proje planı hangi sürede nelerin yapılacağını ihtiva eden bir dokümandır. Her proje öncesi böyle bir planın oluşturulması gerekmektedir. Proje planı olmayan bir proje rotası belli olmayan bir gemi gibidir. Hangi sürede hangi yolun katedileceğini kimsenin kestirmesi mümkün değildir. Bunun yanı sıra hedefin ne olduğu belli değildir.

Proje planının temelinde oluşturulacak yazılım sisteminin vizyonu yatar. Bu vizyondan gereksinimler doğar ve implementasyon bu gereksinimler doğrultuşunda gerçekleşir. İmplementasyon gereksinimlerin program koduna dönüştürüldüğü işlemdir. Programcılar müşteri gereksinimlerinden ve dolaylı olarak vizyondan doğan gereksinimleri implemente ederler. Gereksinimin ne olduğunu müşteri söyler. Proje planı, hangi gereksimin hangi zaman diliminde ve hangi sıraya göre implemente edileceğini ihtiva eden bir dokümandır. Bunun yanı sıra proje planı vizyonun hayata geçirilmesi için zamansal bir sınırlama getirir. Bu hedefe ulaşmak için gerekli zaman biriminin tanımlanmış halidir.

Proje planının eksikliği vizyonun hayata geçirilmesi için konulması gereken zamansal sınırın eksikliği anlamına gelir. Bu durumda hedefin ne olduğu bilinemez. Hedefin bilinmemesi rotası belli olmayan bir gemide seyahat etmek gibidir.

Sürüm Planlaması (Release Planning)

Bir projenin gidişatını kontrol edebilmek için proje planına ihtiyaç duyulmaktadır. XP projelerinde proje planlaması sürüm planlarının oluşturulmasıyla gerçekleşir. Bu konuda detaya girmeden önce bir sürümün ne olduğu konuşuna açıklık getirmemiz gerekiyor.

Sürüm, bir yazılım sisteminin bir veya birden fazla özellik implementasyonunu ihtiva eden bir versiyonudur. Her sürüm bir ile üç aylık bir yazılım sürecinden sonra oluşan özellikleri ihtiva eder. Her sürüm müşteri tarafından produktif kullanılabilir yapıdadır. Bazı teknik sebeplerden dolayı produktif olarak kullanılamayan sürümlerde olabilir. Ama yeni bir sürüm

oluşturulmasının ana amacı, müşteriye kullanabileceği bir sistemin sunulmasının hedeflenmesidir.

Sürüm planlama aktivitesi müşteriye, hangi gereksinimlerinin bir sonraki sürümde olması gerektiği konusunda yardımcı olur. Bunun yanı sıra programcılar sürüm planlaması esnasında implementasyon için gerekli teknik analizi yapma imkanı bulurlar. Programcılar bu süreçte tahminlerde bulunarak sürüm planının oluşumunda aktif rol oynarlar. Sürüm planlama aktivitesi bir ile iki haftalık bir zaman diliminde gerçekleşir.

Sürüm planı projenin yol haritasıdır. Bu planda özelliklerin hangi sıraya göre implemente edileceği ve hangi tarihte yeni sürümlerin oluşturulacağı yer alır.

XP projelerinde sürüm planları **sürüm planlama oyunlarında** oluşturulur. Bu oyunun nasıl oynandığını yakından inceleyelim.

Sürüm Planlama Oyunu (Release Planning Game)

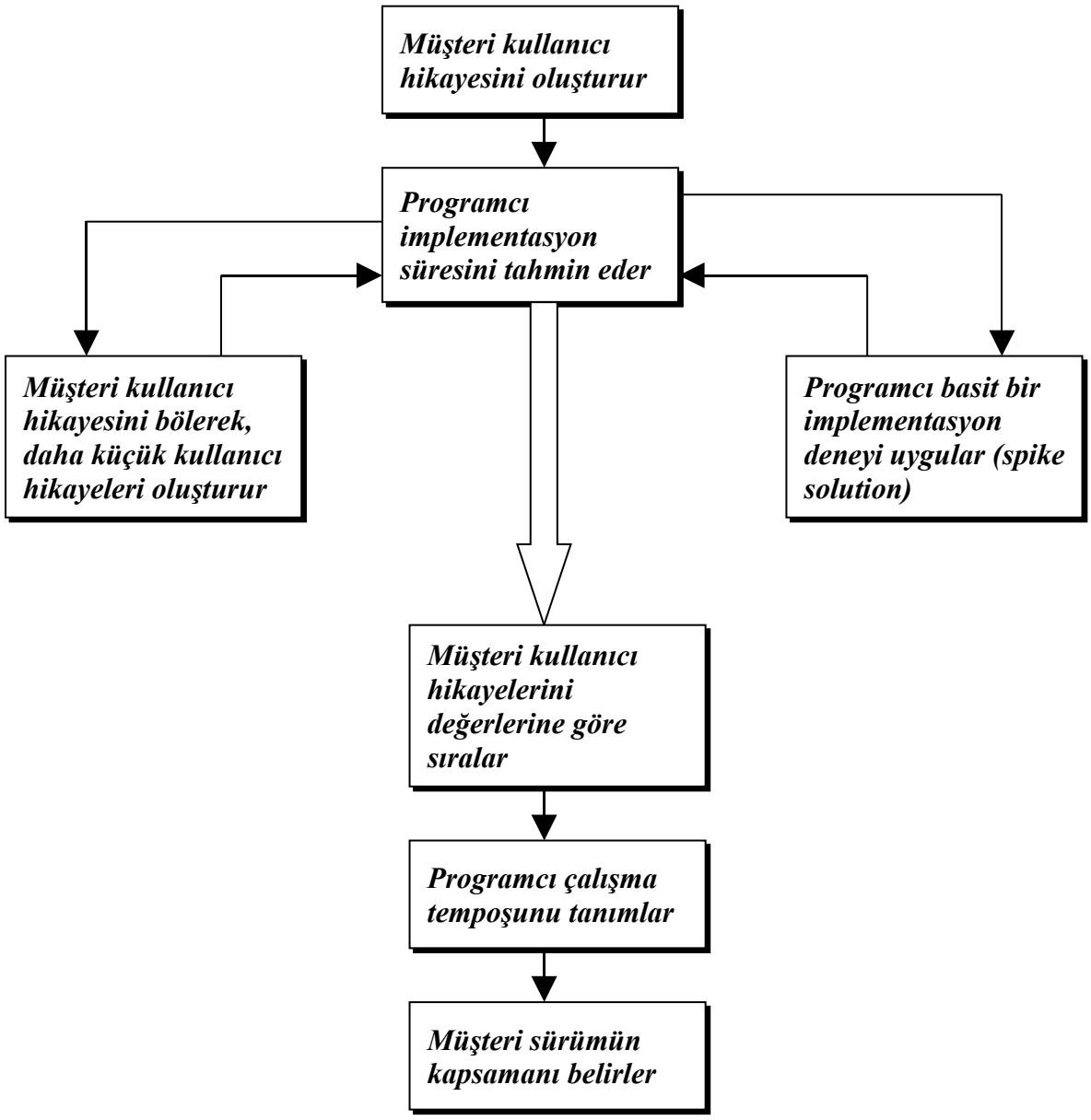
Proje start aldıktan sonra müşteri ve programcılar sürüm planlama oyununda bir sonraki sürümü planlamak için bir araya gelirler.

Bu rekabetin olduğu bir oyun değildir. Daha çok oyuncular birbirlerine yardım ederek, oyunu sonuçlandırmayı hedeflerler. Oyun sürüm planını ihtiva eden dokümanın oluşturulmasıyla son bulur.

Resim 2.1 Programcılar ve müşteri tarafından oynanan sürüm planlama oyunu[1]

[1] Kaynak: http://blog.nayima.be/wp-content/uploads/xp-game-1.jpg

Sürüm planlama oyunu öncesi müşteri gerekli gördüğü tüm kullanıcı hikayelerini oluşturur. Sürüm planlama oyununu için hikaye kartlarına yazılan kullanıcı hikayeleri baz alınır. Oyun içinde yapılabilecek hamleler resim 2.2 de yer almaktadır.



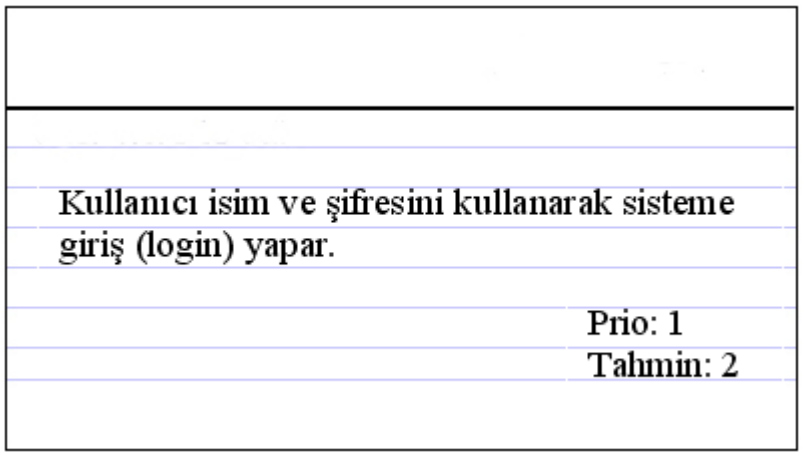
Resim 2.2 Sürüm planlama oyununda yapılabilecek hamleler [2]

Müşteri Kullanıcı Hikayesini Yazar

Sürüm planlama oyunu hikaye kartlarına yazılan (story card) kullanıcı hikayeleri (user story) ile oynanır. Kullanıcı hikayeleri müşteri tarafından oluşturulur ve bir ya da iki cümle ile sistem özelliklerini anlatımda kullanılır. Oluşturulan kullanıcı hikayelerinin programcılar tarafından anlaşılır yapıda olmaları gerekmektedir, çünkü programcılar implementasyonu kullanıcı hikayesini baz alarak yapalar. Bunun yanı sıra oluşturulan kullanıcı hikayelerinin

[2] William C. Wake. Extreme Programming Explored kitabından (s.100) örnek alınmıştır.

test edilebilir yapıda olması gerekir. Kullanıcı hikayesinin bir örneği resim 2.3 de yer almaktadır. Kullanıcı hikayeleri A5 yada A6 formatında kalın kartonda yapılmış kartlar (story card) üzerine yazılır.



Resim 2.3 Kullanıcı hikaye kartı (story card)

Planlama esnasında hikaye kartları önemli bir enstrümandır. Hikaye kartları müşteri ve programcılar arasında gereksinimler hakkında fikir alış verişini destekler. Programcılar kullanıcı hikayeleri hakkında tartışarak implementasyon süreleri hakkında fikir edinebilirler. Oluşan sorular müşteriye hemen yöneltilebilir.

Her kullanıcı hikayesi için müşteri öncelik sırası (Prio) belirler ve hikaye kartını tahmin yapılmak üzere programcılara verir.

Programcı Tahmin Eder

Programcı bir kullanıcı hikayesini implementasyon öncesi tüm detaylarıyla bilmek zorunda değildir. Bir kullanıcı hikayesini en son detayına kadar kavramaya çalışmak zaman kaybı olabilir, çünkü kullanıcı hikayesinde yer alan müşteri gereksinimi değişikliğe uğrayabilir. Bu genelde müşterinin programın ilk sürümlerini görmesiyle gerçekleşir. Çalışır bir program aracılığıyla müşteri gereksinimlerini daha iyi kavrayacak ve gerekli değişiklikleri talep edecektir. Bu sebepten dolayı implementasyona başlamadan önce kullanıcı hikayesinin ihtiva ettiği tüm detayları tespit etmek faydalı olmayacaktır, çünkü kullanıcı hikayesi değişikliğe uğrayabilir. Programcı ekibin kullanıcı hikayesi hakkında implementasyon için gerekli zamanı tahmin edebilecek kadar bilgiye sahip olması yeterli olacaktır.

Programcılar müşteri tarafından seçilen kullanıcı hikayesinin implementasyon süresini tahmin ederler. Bu tahminler **planlama pokerinde** yapılır.

Resim 2.4 Planlama poker kartları[3]

Planlama pokeri yapılmayan tahminler bazen sağlıklı sonuçlar vermeyebilir. Bir programcı tarafından yapılan tahmin diğer programcıları etkileyebilir. Ya da bazı programcılar herhangi bir sebepten dolayı tahmin etme sürecine aktif olarak dahil olmayabilirler. Bu gibi sebeplerden dolayı oluşan tahmin süreleri yanıltıcı olabilir. Daha geçerli tahminler elde edebilmek için planlama pokeri oynanır.

Planlama pokeri için kullanılan kartlar resim 2.4 de yer almaktadır. Her programcı bu kartların bir setine sahiptir. Planlama pokeri şu şekilde oynanır: Bir moderatör ilk kullanıcı hikayesini okur. Müşteri bu kullanıcı hikayesi için implementasyon süresinin ne olduğunu sorar. Programcılar kısa bir zaman düşündükten sonra hep beraber planlama poker kartlarından birisini seçerek gösterirler. Çoğu zaman kullanılan kartlardaki değerler farklı olacaktır. En çok süreyi ve en az süreyi tahmin eden programcılardan bu sonuca nasıl vardıklarının açıklanması istenir. Verilen bilgiler doğrultusunda ortak bir değer bulunur.

[3] Kaynak: http://www.crisp.se/planningpoker/crispdeck.jpg

Resim 2.5 Planlama pokeri oynayan programcılar[4]

Tahminler için **hikaye puanları** (story points) kullanılır. 1 hikaye puanı örneğin 1 iş günü (8 saat) olabilir. Programcılar her kullanıcı hikayesini kendi başına tahmin etmek yerine, kullanıcı hikayelerini birbirleriyle kıyaslayarak tahminde bulunurlar. Örneğin kullanıcı hikayesi A için 2 hikaye puanı tahmin edilmişse, kullanıcı hikayesi B bu değer göz önünde bulundurularak tahmin verilir. Eğer kullanıcı hikayesi B A dan üç katı daha büyükse, o zaman B için tahmin 2x3 = 6 hikaye puanı olarak verilir.

Load Factor

Bir kullanıcı hikayesinin ideal şartlarda implementasyonu için gerekli zaman dilimi ile normal şartlarda implementasyonu için gerekli zaman dilimi farklı olacaktır. Örneğin programcılar gün boyunca yazılım haricinde toplantı, bilgi alış verişi gibi işler için zaman ayırmak zorundadır. Bir programcının sekiz saatlik bir iş gününde sekiz saat program yazabilmesi **ideal zaman dilimi** olarak tanımlanır. Toplantı ve diğer işler için kullanılan zaman ideal zaman diliminde çıkartıldığı zaman **normal zaman dilimi** elde edilir. Kullanıcı hikayelerinin tahminlerinde ideal ve normal zaman dilimlerinin göz önünde bulundurulması gerekmektedir, aksi taktirde kullanıcı hikayesi için yapılan tahmin gerçekleri yansıtmayacaktır. Gerçekçi bir tahmin yapabilmek için **load factor** olarak bilinen değer kullanılır. Bu değer bir kullanıcı hikayesinin implementasyonu için kullanılan zamanın ideal zamana bölünmesiyle elde edilir. Örneğin bir kullanıcı hikayesi için 1 iş günü (8 saat) tahmin edilmiş ve programcı kullanıcı hikayesini 2 iş gününde tamamlamış olsun. Bu durumda load factor 16 / 8 = 2 olacaktır. Load factor için 2 ile 5 arasında bir değer normaldir. Tahmin yapılırken tahmin edilen implementasyon zamanı load factor ile çarpılır. Örneğin programcı bir kullanıcı hikayesini 2 iş gününde implemente edebileceğini düşünüyorsa, kullanıcı hikayesi için tahmin süresi 2 değil, 2 iş günü x 2 load factor = 4 olmalıdır. Bu şekilde daha gerçekçi tahmin elde edilir.

Programcılar load factor değerini göz önünde bulundurarak, tahminde bulunurlar.

Programcı Dener

Eğer programcılar bir kullanıcı hikayesi için tahminde bulunamazlarsa küçük çaplı bir demo implementasyonu yaparak implementasyon süresini tahmin etmeye çalışırlar. XP dilinde bu işe **spike solution** ismi verilir. İdeal şartlarda bu işlemin sürüm planlama oyunundan önce yapılmış olması gerekir. Buradan sürüm planlama oyunu için kullanıcı hikayelerinin oyun öncesi hazırlanmış olması gerektiği sonucunu çıkartabiliriz.

Programcılar yaptıkları denemeler sonunda kullanıcı hikayesi için tahmin verebilecek duruma gelirler.

Müşteri Kullanıcı Hikayesini Böler

Her sürüm birden fazla iterasyondan oluşur. Her iterasyon bir ile iki haftalık zaman dilimini kapsar. Seçilen kullanıcı hikayelerinin bir iterasyon bünyesinde implemente edilebilir büyüklükte olması gerekmektedir. Aksi taktirde implementasyon bir iterasyon bünyesinde

[4] Kaynak: http://farm3.static.flickr.com/2237/2525997374_78ec1a216a.jpg?v=0

yapılamaz. Bu istenmeyen bir durumdur. Bu durumda müşteriden kullanıcı hikayesini iki ya da daha fazla kullanıcı hikayesine bölmesi istenir. Müşteri kullanıcı hikayesini bölerek, yeni oluşan kullanıcı hikayelerini tekrar tahmin edilmek üzere programcılara verir.

Müşteri Kullanıcı Hikayelerinin Sırasını Belirler

Programcılar tarafından her kullanıcı hikayesi için tahminler yapıldıktan sonra, müşteri kullanıcı hikayelerini program açısından sahip oldukları değerlere göre sıraya koyar.

Tablo 2.1: Kullanıcı hikayeleri ve değerlilik sırası

Değer	Kullanıcı Hikayesi
Yüksek	A (3)
	B (2)
	C (1)
Orta	D (3)
	E (3)
	F(2)
Düşük	G (2)
	F (1)
	H (1)

Tablo 2.1 de müşteri tarafından gruplanan kullanıcı hikayeleri yer almaktadır. Her kullanıcı hikayesi için yapılan tahmin parantez içinde yer almaktadır.

İterasyon Süresi Belirlenir

Her sürüm birden fazla iterasyondan oluşur. Bir iterasyon bir ile iki haftalık bir zaman dilimini kapsar. İterasyon sonunda test edilmiş ve çalışır bir sistem müşteriye kullanılmak üzere sunulur.

İterasyonun uzunluğu bellidir. Bu bir hafta, iki hafta ya da dört hafta olabilir. İterasyon için seçilen kullanıcı hikayelerinin bu iterasyon süresinde implemente edilebilir olmaları gerekmektedir. İmplementasyonu bitirebilmek için iterasyon süresi dinamik olarak değiştirilmez. Eğer yetişmeyen kullanıcı hikayeleri varsa, bunlar yarım bırakılır ya da hiç başlanmadan bir sonraki iterasyona devredilir. Bir iterasyonda ne kadar kullanıcı hikayesinin implemente edilebileceği programcıların çalışma hızına (velocity) bağlıdır.

Programcı Çalışma Hızını Bildirir

Programcıların bir iterasyon (1 ile 2 haftalık süreç) bünyesinde implemente edebildikleri kullanıcı hikayelerinin hikaye puan toplamı, ekibin o iterasyondaki hızıdır. (velocity). Örneğin ekip bir iterasyonda 3 kullanıcı hikayesini implemente etmiş ve bu kullanıcı

hikayelerinin hikaye puan toplamı 5 ise (2+2+1), iterasyon çalışma hızı 5 dir. Sürüm planı oluşturulabilmesi için bu hızın bilinmesi gerekmektedir. Her iterasyon süresi belli olduğu için, bu iterasyon süresini aşacak şekilde kullanıcı hikaye seçimi yapılamaz. Eğer bir önceki iterasyon çalışma hızı 5 hikaye puanına eşitse, bir sonraki iterasyon için 5 hikaye puanına eşit değerde kullanıcı hikayesi seçilir. Eğer ilk iterasyon için planlama yapılıyorsa, o zaman programcıların çalışma temposunu tahmin etmeleri gerekmektedir. Her yeni iterasyonla bu değer daha doğru bir hal alacaktır.

Müşteri Sürümün Kapsamını Belirler

Sürüm planlama oyununun son basamağında müşteri sürüm içinde implemente edilecek olan hikayeleri seçer.

Tablo 2.2: Sürüm Planı

İterasyon	Hikaye Puanı	Kullanıcı Hikayesi
1	3	A
	2	B
2	1	C
	3D	
	1H	
3	3	E
	2F	
4	2	G
	1	F

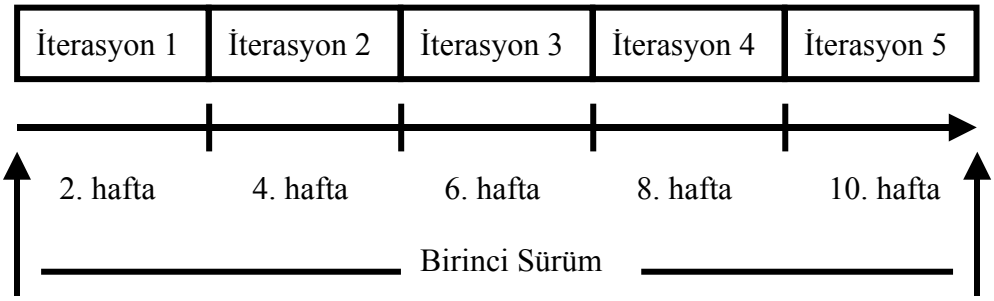
Sürüm planı projenin başlangıcında yapılan ve bir daha değişmeyen bir plan değildir. Müşteri herhangi bir iterasyonda yeni bir kullanıcı hikayesinin implementasyonunu talep edebilir. Bu durumda sürüm planının aktualize edilmesi gerekebilir. Sürüm planının düzenli aralıklarla kontrol edilmesi gerekmektedir. Özellikle her iterasyondan sonra programcıların çalışma hızı (velocity) tekrar hesaplanabilir olduğundan, bir sonraki iterasyon planı bu değer göz önünde bulundurularak tekrar gözden geçirilir.

İterasyon Planlaması (Iteration Planning)

Sürüm planı projenin uzun vadeli planlamasını ihtiva eder. Daha kısa zaman birimlerini daha detaylı planlayabilmek için iterasyon planları oluşturulur.

Yazılım sistemini tek solukta implemente etmek yerine XP projelerinde yapılacak iş belirli uzunlukta olan iterasyonlara bölünür. Her iterasyon iki ile dört haftalık zaman dilimlerini kapsar. İterasyon süresi sabittir. Eğer iterasyon bünyesinde seçilmiş olan kullanıcı hikayeleri

implemente edilemezse, iterasyon süresi değiştirilmez. Bunun yerine implemente edilemeyen kullanıcı hikayeleri bir sonraki iterasyona devredilir.



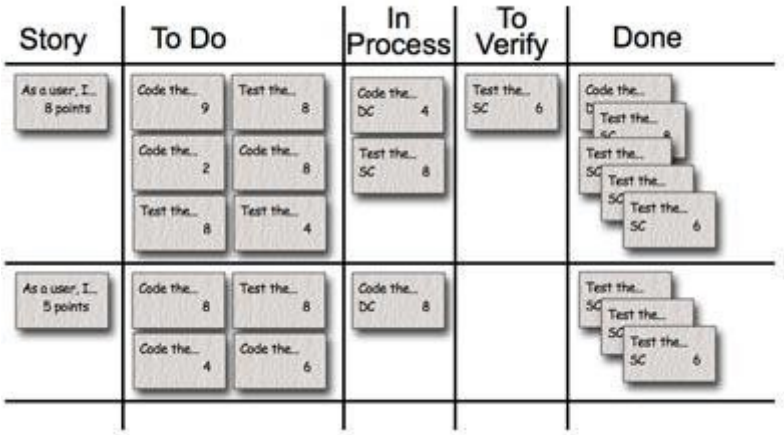
Resim 2.6 Sürüm ve iterasyon

İterasyon planını oluşturmak amacıyla iterasyon öncesi **iterasyon plan toplantıları** düzenlenir. Bu toplantılar bir ile dört saat arasında bir zaman diliminde gerçekleşir. Bu toplantılarda, iterasyon bünyesinde implemente edilmek üzere seçilmiş olan kullanıcı hikayeleri tekrar gözden geçirilir. Diğer iterasyonlarda edinilen tecrübeler doğrultusunda tahminler tekrar gözden geçirilir. İterasyon plan toplantılarına müşteri, programcılar, testçiler, tasarımcılar ve yazılım sisteminin oluşumunda sorumlu herkes katılır. Müşteri kullanıcı hikayelerinin sırasını tekrar gözden geçirebilir, istediği kullanıcı hikayesini iterasyondan çıkartıp, yeni bir kulllanıc hikayesini iterasyon planına ekleyebilir.

Resim 2.7 İterasyon plan toplantısı[5]

Programcılar kullanıcı hikayelerini gözden geçirerek görev (task) listesi oluştururlar. Bu görevler **görev kartlarına** (task card) yazılır. Sürüm planında olduğu gibi görevlerin implementasyon süresi poker kartları kullanılarak programcılar tarafından tahmin edilir. Tahminler saat bazında yapılır. Görev kartları ait oldukları kullanıcı hikayesinin yer aldığı hikaye kartıyla gruplandırılır (resim 2.8).

[5] Kaynak: http://www.industriallogic.com/xp/planning/release.jpg



Resim 2.8 Görev kartları kullanıcı hikaye bazında gruplanır[6]

Tespit edilen görevler doğrudan programcılara atanmaz. Programcılar iterasyon başlamadan önce üzerinde çalışacakları ilişkili bir ya da iki görev kartı seçerler.

Tavsiye

İterasyon planlaması yapılırken, kullanıcı hikayesinden yola çıkarak görevler yerine, kullanıcı hikayesinin implementasyonunu teyit eden akseptans (onay/kabul) test listesi oluşturulmalı ve implementasyon bu testlerle start almalıdır, yani görev kartları (task card) yerine **test kartları** oluşturulmalıdır. Görev listesine bakılarak yazılım sisteminde ne oranda bir gelişme olduğunu anlamak çoğu zaman mümkün değildir. Örneğin aşağıdaki görev listesinde yer alan üç numaralı göreve bir göz atınız. Sizce bu görev, tamamlandığında ne orada bir ilerlemenin kaydedildiğini ifade ediyor? Oluşturulan görevler bazen implementasyonun gerçek amacının gözden kaçırılmasına sebep olabilirler. Görevler bize ne yapılması gerektiğini söylerler, ama somut olarak hangi adımların atılması gerektiğini söylemezler.

Görev listesi:

1. LoginController sınıfı işletme (business) katmanında bulunan LoginManager sınıfını kullanarak login işlemini gerçekleştirir. LoginManager sınıfını oluştur.
2. LoginManager sınıfı LoginDao üzerinden bilgibankası işlemlerini gerçekleştirir. LoginDao isminde bir sınıf oluştur.
3. Üye bilgileri bilgibankasında customer isimli tabloda tutulur. Bu tablo yeterli mi, kontrol et.

Akseptans test listesi:

1. Kullanıcı login sayfasına gider. Email adresi ve şifre alanlarını boş bırakarak login butonuna tıklar. Kullanıcıya "Lütfen email adresinizi ve şifrenizi giriniz!" hata mesajı gösterilir.

[6] Kaynak: http://www.mountaingoatsoftware.com/system/hidden_asset/file/28/MockedTaskBoard_small.jpg

2. Kullanıcı login sayfasına gider. Email adresini girer ve şifre alanını boş bırakarak login butonuna tıklar. Kullanıcıya “Lütfen şifrenizi giriniz!” hata mesajı gösterilir.
3. Kullanıcı login sayfasına gider. Email adres alanını boş bırakarak, şifresini girer ve login butonuna tıklar. Kullanıcıya “Lütfen email adresinizi giriniz!” hata mesajı gösterilir.
4. Kullanıcı login sayfasına gider. Email adresi ve şifreni girer ve login butonuna tıklar. Email adresi ve şifre doğrudur. Login işlemi gerçekleşir. Üye, hoş geldiniz sayfasına yönlendirilir.
5. Kullanıcı login sayfasına gider. Email adresi ve şifreni girer ve login butonuna tıklar. Email adresi geçersizdir. Kullanıcıya “Email adresiniz geçersizdir, lütfen tekrar ediniz!” hata mesajı gösterilir.
6. Kullanıcı login sayfasına gider. Email adresi ve şifreni girer ve login butonuna tıklar. Email adresi geçerli olmasına rağmen şifre hatalıdır. Kullanıcıya “Şifre hatalı, lütfen tekrar deneyiniz!” hata mesajı gösterilir.

Akseptans testleri görevlere kıyasla daha temsili ve somut yapıdadır. Onlar neyin nasıl yapılması gerektiği hakkında bilgi ihtiva ederler ve böylece programcının işini kolaylaştırırlar. Akseptans test listesinde bulunan birinci teste baktığınızda ne yapmanız gerektiğini hemen anladınız değil mi? Akseptans testleri yazılım sistemindeki özelliklerle birebir örtüşür ve bu yüzden ilerlemenin ölçülebilir olmasını sağlar. Ben bu yüzden iterasyon planlamasında görev yerine akseptans testleri oluşturulmasını ve implementasyonunu diğer bölümlerde yakından göreceğimiz gibi top down TDD tarzı yapılmasını tavsiye ediyorum.

Sürüm/İterasyon Planı Nerede?

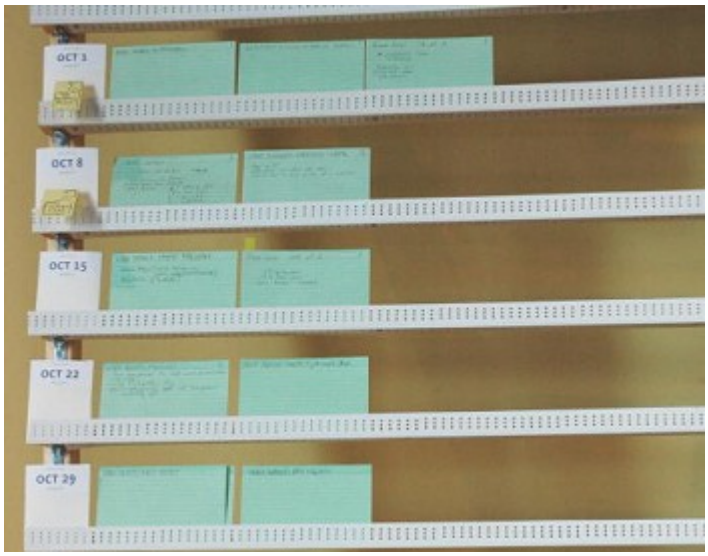
XP projelerinde sürüm ve iterasyon planları tüm proje çalışanları tarafından görülebilecek bir yerdedir. Çoğu zaman duvar panoları ve beyaz tahtalar kullanılarak kullanıcı hikayeleri ait oldukları sürüm ve iterasyon belli olacak şekilde gruplanır. Resim 2.9 de bunun bir örneği yer almaktadır.

Resim 2.9 Sürüm ve iterasyon planını ihtiva eden pano

Panonun ilk üç katında tamamlanmış iterasyonlar ve ihtiva ettikleri kullanıcı hikayeleri yer almaktadır. Tamamlanan iterasyonlar, iterasyonun başlangıç tarihini ve implementasyon için kullanılan zamanını (5 = 5 gün) ihtiva eden beyaz kartlarla birbirini takip etmektedir.

Resim 2.10 Sürüm ve iterasyon planını ihtiva eden pano

Panonun 4-12 katları planlanan 9 iterasyonu ihtiva eder. Her iterasyon 1 hafta olduğu için panonun bu katlarında 9 haftalık çalışma süresi planlanmıştır. 4. kat aktüel iterasyondur. Her katın sol bölümünde o iterasyonda implemente edilecek olan kullanıcı hikayeleri yer alır. Her katın sağ bölümünde önemlilik derecesi düşük olan ve henüz iterasyon planlarında yer almamış kullanıcı hikayeleri bulunur.

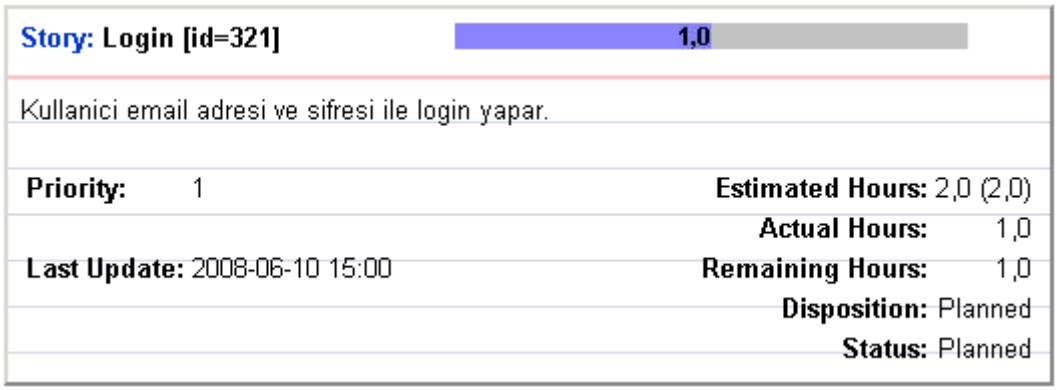


Resim 2.11 Sürüm ve iterasyon planını ihtiva eden pano

Resimlerde görüldüğü gibi hikaye kartları çok basit araçlar kullanılarak oluşturulabilirler. Hikaye kartlarının ana amacı programcı ekip ve müşteri arasındaki diyaloğu kuvvetlendirmek ve projede kaydedilen ilerlemeyi görsel olarak sağlamaktır. Bir bilgisayar programı kullanılarakta hikaye kartları oluşturulabilir. Lakin bu yöntem hikaye kartları kadar çevik olmayacaktır, çünkü bir bilgisayar ya da notebook çalıştırılıp, hikaye kartlarına ulaşılana kadar az denilmeyecek bir zaman geçebilir. Oysa hikaye kartları bir pano üzerinde herkesin görebileceği şekilde dizilebilir ve kullanılabilirler. Değişik lokasyonlarda beraber çalışmak zorunda olan ekipler için dijital hikaye kart yönetim sistemleri düşünülebilir.

Dijital Hikaye Kartları

Hikaye kartları dijital ortamda da oluşturulabilir. Bu işlem için açık kaynaklı olan XPlanner[7] programı kullanılabilir. XPlanner web tabanlı proje yönetim ve takip programıdır. XPlanner ile iterasyonlar ve ihtiva ettikleri dijital hikaye kartları oluşturulur. Her dijital kart bünyesinde görev (task) listesi oluşturulabilir. Ayrıca bu kartlar üzerine unit ve akseptans testleri not edilebilir. Böyle bir XPlanner dijital hikaye kartını bir sonraki resimde görmekteyiz.



[7] http://www.xplanner.org

Resim 2.12 Dijital hikaye kartı

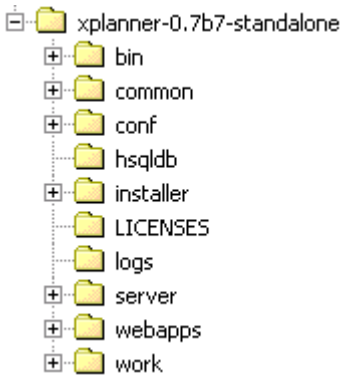
Sol üst kösede kullanıcı hikayesinin ismi (Story: Login) ve numarası (id=320) bulunmaktadır. Kartın ilk satırında kullanıcı hikayesi hakkında açıklama yer almaktadır: "Bir sistem kullanıcısı sahip olduğu email adresi ve şifresi ile sisteme login yapabilir". Kullanıcı hikayesinin öncelik sırası müşteri tarafından 1 (Priority:1) olarak tanımlanmıştır. Hikaye kartının üzerinde bulunan diğer bölümler sırasıyla şöyledir:

- **Estimated Hours:** Bu Login isimli kullanıcı hikayesi için tahmin edilen implementasyon süresidir. Burada programcı ekip 2 saat implementasyon süresi belirlemiştir.
- **Actual Hours:** İmplementasyon için programcı ekip tarafından harcanan zamandır. Programcı ekip bu kullanıcı hikayesi için 1 saatlik bir çalışma yapmıştır.
- **Remaining Hours:** İmplementasyon için geriye kalan zaman dilimidir. Programcı ekip kalan 1 saat içinde bu kullanıcı hikayesini implemente etmelidir.
- **Disposition:** Burada Planned, Carried Over ve Added statüleri yeralabilir. Planned kullanıcı hikayesinin iterasyona, iterasyona başlanmadan önce eklenmiştir anlamına gelmektedir. Carried Over kullanıcı hikayesinin bir önce iterasyondan alındığını ve Added statüsü kullanıcı hikayesinin bu iterasyona başlandıktan sonra eklendiğini gösterir.
- **Status:** Kullanıcı hikayesinin belli aşamalardaki statüsünü gösterir. Burada Draft, Defined, Estimated, Planned, Implemented, Verified ve Accepted gibi bir değerler yer alabilir. Kullanıcı hikayesi ilk oluşturulduğunda Draft statüsündedir. Programcı ekip tarafından implementasyon tahminleri yapıldığı zaman Estimated statüsüne geçer. Kullanıcı hikayesi bir iterasyon içinde eklendiğinde Planned, programcı ekip tarafından implemente edildikten sonra Implemented, unit testlerinin doğru çalışması sonucunda Verified ve akseptans testleri olumlu sonuç verdiğinde Accepted statüsüne geçer. Her kullanıcı hikayesinin ulaşması gerektiği en son statüs Accepted olmalıdır. Bu yüzden her kullanıcı hikayesi için en az bir adet akseptans testinin hazırlanması gerekmektedir.

XPlanner

XPlanner'in aktüel sürümünü bu[8] adresten temin edebilirsiniz. Hızlı kurulum ve kullanım için Standalone (xplanner-0.7b7-standalone.zip) versiyonunu indirmenizi tavsiye ederim. Bu versiyon içinde Tomcat Serveri ve HSQL bilgibankası bulunmaktadır. Paketi herhangi bir dizine açtıktan sonra, aşağıdaki dizin yapısı oluşacaktır:

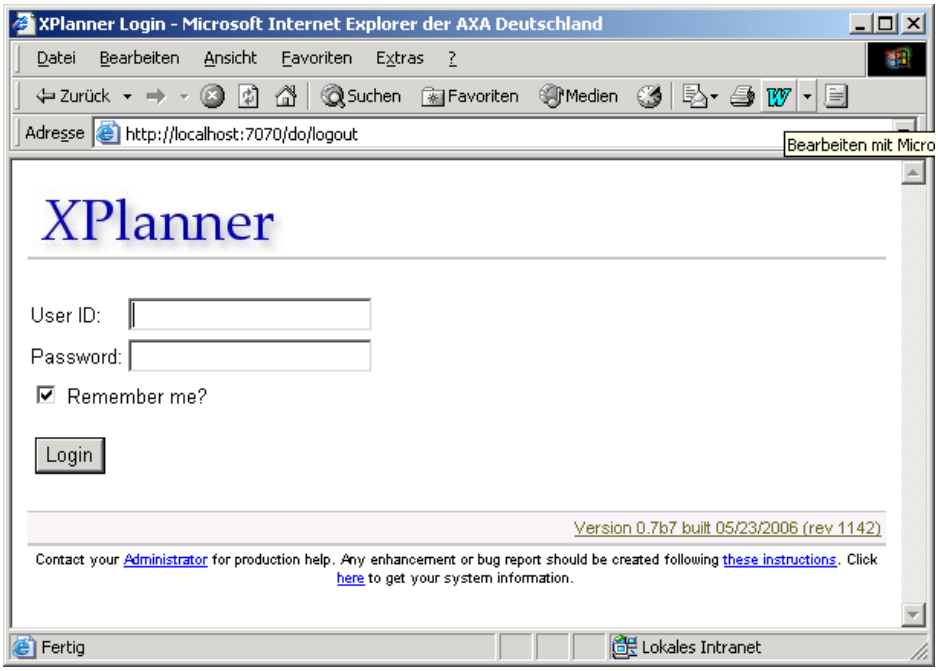
[8] Bakınız: http://sourceforge.net/project/showfiles.php?group_id=49017



Resim 2.13 XPlanner dizin yapısı

Bin dizininde yer alan startup.bat ile XPlanner çalıştırılır. Bilgibankası olarak MySQL ya da HSQL kullanılabilir. JDK 1.4 ve üstü gerekmektedir. XPlanner standalone sürümü HSQL ile çalışacak şekilde konfigüre edilmiştir. Bu yüzden startup.bat ile kurulu bir XPlanner sürümü çalıştırılabilir. Kurulum için detaylı bilgiyi bu[9] adres altında bulabilirsiniz.

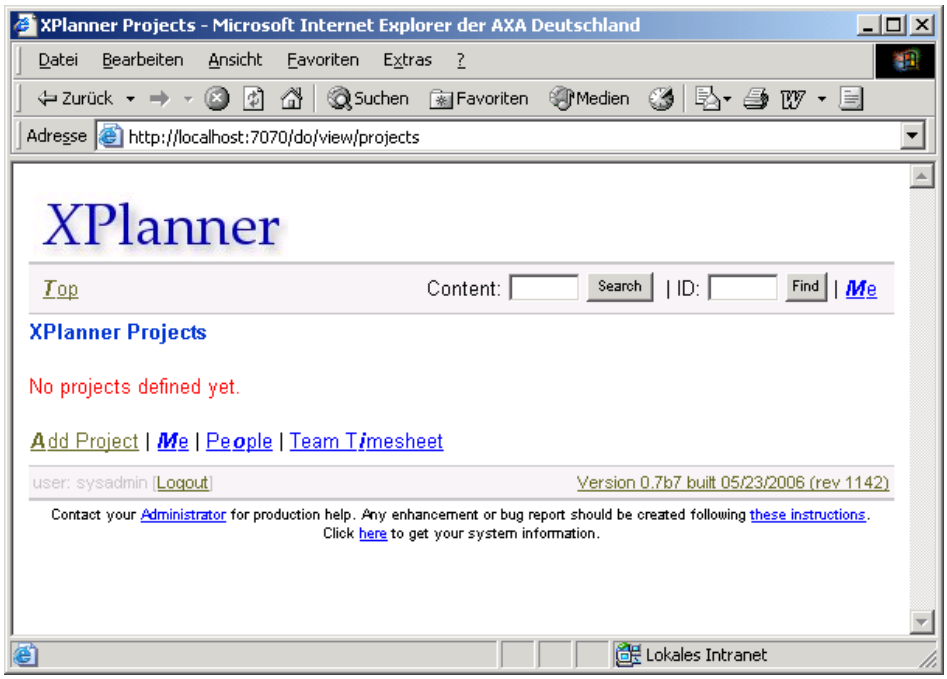
startup.bat ile lokal Tomcat çalıştırıldıktan sonra, http://localhost:7070 adresinden XPlanner'in web tabanlı arayüzüne bağlanabiliriz.



Resim 2.14 XPlanner login sayfası

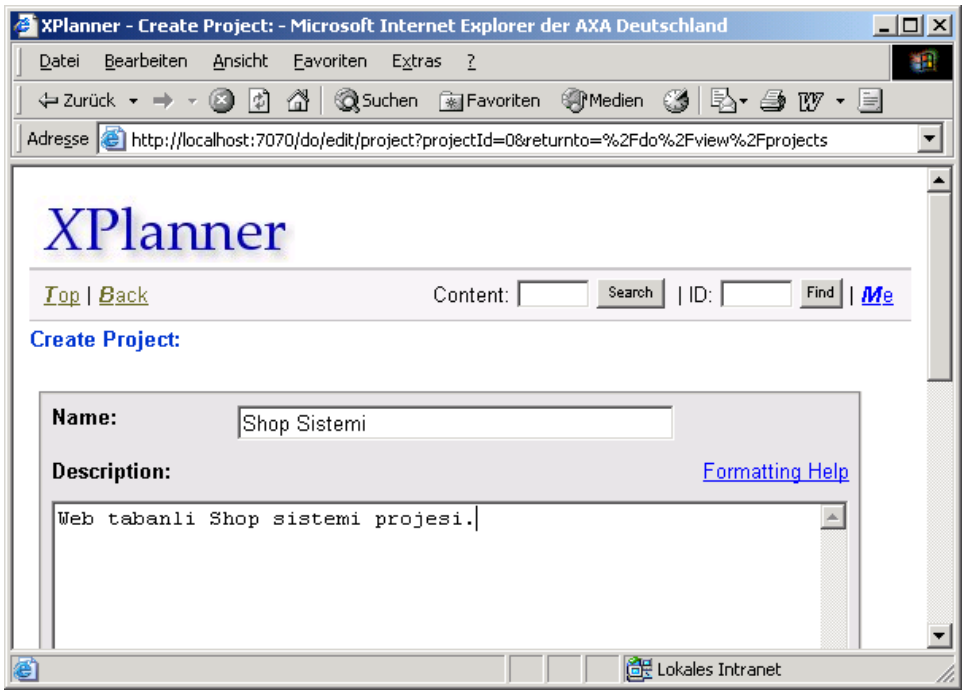
sysadmin/admin kullanıcı ismi ve şifresi ile login yapılabilir. Login işleminin ardından yönetilen projelerin yer aldığı ilk sayfa görüntülenecektir. Henüz bir proje tanımlamadığımız için bu sayfada "No projects defined yet (henüz proje tanımlanmadı)" mesajı yer almaktadır.

[9] Bakınız: http://www.xplanner.org/install.html



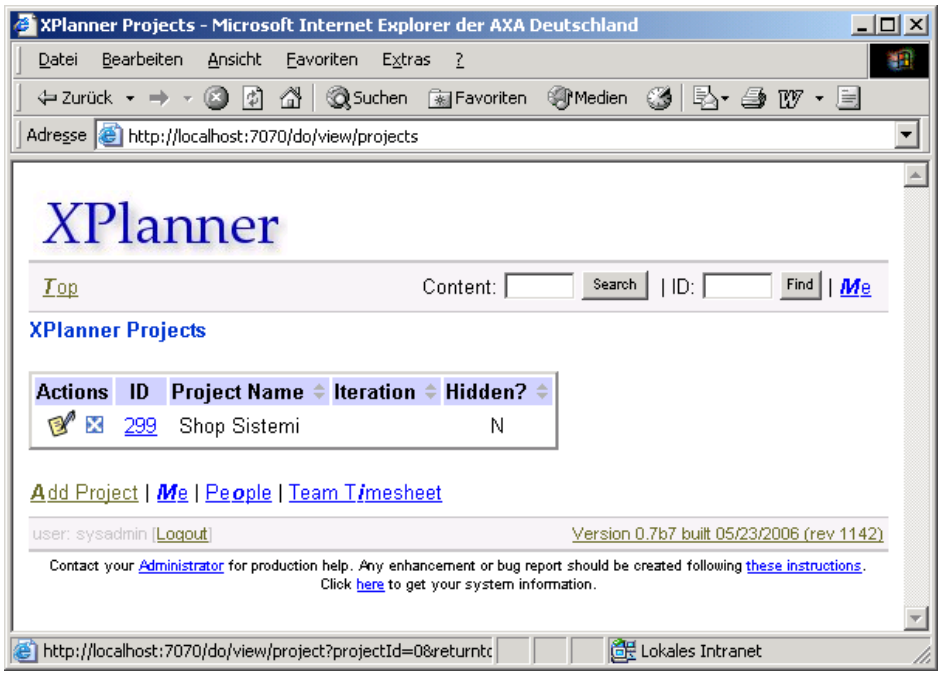
Resim 2.15 XPlanner ana sayfa

Add Project linkine tıklayarak projemizi ekleyebiliriz.



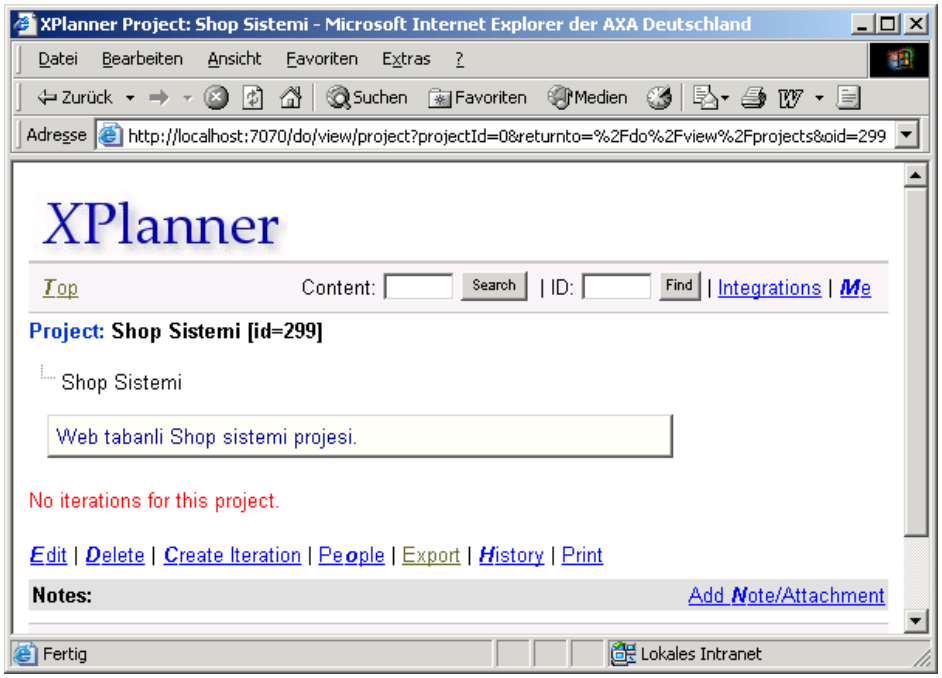
Resim 2.16 XPlanner proje oluşturma sayfası

Bu işlemin ardından Shop Sistemi isminde bir proje oluşturulacaktır. Proje aşağıdaki şekilde proje ana sayfasında görüntülenir.



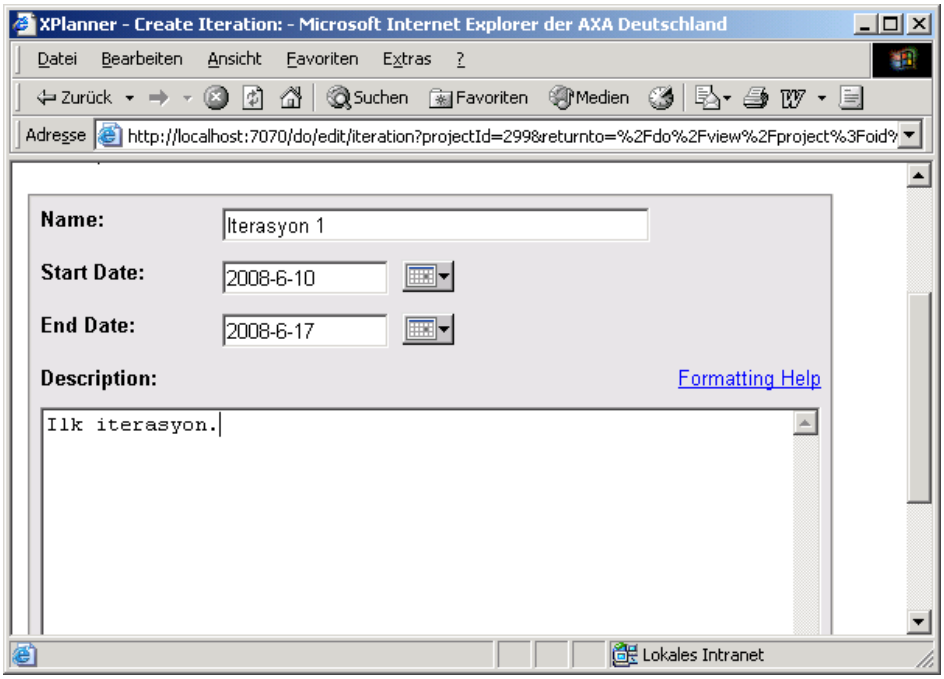
Resim 2.17 XPlanner ana sayfa

XPlanner Shop Sistemi isminde bir proje oluşturmuş ve bu projeye 299 numarasını atamıştır. 299 rakamı üzerindeki linke tıklayarak proje detaylarını görebiliriz.



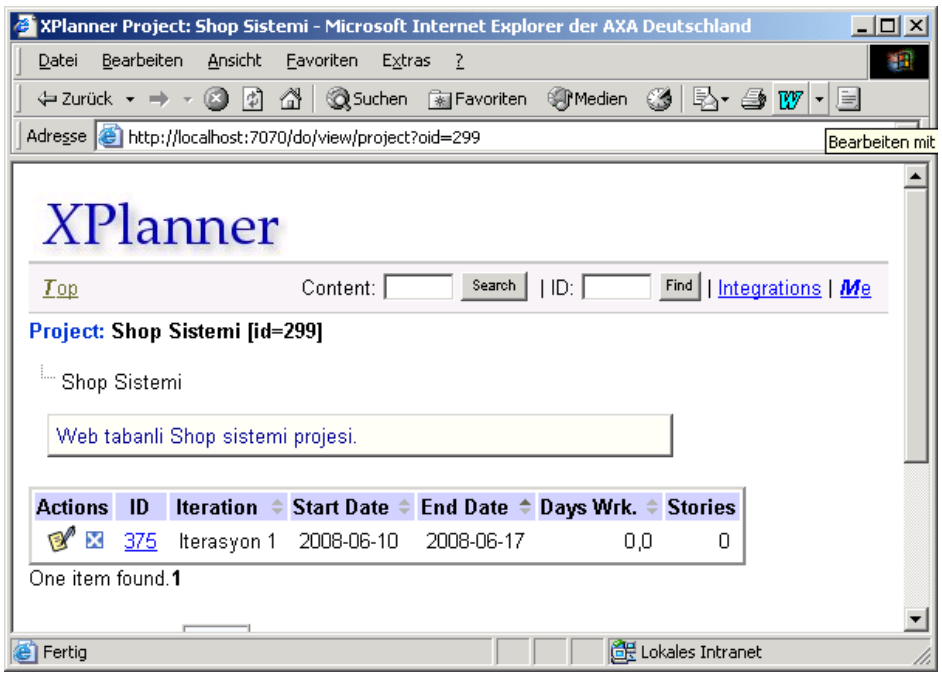
Resim 2.18 XPlanner proje detay sayfası

Her proje için bir veya birden fazla iterasyon oluşturulması gerekmektedir. Her iterasyon implemente edilecek kullanıcı hikayelerini, yani hikaye kartlarını ihtiva eder. Bir iterasyon oluşturmak için Create Iteration linkine tıklıyoruz.



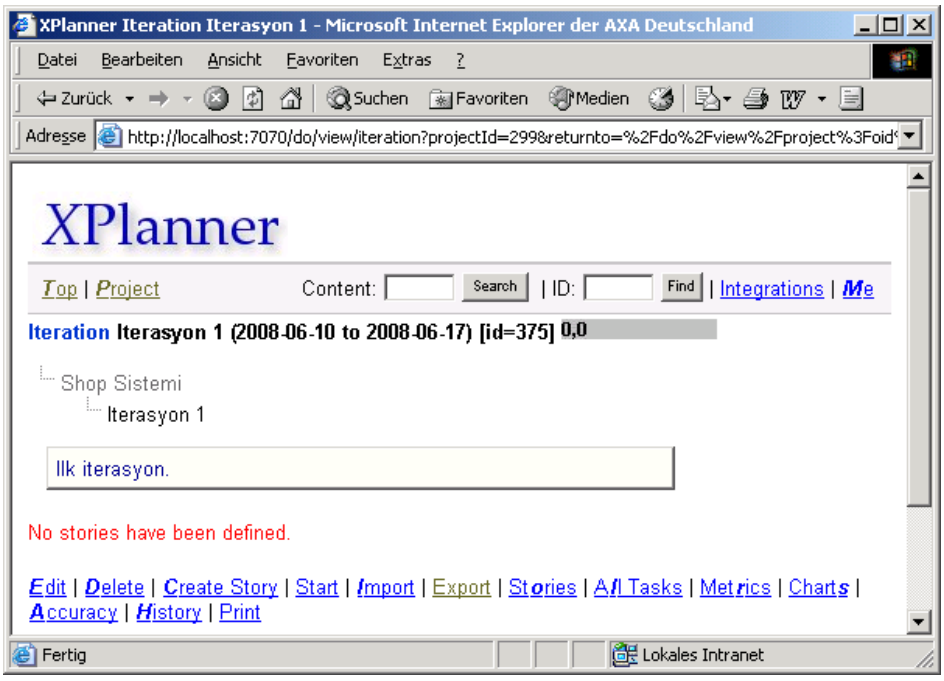
Resim 2.19 XPlanner iterasyon oluşturma sayfası

İlk iterasyonumuzu **İterasyon 1** olarak isimlendiriyoruz. Her iterasyonun bir başlangıç ve bitiş tarihi vardır. Başlangıç tarihi olarak 10.6.2008, bitiş tarihi olarak 17.6.2008 seçiyoruz. Buradan da anlaşılacağı gibi iterasyon sürece 1 hafta olarak seçilmiştir.



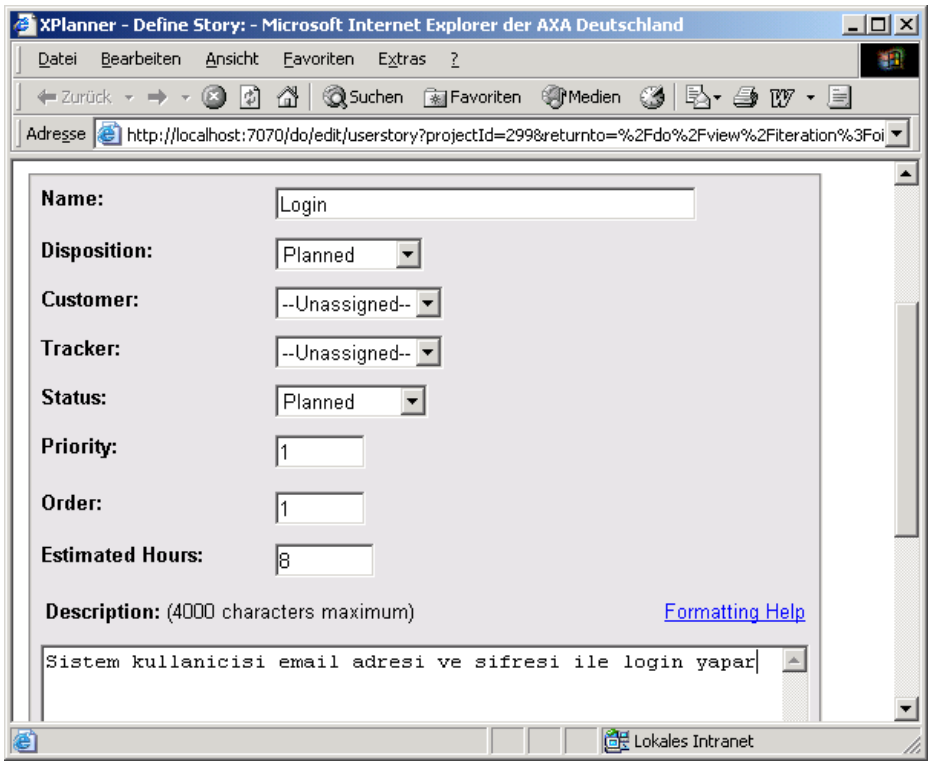
Resim 2.20 XPlanner proje ana sayfa

Bu işlemin ardından XPlanner 375 numaralı ve **İterasyon 1** ismini taşıyan ilk iterasyonu oluşturur. Şimdi bu iterasyon bünyesinde implementasyonu yapılacak olan kullanıcı hikayelerini oluşturmamız gerekiyor. 375 rakamı üzerindeki linke tıklayarak, iterasyon bölümüne geçiyoruz.



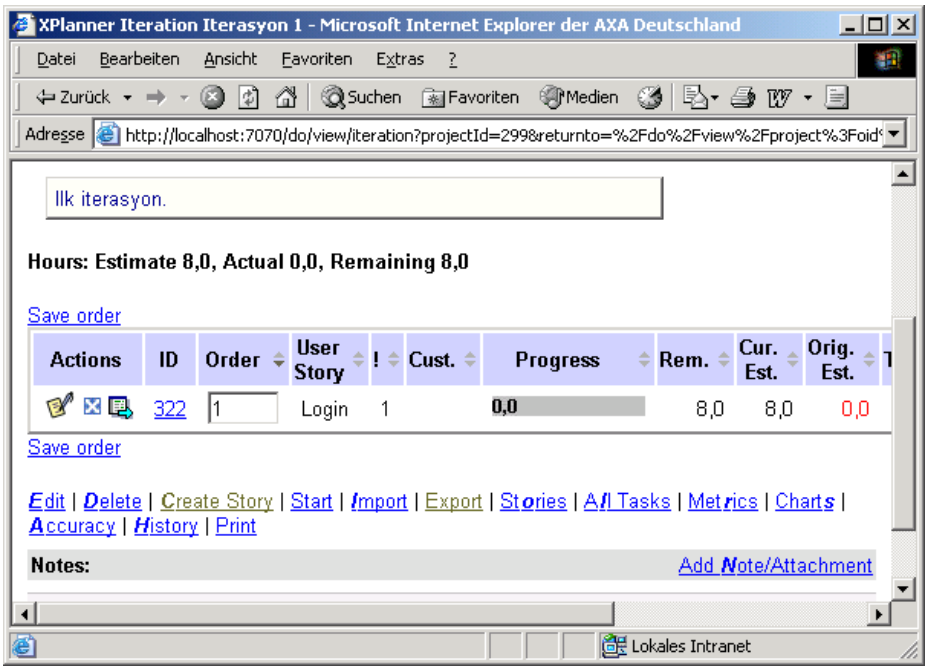
Resim 2.21 XPlanner iterasyon listesi

Bu iterasyon bünyesinde henüz kullanıcı hikayesi tanımlanmadığı için “No stories have been defined (kullanıcı hikayesi tanımlanmadı)” mesajı yer almaktadır. Create story linkine tıklayarak, ilk kullanıcı hikayesini oluşturuyoruz.



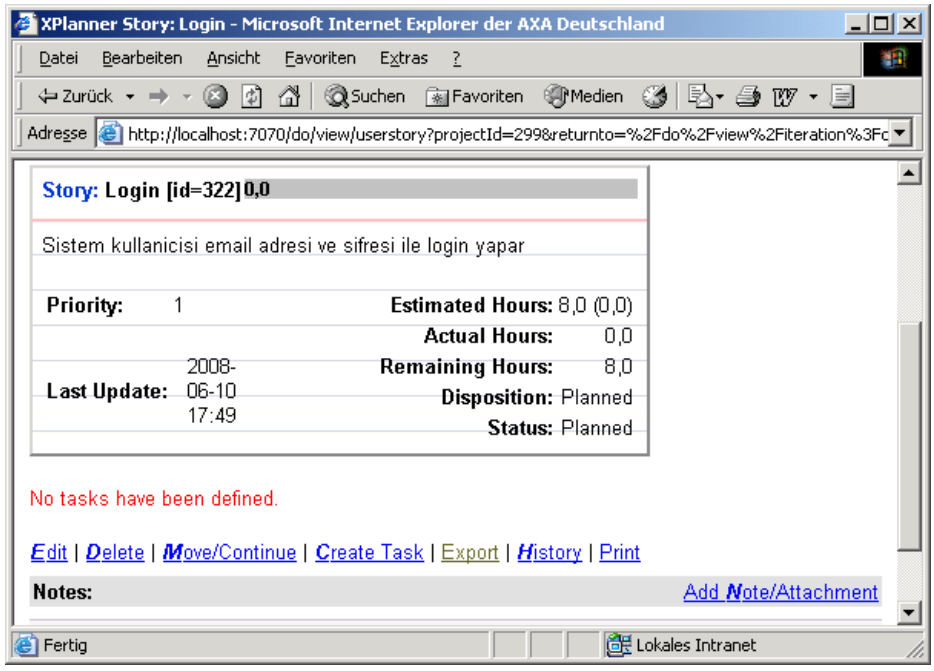
Resim 2.22 XPlanner kullanıcı hikayesi oluşturma sayfası

Bu işlemin ardından XPlanner ilk kullanıcı hikayesini aşağıdaki şekilde oluşturur.



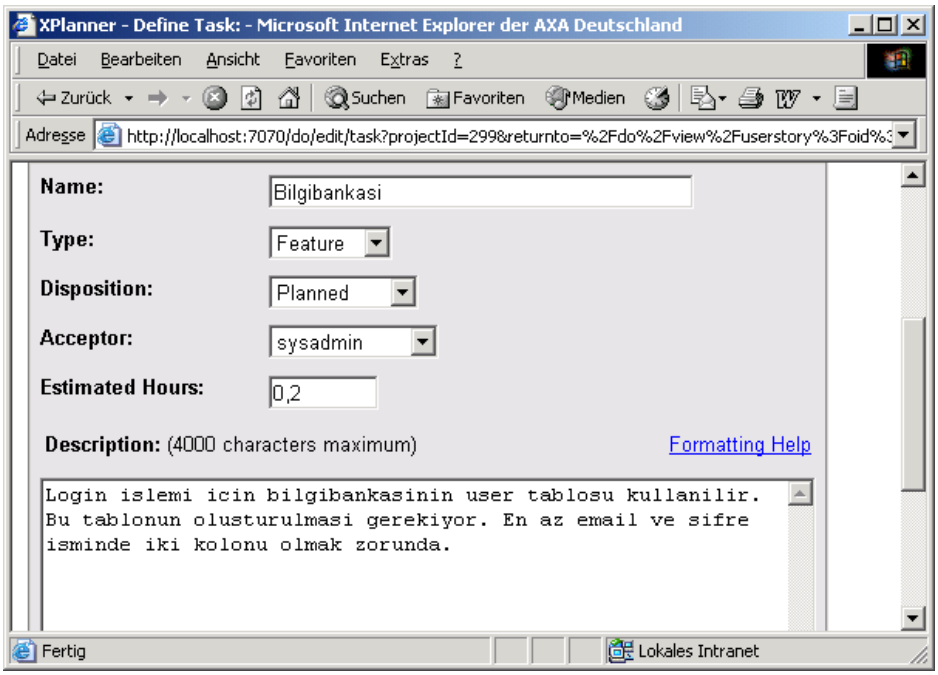
Resim 2.23 XPlanner kullanıcı hikaye listesi

Kullanıcı hikayesi 322 rakamını taşımaktadır. Bu rakamın üzerinde bulunan linke tıklayarak, kullanıcı hikayesinin detaylarını görebiliriz.

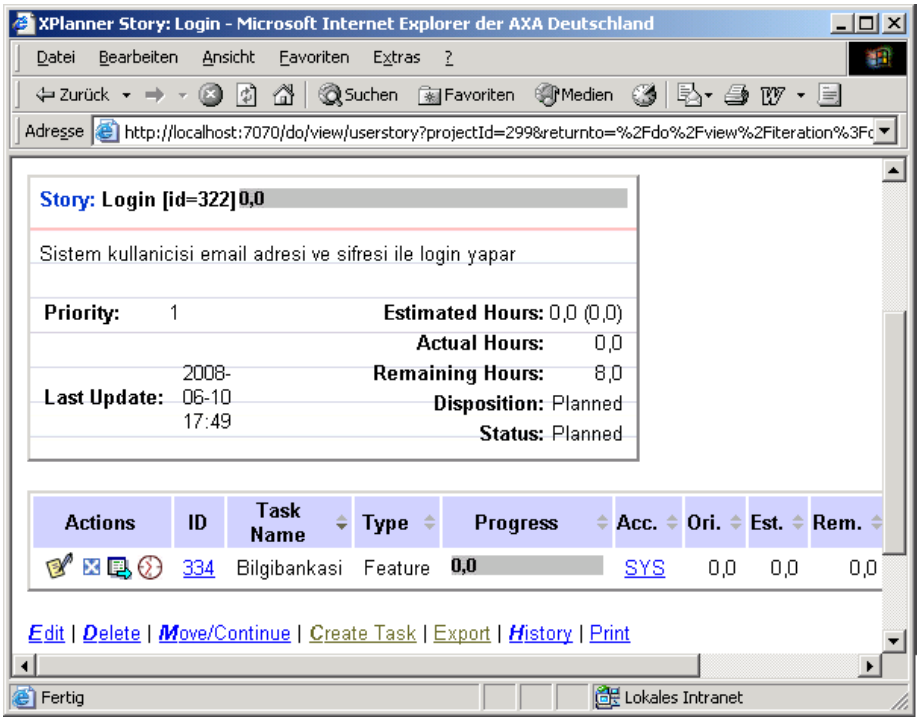


Resim 2.24 XPlanner kullanıcı hikaye detay sayfası

Burada 322 rakamını taşıyan bir hikaye kartı görmekteyiz. Login ismini taşıyan bu hikaye kartı, kullanıcı hikayesi için 8 saatin tahmin edildiğini ve öncelik derecesinin 1 olduğunu göstermektedir. Bu kullanıcı hikayesi bünyesinde görev listesi tanımlanmadığı için "No tasks have been defined (görev listesi tanımlanmamıştır)" mesajı görülmektedir. Programcılar kullanıcı hikayesinin implementasyon zamanını tahmin edebilmek için, bu kullanıcı hikayesi için gerekli görev (task) listesi oluştururlar. Her görev için bir zaman dilimi tespit edilir. Bunların toplamı, kullanıcı hikayesinin tahmin edilen implementasyon (Estimated Hours) zamanıdır. Create Task linkine tıklayarak ilk görevi oluşturabiliriz.

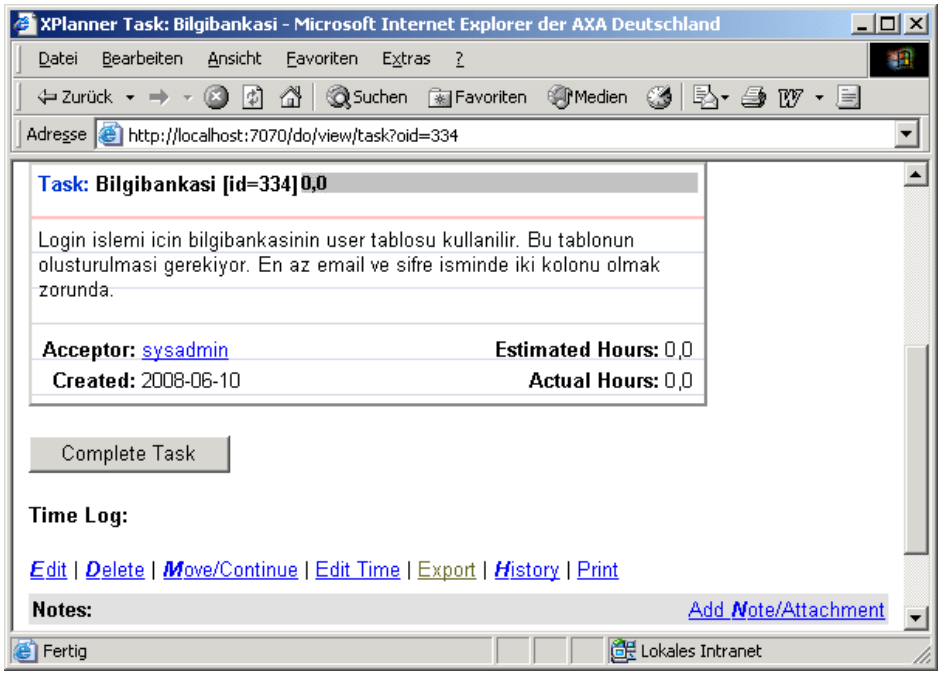


Resim 2.25 XPlanner görev detay sayfası



Resim 2.26 XPlanner kullanıcı hikaye detay sayfası

Hikaye kartına geri döndüğümüzde yeni görevin kartın alt bölümüne eklendiğini görebiliriz. Yeni göreve 334 rakamı verilmiştir. Bu linke tıklayarak görev detay bölümüne geçebiliriz.



Resim 2.27 XPlanner görev detay sayfası

Görev kartları (task card) hikaye kartlarına (story card) benzemektedir. Her görev kartının sahip olduğu başlığı (Task:Bilgibankası) ve numarası ([id=334]) bulunmaktadır. Görev kartına görev hakkında açıklama eklenir. Programcı ekip görevin hangi zaman diliminde bitirilebileceğini tahmin eder. Complete Task butonuna tıklayarak, görevin tamamlandığı belirtilir.

Gördüğümüz gibi XPlanner kullanıcı hikayelerinin ve hikaye kartlarının dijital olarak oluşturulmasını kolaylaştırmaktadır. Tüm ekip bir web tarayıcısıyla projenin gidişatı hakkında bilgi sahibi olabilir. Lakin aynı odada çalışan bir ekip için XPlanner gibi bir program tavsiye edilmemektedir. Aynı odada çalışan programcı ekip kalemle yazılan hikaye kartları tercih etmelidir. Bu şekilde projenin ve aktüel iterasyonun hangi safhada olduğu çok daha kolay anlaşılabilir. Programcılar kartondan olan kullanıcı hikaye kartlarını ellerine alarak, bilgi alış verişinde bulunabilirler ve kartlar üzerinde değişiklik yapabilirler. Bu şekilde bir çalışma sistemi, projenin daha hızlı bir şekilde gerçekleştirilmesini sağlayacaktır. Aynı odada çalışma imkanı bulamayan, değişik bina yada şehirlerde bulunan programcı ekipler için XPlanner gibi bir sistemin kullanılması avantajlı olacaktır. Bu şekilde İnternet üzerinden çalışmalar koordine edilebilir.