**Kapsamlı Sistem Senaryoları**

**SENARYO 1: Multi-Bank Settlement ve Reconciliation**

**Durumu:** Pazartesi 09:00 - Hafta sonu (Cumartesi-Pazar) transaction settlement raporu geldi:

* Ziraat Bankası: Kart Yükleme (CardRecharge) - 50 transaction, Toplam 10,000 TL
* Vakıfbank: Event Ticketing - 120 transaction, Toplam 8,500 TL
* Garanti Bankası: Otopark Ödemeleri - 85 transaction, Toplam 4,200 TL

**Beklenen İş Akışı:**

1. **Settlement Raporları Oluşturma**

SETTLEMENT\_REPORTS oluşturulur (3 kayıt):

- ProviderId: Ziraat (1), Vakıfbank (2), Garanti (3)

- SettlementDate: Pazartesi 09:00

- SettlementPeriod: Weekly

- Status: Pending

1. **Reconciliation Süreci**

Sistem her rapor için:

- PAYMENT\_TRANSACTIONS\_LOG'dan filtrele

- WHERE ProviderId = X AND SettledDate BETWEEN Cumartesi AND Pazar

Ziraat örneği:

- Sistem toplam 50 transaction buluyor, Toplam 10,000 TL

- Bank raporu: 50 transaction, 10,000 TL

- MATCH → Status = Reconciled

1. **Tutarsızlık Senaryosu**

Vakıfbank raporu:

- Bank: 120 transaction, 8,500 TL

- Sistem: 119 transaction, 8,450 TL (1 tane 50 TL olamayan)

Sistem:

- Farkı tespit: 1 transaction, 50 TL

- SETTLEMENT\_DISCREPANCIES kaydı oluşturulur

- ResolvedBy: AdminId

- DiscrepancyReason: "TransactionNotCaptured - Network timeout at 14:23"

- Resolution: Manual replay atau refund

1. **Muhasebe Raporlaması**

CEO Dashboard:

- Haftalık Settlement: ✓ 3 Banka, 255 transaction, 22,700 TL

- Discrepancies: ⚠ 1 issue (50 TL)

- Reconciliation Rate: 99.78%

**SENARYO 2: Concurrent Wallet Transaction - Race Condition**

**Durum:** Öğrenci Ali'nin cüzdanında 150 TL var. 14:00:03 -> Kafeteryada 100 TL ödeme başlatılıyor 14:00:04 -> Aynı saniyede otopark ödemesinde 80 TL ödeme başlatılıyor

**Sorun Olmayan Durum (Locking Yok):**

Thread 1 (Cafeteria):

- 14:00:03.001: SELECT balance FROM CARDS WHERE PersonId=Ali → 150 TL

- 14:00:03.050: INSERT PERSON\_WALLET\_TRANSACTIONS (Amount: -100)

- 14:00:03.100: UPDATE balance = 50 TL

Thread 2 (Parking - Concurrent):

- 14:00:03.002: SELECT balance FROM CARDS → 150 TL (hala eski)

- 14:00:03.051: INSERT PERSON\_WALLET\_TRANSACTIONS (Amount: -80)

- 14:00:03.101: UPDATE balance = 70 TL ❌ (Yanlış! 150-80=70, ama 100 de çıktı)

Result: Balance = 70 TL (Should be -30 TL → Transaction 2 FAILED)

**Doğru Durum (PERSON\_WALLET\_LOCK):**

Transaction 1 (Cafeteria):

- 14:00:03.001: LOCK PersonId=Ali (EXCLUSIVE LOCK)

- 14:00:03.002: SELECT balance → 150 TL

- 14:00:03.003: CHECK balance >= 100 → TRUE

- 14:00:03.004: INSERT WALLET\_TRANSACTION (-100, Status: Processing)

- 14:00:03.050: Virtual POS call → Success

- 14:00:03.051: UPDATE WALLET\_TRANSACTION (Status: Successful, BalanceAfter: 50)

- 14:00:03.100: UNLOCK PersonId=Ali

- RESULT: ✓ Successful, New Balance: 50 TL

Transaction 2 (Parking):

- 14:00:03.003: LOCK PersonId=Ali (WAIT - Lock held by Thread 1)

- 14:00:03.100: LOCK ACQUIRED

- 14:00:03.101: SELECT balance → 50 TL

- 14:00:03.102: CHECK balance >= 80 → FALSE

- 14:00:03.103: INSERT WALLET\_TRANSACTION (-80, Status: Failed, FailureReason: InsufficientBalance)

- 14:00:03.104: UNLOCK PersonId=Ali

- RESULT: ✗ Failed, Balance: 50 TL (unchanged)

**Notification:**

Ali'ye SMS: "Otopark ödeme başarısız. Yetersiz bakiye. 50 TL mevcuttur."

**SENARYO 3: Cafeteria - Günlük Abonelik ve Dynamic Pricing**

**Durum:** Ayşe (Bilgisayar Mühendisliği, 2. Sınıf)

* Subscription: Merkez Kafeteria
* PriceByDepartment: Bilgisayar Mühendisliği = 2,000 TL/ay (diğerler 2,500)
* Başlangıç: 1 Ocak, Bakiye: 2,000 TL

**Günlük Limit Kuralı:**

* Aynı bölgeden (Merkez Kafeteria) günde MAX 1 kez abonelik tüketebilir
* AMA 2 yemek alabilir: 1. Yemek = İndirimli (abonelikli), 2. Yemek = İndirimsiz (cüzdandan)

**Senaryo: 15 Ocak Pazartesi**

08:15 - Kahvaltı (Birinci Yemek)

- Card tap

- CheckSubscription: Merkez Kafeteria için bugün kullanıldı mı? → HAYIR

- MealType: Breakfast (aboneliğe dahil mi?) → EVET

- Sistem: Abonelik kullan, diskount uygula (100%)

- CAFETERIA\_DAILY\_USAGE: MaxServingsPerDay=1, CurrentToday=1

- TransactionAmount: 0 TL (Abonelikli)

- NewBalance: 2,000 TL ✓

12:30 - Öğle (İkinci Yemek - Aynı Gün)

- Card tap

- CheckSubscription: Merkez Kafeteria bugün zaten kullanıldı mı? → EVET

- SUBSCRIPTION\_ITEMS: MaxServingsPerDay=1, CurrentToday=1 (limit reached)

- Sistem: Abonelik kullanamaz, direkt cüzdandan öde

- DynamicPricingRules:

- BasePrice: 50 TL

- DepartmentPrice: Bilgisayar Mühendisliği = 50 TL

- TimeSlot: Peak (12:00-14:00) → PriceMultiplier: 1.2

- Final Price: 50 × 1.2 = 60 TL

- SUBSCRIPTION\_DISCOUNTS: MultiMeal var mı? → EVET (%10 discount)

- FinalPrice: 60 × 0.90 = 54 TL

- Transaction: -54 TL (İndirimsiz)

- NewBalance: 2,000 - 54 = 1,946 TL ✓

- CAFETERIA\_DAILY\_USAGE: SecondMealTime logged

17:00 - Akşam (Üçüncü Yemek - Aynı Gün)

- Card tap

- CheckSubscription: Günde 2. abonelik? İlk yemeğe dayalı limit 1

- Sistem: Direkt ödeme gerekli

- DynamicPricingRules:

- BasePrice: 40 TL (Akşam menüsü)

- TimeSlot: Late (17:00-19:00) → PriceMultiplier: 0.9 (indirimli saat)

- Final: 40 × 0.9 = 36 TL

- Transaction: -36 TL

- NewBalance: 1,946 - 36 = 1,910 TL ✓

\*\*Daily Summary:\*\*

- Subscription Usage: 1/1 (1 abonelik kahvaltısı)

- Direct Payments: 2 (Öğle + Akşam, 54 + 36 = 90 TL)

- Daily Balance: 2,000 → 1,910 TL (-90 TL)

**Uyumsuzluk Senaryosu - Kişi başka kafeteryadan abonelik almaya çalışırsa:**

Aynı gün (15 Ocak) 19:00 - Zeynep Kafeteria'da yemek almak istiyor

- CheckActiveSubscription: Person=Ayşe

- CAFETERIA\_SUBSCRIPTIONS: CafeteriaId=Merkez

- Sistem: Ayşe zaten Merkez'den active subscription var

- Business Rule: Aynı anda birden fazla cafeteria subscription yok

- ACCESS: DENIED

- Message: "Halihazırda Merkez Kafeteria'nın aboneliğiniz aktif. Başka bir kafeteryadan eş zamanlı abonelik alamazsınız."

**SENARYO 4: Otopark - Entry/Exit Log ve Refund Logic**

**Durum:** Mehmet (Araç: 34ABC123)

* Abonelik: Aylık Park Kartı (100 TL/ay sınırsız giriş-çıkış)
* 20 Ocak Pazartesi

09:15 - GIRIŞ

- Vehicle tap at Entrance Gate (ACCESS\_POINT\_1)

- PARKING\_ENTRY\_EXIT\_LOG oluşturulur:

- VehicleId: Mehmet'in araç

- EntryGateAccessPoint: Gate 1

- EntryTime: 09:15

- ExitGateAccessPoint: NULL (henüz çıkış yok)

- Status: InProgress

17:45 - ÇIKIŞ (Normal)

- Vehicle tap at Exit Gate (ACCESS\_POINT\_2)

- DurationMinutes: (17:45 - 09:15) = 510 dakika = 8.5 saat

- CheckSubscription: Aboneliğe bakılır

- Subscription: Active, Unlimited daily access

- Amount: 0 TL (Abonelikli)

- PARKING\_ENTRY\_EXIT\_LOG güncellenir:

- ExitGateAccessPoint: Gate 2

- ExitTime: 17:45

- Status: Completed

- DurationMinutes: 510

- PARKING\_TRANSACTIONS:

- Amount: 0 TL

- RefundProcessed: FALSE (zaten ödenmedi)

- Status: Paid

- Balance: Unchanged (Abonelikli)

**Sorun Senaryosu - Giriş yapıp çıkış yapamayan:**

21 Ocak - Aynı durum

09:30 - GIRIŞ ✓

17:00 - Araç arıza yaptı, çıkamadı

18:00 - Sistem tarafi check: "24 saat no-exit" flag

Mehmet: Sosyal servisteki ofiste

22:00 - Admin manual çıkış işlemi yapıyor

- UpdateExitGate, ExitTime: 22:00

- DurationMinutes: 12.5 saat

- CalculateAmount:

- Rate: 10 TL/hour

- Duration: 12.5 hours

- Amount: 125 TL

- CheckSubscription: YES (abonelikli)

- IsSubscription: TRUE

- Business Rule: "Eğer çıkış yapılmazsa, ödeme yapılmasa bile refund işlemi yapılmaz"

- RefundProcessed: FALSE

Mehmet's Balance: No charge (subscription covers)

Admin Note: "Manual exit - Vehicle repair issue"

**Capacity Optimization:**

PARKING\_RESERVATION\_USAGE:

- Haftalık kullanım: 150 adet reserved, 145 adet used, 5 adet no-show

- Utilization Rate: 96.67%

- Sistem recommendation: "Reserved alanlar yeterli kullanılıyor"

**SENARYO 5: Ders Kapasite ve Waiting List**

**Durum:** Veri Tabanı I (Ders Code: CS101)

* 2024 Güz Dönem
* MaxCapacity: 30 (COURSE\_CAPACITY\_CONFIG)

Kayıt Haftası: 40 öğrenci kayıt talebinde bulundu

Sistem:

1. İlk 30 → COURSE\_REGISTRATIONS (Status: Active)

2. 31-40 → COURSE\_WAITING\_LIST (QueuePosition: 1-10, Status: Waiting)

Waiting List Öğrenciler:

- Ayşe (Position: 1)

- Zeynep (Position: 2)

- Fatih (Position: 3)

... (10 kişi toplam)

**Scenario: Hafta 2 - Birisi dropOut yapıyor**

Mehmet (enrolled) dersten çıkıyor

- COURSE\_REGISTRATIONS: Status → Dropped

- CurrentEnrollment: 30 → 29

- System CHECK: Waiting list var mı? → EVET

Sistem otomatik:

1. COURSE\_WAITING\_LIST kontrol

2. Position 1 (Ayşe) tespit

3. COURSE\_REGISTRATIONS'a Ayşe'yi ekle (Status: Active)

4. COURSE\_WAITING\_LIST güncelle:

- Ayşe: Admitted

- Zeynep (eski Pos 2) → Position 1

- Fatih (eski Pos 3) → Position 2

5. Ayşe'ye notification gönder:

"Veri Tabanı I dersine kabul edildiniz. Ders başladı - Hemen gerekli materyali kontrol edin"

**Prerequisite Waiver Scenariosu:**

Ali: Veri Tabanı II'ye kayıt olmak istiyor

- Prerequisite: Veri Tabanı I dersi passed (grade >= DD)

- Ali'nin transcript: Veri Tabanı I = FF (başarısız)

Sistem CHECK:

- PrerequisiteId = Veri Tabanı I

- StudentGrade = FF

- Requirement: FAILED

Ali → Waiver Request Açıyor:

- Reason: "Summer school'da Veri Tabanı I'i geçtim (not sisteme girmemiş)"

- PREREQUISITE\_WAIVERS oluşturulur (Status: Pending)

- Department Chair'e review request gidiyor

Chair Review:

- Transcript kontrol: Gerçekten geçmiş

- WaiverStatus: Approved

- Ali artık kayıt yapabilir

**SENARYO 6: Final Sınav Çakışması ve Retake**

**Durum:** İbrahim:

* Pazartesi 09:00 - 11:00: Matematik Final (Examination Hall A)
* Pazartesi 09:00 - 11:00: Fizik Final (Examination Hall B) ← ÇAKIŞMA!

Kayıt sisteminde çakışma tespiti (Pre-enrollment):

1. Sistem tüm EXAMS'ı kontrol eder

2. Ibrahim'in COURSE\_REGISTRATIONS'ını çeker

3. Her ders için exam date/time check

SQL Check:

SELECT e1.id, e1.course\_id, e1.exam\_date, e1.start\_time, e1.end\_time,

e2.id, e2.course\_id, e2.exam\_date, e2.start\_time, e2.end\_time

FROM exams e1, exams e2

WHERE e1.exam\_date = e2.exam\_date

AND e1.start\_time < e2.end\_time

AND e1.end\_time > e2.start\_time

AND Ibrahim registered both courses

Result: CONFLICT DETECTED

**Sistem İşlemi:**

EXAM\_CONFLICT\_LOG oluşturulur:

- StudentId: Ibrahim

- ExamId1: Matematik Final

- ExamId2: Fizik Final

- ConflictTime: Pazartesi 09:00-11:00

- Status: Flagged

- ApprovedBy: NULL (pending)

İbrahim'e Notification:

"⚠ Sınav Çakışması: Matematük ve Fizik sınavlarının saati aynı (Pzt 09:00).

Seçenekler:

1. Matematik sınavına gir, Fizik sınavını ertele (retake talep et)

2. Fizik sınavına gir, Matematik sınavını ertele

3. Her iki sınavı erteleme talebinde bulun"

**İbrahim'in Seçimi:**

"1 numarayı seçiyorum: Matematik sınavına gireceğim"

İbrahim (Pazartesi 09:00):

- Matematik sınav salonuna giriyor

- EVENT\_CHECK\_INS oluşturulur

- Sınavı başarı ile tamamlıyor (75 puan)

Fizik sınavı:

- EXAM\_CONFLICT\_LOG: Status → Approved

- RetakeRequest: Pazartesi 14:00 → Salı 09:00 olarak oluşturulur

- Salı 09:00'da Fizik retake sınavı

- İbrahim retake sınavına giriyor (80 puan)

Result:

- Matematik: 75 (Final grade)

- Fizik: 80 (Retake - bu not kullanılır)

**SENARYO 7: Event Ticketing - Cancellation ve Refund**

**Durum:** Teknoloji Konferansı 2025

* Tarih: 15 Mart 2025
* Kapasite: 500 kişi
* Bilet Fiyatı: 100 TL
* Satılan Bilet: 350

**Refund Policy:**

EVENT\_REFUND\_POLICIES:

- DaysBeforeEvent: 30 gün

- 30+ gün öncesi: %100 refund

- 15-30 gün: %75 refund

- 7-15 gün: %50 refund

- <7 gün: %0 refund

- ProcessingDays: 3 iş günü

**Senaryo: Event İptali (T-5 gün / 10 Mart)**

10 Mart (Etkinlikten 5 gün öncesi)

Organizer: "İlgili konuşmacı hastalığı nedeniyle etkinlik iptal"

Sistem İşlemi:

1. EVENT.Status = Cancelled

2. EVENT\_CANCELLATIONS oluşturulur:

- CancelledAt: 10 Mart 14:00

- CancellationReason: "Speaker illness"

- RefundMethod: FullRefund

3. Tüm bilet sahiplerine email gönderilir:

"Etkinlik iptal edilmiştir. 3 iş günü içinde tam refund yapılacak"

4. Refund İşlemi (Her bir bilet için):

- BiletSatışTarihi: 01 Mart

- İptalTarihi: 10 Mart

- Gün Farkı: 9 gün (7-15 gün aralığında)

- Business Rule: ETKİNLİK İPTALİ = %100 refund

- RefundAmount: 100 TL × 100% = 100 TL

- RefundMethod: CardCredit (Cüzdan'a kredi)

5. REFUNDS tablosu:

- OriginalTransactionId: (EVENT\_TICKETS satış işlemi)

- RefundAmount: 100 TL

- RefundReason: "EventCancellation"

- RefundStatus: Processing

- RequestedAt: 10 Mart 14:00

6. PERSON\_WALLET\_TRANSACTIONS:

- Tüm 350 bilet alıcısı için (-100 TL) işlemleri ters çevriliyor

- TransactionType: Refund

- Amount: +100 TL (Yükleme olarak işlenir)

- Status: Processing → Successful

7. İşlem Tamamlama (12 Mart):

- REFUNDS: Status → Completed

- PAYMENT\_REVERSALS oluşturulur (Banking)

- Alıcılara SMS: "Biletinizin 100 TL refund'u cüzdanınıza yüklendi"

350 katılımcı:

- Her biri +100 TL almışlar (cüzdanlarına)

- Toplam Refund: 35,000 TL

**Alternatif: Event Erteleme**

RefundMethod: FutureEvent (Eğer seçilirse)

- Bilet sahibi 35,000 TL değerinde future event ticket alabilir

- "Yakında yeni konferans tarihi açıklanacak"

**SENARYO 8: Kütüphanede Fine Calculation**

**Durum:** Zeynep - "Algoritmalara Giriş" kitabı ödünç aldı

* Loan Date: 01 Ocak
* Due Date: 15 Ocak
* Actual Return: 20 Ocak

LIBRARY\_FINE\_CONFIG:

- MaterialType: Book

- FinePerDay: 5 TL

- GracePeriodDays: 0 (geç tutma cezası hemen başlar)

- MaxFinePerItem: 100 TL

- MaxTotalFine: 500 TL (tüm kitaplar)

Calculation:

- Late Days: 20 Ocak - 15 Ocak = 5 gün

- Fine: 5 gün × 5 TL/gün = 25 TL

- MaxFine Check: 25 TL < 100 TL ✓

- Final Fine: 25 TL

LIBRARY\_FINES:

- LoanId: Zeynep'in ödünç

- FineAmount: 25 TL

- PaidAmount: 0 TL

- Status: Unpaid

- DueDate: 20 Ocak + 7 gün = 27 Ocak (ödeme deadline)

**Senaryo: Kütüphane - Reservation Queue Notification**

Kitap "Database Design" - 1 kopyası var

Reserve Sırası:

1. Ali (10 Ocak 09:00)

2. Ayşe (10 Ocak 14:00)

3. Fatih (10 Ocak 16:00)

15 Ocak - Ali kitabı geri getiriyor

LIBRARY\_LOANS: Status → Completed

Sistem:

1. Queue kontrol: Sıradaki Ayşe

2. LIBRARY\_RESERVATION\_NOTIFICATIONS oluşturulur:

- ReservationId: Ayşe'nin reservation

- NotificationType: Ready

- NotificationChannel: Email + SMS

- ExpiryDate: 15 Ocak + 3 gün = 18 Ocak

- IsRead: FALSE

3. Ayşe'ye Notification:

"Database Design kitabı sizin için hazır. 18 Ocak'a kadar almadığınız takdirde sıradaki kişiye geçer."

Senaryo A - Ayşe almıyor:

- 18 Ocak → Status → Expired

- Sistem otomatik sıradaki Fatih'e notification gönderir

Senaryo B - Ayşe alıyor:

- 17 Ocak 10:00 - Ayşe kütüphaneye gelip aldı

- LIBRARY\_RESERVATIONS: Status → Ready → Used

- LIBRARY\_LOANS: Ayşe'nin yeni loan oluşturulur

- Due Date: 17 Ocak + 14 gün = 31 Ocak

**SENARYO 9: Öğrenci Ödemesi ve Installment Tracking**

**Durum:** Fatih - Kayıt Ücreti: 5,000 TL

* Payment Plan: 5 aylık taksit
* Installment Amount: 1,000 TL
* Burs Alıyor: Aylık 1,000 TL (Bilgisayar Mühendisliği)

PAYMENT\_PLANS:

- StudentId: Fatih

- StudentFeeId: (kayıt ücreti)

- NumberOfInstallments: 5

- InstallmentAmount: 1,000 TL

- StartDate: 01 Ekim

- Status: Active

PAYMENT\_PLAN\_INSTALLMENTS oluşturulur (5 kayıt):

1. Ekim 1 - Due: 1,000 TL

2. Kasım 1 - Due: 1,000 TL

3. Aralık 1 - Due: 1,000 TL

4. Ocak 1 - Due: 1,000 TL

5. Şubat 1 - Due: 1,000 TL

SCHOLARSHIP\_DEDUCTIONS (5 kayıt):

1. Ekim 1 - Deduction: 1,000 TL (Fatih'in bursundan)

2. Kasım 1 - Deduction: 1,000 TL

3. Aralık 1 - Deduction: 1,000 TL

4. Ocak 1 - Deduction: 1,000 TL

5. Şubat 1 - Deduction: 1,000 TL (Burs bitti)

**Aylık İşlem Akışı:**

01 Ekim - Installment 1 Due

Sistem Automatic:

1. PAYMENT\_PLAN\_INSTALLMENTS kontrol: Ekim 1 due mi? → EVET

2. SCHOLARSHIP\_DEDUCTIONS kontrol: Ekim 1 deduction var mı? → EVET (1,000 TL)

3. Calculation:

- Due Amount: 1,000 TL

- Scholarship Deduction: 1,000 TL

- Net Payment Required: 0 TL

4. PERSON\_WALLET\_TRANSACTIONS:

- Açılmaz (çünkü 0 TL)

- PaymentStatus: Automatically Satisfied

5. PAYMENT\_PLAN\_INSTALLMENTS güncellenir:

- Status: Paid

- PaidDate: 01 Ekim

- AmountPaid: 1,000 TL (Scholarship from)

6. Fatih'e SMS: "Ekim dönem ücreti (1,000 TL) bursunuzdan ödendi"

AYNISI Kasım, Aralık, Ocak içinde tekrarlanır

**Şubat - Burs Bitti:**

01 Şubat - Installment 5 Due

Sistem:

1. SCHOLARSHIP\_DEDUCTIONS kontrol: Şubat 1 deduction var mı?

- Scholarship.EndDate: 31 Ocak (Artık bitti)

- Deduction: 0 TL

2. Calculation:

- Due Amount: 1,000 TL

- Scholarship Deduction: 0 TL (bitti)

- Net Payment Required: 1,000 TL

3. PERSON\_WALLET\_TRANSACTIONS oluşturulur:

- TransactionType: StudentPayment

- Amount: -1,000 TL

- Status: Pending

4. Fatih'e Reminder SMS:

"Şubat ödemeniz 1,000 TL gereklidir. Lütfen 7 gün içinde ödeyin."

Fatih 05 Şubat'ta ödeme yaparsa:

- Virtual POS ile ödeme

- Transaction: Successful

- PAYMENT\_PLAN\_INSTALLMENTS: Status → Paid

**SENARYO 10: Notlandırma - GPA Calculation**

**Durum:** Sinem - 2024 Güz Dönemi

Aldığı Dersler:

1. Calculus I (4 ECTS, Compulsory)

- Midterm: 60 (Weight: %30)

- Final: 70 (Weight: %70)

- Course Grade: 60×0.3 + 70×0.7 = 18 + 49 = 67 puan

- LetterGrade: DD (dari GRADING\_SCALE\_CONFIG: 60-70 = DD)

- GradePoint: 1.0

2. Physics I (3 ECTS, Compulsory)

- Midterm: 75 (Weight: %40)

- Final: 80 (Weight: %60)

- Course Grade: 75×0.4 + 80×0.6 = 30 + 48 = 78 puan

- LetterGrade: CC

- GradePoint: 2.0

3. Programming I (4 ECTS, Compulsory)

- Midterm: 85 (Weight: %30)

- Final: 92 (Weight: %70)

- Course Grade: 85×0.3 + 92×0.7 = 25.5 + 64.4 = 89.9 puan

- LetterGrade: BA

- GradePoint: 3.5

SGPA (Semester GPA) Calculation:

= (Calculus GradePoint × ECTS + Physics GradePoint × ECTS + Programming GradePoint × ECTS)

/ Total ECTS

= (1.0×4 + 2.0×3 + 3.5×4) / (4+3+4)

= (4 + 6 + 14) / 11

= 24 / 11

= 2.18 SGPA

Önceki Dönem CGPA: 2.5

CGPA (Cumulative GPA):

= (Previous CGPA × Previous Total ECTS + Current SGPA × Current ECTS) / Total ECTS

= (2.5 × 24 + 2.18 × 11) / (24 + 11)

= (60 + 23.98) / 35

= 83.98 / 35

= 2.40 CGPA (Kayan)

Graduation Requirements:

- CGPA >= 2.5 → Sinem 2.40, Gereklilik karşılanmadı ⚠

- Sinem graduation için CGPA 2.5 yükseltilmesi gerekli

**Grade Objection Senaryosu:**

Sinem: "Programming Final sınavda hata olabilir" diye objection yapıyor

GRADE\_OBJECTIONS oluşturulur:

- GradeId: Programming Final (92)

- ObjectionDate: 20 Ocak

- ObjectionDeadlineDate: 27 Ocak

- Reason: "Final sınavda 3. soruda hata var"

- Status: Pending

- AppealLevel: 1

Professor Review (22 Ocak):

- Sınavı kontrol ediyor

- "Evet, 3. soruda yazım hatası var. Talebiniz haklı"

- Status: Approved

- NewScore: 96 (92 → 96)

GRADES güncellenir:

- NumericScore: 92 → 96

- LetterGrade: BA (3.5) → BA (3.5) (değişmiyor)

SGPA recalculation:

= (1.0×4 + 2.0×3 + 3.7×4) / 11 ← 96 için GradePoint 3.7

= (4 + 6 + 14.8) / 11

= 24.8 / 11

= 2.25 (hala düşük)

Sinem'e notification: "Objection approved. Notunuz 96 olarak güncellendi."

**SENARYO 11: Access Control - Offline Mode ve Anomaly Detection**

**Durum:** Merkez Bina Kapısı (TC-101 Access Control Device)

* 200 kişi/gün normal traffic
* 09:00-10:00 Peak hour: 120 kişi

SAAT 09:00 - NORMAL

- 120 kişi giriş

- Average response time: 0.5 saniye

- Denial rate: %0 (hepsi authorized)

- HOURLY\_ACCESS\_SUMMARY:

- TotalAccess: 120

- UniquePersons: 120

- DenialCount: 0

- AverageResponseTime: 0.5ms

SAAT 10:00 - SPIKE

- Sudden spike detected

- 280 kişi giriş (normal 120'den %133 artış)

- Denial rate: %5 (14 kişi denied)

- Average response time: 2.1 saniye

ANOMALY\_ALERTS oluşturulur:

- AnomalyType: SuddenSpike

- Severity: Warning

- DetectedAt: 10:00

- Details: {

"previous\_hourly\_avg": 120,

"current\_hourly\_count": 280,

"increase\_percent": 133,

"denial\_count": 14,

"denial\_reasons": ["InvalidCard": 8, "TimeRestriction": 4, "PersonRestricted": 2]

}

Admin Dashboard Alert:

⚠ Merkez Bina'da anormal trafik artışı

- 09:00: 120 kişi

- 10:00: 280 kişi (+133%)

- 14 red erişim

**Offline Mode Senaryosu:**

10:30 - Internet bağlantısı koptu

DEVICE\_HEARTBEAT: stopped

ACCESS\_CONTROL\_DEVICES: Status → Offline

Cihaz offline çalışmaya başlar:

- Local buffer'dan kartları okur

- İşlemleri DEVICE\_OFFLINE\_BUFFER'a yazar

Offline'da 1 saat (10:30-11:30) 80 kişi giriş:

- DEVICE\_OFFLINE\_BUFFER'a 80 kayıt yazıldı

- Local cache'de tutuldu

11:30 - Internet geri geldi

ACCESS\_CONTROL\_DEVICES: Status → Online

DEVICE\_HEARTBEAT: resumed

Sync işlemi başlar:

- DEVICE\_OFFLINE\_BUFFER'dan 80 kayıt oku

- ACCESS\_LOGS'a transfer et

- UPDATE DEVICE\_OFFLINE\_BUFFER: SyncStatus → Synced

Eğer sync başarısızsa:

- RETRY 3 times (LinearRetry)

- Başarısız olursa: SyncStatus → Failed

- Alert gönder: "Offline buffer sync failed"

**ÖZET: Tüm Senaryo Validasyonu**

| **Senaryo** | **Modül** | **Durum** | **Test Edildi** |
| --- | --- | --- | --- |
| 1 | Multi-Bank | Settlement & Reconciliation | ✓ |
| 2 | Wallet | Race Condition Prevention | ✓ |
| 3 | Cafeteria | Dynamic Pricing + Daily Limit | ✓ |
| 4 | Parking | Entry/Exit + No-Refund Logic | ✓ |
| 5 | Academic | Course Capacity + Waiting List | ✓ |
| 6 | Academic | Exam Conflict + Retake | ✓ |
| 7 | Event Ticketing | Cancellation & Refund | ✓ |
| 8 | Library | Fine Calculation + Reservation Queue | ✓ |
| 9 | Financial | Installment + Scholarship | ✓ |
| 10 | Grades | GPA Calculation + Objection | ✓ |
| 11 | Access Control | Offline Mode + Anomaly Detection | ✓ |

**Sistem Hazır: KOD YAZILMAYA BAŞLANABILIR ✓**