Yazılım Mühendisliği Simülasyonu Projesi

190508001 Muhammed Yusuf Akçadağ
Bilgisayar Mühendisliği Bölümü Öğrencisi
Bilgisayar Mühendisliği Uygulamaları I, II Dersi Projesi
Bitirme Projesi

İçerik

Bölüm 0: Giriş

0.1: Yazılım Mühendisliği ve Yazılım Mühendisi Nedir

Kariyer hedefinde **Yazılım Mühendisliğini** seçmiş biri olarak projeye başlamadan önce **"Yazılım Mühendisliği ve Yazılım Mühendisi"** terimlerinden bahsetsem iyi olacak.

Yazılım Mühendisliği, karmaşık yazılım sistemlerinin iş bölümü yapılarak; belirli ilkelerle, yöntemlerle ve araçlarla belirli bir hedefe doğru geliştirilmesi ve geliştirildikten sonra bakımının yapılmasıdır. Yani **Yazılım Mühendisliği** yazılım geliştirme yaşam döngüsüyle ilgilenen bir mühendislik alanıdır.

Mühendis sistem tasarlayan, geliştiren ve geliştirdiği sistemi optimize eden personeldir. **Yazılım Mühendisi** ise, yazılım sistemini tasarlayan, geliştiren ve geliştirdiği sistemi optimize eden personeldir. Yani **Yazılım Mühendisi** yazılım geliştirme yaşam döngüsü aşamalarında sorumluluklar alan ya da daha iyisi olarak bu döngüyü yöneten mühendislik personelidir.

0.2: Yazılım Geliştirme Yaşam Döngüsü Nedir ve Yazılım Geliştirme Modelleri Nelerdir

İnsanlık kendini bildi bileli herhangi bir sistem geliştiriyor ve mühendislik uyguluyor. Tarım sistemleri, inşaat sistemleri, gemi sistemleri, askeri sistemler… Başarılı olabilmek; yani kaynakları doğru kullanmak ve zamanında işi bitirebilmek için hepsinin ortak bir noktası var. Bir plana sadık kalmak. Eğer yazılım sistemi geliştirilmesinde de başarılı olmak istiyorsak yazılım sistemimize uygun bir plan seçip bu plana sadık kalmalıyız.

Projelerdeki yazılım sistemlerinin geliştirilmesi için sadık kaldığımız genel plana **Yazılım Geliştirme Yaşam Döngüsü** diyoruz.

Yazılım Geliştirme Yaşam Döngüsünün temel aşamaları

şunlardır:

Yapılabilirlik Analizi Aşaması: Bu aşamayı uygulayabilmek için geliştirilecek yazılım sistemininin işlevselliği detaylandırılmadan tarif edilirek ön gereksinimler belgesi oluşturulur. Yazılım sisteminin mevcut teknolojiyle, mevcut bütçeyle ve hedeflenen zamanla yapılabilirliğini analiz etme aşamasıdır.

Planlama Aşaması: Yazılım sistemini geliştirmeye başlamadan önce kaynaklarımızı (bütçe, zaman, personel vb.) baz alarak yazılım sisteminin geliştirilmesi hakkında detaylı bir plan oluşturma aşamasıdır.

Analiz Aşaması: Yazılım sistemini hayata geçirmek için yazılım sisteminin tasarımını yapmadan ve yazılım sistemini geliştirmeye başlamadan önce yazılım sisteminin işlevselliklerini kesin ve anlaşılır bir şekilde hem kullanıcılar hem de geliştiriciler için tanımlama aşamasıdır. Bu aşamada yazılım sistemi analizi yapılır. Yani gereksinimler belgesi oluşturulur.

Tasarım Aşaması: Çıkarılan gereksinimler belgesine göre yazılım sisteminin bileşenlerinin belirli ilkelerle, yöntemlerle ve araçlarla tasarlanması aşamasıdır.

Gerçekleştirme Aşaması: Tasarlanan yazılım sistemi bileşenlerinin tasarım planlarına sadık kalarak geliştirilmesi aşamasıdır.

Doğrulama Aşaması: Verilen gereksinimler belgesi aracılığıyla geliştirilen yazılım sisteminin işlevselliklerinin doğruluğunun kontrol edilmesi, yazılım sisteminin bileşenlerinin hatasız ve arızasız çalıştığının belirli ilkelerle, yöntemlerle ve araçlarla test edilmesi, yazılım sisteminin hatasız ve arızasız çalıştığının belirli ilkelerle, yöntemlerle ve araçlarla test edilmesi aşamasıdır.

Yayımlama Aşaması: Doğrulanan yazılım sisteminin, yazılım sistemini isteyen şirkete teslim edilmesi ya da kullanıcıların

kullanımına sunulması aşamasıdır.

Bakım Aşaması: Yazılım sistemini isteyen şirkete teslim edilen ya da kullanıcıların kullanımına sunulan yazılım sisteminin arızalarının giderilmesi, hatalarının giderilmesi, güvenilirliğinin sağlanması, emniyetinin sağlanması, güvenliğinin sağlanması, kullanışsız alanların yenilenmesi, hatalı işlevselliklerin değiştirilmesi, yeni işlevselliklerin eklenmesi ve optimizasyon yapılması aşamasıdır.

Yazılım Geliştirme Modelleri ise yukarıda verilen yazılım geliştirme yaşam döngüsü aşamalarının verilen öneme göre uygulanmasının, tekrar edilme sayılarının, ayrılan sürenin, uygulanacağı zamanın ve içeriklerinin farklılıklarına göre oluşan modellerdir. Bu modeller yazılım sistemi geliştirenler tarafından yukarıda verilen aşamlara doğal olarak bağlı kalacak şekilde tasarlanmışlardır. Çoğu yazılım sistemi geliştirenler kendi yazılım geliştirme modellerini aşağıda verilecek modellerden ilham alacak şekilde tasarlayıp uygulamaktadır.

Şu bilinmelidir ki hiçbir model ve süreç kesin doğru ve 100% etkili değildir. Farklı yazılım sistemi türlerine göre farklı **Yazılım Geliştirme Modelleri** uygulanır. Mesela bir sağlık makinesi yazılım sisteminin geliştirme yaşam döngüsünün analiz, tasarım ve doğrulama aşamaları en önemlidir. Bu aşamalar 100% doğrulukla ve bütünlükle uygulanmalı ki insan canına mâl olacak bir hata meydana gelmesin (solunum cihazı vb.). Lakin bir elektronik ticaret yazılım sistemindeki bir hata en fazla şirketin hizmetini aksatabilir. Ama bir şirket için elektronik ticaret pazarıyla ve müşterileriyle bir an önce buluşmak önemli olduğu için yazılım sisteminin geliştirme yaşam döngüsünün geliştirme, yayımlama ve bakım aşamaları en önemlidir. Son olarak şunu da belirtelim, bir askeri füze projesinin yazılım sistemini geliştirme yaşam döngüsünün bakım süreci hiç önemli olmayabilir çünkü yazılım sistemi 100% doğrulukla çalışır ve donanım değişmeyeceği için yazılım sistemi de değişmez. Eğer farklı gereksinimlere ihtiyaç olursa yeni bir füze projesine başlanır.

Sonuç olarak “**Yazılım Geliştirme Modelleri”** yazılım sistemi geliştirenler tarafından ihtiyaca göre kullanılabilir ya da geliştirilecek yazılım sistemine uyumlu aşağıda verilecek modellerden ilham alacak şekilde tasarlayıp yeni bir model oluşturulabilir. Gelin bu modellerden bazılarına bir göz atalım.

Şelale Model (Waterfall Model) – İlk uygulanan modeldir diyebiliriz. Yeni olan yazılım mühendisliği uygulamasına diğer mühendisliklerden (inşaat vb.) alınan ilhamla geçen yazılım geliştirme modelidir. **Planlama** → **Analiz** → **Tasarım** → **Gerçekleştirme** → **Doğrulama** → **Yayımlama** → **Bakım** aşamalarının sırasıyla ve yüksek doğrulukla yapılması gereken bir modeldir. Bir aşama bitmeden diğer bir aşamaya geçilemeyeceği için aşamalar uzun zaman alır. Herhangi bir aşamada ve özellikle başlardaki aşamalarda yapılan ve son aşamalarda fark edilen bir hata yüksek maliyetlere sebep olur (hem parasal hem zamansal).

Bu modeli bir inşaat sistemine uyguladığımızda mükemmel çalışır ama her zaman ihtiyaçların, isteklerin değiştiği ve kullanıcıyla ya da müşteriyle bir an önce buluşması gereken yazılım sistemlerinde tam bir hayal kırıklığıdır. Bunun için yazılım mühendisleri bu aşamaların sürekli ve sık zamanlarda tekrar edilerek uygulandığı, belgelemenin ve prosedürlerin en aza indirildiği, aşamaların çoğunlukla senkronize götürüldüğü çevik modelleri (bu projede bizi ilgilendirmeyen geniş bir konu) geliştirmişlerdir.

Lakin bu model hala askeri yazılım sistemleri, sağlık yazılım sistemleri gibi yüksek doğruluk gerektiren yazılım sistemlerinde ve yap bitir tarzı karmaşık olmayan yazılım sistemlerinde popülerdir.

Tekrarlı Model (Iterative Model) – Şelale modelin katı kurallarından kurtulmak isteyen yazılım mühendislerinin uyguladığı bir modeldir. **Planlama** → **Analiz** aşamaları uygulandıktan sonra **Tasarım** → **Gerçekleştirme** → **Doğrulama** aşamalarının gereksinimler belgesiyle döngü şeklinde daha kısa zaman aralıklarında uygulandığı ve son olarak da **Yayımlama** → **Bakım**

aşamalarının uygulandığı bir modeldir.

Uygulanan döngüler sayesinde parça parça inşa edilen yazılım sistemindeki hatalar ve arızalar diğer parçalar inşa edilmeye başlanmadan fark edilip düzeltilebilecek ve hata maliyeti en aza indirilmiş olacaktır.

Tekrar Yayımlamalı Model (Incremental Model) – Tekrarlı model gibi lakin her döngüde müşteriye ya da kullanıcılara verilen en önemli işlevselliğiyle çalışan yazılım sisteminin olduğu bu modelde; **Planlama** → **Analiz** aşamaları uygulandıktan sonra **Tasarım** → **Gerçekleştirme** → **Doğrulama** → **Yayımlama** aşamalarının gereksinimler belgesiyle döngü şeklinde daha kısa zaman aralıklarında uygulandığı ve son olarak da **Bakım** aşamasının uygulandığı bir modeldir.

Bu modelle birlikte müşteriye ya da kullanıcılara yazılım sisteminin geliştirme ilerleyişini gösteriyoruz. Müşterilerin ya da kullanıcıların en önemli işlevselliğiyle yazılım sistemini kullanmasını sağlıyoruz.

Evrimsel Model (Evolutionary Model) – Artık esnek ve müşterilerin ya da kullanıcıların değişen isteklerine yazılım sistemini geliştirme sürecinde cevap veren bir modele geldik. Benim projemde uygulayacağım model de budur.

Planlama aşaması uygulandıktan sonra **Analiz** → **Tasarım** → **Gerçekleştirme** → **Doğrulama** → **Yayımlama** aşamalarının döngü şeklinde daha kısa zaman aralıklarında uygulandığı ve son olarak da **Bakım** aşamasının uygulandığı bir modeldir.

Her döngüde tekrar tekrar yapılan analizin kullanıcıların istekleriniin en iyi şekilde alınmasını, en gerçekçi ve uygulanabilir gereksinimler belgesinin çıkmasını, çıkan gereksinimler belgesinin tasarım ve gerçekleştirme ekipleri tarafından daha iyi anlaşılabilmesini ve daha iyi anlaşılan gereksinimlerin daha iyi uygulanmasını sağlar.

Geliştirilen yazılım sistemi kullanıcılara yayımlanır.

Modelimin aşamalarından proje raporunun içinde daha detaylı bir şekilde bahsedeceğim için son olarak çevik modellerden kısaca devam edeyim.

Çevik Model – Analizin en hızlı şekilde ve en önemli işlevselliklere göre parça parça yapıldığı; parça parça alınan gereksinimlerin parça parça tasarlanıp gerçekleştirildiği; ürünün en hızlı şekilde yayımlandığı; hızlı bir şekilde hataların, arızaların ve güvenlik açıklarının kapatıldığı; kodun ve mimarinin sürekli optimize edildiği; senkronize bir şekilde tüm aşamaların uygulandığı bir modeldir. **Uç Programlama** ve **Scrum** gibi uygulamalar içerir. Modelim olan **Evrimsel Model** çevik modele en yakın plan odaklı modellerden biridir.

Bölüm 1: Simülasyonumun Konusu

1.1: Simülasyonum

Simülasyonum yaygın olan İnternet üzerinden hizmet vermenin ve iş süreçlerinin yazılım sistemi üzerinden uygulanmasının daha da yaygınlaşmaya başladığı bir zamanda teknolojiye ayak uydurmaya çalışan ve büyüyen genç bir taşıt kiralama şirketinin taşıt kiralama hizmeti için eski, kullanışsız ve iş ihtiyaçlarını karşılamayan yazılım sistemi yerine yeni bir yazılım sistemi istemesinden oluşuyor.

Şirket, ilk başta web uygulaması şeklinde iş süreçlerini uygulamak ve internet üzerinden çevrim içi hizmet vermek istiyor. Yazılım sistemi geliştirme firmalarından aldığı teklifler bütçesini çok çok aşıyor. Bütçesine sınırdan uyan tekliflerin ise yazılım sistemini teslim edecekleri tarih çok uzun geliyor. Yedi (7) şubesi bulunan şirket şimdilik geliştirilmesi daha ucuz olan masaüstü uygulaması teknolojisine yöneliyor ve çevrim içi hizmetten vazgeçiyor. Kendisi gibi büyüyen genç bir yazılım geliştirme şirketinden uygun bir teklif alıyor. Kısa vadede (~8 ay) iş süreçlerini uygulamak için masaüstü

uygulaması geliştirilmesinde anlaşıyorlar. Uzun vadede ise (~16 ay) bu yazılım sisteminin web üzerine geçirilmesinde ve çevrim içi hizmet verilmesinde anlaşıyorlar.

Yazılım sistemini istediğinde yedi (7) şubesi bulunan şirket, bu şube sayılarını masaüstü uygulamayı teslim aldığında dokuza (9) ve web uygulamasını teslim alıp çevrim içi hizmet vermeye başladığında on altıya (16) çıkardı. Şimdilerde ise yirmi dört şubesi (24) bulunuyor.

Bu simülasyondaki her veri **Muhammed Yusuf Akçadağ** yani benim tarafımdan düşünülmüş, tasarlanmış, oluşturulmuş ve yazılmıştır.

1.2: Simülasyonda Yazılımı İsteyen ve Yazılımı Yapan Şirketler

Simülasyonda **"MyCar Rent"** taşıt kiralama şirketi tarafından **"YuSoft"** yazılım sistemi geliştirme şirketinden taşıt kiralama sürecini yönetmek ve otomatize etmek, taşıt takibini yapmak, bütçeyi yönetmek vb. için yazılım sistemi istenmiştir.

1.3: MyCar Rent Şirketi Hakkında

"MyCar Rent" yönetim şubesi **Tokat Şubesi** olan; Tokat (1), Malatya (1), Ankara (2), İzmir (1) ve İstanbul'da (2) hizmet veren; birçok taşıt türünde kişisel kiralama hizmeti sunan bir şirkettir. Şirket büyüyen hizmetlerini daha rahat yönetebilmek için işlevsel ve güvenilebilir yeni bir yazılım sistemine ihtiyaç duymaktadır. Eski yazılım sistemi artık gereksinimleri karşılayamamaktadır. (Parantezler şehirlerde bulunan şube sayıları.)

1.4: YuSoft Şirketi Hakkında

"YuSoft"; 4 mühendislik personeli (1 Yazılım Mühendisi, 1 Kullanıcı Arayüzü Geliştirici, 1 Uygulama Geliştirici, 1 Veritabanı Geliştirici) bulunan bir yazılım sistemi geliştirme şirketidir.

Bölüm 2: Proje Aşamaları ve Yapılanlar

2.1: Bilgisayar Mühendisliği Uygulamaları I Aşaması

2.1.1: Proje Raporu Yazılmaya Başlandı

Projemizin üzerinde kurulu olduğu derin olmayan bilgilerin yer aldığı, simülasyon hakkında bilgilerin yer aldığı, proje takibini kolaylaştıran kayıtların tutulduğu, projeyi sunmamıza yardımcı olan **Proje Raporu**'nu yazmaya başladım ve **Proje Raporu PDF** adlı belgeyi oluşturdum.

Bölüm 0'da Yazılım Mühendisliği, Yazılım Mühendisi, Yazılım Geliştirme Yaşam Döngüsü ve Yazılım Geliştirme Modelleri hakkında derin olmayan bilgilerden bahsettim.

Bölüm 1'de Simülasyonumun konusundan bahsettim.

Bölüm 2'de proje yapım aşamalarının kayıtlarını tutmaya başladım.

2.1.2 MyCar Rent Yazılım Sistemi "Yapılabilirlik Analizi Aşaması" Uygulanmaya Başlandı

Yapılabilirlik Analizi Aşaması, geliştirilecek yazılım sistemininin işlevselliği detaylandırılmadan tarif edilirek oluşturan ön gereksinimler belgesi aracılığıyla yazılım sisteminin mevcut teknolojiyle, mevcut bütçeyle ve hedeflenen zamanla yapılabilirliğini analiz etme aşamasıdır.

Bu proje için elimde yapılabilirliği analiz edecek verilerin olmaması; bu verilere sahip olsam bile bu verileri analiz edecek teorik bilgimin olmamasından dolayı bu aşamada sadece **Ön Gereksinimler Raporu PDF** adlı dökümanı oluşturdum. **Yapılabilirlik Analizi Aşaması**nı yapmış gibi **Planlama Aşaması**na dahil ettim.

2.1.3: MyCar Rent Yazılım Sistemi “Planlama Aşaması” Uygulanmaya Başlandı

Planlama Aşaması, kaynaklar ve hedefler baz alınarak yazılım sistemini geliştirmeye başlamadan önce tüm sürecin titizlikle planlandığı aşamadır. Bu aşamada elde edilen ön bilgilerle **Yazılım Geliştirme Model**leri baz alınarak **Yazılım Geliştirme Modeli** ve **Süreçleri** tasarlanır; **İş Yükü Dağıtımı** yapılır; **Geliştirme Takvimi** oluşturulur; **Yasal Gereksinimler** belirlenir; **Yaklaşık Zamansal Maliyet, Yaklaşık Parasal Maliyet** hesaplanır. Yazılım sistemine göre çeşitli başlıklar tanımlanabilir.

Planlama Aşaması için **MyCar Rent Yazılım Sistemi Geliştirme Planlaması PDF** adlı belgeyi oluşturdum.

Bu proje için elimde olan veriler ve teorik bilgiyle yukarıda koyu olarak yazılan aşamaların **Yazılım Geliştirme Model**ini ve **Süreçleri**ni tasarladım; **İş Yükü Dağılımı**nı oluşturdum. **Geliştirme Takvimi**ni oluşturmak, **Yasal Gereksinimler**ini belirlemek, **Yaklaşık Zamansal Maliyet ve Yaklaşık Parasal Maliyet**ini hesaplamak eylemlerini ise elimde verilerin olmaması; bu verilere sahip olsam bile bu verileri kullanacak teorik bilgimin olmamasından dolayı gerçekleştiremedim. Bu eylemleri gerçekleştirmiş gibi **Planlama Aşaması**na dahil ettim.

2.1.4: MyCar Rent Yazılım Sistemi “Döngü I : Aşama II” Uygulanmaya Başlandı

Analiz Aşaması, yazılım sisteminin işlevselliklerinin kesin ve anlaşılır bir şekilde hem kullanıcılar hem de geliştiriciler için tanımlandığı aşamadır. Bu aşamada yazılım sistemi analizi yapılır. Yani gereksinimler belgesi oluşturulur. Eğer yazılım sistemi **Kullanıma Hazır Yazılım** ise gereksinimler belgesi şirket tarafından öznel olarak oluşturulur. Eğer yazılım sistemi **Özel Yazılım** ise gereksinimler belgesi yazılım sistemini isteyen şirketin gereksinimlerine göre oluşturulur. Özel yazılımlarda gereksinimler

belgesi iki şirket arasında (yazılım sistemini geliştiren şirket ve yazılım sistemini isteyen şirket) hukuksal anlaşma yerine de geçer.

Döngü I : Aşama II için **Gereksinimler Raporu I PDF** adlı belgeyi oluşturdum.

Gereksinimler Raporu I PDF adlı belge yazılım sisteminin geliştirilmesine ve geliştiriciler için gereksinimlerin oluşturulmasına kılavuzluk eden, az biçimsel olarak doğal dille yazılmış bir belgedir. Bu belge işlevsellikler ve varlıklarla ilgili tam ayrıntıları içermeyen, hem yazılım sistemini isteyenler için hem de geliştirenler için anlaşılır olan, yazılım sisteminin sağlayacağı işlevselliklerin ve tutacağı verilerin sınırını çizen ve tanımlayan soyut bir belgedir.

Döngü I : Aşama II için **Geliştiriciler İçin Gereksinim Kartları I** adlı dosyayı oluşturdum.

Kartlar, sınırı çizilen ve tanımlanan işlevsellikler ve verileri yazılım sistemini isteyenler için tam anlaşılır olmayan ama geliştiriciler için anlaşılır olacak şekilde tanımlayan biçimsel olarak doğal dille yazılmış **Varlıklar, Fonksiyonel Gereksinimler** ve **Fonksiyonel Olmayan Gereksinimlerden** oluşan bir somut koleksiyondur. Fonksiyonel Gereksinimler yazılım sistemini isteyen ve geliştiren için anlaşılır olan bir anlaşmadır.

Gerçek hayatta çok daha ayrıntılı olması gereken bu belgeleri yazılım sisteminin çalışacağı alanı bilmemem, tecrübesiz olmam ve bunun bir simülasyon olmasından dolayı göstermelik şekilde ayrıntısız olarak oluşturdum. Geriye dönük baktığımda da yazılım sistemini isteyen müşterinin hoşuna gitmeyen birçok hatanın iyi bir şekilde tanımlanmayan veya mantıksal hataları bulunan veya hiç yazılmayan gereksinimlerden kaynaklandığını anladım. Plan odaklı geliştirmeler için iyi bir şekilde tespit edilip tanımlanan ve mantıksal hataların en aza indirildiği gereksinimlere ihtiyaç vardır.

2.1.5: MyCar Rent Yazılım Sistemi “Döngü I : Aşama III”

Uygulanmaya Başlandı

Tasarım Aşaması, çıkarılan gereksinimler belgesine göre yazılım sisteminin bileşenlerinin belirli ilkelerle, yöntemlerle ve araçlarla tasarlanması aşamasıdır. Plan odaklı geliştirmede tasarım aşaması çok önemlidir çünkü yazılım sistemi çevik geliştirmeler gibi kodlama aşamasında tasarlanıp ardından düzeltilemezler. Yanlış tasarlanan bir yazılım sistemi pahalı bir bakım aşamasına neden olur.

Döngü I : Aşama III için **Yazılım Sistemi Gösterimi I PDF** adlı belgeyi oluşturdum.

Yazılım Sistemi Gösterimi I PDF adlı belgenin tasarım standartları benim tarafımdan oluşturulmuş olup yazılım sisteminin genel bir gösterimidir.

Döngü I : Aşama III için **Use Case Diagrams I PDF** adlı belgeyi oluşturdum.

Use Case Diagrams I PDF adlı belge yazılım sistemini kullanan aktörlerin ve eriştikleri işlevselliklerin gösterimidir.

Döngü I : Aşama III için **Yazılım Sistemi Mimarisi I PDF** adlı belgeyi oluşturdum.

Yazılım Sistemi Mimarisi I PDF adlı belge yazılım sistemini geliştirirken kullanılacak mimariyi gösterir. Kullandığım **3-Katmanlı Mimari**de **Veri Tabanı** yazılım sistemine ait bir katman değildir. Yazılım sistemi **Veri Tabanı**ndaki verileri işleyerek kullanıcıya gösteren **Sunum Katmanı, İş Katmanı** ve **Veri Erişim Katmanı**ndan oluşur. **Veri Erişim Katmanı Veri Tabanı**ndan verileri **İş Katmanı**na aktarır. **İş Katmanı** verileri işleyerek **Sunum Katmanı**na aktarır ve **Sunum Katmanı**nda işlenmiş veriler kullanıcı ile buluşturulur. Bu katmanlar sayesinde yazılım sisteminin geliştirilmesi, planma yönetimi ve bakımı kolaylaşır.

Döngü I : Aşama III için **Class Diagram I IMG** adlı belgeyi oluşturdum.

Class Diagram I IMG adlı belge yazılım sistemini oluşturan sınıfları, sınıfların özelliklerini ve birbirlerinin arasındaki ilişkileri göstererek yazılım sisteminin statik iskeletini ortaya koyar. Yazılım sisteminin dinamik iskeleti ise **Fonksiyonel Gereksinim Kartları I** ve **Use Case Diagrams I**'de orttaya konmaktadır.

Veri Tabanı Tasarımı***