Yazılım Mühendisliği Proje Raporu

Rapor-2: SİSTEM TASARIMI - Yineleme 1 (b)

Bu rapor üç adımda sunulacaktır:

1. **Bölüm 1** (Etkileşim Diyagramları)
2. **Bölüm 2** (Sınıf Diyagramı ve Sistem Mimarisi)
3. **Tüm Rapor #2** (tüm bölümler dahil)

Not: Bu raporu ileride kullanmak üzere saklayın çünkü nihai raporun bir parçası olacaktır (**Rapor-3**). Bu nedenle, dönem boyunca ilerledikçe, gözden geçirmek, güncellemek ve/veya düzeltmek için her fırsatı değerlendirin.

Bu raporu hazırlarken, değerlendirme kontrol listesini kontrol etmek ve raporunuzda bazı tipik hatalardan kaçınmak isteyebilirsiniz.

---

1. Rapor Formatı

Bir bölümü veya raporun tamamını gönderirken, aşağıdaki bölümleri ve alt bölümleri ekleyin. Her bölüm/alt bölüm uygun bir başlık ile açıkça tanımlanmalıdır.

- **Kapak Sayfası** ve **Bireysel Katkı Dağılımı**

  Yazılım mühendisliği öneri metninin 5. bölümünde belirtildiği gibi hazırlayın. Katkı dağılımı, **Rapor-1**'de açıklandığı gibi sorumluluk matrisini ve sorumluluk tahsis tablosunu içermelidir.

- **İçindekiler**

  Sayfa numaralarının doğru olduğundan emin olun.

Bölüm 1: Etkileşim Diyagramları

- **Etkileşim Diyagramları**
  - **Rapor-1**'de detaylandırdığınız ("ayrıntılı") kullanım durumları için etkileşim diyagramları (sequence veya communication diyagramları) oluşturun. En azından sequence diyagramları hazırlamalısınız, ancak diğer UML etkileşim diyagramlarını da kullanabilirsiniz.
  - Nesnelere sorumluluk verme sürecinde hangi tasarım ilkelerini kullandığınızı açıklayın. Bu açıklamaları diyagramda "yorum baloncukları" olarak veya diyagram başlığında yapabilirsiniz.

    Tasarım ilkeleri hakkında bilgi edinmek için ders notlarındaki etkileşim diyagramlarını inceleyin.

- **Proje Yönetimi** (aşağıdaki **Bölüm 7**'de açıklanmıştır) ve **Kaynakça** (aşağıdaki **Bölüm 8**'de açıklanmıştır).

Bölüm 2: Sınıf Diyagramı ve Arayüz Özellikleri

a. **Sınıf Diyagramı**

- Tüm sınıfları ve ilişkilendirmelerini gösterin. Yalnızca niteliklerin ve işlemlerin görünürlüğünü belirtin; tipler ve imzalar hakkındaki detaylar bir sonraki maddede verilmelidir.
- Sınıf diyagramını tek bir sayfaya sığdıramıyorsanız veya çok karmaşık görünüyorsa, tüm sınıfları ve ilişkilerini gösteren bir "genel bakış" sınıf diyagramı oluşturun. Bu diyagramda her sınıf için yalnızca sınıf adıyla tek bir bölme gösterin (nitelikleri ve işlemleri dışarıda bırakın).
- Daha sonraki sayfalarda, üç bölmeli ve bir sınıfın tüm nitelikleri ve işlemleri ile kısmi sınıf diyagramları gösterin. Diyagramlarda belirttiğinizden ve kısmi diyagramların genel sınıf diyagramına nasıl uyduğunu açıkladığınızdan emin olun.

b. **Veri Türleri ve İşlem İmzaları**

- Sınıf diyagramından bağımsız olarak, UML gösterimindeki sınıf tanımını yazın. Her sınıf için tüm niteliklerin ve işlem imzalarının veri türlerini belirtin.
- Her sınıfın, işlemin ve niteliğin anlamını açık bir dille tanımlayın.

c. **İzlenebilirlik Matrisi**

- Sınıflarınızın alan modeli kavramlarınızdan nasıl geliştiğini gösterin. Tek bir kavramdan evrimleşen değiştirilmiş adlar veya birden çok sınıf için açıklama sağlayın; yalnızca onay işaretli bir matris yapmak yeterli değildir.
- Sisteminizde çok sayıda sınıf varsa, matrisin okunmasını kolaylaştırmak için tüm alan kavramlarınızı satır satır listeleyebilir ve her kavramdan hangi sınıfların türetildiğini ve nedenini açıklayabilirsiniz.

Sistem Mimarisi ve Sistem Tasarımı

a. **Mimari Tarzlar**

- Tasarımınızda kullanılan mimari stilleri açıklayın (Google'da "mimari yazılım stilleri" örneklerini araştırabilirsiniz).

b. **Alt Sistemlerin Tanımlanması**

- Sisteminizdeki alt sistemlerin UML paket diyagramını çizin ve açıklayın.

c. **Alt Sistemleri Donanıma Eşleme**

- Sisteminizin birden fazla bilgisayarda çalışması gerekiyor mu? Örneğin, farklı makinelerde çalışan bir istemci (web tarayıcı) ve bir sunucu (web sunucusu) alt sisteminiz olabilir.
    - **HAYIR** ise, bir sonraki bölüme geçin.
    - **EVET** ise, hangi alt sistemlerin hangi makinede çalıştığını belirtin.

d. **Kalıcı Veri Depolama**

- Sisteminiz, sistemdeki bir komutu çalıştırmak için verileri kaydetmesi gerekiyor mu?
    - **HAYIR** ise, bir sonraki bölüme geçin.
    - **EVET** ise, kalıcı nesneleri belirleyin ve depolama yönetimi stratejinizi seçin (örneğin düz dosyalar, ilişkisel veri tabanı vb.).

      Dosya biçimi ve/veya veri tabanı şemasının açıklamasını ekleyin (veri tabanı tablolarının yapısı, komut açıklamalarıyla birlikte verilmelidir).

e. **Ağ Protokolü**

- Sisteminiz tek bir makinede çalışıyorsa, bu adım muhtemelen sizin için geçerli **DEĞİLDİR**; sonraki bölüme geçin.
- Aksi takdirde, hangi iletişim protokolünü kullanacağınızı ve bu seçimi neden yaptığınızı açıklayın (örneğin, .NET web servisleri, Java RMI, Java JDBC, HTTP vb.).

f. **Genel Kontrol Akışı**

- **Yürütme Düzeni:** Sisteminiz yordamsal olarak mı yönlendiriliyor ve her kullanıcının aynı adımlardan geçmesi gereken "doğrusal" bir şekilde mi çalışıyor, yoksa olaylar için bir döngüde bekleyen ve her kullanıcı için olaya dayalı bir sistem mi?
- **Zamana Bağlılık:** Sisteminizde zamanlayıcı var mı?

  Olay yanıtlı bir sistem mi yoksa gerçek zamanlı bir sistem mi? Gerçek zamanlı ise, periyodik midir ve dönemler için zaman kısıtlamaları nelerdir?

- **Eşzamanlılık:** Sisteminiz birden çok iş parçacığı kullanıyor mu?
    - **HAYIR** ise, bir sonraki bölüme geçin.
    - **EVET** ise, ayrı kontrol iş parçacığı olan nesneleri tanımlayın ve iş parçacıkları arasındaki senkronizasyonu açıklayın.

g. **Donanım Gereksinimleri**

- Sisteminiz hangi kaynaklara bağlıdır? Örneğin, ekran görüntüsü, disk depolama, iletişim ağı veya bazı özel sensörlere/cihazlara erişim gerekebilir.
- Sisteminizin çalışması için bu kaynaklara ilişkin tam gereksinimleri açıklayın. Örneğin, minimum 640×480 piksel çözünürlüğe sahip renkli ekrana ihtiyacınız vardır; en az 2 GByte sabit disk alanı; minimum ağ bant genişliği 56 Kb/sn vb.
- **Proje Yönetimi** (aşağıdaki **Bölüm 7**'de açıklanmıştır) ve **Kaynakça** (aşağıdaki **Bölüm 8**'de açıklanmıştır).

Bölüm 3: Algoritmalar ve Veri Yapıları (Varsa)

a. **Algoritmalar**

- **Rapor-1**'inizden matematiksel modeller uygulayan algoritmaları açıklayın. Sisteminiz başka karmaşık algoritmalar kullanıyor mu? Örneğin, bir oyunda bir animasyon figürü için hareket yörüngesini hesaplarken bazı sayısal veya bilgisayar grafik algoritmaları kullanabilirsiniz. Veya borsa hareketlerini değerlendirirken istatistiksel algoritmalar kullanıyor olabilirsiniz.
    - **HAYIR** ise, sonraki bölüme geçebilirsiniz.
    - **EVET** ise, algoritmalarınızı tanımlayın. Örneğin, animasyonlu bir figürün hareketi için yol koordinatları önceden hesaplanıp bir arama tablosunda mı saklanacak yoksa bir spline enterpolasyon algoritması kullanılarak mı hesaplanacak?

      Algoritma tasarımını tanımlamak için aktivite diyagramlarını kullanabilirsiniz.

b. **Veri Yapıları**

- Sisteminiz diziler, bağlı listeler, hash tablolar veya ağaçlar gibi karmaşık veri yapıları kullanıyor mu?
    - **HAYIR** ise, sonraki bölüme geçiniz.
    - **EVET** ise, hangi veri yapısını kullanacağınıza karar verirken hangi kriterleri kullandığınızı (örneğin performans ve esneklik) açıklayın.

Kullanıcı Arayüzü Tasarımı ve Uygulaması

- **Rapor-1** için geliştirilen ilk ekran tasarımlarını değiştirip değiştirmediğinizi ve uygulayıp uygulamadığınızı açıklayın. Yalnızca kullanıcı arayüzünüzde kullanıcının çabalarını azaltan (veya artıran) önemli değişiklikler hakkında yorum yapın. Renk veya stil değişiklikleri önemli değildir ve raporunuzdan çıkarılmalıdır.
- **"Kullanım kolaylığı"**, genellikle kullanıcı arayüzünün temel bir özelliği olarak kabul edilir. Bu kavram, çok sayıda renk, resim veya grafikle süslenmiş bir arayüzle karıştırılmamalıdır. Aksine, gösterişli kullanıcı arayüzlerinden kaçınmalısınız. "Kullanım kolaylığı", arayüzün pek çok soru sormak veya hacimli belgeleri okumak zorunda kalmadan sezgisel, anlaşılması ve kullanımı kolay olduğu anlamına gelir. İyi organize edilmiş, minimal bir kullanıcı arayüzü yeterli olmalıdır. Kullanıcı çabasını **Rapor-1**'in bir parçası olarak zaten düşündünüz ve burada kullanıcı çabasını en aza indirerek "kullanım kolaylığı"nı en üst düzeye çıkarmaya çalışmalısınız.

Test Tasarımı

Bu rapor için sadece testlerinizi tasarladığınızı unutmayın; uygulama aşamasında bu testleri programlayacak ve çalıştıracaksınız.

a. **Yazılımınızın Birim Testi**

- Programlanacak ve kullanılacak test senaryolarını listeleyin ve açıklayın.

b. **Testlerinizin Kapsamı**

- Testlerinizin ne kadarını kapsadığını tartışın.

c. **Entegrasyon Testi Stratejisi**

- Entegrasyon testi stratejinizi ve bunu nasıl uygulayacağınızı planlayın.
- **Rapor-1**'inizde belirtmiş olabileceğiniz algoritmaları, işlevsel olmayan gereksinimleri veya kullanıcı arayüzü gereksinimlerini test etme planlarınızı da açıklayın.

Proje Yönetimi ve Çalışma Planı

a. **Ekip Üyelerinin Bireysel Katkılarını Birleştirme**

- Raporun son kopyasını herkesin çalışmasından derleyerek tutarlılık, düzgün biçimlendirme ve görünüm sağlayın.
- Hangi sorunlarla karşılaşıldığını ve nasıl ele alındıklarını açıklayın.

b. **Proje Koordinasyonu ve İlerleme Raporu**

- Hangi kullanım durumları uygulandı?
- Halihazırda işlevsel olanlar nelerdir, şu anda neler üzerinde çalışılmaktadır?
- İlgili diğer proje yönetimi faaliyetlerini listeleyin ve açıklayın.

c. **Çalışma Planı**

- Planladığınız aşamaları ve bunları gerçekleştirmeyi hedeflediğiniz tarihleri listeleyin.
- Projenizi planlamak için tercihen Gantt şemalarını kullanmalısınız.

d. **Sorumlulukların Dağılımı**

- Her ekip üyesinin şu anda geliştirme, kodlama ve testten sorumlu olduğu modüllerin ve sınıfların adlarını listeleyin.
- Entegrasyonu kim koordine edecek?
- Entegrasyon testini kimler yapacak? (Varsayım, birim testi her bir birim için o birimi geliştiren öğrenci tarafından yapılacaktır.)

---

8. Kaynakça

- Kaynaklar listesi, projede kullanılan herhangi bir materyalin kesin referanslarını ve URL'lerini içermelidir.

NOT: Yazılımınızı uygulamak için hangi programlama dilini kullanırsanız kullanın (Java, PHP, C# veya başka bir dil), önce UML'de genel, dilden bağımsız tasarım diyagramları (etkileşim diyagramları) oluşturmanız gerekir.

- Eğer yazılımınızı nesne yönelimli olmayan bir dilde uyguluyorsanız, ayrıca uygulamaya özel sıra diyagramları sağlamalısınız. Bu durumda, uygulama diyagramlarınızın genel UML tasarımınıza nasıl karşılık geldiğini adım adım açıklayın.
- Bu, genel tasarımdaki sınıfların ve yöntemlerin, nesne yönelimli olmayan uygulamanızın işlevlerine göre izlenebilir olması gerektiği anlamına gelir.
- Kaynak kodu bu raporla birlikte sunulmamalı, ilk tanıtım materyallerinize (**Demo-1veya 2**) eklenmelidir.

NOT: Diyagramlarınızı yorumlayın! Diyagramlarda açık olmayan tüm tasarım kararlarını ve diğer önemli noktaları açıklayın. Yararlı herhangi bir bilgi kabul edilir. Rapor için sayfa sayısında bir sınırlama yoktur. İyi yorum ve açıklamalara sahip olmak, projenin anlaşılmasına ve değerlendirilmesine büyük ölçüde yardımcı olur ve notunuza olumlu katkıda bulunur.

Etkileşim Diyagramları Çizme

- Teorik olarak, her sistem fonksiyonu için tek bir etkileşim diyagramı çizilmelidir (sistem sekans diyagramınızı kontrol edin - SSD). Bazı sistem fonksiyonları önemsiz olduğundan, aşağıdaki pratik tekniği öneririz:
    - Her kullanım durumu için bir etkileşim diyagramı çizmeye başlayın. Diyagram bir noktada çok karmaşık hale gelirse veya bir sonraki sayfaya taşarsa, etkileşim diyagramınızı bölün ve her sistem fonksiyonu için ayrı bir diyagram çizin.

Testler Hakkında Not

- **Bölüm 6**'daki testler, ilk tanıtım için gerçek testler yapacağınız zaman değiştirilebilen ve geliştirilebilen ön tasarımlardır.
- Bu raporun birim testi için herhangi bir program kodu yazmanız beklenmemektedir. Tanıtım için test kodunun yazılması ve testlerin yürütülmesi beklenmektedir (bakınız 2. madde).
- **Rapor-2** için, hangi sınıfların/fonksiyonların test edilmesi gerektiğini ve bunu nasıl yapacağınızı açıklamanız gerekir. Durum diyagramlarını oluşturmalı ve test senaryo tablolarını çizmelisiniz.
- Ayrıca birim testlerinizin kapsamını tartışın. Durum tabanlı test yaptığınızı hatırlayın; bu nedenle, kapsamınızın durumlar ve geçişler açısından analiz edilmesi gerekir. Bir durum diyagramının sorumlu bir şekilde test edilmesi için kabul edilebilir minimum strateji şunları içerir:
    - **Tanımlanan tüm durumları** en az bir kez kapsamak (her durum en az bir test vakasının parçasıdır).
    - **Geçerli tüm geçişleri** en az bir kez kapsamak.
    - **Tüm geçersiz geçişleri** en az bir kez tetiklemek.

- Yazılımınızın genel UML tasarımı için testlerinizi tasarlayın.
- Yazılımınızı nesneye yönelik olmayan bir dilde uyguluyorsanız, uygulamanız için test tasarımı hakkında rapor vermeyin. Uygulama testleri, ilk tanıtım katkılarınızın bir parçası olarak rapor edilecektir. Birim testleri (C#, PHP, ASP.NET vb.) için internet üzerinde kaynaklar bulabilirsiniz.

UML Diyagramları ve Uygulama Araçları

- UML diyagramlarınızı sunarken, metninizde sisteminizi uygulamak için hangi araçları, platformları veya dilleri kullandığınızı veya kullanacağınızı belirtin.
- Kullandığınız yazılım araçları, platformları veya dilleri hakkında daha fazla bilgiyi nerede bulacağınızı belirtin ve ilgili bağlantıları **Kaynakça** bölümünde listeleyin.

1.1. Sınıfların Alan Modeli Kavramlarına Göre İzlenebilirliği

- En büyük sorun, **Rapor-2**'nin genellikle, **Rapor-1**'inizden neredeyse bağımsız olarak, herhangi bir referans göstermeden oluşturulmuş gibi görünmesidir. İyi yazılım mühendisliğinin anahtarı, eserler arasında mantıksal ilerlemeyi (yani izlenebilirliği) korumaktır: gereksinimlerden kavramsal modellemeye, sınıf tasarımına ve kod uygulamasına kadar.
- Bu genellikle etkileşim diyagramlarınızın **Rapor-1**'inizden, yani sistem sekans diyagramlarınızdan ve alan modeli analizinizden nasıl geliştiğini yansıtmasıyla ilgilidir.
- Genellikle net bir bağlantı yoktur! **Rapor-2**'nizdeki sınıfların/nesnelerin **Rapor-1**'deki alan modelinizden nasıl evrimleştiğini izlemek mümkün değildir. Diyagramlar, **Rapor-1** için yaptığınız tüm analizlerden bağımsız olarak ortaya çıkmış gibi görünebilir. Bu durum, **Rapor-1**'inizi işe yaramaz hale getirir ve yazılım mühendisliğine kötü bir örnektir.
- **Rapor-2**'yi düzgün yapmak için aşağıdakileri yapmalısınız:
    1. Her etkileşim diyagramı için hangi sistem sekans diyagramından evrimleştiğini ve hangi sistem yönteminin detaylandırıldığını açıkça belirtin.
    2. Her etkileşim diyagramında, alan modelinizdeki kavramların en azından bir kısmını kullanın. **Rapor-2**'yi hazırlarken (Rapor-1 gönderildikten sonra), bu kavramların çoğunun değiştirilmesi veya yeni kavramların sunulması gerektiğini keşfetmiş olsanız bile, hangi yeni nesnelerin (etkileşim diyagramlarınızda) eski kavramlardan geliştiğini açıkça belirtin. Yeni nesneler tanıtmanız gerekiyorsa (ve alan modelinizde **Rapor-1**'de bu nesnelere karşılık gelen hiçbir şey yoksa), bu yeni nesnelerin neden gerekli olduğunu ve **Rapor-1**'deki alan analizi aşamasında neden bulunmadığını en az bir cümle ile açıklayın.
- **Rapor-2**'nizdeki sınıfları/nesneleri **Rapor-1**'den alan kavramlarına geri izlemek mümkün olmalıdır. Bu izlemeyi raporlarınız aracılığıyla sunmak yerine açık hale getirin ve hangi etkileşim diyagramının hangi sistem sekans diyagramından ve hangi sınıfın alan modelinizdeki hangi kavramdan kaynaklandığını belirtin. İlerleme, yeni tasarımınızın **Rapor-1**'de yapılan analizle net bir bağlantısı olmadığı ani sıçramalardan ziyade evrimsel olmalıdır. **Rapor-1**'den **Rapor-2**'ye süreklilik eksikliği, **Rapor-1**'i işe yaramaz hale getirir ve bu sadece zaman kaybıdır.
- Öğrencilerden duyulan yaygın bir mazeret, yazılımlarının PHP veya nesneye yönelik olmayan başka bir dil kullanılarak uygulanmasıdır, bu nedenle sınıflar ve PHP komut dosyaları arasında net bir ilişki yoktur. Bu geçerli bir bahane değildir. Uygulamanız

kavramsal analizinize dayanmalıdır (aksi takdirde kavramsal analiziniz işe yaramaz). Uygulamanız, seçtiğiniz programlama dilini kullanarak kavramsal modelinizi uygulamanın bir yoludur. Kavramsal analizinizde iyi bir iş çıkardıysanız, başka bir geliştiricinin kavramsal modelinizi alıp farklı bir programlama dilinde uygulaması mümkün olmalıdır. Bu nedenle, PHP komut dosyanızın (veya diğer program birimlerinin) kökenini kavramsal modelinizden gelen kavramlara kadar izleyebilmelisiniz. Eğitmeninizin raporlarınız aracılığıyla bu bağlantıyı anlamasını beklemek yerine, bu izlemeyi açık hale getirin ve alan modelinizdeki hangi kavramdan hangi PHP komut dosyasının kaynaklandığını açıklayın.

1.2. Rapor-1'deki Değişiklikler

- Sorununuzu daha iyi anladıkça veya daha iyi bir çözüm yolu buldukça, belgelerinizde gerekli tüm değişiklikleri yapmalısınız. Yinelemeli yaklaşımın amacı budur: tekrarlanan döngüler (yinelemeler) yoluyla bir sistem geliştiririz ve sistemin önceki bölümlerinin veya sürümlerinin geliştirilmesi sırasında öğrenilenlerden yararlanırız. Bu nedenle, daha fazla bilgi edindikçe geçmiş belgelerinizde (**Rapor-1**) revizyonlar ve iyileştirmeler yapmanız önerilir.
- **Rapor-1'deki Materyalleri Güncelleme:** Kullanım durumlarını, alan modelini ve hatta gereksinimlerinizi proje geliştirme sürecine devam ederken güncellemeniz gerekebilir. Yazılım projelerinde bu normaldir ve ilk raporunuzu mevcut proje durumunuza doğru şekilde yansıtacak şekilde güncellemeniz önemlidir. Önemli değişiklikler yaptıysanız ve gözden geçirilmiş ilk raporunuzu görmeden **Rapor-2**'nizi anlamak zor olabilecekse, güncellenmiş raporunuzu da vermelisiniz. Güncellenmiş rapor sürümünüzde bir yerde değişikliklerin kısa bir özetini vermeniz faydalı olacaktır, böylece projenizin gelişimini daha iyi anlayabiliriz.
- Revize edilmiş **Rapor-1**'i derecelendirmeyeceğiz veya bu konuda geri bildirimde bulunmayacağız, ancak **Rapor-2**'yi derecelendirirken ona bakacağız. Gözden geçirilmiş **Rapor-1**'inizi **Rapor-3**'ün bir parçası olarak gönderdiğinizde değerlendireceğiz.

---

2. Rapor Hazırlama

- Kurallar, **Rapor-1** ile aynıdır.
- UML diyagramı oluşturmak için internetten yazılım araçları araştırabilir ve uygun olanları kullanabilirsiniz. UML sembollerini destekleyen herhangi bir araç kabul edilebilir.

---

3. Rapor Değerlendirmesi

- Bireysel üyeler için genel takım notu ve notları atamak için değerlendirme politikasına bakınız.
- Bu rapor ancak raporun tamamı gönderildikten sonra notlandırılacaktır.
- Raporun bölümleri sunumdan hemen sonra notlandırılmayacaktır.
- Ancak, puanların %50'si zamanında teslim edilmeyen kısımlardan düşülecektir.
- Zaman izin verirse, raporun tamamının tamamlanmasından önce tek tek parçalar hakkında geri bildirim sağlayabiliriz.

- İlk olarak, tüm raporlar aşağıdaki gibi birbirinden bağımsız olarak derecelendirilecektir:

  (**NOT:** Ayrıca gerçek derecelendirme kontrol listesini de kontrol edin.)

Özellikler	Etkileşim Diyagramları			Sınıflar + Tanımlamalar		Sistem Mimarisi ve Tasarımı					Algoritmalar Ve veri yapıları	Kullanıcı Arayüzü		Test Tasarımı	Proje Yönetimi			Kaynakça
	UML Diyagramları	Şemanın yazılı açıklamaları	Alternatif Açıklamalar (#)	Sınıf diyagramları ve açıklamaları	İşaretler	Stil	Paket Şeması	Donanım Haritası	Veri Tabanı	Diğer ($)		Görünüm	Yazılı Açıklama (+)		Belge Birleştirme	Proje Koordinasyonu / ilerlemesi	Çalışma Planı	
Puanlar	10	10	10	5	5	5	2	2	3	*3	*4	6	5	12	11	5	2	(-5)
Toplam Puan	100																	

* Yıldız işareti olan yerler yalnızca uygulanabilirse derecelendirilir. Aksi takdirde, bu yerler bu raporun 5. Bölümüne atanacaktır: Kullanıcı Arayüzü Tasarımı.
(#) Bu, düşündüğünüz alternatif tasarım seçeneklerini ve bunların etkileşim diyagramlarını açıklayan anlatıyı ifade eder.
Hangi tasarım seçimlerinin yapıldığını, hangi tasarım ilkelerinin uygulandığını açıkça belirtin ve seçimlerinizin mantığını sağlayın.
($) Ağ protokolü, global kontrol akışı ve donanım gereksinimleri gibi yukarıdaki listede verilen diğer tasarım yönleri.
(+) Bu, kullanıcı ara yüzü tasarımınızı açıklayan anlatıyı ifade eder. Hangi tasarım seçimlerinin yapıldığını açıkça belirtin ve seçimlerinizin mantığını sağlayın.
Eksik veya tamamlanmamış kaynakça için 5 puana kadar düşülecektir

İkinci olarak, tüm raporlar karşılaştırılır ve sıralanır. Bu ikinci adımın açıklaması için lütfen **Rapor 1**'e bakınız.

---

4. Rapor Teslimi

Eğer önemli değişiklikler yaptıysanız ve güncellenmiş **Rapor-1**'inizin **Rapor-2**'nizi daha iyi anlamamıza yardımcı olabileceğini düşünüyorsanız, gözden geçirilmiş **Rapor 1**'inizi **Rapor-2**'nize ekleyebilirsiniz.

Ancak, güncellenmiş **Rapor-1**'i derecelendirmeyeceğiz ve düzeltmelerle ilgili herhangi bir geri bildirim vermeyeceğiz. Gözden geçirilmiş **Rapor-1**'i, **Rapor-3**'ün bir parçası olarak gönderdiğinizde değerlendireceğiz.

Bölüm 1'i gönderirken, belgenizin aşağıdaki bölümleri içermesi gerekmektedir:

- **Kapak Sayfası** ve **Bireysel Katkı Dağılımı**
- **İçindekiler**

- **Bölüm 1: Etkileşim Diyagramları**
- **Proje Yönetimi**
- **Kaynakça**

Bölüm 2'yi gönderirken, belgenizin aşağıdaki bölümleri içermesi gerekmektedir:

- **Kapak Sayfası** ve **Bireysel Katkı Dağılımı**
- **İçindekiler**
- **Bölüm 1: Etkileşim Diyagramları** (gerekirse düzenlenebilir)
- **Bölüm 2: Sınıf Diyagramı**
- **Bölüm 3: Sistem Mimarisi**
- **Proje Yönetimi**
- **Kaynakça**

Tam raporunuz, **Rapor Formatı**'nda belirtilen tüm bölümleri içermelidir (daha önce gönderilen bölümler gerektiğinde revize edilebilir).

Her grup, raporlarını son tarihte veya öncesinde öğretmeninize **classroom** ödevine yanıt olarak göndermelidir (**yalnızca PDF dosyası olarak**).

Rapor ayrıca proje web sitenize indirilmek üzere yüklenmelidir (ücretsiz web sunucusu için **Yazılım Mühendisliği Öneri Yazısı**'nı okuyunuz).

Gerçek Derecelendirme Kontrol Listesi

Proje Yönetimi (Maksimum Puan: 16 puan)

1. **Raporun kalitesi düşük olduğu için proje kötü yönetilmiş görünüyor.**
2. **Projenin algılanan yeniliği düşük.**
3. **Gelişigüzel düşünülmüş ve yüzeysel olarak tasarlanmış, çok sayıda (ilgisiz) özelliğe sahip odaklanmamış proje.**
4. **Zayıf ve tutarsız yazım; farklı stiller, okunması ve anlaşılması zor.**
5. **Şemalarında ve metninde raporun diğer bölümleriyle bağlantı kurulmamış; farklı bölümler farklı öğrenciler tarafından oluşturulmuş.**
6. **Farklı araçlarla oluşturulan, belirsiz ve anlaşılmaz UML diyagramları.**
7. **Gereksinimler, kullanım örnekleri, kullanıcı arayüzü ve alan modeli arasında tutarlılık ve izlenebilirlik eksikliği.**
8. **Rapor eksik, bağımsız değil ve ilgili ek bilgilere referans verilmiyor.**
9. **Eksik sayfa numaraları, bölüm başlıkları karışık; şekiller ve tablolar etiketlenmemiş, eksik altyazılar veya metinde açıklanmamış ve referans verilmemiş.**
10. **Rapor belirtilen formatı takip etmiyor.**
11. **Farklı UML diyagramları için farklı araçlar kullanılmış, ancak bunun neden gerekli olduğu açıklanmamış.**
12. **Bazı diyagramları okumak imkânsız; ayrıca, çevrelerindeki kenar boşluklarında çok fazla boşluk bırakılmış.**

13. **Proje büyük ölçüde üçüncü taraf bir donanım veya yazılım platformuna dayanıyor, ancak bu platform hakkında yeterli bilgi sağlamıyorsunuz; bu da raporunuzu okumayı zorlaştırıyor. Üçüncü taraf sistemler için öğretici vermeniz gerekmez, ancak en azından bu tür öğreticilerin bulunabileceği referansları kaynakçada belirtin. Ayrıca, üçüncü taraf sistemlerin her bir kavramı veya yapısı üzerine kısa bir yorum (bir cümle) eklemek raporunuzun okunabilirliğini artırmaya yardımcı olur.**
14. **Diğer: _____**

Bölüm 1: Etkileşim Diyagramları (Maksimum Puan: 30 puan)

1. **Rapor-2'nizdeki diyagramlar, Rapor-1'de yaptığınız analizlerden bağımsız olarak oluşturulmuş gibi görünüyor; özellikle, alan modelindeki bazı kavramlar etkileşim diyagramlarında görünmüyor ve neden atlandıklarına dair bir açıklama yok.**
2. **Belirli bir UML sıralama diyagramının nasıl ilişkilendirildiği açık değil; özellikle, bu diyagramların kaynağı ve sistem sırası diyagramları bağlamında nerede bittiği ve proje için nasıl kullanıldığı belirsiz.**
3. **Diyagramlar yeterli açıklama olmadan sunulmuş.**
4. **Belirli bir sıralama diyagramının hangi sistem yöntemiyle (sistem sırası diyagramları bağlamında) ortaya çıktığı açık değil—başlangıç olayı nedir? Rapor-1'deki bir sistem sırası diyagramına bakın ve mevcut etkileşim diyagramınızla hangi sistem yöntemini geliştirdiğinizi belirtin.**
5. **Alan modelinizde (Rapor-1) tanımladığınız kavramlar, sıralama diyagramlarınızda kullanılmıyor; yeni kavramlar, Rapor-1'in alan modeliyle nasıl ilişkili oldukları açıklanmadan tanıtılıyor.**
6. **Bazı nesneler veya sınıflar hiç açıklanmamış, sanki bir anda ortaya çıkmışlar; raporunuzu anlamak için çok fazla tahmin gerekiyor. Alan modelinde (Rapor-1) kullanılmayan yeni kavramların veya nesnelerin tanıtılması kabul edilebilir olsa da, bu yeni kavramların tanımlanması ve gerekçelendirilmesi gerekir.**
7. **Sınıfların, nesnelerin, yöntemlerin veya değişkenlerin adları sezgisel değil ve tanımlanmamış.**
8. **Bir UML diyagramının açıklaması, diyagramın belirli öğelerine değinmiyor; bu nedenle, açıklanan şemayla bağlantı kurmak zorlaşıyor.**
9. **Proje, web mimarisi stilini (tarayıcıda çalışan ve bir sunucuya bağlanan istemci) kullanıyor, ancak sıralama diyagramlarında istemci tarafının nerede bittiği ve sunucu tarafının nerede başladığı net değil.**
10. **Proje üçüncü taraflardan sınıflar kullanıyor, ancak atıf net değil ve kaynaklar doğru şekilde belirtilmemiş.**
11. **Diğer: _____**

Bölüm 2: Sınıf Diyagramı ve Arayüz Özellikleri (Maksimum Puan: 10 puan)

1. **Sınıf diyagramı ve yöntem imzaları, sınıfların, yöntemlerin ve niteliklerin metinsel açıklaması olmadan sunulmuş; her sınıf, yöntem ve nitelik tanımlanmalıdır.**

2. **Okunabilirliği artırmak için sınıf diyagramı bölünmüş ve birkaç sayfada gösterilmiş (bu iyi), ancak bağlantı noktaları gösterilmemiş ve diyagramların nasıl birbirine bağlandığı ve tek bir program oluşturduğu belirsiz.**
3. **Birden fazla sınıf diyagramı gösterilmiş, ancak birbirleriyle nasıl ilişkili oldukları açıklanmamış.**
4. **Sınıf diyagramlarındaki ilişkilendirme bağlantıları (kalıtım, toplama, gezinme vb.) gösterilmemiş, yetersiz etiketlenmiş veya tanımlanmamış—diyagramlarınızı açıklamak için metinler sağlayın.**
5. **İzlenebilirlik matrisi eksik veya açıklanmamış.**
6. **Bazı sınıflar bir alan kavramına kadar izlenemiyor ve yeni sınıfların neden tanıtıldığına dair açıklama eksik.**
7. **Alan modelindeki bazı kavramların sınıf diyagramında karşılık gelen sınıfları yok ve bunun nedenine dair bir açıklama yok.**
8. **Diğer: _____**

---

*Bölüm 3: Sistem Mimarisi ve Sistem Tasarımı (Maksimum Puan: 15 (*) puan; * = değişken)*

1. **Her paketin ne yaptığı ve sınıfların neden belirli şekilde gruplandığı açıklanmamış; örneğin, neden üç katmanlı bir mimari sisteminiz için uygundur?**
2. **MVC çerçevesinin bir parçası olan "Model" gibi iyi bilinen bir kavramdan bahsediyorsunuz, ancak bu kavramın yazılımınızla nasıl eşleştiğini açıklamıyorsunuz ve açıklamalarınızdan bunu anlamak mümkün değil—bu kavramların özel bağlamınıza nasıl uyduğunu açıklayın.**
3. **Örneğin, yazılımınızdaki "Model" nedir? Belirli bir sınıf mı yoksa bir grup sınıf mı? Hangileri? Rapor-1'deki alan modelinizden başlayarak "Model" bölümünü nasıl oluşturdunuz?**
4. **Donanım sınırları açıkça belirtilmediğinden, üçüncü taraf sistemlere göre kendi yazılımınızın hangi parçasının net olmadığı anlaşılıyor.**
5. **Diğer: _____**

Bölüm 4: Algoritmalar ve Veri Yapıları (Maksimum Puan: Varsa 4 puan)

1. **Rapor-1'deki matematiksel modeller için algoritmalar eksik.**
2. **Projenizde kullanılan algoritmalar yetersiz tanımlanmış; bazı şeyler basit bir dille tam olarak açıklanamaz—net diyagramlar, klasik akış şemaları veya tercihen UML etkinlik diyagramları sağlamalısınız.**
3. **Diğer: _____**

Bölüm 5: Kullanıcı Arayüzü Tasarımı ve Uygulaması (Maksimum Puan: 11 puan)

1. **Rapor-1'inizdeki kullanıcı arayüzü açıklamasına kıyasla hiç veya çok az ilerleme kaydedilmiş.**
2. **Farklı ekranlar için görsel tasarımlar sağlanmamış.**
3. **Bazı ekranlarda açıklanamayan veya net olmayan ifadeler var.**

4. **Ekranlar arasındaki gezinme yolu belirsiz veya eksik.**
5. **Kullanıcı arayüzünün her bir bileşeninin (etiketleriyle belirtilen) belirli gereksinimleri nasıl karşıladığı açıklanmamış; kullanıcı arayüzünün kullanım durumlarınızın uygulanmasını nasıl desteklediğini görmek zor.**
6. **Diğer: _____**

Bölüm 6: Test Tasarımı (Maksimum Puan: 12 puan)

1. **"Sisteminizin test gerektirmediği" iddiası—projenizin işe yaradığını başka nasıl bilebilirsiniz?!**
2. **Hiçbir sistematik test yaklaşımı sunulmamış; testler rastgele gibi görünüyor.**
3. **Yalnızca kullanım durumları için kabul testleri sağlanmış, ancak modüller veya sınıflar için birim testleri eksik.**
4. **Test kapsamı analiz edilmemiş veya yetersiz analiz edilmiş.**
5. **Diğer: _____**

Bölüm 7: İş Planı (Maksimum Puan: 2 puan)

1. **Zaman çizelgesi diyagramı yok veya diyagram karmaşık ve okunması zor.**
2. **Eksik veya düzensiz sorumluluk dağılımı (bazı öğrenciler orantısız sorumluluklar üstlenmiş) veya ortam çok karışık.**
3. **Diğer: _____**

Bölüm 8: Kaynakça (Eksi 5 puana kadar)

1. **Kaynaklarınız tamamen yetersiz; raporunuzda çok sayıda kaynağa atıfta bulunmanız gerekirken, bunlardan hiç bahsetmemişsiniz.**
2. **Raporunuzda genel olarak bir atıf sorunu var. Kaynakça listeniz çok sınırlı, ancak büyük fikirlerden bahsediyorsunuz ve hepsini sizin icat ettiğinize dair bir kanıt sunmuyorsunuz.**
3. **Sadece başlığı gösterilen, ancak URL'si olmayan kaynaklar var.**
4. **Metnin bazı bölümleri veya şekiller veya tasarım fikirleri başka bir yerden alınmış gibi görünüyor, ancak alıntılar eksik; icat etmediğiniz ve genel bilginin bir parçası olmayan her şey için bir referans göstermelisiniz.**
5. **Numaraya veya yazar adına göre belirtilmeyen veya ana metinde başka şekilde atıf yapılmayan kaynaklar.**
6. **Diğer: _____**