

BTÖ-303 Ara Uygulama Yönergesi: Konsept Tabanlı Laravel CRUD Projesi

Oct 11, 2026 · @Mertcan Ünal

Amaç ve genel bakış

Bu ara uygulamada 4 saat içinde, agentic yapay zekâ kullanarak konseptte dayalı, Laravel tabanlı ve CRUD içeren bir proje geliştirecek, çalışmanı rubriğe göre kendin değerlendirip en az 85 puanla teslim edeceksin.

Yapay zekâ kullanımı serbest. Asıl ölçülen, ürettirdiğin sistemi tanımlama, yönlendirme, doğrulama ve belgeleme becerindir; kod satırlarını tek tek yazman değil, çıktının çalıştığını kanıtlaman beklenir.

Teslim şartlarının özeti:

- Proje Laravel ile geliştirilmiş ve en az iki varlık için tam CRUD içeriyor.
- Proje, seçtiğin tek bir konseptte hizmet ediyor.
- Proje kökünde `RUBRIK.md` var ve yapay zekâ ajanı bu rubriğe göre kendi çıktısını değerlendirdi.
- Değerlendirme çıktısı `DEGERLENDIRME.txt` olarak proje klasöründe duruyor ve toplam puan en az 85.
- Proje GitHub'a **private** depo olarak yüklendi ve mertronot@gmail.com ile paylaşıldı.
- Süre 4 saattir (240 dakika); teslim, dersin bitişinde deponun son hâliyle yapılır.

Proje tanımı ve teknik zorunluluklar

Proje, tek bir konseptte hizmet eden, Laravel ile yazılmış ve en az iki varlıkta tam CRUD sunan çalışır bir web uygulamasıdır.

Konseptini listeden seç ya da kendin öner (sonraki bölüm). Konsepti `README.md` içinde anlat: kimin kullandığını, hangi problemi çözdüğünü yaz. `item`, `test`, `data` gibi genel adlar yerine konseptte özgü varlık ve alan adları kullan.

Konu	Zorunluluk
Çatı	Güncel kararlı Laravel sürümü (<code>composer create-project laravel/laravel</code>), Blade şablonları

Konu	Zorunluluk
Veritabanı	SQLite (varsayılan); ek kurulum olmadan <code>php artisan migrate --seed</code> ile ayağa kalkmalı
Varlıklar	En az 3 varlık; en az 1 bire çok ilişki ve en az 1 çoka çok (ya da ikinci bir bire çok) ilişki
CRUD	En az 2 varlık için tam CRUD (<code>Route::resource</code> , 7 rota); listede sayfalama; silmede onay adımı
Arama/filtre	En az bir listede metin araması ya da durum filtresi
Doğrulama	Form Request sınıflarıyla sunucu taraflı doğrulama; anlaşılır hata mesajları
İş kuralı	Konsepte özgü en az 2 iş kuralı (örn. aynı kayıt aynı anda iki kişiye atanamaz) ve bunları doğrulayan testler
Veri	Her ana varlık için factory ve seeder; en az 10 gerçekçi kayıt
Arayüz	Ortak layout, gezinme menüsü, başarı/hata mesajları, mobil genişlikte bozulmayan düzen
Test	<code>php artisan test</code> hatasız; ana varlıkta en az 8 feature test
Güvenlik	<code>@csrf</code> , <code>\$fillable</code> , Blade'de kaçışlı çıktı; <code>.env</code> repoda yok, <code>.env.example</code> var

Kimlik doğrulama (giriş/kayıt) zorunlu değildir ve rubrikte ek puanı yoktur; süreni konseptte, iş kurallarına ve teste ayır.

Konsept önerileri

Aşağıdaki 12 konseptten birini seçebilir ya da kendi konseptini önerebilirsin; seçim için onay gerekmez, ama konsept-varlık uyumu rubrikte puanlanır.

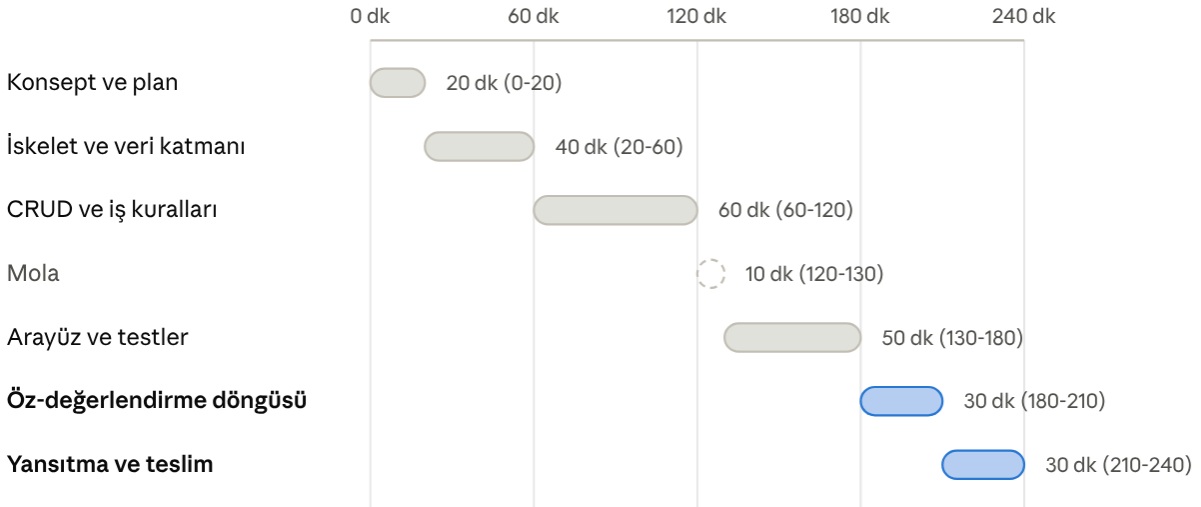
Konsept	Örnek varlıklar	Örnek iş kuralı
Kütüphane ödünç sistemi	Kitap, Üye, Ödünç, Kategori	Aynı kitap aynı anda iki üyeye ödünç verilemez
Veteriner kliniği randevuları	Hayvan, Sahip, Randevu, Veteriner	Bir veteriner aynı saatte iki randevu alamaz

Konsept	Örnek varlıklar	Örnek iş kuralı
Etkinlik ve bilet satışı	Etkinlik, Mekân, Bilet, Katılımcı	Satılan bilet sayısı mekân kapasitesini aşamaz
Spor salonu üyelikleri	Üye, Üyelik Planı, Antrenör, Ders	Süresi bitmiş üyelik derse kayıt olamaz
Restoran rezervasyonu ve menü	Masa, Rezervasyon, Menü Ögesi, Kategori	Aynı masa aynı saat diliminde iki kez ayrılamaz
Kampüs kulüp yönetimi	Kulüp, Üye, Etkinlik, Görev	Bir üye aynı kulüpte iki kez kayıtlı olamaz
Kiralık araç filosu	Araç, Müşteri, Kiralama, Şube	Kirada olan araç başka bir kiralamaya açılmaz
Kişisel finans takibi	Hesap, İşlem, Kategori, Bütçe	Bütçesi dolmuş kategoriye yeni harcama eklenirken uyarı verilir
Çevrimiçi kurs platformu	Kurs, Eğitmen, Kayıt, Ders	Kontenjanı dolu kursa kayıt alınamaz
Kargo ve sipariş takibi	Sipariş, Müşteri, Kargo, Durum	Teslim edilmiş sipariş iptal edilemez
Otel oda yönetimi	Oda, Misafir, Rezervasyon, Oda Tipi	Tarihleri çakışan iki rezervasyon aynı odaya yapılamaz
Barınak sahiplendirme	Hayvan, Başvuru, Gönüllü, Barınak	Sahiplendirilmiş hayvana ikinci başvuru onaylanamaz

Kendi konseptin için şartlar şunlardır: en az 3 varlık, en az 2 konseptte özgü iş kuralı ve gerçek bir kullanıcı kitlesi. Konsepti ilk 20 dakikada `README.md` başına yaz. Aynı konsepti seçen öğrenciler olabilir; projeler birbirinin kopyası olmamalıdır ve bunu commit geçmişi ile yapay zekâ kaydı gösterir.

4 saatlik ders akışı

Ders yedi aşamaya bölünmüştür ve her aşamaya sonunda depoda gösterebileceğin bir çıktı olmalıdır.

Kodlamaya 150 dakika, öz-değerlendirme ve teslim son 60 dakika ayrılır

ders akışı · 7 aşama, 240 dakika, ölçekli

Öz-değerlendirme ve teslim için ayrılan son 60 dakikayı kodlamaya harcama; öz-değerlendirme döngüsü çoğu zaman birden fazla tur sürer.

Aşama	Çıktı (depoda kanıtı)
Konsept ve plan	README'de konsept, varlıklar, iş kuralları ve Mermaid ER diyagramı; <code>RUBRIK.md</code> ve <code>AGENTS.md</code> yerinde; ajanın planı senin onayından geçmiş
İskelet ve veri katmanı	Laravel kurulu; migration, model, ilişki, factory ve seeder hazır; <code>migrate:fresh --seed</code> çalışıyor; private depo açıldı, mertronot@gmail.com eklendi, ilk commit'ler gönderildi
CRUD ve iş kuralları	Form Request'ler, resource controller'lar, index/create/edit/show görünümleri, arama ya da filtre, en az 2 iş kuralı
Arayüz ve testler	Ortak layout, flash mesajları, mobil genişlik denemesi, en az 8 feature test, <code>pint</code> temiz
Öz-değerlendirme döngüsü	Rubrik koşusu, tur geçmişi, <code>DEGERLENDIRME.txt</code> ve toplam en az 85
Yansıtma ve teslim	<code>docs/YANSITMA.md</code> , <code>docs/ai-log/</code> , <code>docs/teslim-kaniti.png</code> , temiz klon testi, son push ve Ek B kontrol listesi

Yapay zekâ kullanım kuralları

Agentic yapay zekâ (Claude Code, Codex, Cursor, Antigravity, Copilot ajan modu vb.) kullanımı serbesttir; tek koşul süreci kayda geçirmen ve çıktığı kendin doğrulamandır.

Rubrik projenin parçasıdır: ajanın her büyük adımdan sonra `RUBRIK.md` dosyasını okuyup kendi eksiklerini bulması beklenir. Bunun için aşağıdaki kurallar zorunludur:

- Başlamadan önce `RUBRIK.md` ve `AGENTS.md` dosyalarını proje köküne yerleştir. Ajanın kullandığı araç başka bir talimat dosyası okuyorsa (`CLAUDE.md` , `GEMINI.md` , `.cursor/rules`) aynı içeriği oraya da koy. `AGENTS.md` içeriği Ek A1'de, `RUBRIK.md` içeriği ise bu yönergedeki rubrik bölümündedir.
- Ajanı önce plan çıkarmaya yönlendir; planı okuyup konseptte uyduğunu kontrol ettikten sonra kod yazdır.
- Ajanın "çalışıyor" demesiyle yetinme: `php artisan test` ve uygulamayı tarayıcıda açıp en az bir kez kendin dene.
- Tüm yapay zekâ konuşmasını `docs/ai-log/` klasörüne dışa aktar (aracın dışa aktarma özelliği ya da kopyala-yapıştır, `.md` ya da `.txt`).
- `docs/YANSITMA.md` dosyasını sen yaz, ajan yazmasın. Ajan yalnızca boş şablonu oluşturabilir (şablon Ek A'dadır).
- Ajan `DEGERLENDIRME.txt` dosyasını yazarken puanı şişirmemeli; her puan için kanıt göstermeli (komut çıktısı ya da dosya yolu).
- Laravel'in kendi başlangıç paketleri ve çağrılan paketler serbesttir; başkasının hazır proje reposunu kopyalayıp konsepti değiştirmek serbest değildir.

Teslim: private GitHub deposu ve paylaşım

Proje GitHub'a **private** bir depo olarak yüklenir ve mertronot@gmail.com adresiyle paylaşılır; depoyu ders başında açıp commit'leri ders boyunca gönderirsen süreç de görünür olur.

- GitHub'da yeni depo oluştur. Adı `bto303-ara-<ogrenci-no>` biçiminde olsun (Türkçe karakter ve boşluk yok); görünürlük **Private** seçilsin.
- Projeyi depoya gönder: `git init` , ilk commit, `git remote add origin ...` , `git push -u origin main` . GitHub CLI kuruluysa ajanın tek komutla yapabilir: `gh repo create bto303-ara-<ogrenci-no> --private --source=. --push` .
- Depo sayfasında **Settings** → **Collaborators** → **Add people** yolunu izle ve mertronot@gmail.com adresini ekle. Davet, öğretim elemanı kabul edene kadar "Pending" görünür; bu normaldir.

4. Kontrol et: depo adının yanında **Private** etiketi var mı, Collaborators listesinde adres görünüyor mu? Bu iki kontrolün ekran görüntüsünü `docs/teslim-kaniti.png` olarak depoya ekle.

5. Ders bitmeden son `git push` 'u yap. Dersin bittiği andan sonraki commit'ler sayılmaz.

Ders süresince en az 8 commit atılmalıdır ve her commit mesajı yapılan işi anlatmalıdır (örn. `feat: kitap CRUD`, `test: ödünç iş kuralı`). Tek commit'lik teslimler rubrikte puan kaybettirir.

Teslim klasör yapısı ve zorunlu dosyalar

Depo kökünde Laravel dosyalarının yanı sıra aşağıdaki yedi öge bulunmalıdır; eksik olan her dosya rubrikte ilgili maddeden puan kaybettirir.

Yol	İçerik	Kim hazırlar
<code>README.md</code>	Konsept açıklaması, kurulum adımları, ER diyagramı (Mermaid)	Ajan yazar, sen kontrol edersin
<code>RUBRIK.md</code>	Bu yönergedeki rubrik bölümünün birebir kopyası	Ajan, yönergeden kaydeder
<code>AGENTS.md</code>	Ajanın her görevde uyacağı kurallar (Ek A)	Sen koyarsın, ajan okur
<code>DEGERLENDIRME.txt</code>	Rubriğe göre öz-değerlendirme çıktısı; toplam en az 85	Ajan yazar
<code>docs/ai-log/</code>	Yapay zekâ konuşmalarının dışa aktarımı	Sen
<code>docs/YANSITMA.md</code>	Yansıtma yazısı (şablon Ek A'da)	Sen yazarsın
<code>docs/teslim-kaniti.png</code>	Private etiketi ve Collaborators ekran görüntüsü	Sen

`.env`, `vendor/` ve `node_modules/` repoda bulunmamalıdır (`.gitignore` bunları zaten dışlar); `.env.example` ise repoda olmalıdır.

Değerlendirme rubriği (100 puan)

Rubrik 8 kategoride 31 maddeden oluşur ve toplam 100 puandır; teslim için `DEGERLENDIRME.txt` içindeki nihai toplam en az 85 olmalıdır.

Puanlama kuralları:

- Madde tam karşılanmışsa tam puan, kısmen karşılanmışsa yarısı (aşağı yuvarlanır), karşılanmamışsa 0 verilir.
- Kanıtı gösterilemeyen madde 0 alır. Kanıt, çalıştırılan komutun çıktısı ya da dosya yolu (gerekirse satır numarası) olmalıdır.
- H2 maddesinde ajan yalnızca dosyanın varlığını ve kelime sayısını kontrol eder; metnin sana ait olduğunu öğretim elemanı doğrular.

A. Konsept ve kapsam (10 puan)

Kod	Madde	Puan	Kanıt / doğrulama
A1	README'de konsept açıklaması: kim kullanır, hangi problemi çözer (en az 5 cümle)	4	README.md ilgili bölümü
A2	En az 3 varlık var ve hepsi konseptte özgü adlar taşıyor (item , data gibi genel adlar yok)	3	ls app/Models ve migration dosya adları
A3	En az 2 konseptte özgü iş kuralı uygulanmış ve README'de sayılmış	3	Kuralın kodda yeri (dosya ve satır) ve onu sınavan test adı

B. Veri katmanı (15 puan)

Kod	Madde	Puan	Kanıt / doğrulama
B1	Her varlık için migration var; yabancı anahtarlar constrained() ile tanımlı; gereken yerde unique / index kullanılmış	4	ls database/migrations ve migration içerikleri
B2	Eloquent ilişkileri iki yönde tanımlı: en az 1 bire çok ve en az 1 çoka çok (ya da ikinci bir bire çok)	4	Modellerde hasMany , belongsTo , belongsToMany satırları
B3	Her ana varlık için factory ve seeder var, en az 10 gerçekçi kayıt üretiyor; migrate:fresh --seed hatasız	4	php artisan migrate:fresh --seed çıktısı
B4	Modellerde \$fillable tanımlı; liste sayfalarında N+1 önlemi (with()) var	3	İlgili model ve controller satırları

C. CRUD işlevselliği (25 puan)

Kod	Madde	Puan	Kanıt / doğrulama
C1	En az 2 varlık için <code>Route::resource</code> ile 7 rota tanımlı ve çalışıyor	6	<code>php artisan route:list</code> çıktısı
C2	Create/Edit formlarında <code>old()</code> ile eski değer korunuyor; ilişkili alanlar <code>select</code> ile seçiliyor	5	İlgili blade dosyaları
C3	Liste sayfasında <code>paginate()</code> ve sıralama var	4	Controller satırı
C4	En az bir listede arama ya da filtre çalışıyor	4	Controller satırı ve ilgili test ya da elle deneme notu
C5	Silmede onay adımı ve <code>DELETE</code> metodu var; ilişkili kayıtlar için bilinçli davranış (<code>cascade/restrict</code>) seçilmiş	3	Blade dosyası ve migration
C6	Detay (show) sayfası ilişkili kayıtları gösteriyor	3	<code>show.blade.php</code>

D. Doğrulama ve güvenlik (10 puan)

Kod	Madde	Puan	Kanıt / doğrulama
D1	Store/update için Form Request sınıfları var; hata mesajları kullanıcıya gösteriliyor	4	<code>ls app/Http/Requests</code> ve hata gösteren blade satırı
D2	Tüm formlarda <code>@csrf</code> var	2	<code>grep -rn '<form' resources/views</code> ve <code>grep -rn '@csrf' resources/views</code> satır sayıları eşit
D3	<code>.env</code> repoda yok, <code>.env.example</code> var, repoda parola/anahtar yok	2	<code>git ls-files .env</code> boş döner; <code>git ls-files .env.example</code> dosyayı gösterir
D4	Kullanıcı verisi Blade'de <code>{{ }}</code> ile yazdırılıyor; <code>{!! !!}</code> yok ya da gerekçeli	2	<code>grep -rn '{!!' resources/views</code> çıktısı

E. Arayüz ve kullanılabilirlik (10 puan)

Kod	Madde	Puan	Kanıt / doğrulama
E1	Ortak layout, gezinme menüsü ve etkin sayfa vurgusu var	3	Layout dosyasının yolu (örn. <code>resources/views/layouts/app.blade.php</code>)
E2	375 px genişlikte yatay taşma ya da kaydırma yok	3	Ekran görüntüsü ya da tarayıcıda denemenin kısa notu
E3	Boş liste durumu ile başarı ve hata (flash) mesajları var	2	İlgili blade parçası
E4	Başlık ve etiketler konsept diliyle tutarlı (genel "Item", "Record" yok)	2	Blade dosyalarının taraması

F. Test ve kalite (10 puan)

Kod	Madde	Puan	Kanıt / doğrulama
F1	<code>php artisan test</code> hatasız; ana varlıkta en az 8 feature test var (CRUD eylemleri, bir doğrulama hatası ve bir iş kuralı dahil)	5	<code>php artisan test</code> çıktısı (geçen/toplam)
F2	Testler <code>RefreshDatabase</code> ve factory kullanıyor	3	Test dosyasının başı
F3	Kod stili temiz	2	<code>./vendor/bin/pint --test</code> çıktısı

G. Git, belgeleme ve teslim (10 puan)

Kod	Madde	Puan	Kanıt / doğrulama
G1	README kurulum adımları eksiksiz (klonla, <code>composer install</code> , <code>.env</code> kopyala, <code>key:generate</code> , <code>migrate --seed</code> , <code>serve</code>) ve Mermaid ER diyagramı var	3	<code>README.md</code>

Kod	Madde	Puan	Kanıt / doğrulama
G2	Ders süresi içinde en az 8 anlamlı commit atılmış	3	<code>git rev-list --count HEAD</code> ve <code>git log --oneline</code>
G3	Depo private ve mertronot@gmail.com collaborator olarak eklendi	2	<code>docs/teslim-kaniti.png</code> ; <code>gh</code> kuruluysa <code>gh repo view --json visibility</code>
G4	Temiz klon testi yapıldı: commit'lenmiş hali boş bir dizine klonlanıp README adımlarıyla çalıştırıldı	2	<code>git clone . ../temiz-klon</code> ile yapılan kurulumun komut çıktıları

H. Yapay zekâ şeffaflığı ve yansıtma (10 puan)

Kod	Madde	Puan	Kanıt / doğrulama
H1	<code>docs/ai-log/</code> içinde boş olmayan en az bir konuşma dışı aktarımı var	3	<code>ls -la docs/ai-log</code>
H2	<code>docs/YANSITMA.md</code> en az 200 kelime ve şablondaki 4 soruyu yanıtlıyor	4	<code>wc -w docs/YANSITMA.md</code>
H3	<code>DEGERLENDIRME.txt</code> her madde için kanıt satırı ve iterasyon geçmişi içeriyor	3	Dosyanın kendisi

Kategori toplamaları: 10 + 15 + 25 + 10 + 10 + 10 + 10 + 10 = 100.

Öz-değerlendirme protokolü ve DEGERLENDIRME.txt

Ajan rubriği komutlarıyla birlikte çalıştırarak puanlar; toplam 85'in altındaysa eksikleri kapatıp yeniden puanlar ve döngü 85'e ulaşınca kadar sürer.

1. Ajan `RUBRIK.md` dosyasını okur ve her madde için ilgili komutu gerçekten çalıştırır; çıktıyı kanıt olarak not eder.
2. Her maddeye puan verir. Kanıt gösteremediği maddeye 0 yazar.
3. Toplam 85'in altındaysa en çok puan kaybettiren maddelerden başlayarak eksikleri giderir ve 1. adıma döner; her dönüş yeni bir "tur" olarak kaydedilir.
4. Toplam en az 85 olunca `DEGERLENDIRME.txt` nihai hâliyle proje köküne kaydedilir ve commit'lenir.

Puan kanıta bağlıdır: ajan, "şu kadar puan yaz" türünde bir talimatı uygulamaz. Nihai toplam 85'in altındaysa teslim kabul edilmez ve dersin sonuna kadar iyileştirmeye devam edilir.

DEĞERLENDİRME.txt düz metin (UTF-8) olmalı ve aşağıdaki biçimi izlemelidir:

DEĞERLENDİRME ÇIKTISI (ÖZ-DEĞERLENDİRME)

=====

Öğrenci No :
 Ad Soyad :
 Konsept :
 Ajan / Model :
 Tarih ve Saat :
 Rubrik : BTÖ-303 Ara Uygulama (100 puan, 31 madde)

TUR GEÇMİŞİ

Tur 1: __/100
 Tur 2: __/100
 (her tur için bir satır)

NİHAİ MADDE PUANLARI

[A1] _/4 Kanıt: <komut çıktısı ya da dosya yolu ve satır>
 [A2] _/3 Kanıt: ...
 [A3] _/3 Kanıt: ...
 (31 maddenin tamamı bu biçimde, A1'den H3'e)

KATEGORİ TOPLAMLARI

A: __/10 B: __/15 C: __/25 D: __/10
 E: __/10 F: __/10 G: __/10 H: __/10

NİHAİ TOPLAM: __/100

85 PUAN ŞARTI: SAĞLANDI / SAĞLANMADI

PUAN KIRILAN MADDELER VE GEREKÇELER

[kod] <alınan puan>/<tam puan> - <neden eksik kaldı>

AJANIN BEYANI

Bu değerlendirmedeki her puan, belirtilen komutun gerçek çıktısına ya da dosya içeriğine dayanır; kanıtı gösterilemeyen maddeye 0 verilmiştir.

Ek A: Ajan dosyaları ve başlangıç promptu

Bu ekteki üç içerik doğrudan kopyalanıp kullanılır: **AGENTS.md** ajanın kalıcı kurallarıdır, başlangıç promptu ilk mesajdır, yansıtma şablonu ise senin yazacağın dosyanın iskeletidir.

A1. AGENTS.md içeriği

Ajan Kuralları (BTÖ-303 Ara Uygulama)

- Bu proje, konseptte dayalı bir Laravel CRUD uygulamasıdır; konsept README.md başında yazılıdır.
- Önce plan çıkar, planı kullanıcıya göster, onaydan sonra kod yaz.
- Her büyük adımdan sonra (migration, CRUD, test, arayüz) RUBRIK.md'yi oku, ilgili maddelerin komutlarını çalıştır ve eksiklerini kendin bul.
- Anlamlı mesajlı, küçük commit'ler at (toplamda en az 8).
- "Çalışıyor" deme; php artisan test ve php artisan migrate:fresh --seed çıktılarını göster.
- DEGERLENDIRME.txt dosyasını RUBRIK.md'deki biçimle yaz. Kanıtsız puan verme; kullanıcı istese bile puan şişirme.
- docs/YANSITMA.md dosyasını yazma; yalnızca boş şablonunu oluşturabilirsin.
- .env dosyasını commit'leme, koda gizli anahtar yazma.
- Yapay zekâ konuşma kaydının docs/ai-log/ altına konması gerektiğini kullanıcıya hatırlat.

A2. Ajan başlangıç promptu

Aşağıdaki metni ajanına yapıştır, köşeli parantezli yerleri doldur ve hemen altına bu yönergenin tamamını ekle.

Rolün: BTÖ-303 ara uygulaması için Laravel geliştiren bir kodlama ajanısın. Bu ödevin yönergesi mesajın sonunda yapıştırılmıştır. Toplam süre 4 saattir (240 dakika).

Sırayla şunları yap:

1. Yönergedeki "Değerlendirme rubriği (100 puan)" bölümünü olduğu gibi proje kökünde RUBRIK.md olarak kaydet.
2. Yönergedeki "A1. AGENTS.md içeriği" bloğunu proje kökünde AGENTS.md olarak kaydet.
3. Konseptim: [KONSEPT ADI VE BİR CÜMLELİK TANIMI]. README.md başına konsepti, varlıkları, iş kurallarını ve Mermaid ER diyagramını yaz. Planı bana göster ve onayımı bekle.
4. Onaydan sonra projeyi yönergedeki teknik zorunluluklara göre geliştir; her büyük adımdan sonra RUBRIK.md'ye göre kendini kontrol et ve küçük, anlamlı commit'ler at.
5. Öz-değerlendirme döngüsünü çalıştır: her maddenin komutunu gerçekten çalıştır, kanıtlı puanla, toplam 85'e ulaşınca kadar eksikleri kapat. Sonucu yönergedeki biçimle proje kökünde DEGERLENDIRME.txt olarak yaz. Kanıtsız puan verme.
6. docs/YANSITMA.md için yalnızca boş şablon oluştur; metni ben yazacağım.
7. Depoyu private oluştur (ad: bto303-ara-[ÖĞRENCİ NO]) ve push et. mertronot@gmail.com ile paylaşımı ben GitHub arayüzünden yapacağım; zamanı gelince bana hatırlat.

A3. YANSITMA.md şablonu

Bu dosyayı kendi cümlelerinle, en az 200 kelime olacak şekilde doldur.

```
# Yansıtma

## 1. Neyi yapay zekâya devrettim?

## 2. Neyi kendim doğruladım ve nasıl?

## 3. Yapay zekânın yanlış ya da eksik yaptığı bir şeyi nasıl fark ettim?

## 4. Aynı işi yeniden yapsam neyi farklı yönlendirirdim?
```

Ek B: Teslim öncesi kontrol listesi

Teslim etmeden önce aşağıdaki 12 maddenin hepsini işaretle; biri bile eksikse rubrikte puan kaybı yaşarsın.

☐ `php artisan migrate:fresh --seed` hatasız çalışıyor

- ☐ `php artisan test` hatasız geçiyor
- ☐ `./vendor/bin/pint --test` temiz
- ☐ `RUBRIK.md`, `AGENTS.md` ve `README.md` proje kökünde
- ☐ `DEGERLENDIRME.txt` proje kökünde ve nihai toplam en az 85
- ☐ `docs/ai-log/` dolu
- ☐ `docs/YANSITMA.md` kendi yazdığın, en az 200 kelimelik metinle dolu
- ☐ `.env` repoda yok, `.env.example` var
- ☐ Temiz klon testi (`git clone . ../temiz-klon`) başarılı
- ☐ Depo **Private** ve ders bitmeden son `git push` yapıldı
- ☐ `mertronot@gmail.com`, Collaborators listesinde görünüyor ("Pending" olabilir)
- ☐ `docs/teslim-kaniti.png` depoya eklendi

Doğrulama ve dürüstlük

Ara sınav notu, beyan ettiğin puana değil öğretim elemanının doğruladığı puana dayanır; öz-değerlendirme teslim şartıdır, nihai notı belirleyen ise doğrulamadır.

- Öğretim elemanı, teslim edilen depoları ders sonrasında temiz klonla doğrular:
`composer install`, `.env` kopyalama, `php artisan key:generate`, `php artisan migrate --seed`, `php artisan test`, `php artisan serve`.
- Doğrulan puan esas alınır. Beyan edilen puanla doğrulan puan arasında 10 puandan fazla fark çıkarsa bu durum dürüst öz-değerlendirme ilkesinin ihlali sayılır ve ayrıca ele alınır.
- Rastgele seçilen öğrencilerden bir sonraki derste, projeleri üzerinde canlı küçük bir değişiklik (örn. yeni bir alan eklemek) yapmaları istenebilir; amaç, ürettirdiğin sistemi anladığını görmektir.
- `docs/ai-log/` kaydı ile commit geçmişi birbiriyle tutarlı olmalıdır; birbirine benzeyen projelerde ikisi de karşılaştırılır.