



**Yıldız Teknik Üniversitesi Bilgisayar Mühendisliği Yazılım  
Mühendisliği Dersi Proje Ödevi**

**Dersin Yürütücüsü: Prof. Dr. Mehmet Sıddık Aktaş**

**Mayıs, 2024**

**Konu 2. EMEEK El Sanatları Kursu Programı**

**Grup No:11**

Esma Nur Ekmekci - 20011620

Anıl Kutay Uçan - 20011025

Mehmet Keçeci - 20011103

Emir Çağrı Aykın - 20011062

Efe Girgin - 19011095

1.	<u>Proje Alan Tanımı.....</u>	3
2.	<u>Kabul ve Kısıtlar.....</u>	4
3.	<u>Proje İş-Zaman Çizelgesi.....</u>	4
4.	<u>Ekip Organizasyon Şeması ve Görev Dağılımları.....</u>	5
5.	<u>Proje Risk Tablosu.....</u>	6
6.	<u>Analiz.....</u>	7
	6.1. <u>Kullanım Senaryosu Diyagramı.....</u>	7
	6.2. <u>Kullanım Senaryoları.....</u>	8
	a. <u>Senaryo-Kursiyer Kaydı.....</u>	8
	b. <u>Senaryo-Öğretmen Kaydı.....</u>	8
	c. <u>Senaryo-Ders Programı Oluştur.....</u>	9
	d. <u>Senaryo-Kurs Programı Oluştur.....</u>	9
	6.3. <u>İzlenebilirlik Tablosu.....</u>	10
	6.4. <u>Kavramsal Sınıf Diyagramı.....</u>	11
7.	<u>Tasarım.....</u>	11
	7.1. <u>Sıralama(Sequence)Diyagramı.....</u>	11
	7.2. <u>Durum(State)Diyagramı.....</u>	12
	7.3. <u>Etkinlik(Activity)Diyagramı.....</u>	13
8.	<u>Birim Test Sınamaları.....</u>	15
9.	<u>Uygulama Görüntüleri.....</u>	21
	9.1. <u>Uygulama-Giriş.....</u>	21
	9.2. <u>Uygulama-Hakkımızda.....</u>	21
	9.3. <u>Uygulama-Kurslar.....</u>	22
	9.4. <u>Uygulama-İşlemler.....</u>	22
	9.5. <u>Uygulama-Ders Ekleme.....</u>	22
	9.6. <u>Uygulama-Kurs Ekle.....</u>	23
	9.7. <u>Uygulama-Kursiyer Ekle.....</u>	23
	9.9. <u>Uygulama-Eğitmen Ekle.....</u>	24
	9.10. <u>Uygulama-Kullanıcı Bilgileri Sorgulama.....</u>	24
	9.11. <u>Uygulama-Kurs Satın Al.....</u>	25

## 1.Proje Alan Tanımı

Çeşitli el sanatları kursları sunan firmamız, kurs işlemlerini ve kursiyer yönetimini daha verimli hale getirecek bir yazılım sistemine ihtiyaç duymaktadır. Bu bilgi sistemi, aşağıda belirtilen özelliklere sahip olmalıdır.

### 1. Kurs Programları Oluşturma:

Yazılım ile çeşitli el sanatları kursları programlanabilmelidir.

Hizmet verilen farklı el sanatları kursları ve öğretmen kayıtları sistemde tutulmalıdır. Ayrıca kursiyerlerin daha önce katıldıkları kurslar ve gelecekte kayıt oldukları kurslar görülebilmelidir.

### 2. Hafta İçi/Hafta Sonu Programları ve Ücret Farkları:

Kurs programları hafta içi veya hafta sonu düzenlenir.

Öğretmenlerin hafta içi ve hafta sonu için farklı ücretleri olduğundan, kurs programını hazırlayan personel, kursun maliyetini hesaplayabilmelidir. Kurslar genellikle sabit kar payı ile satılır, ancak dönemsel ücret değişiklikleri veya doluluk durumuna göre fiyat farklılıkları uygulanabilir. Bu nedenle kursiyerlerin kurs başına ne kadar ücret ödediği sistemde kaydedilmelidir.

### 3. Kurs İçerikleri ve Detaylar:

Öğretmenler: Öğretmenlerin adı, telefon numaraları, adresi ve e-posta adresi ile birlikte çalışabildikleri saatler ve verdikleri derslerin detayları ve bedelleri sistemde tutulmalıdır.

Kursiyerler: Kursiyerlerin adı, telefon numaraları, adresi, e-posta adresi ve daha önce katıldıkları kurslar ile katılmak istedikleri kursların detayları sistemde kaydedilmelidir.

### 4. Kurs Programı Hazırlama Süreci:

Personel, kurs içinde hangi el sanatlarının olacağına karar verir. Ayrıca kursun hafta içi veya hafta sonu olup olmayacağını belirler. Sistem, kursa eklenebilecek uygun el sanatları derslerini listelerek yardımcı olur.

Personel, el sanatlarını ve zamanı belirledikten sonra, dersleri verecek öğretmenleri belirler. Sistem, seçilen zamanlarda müsait olan öğretmenlere göre seçim yapma imkanı sunar.

Dersler ve öğretmenler belirlendikten sonra, sistem kursun maliyetini hesaplar.

### 5. Satış Süreci:

Personel, kursiyerin öğrenmek istediği el sanatlarını, uygun zamanını (hafta içi veya hafta sonu) ve bütçesini sisteme girer. Sistem, kursiyer için uygun kursları listeler.

Kursiyer, uygun kurs programlarından birini seçerek ödemeyi yapar (nakit veya kredi kartı).

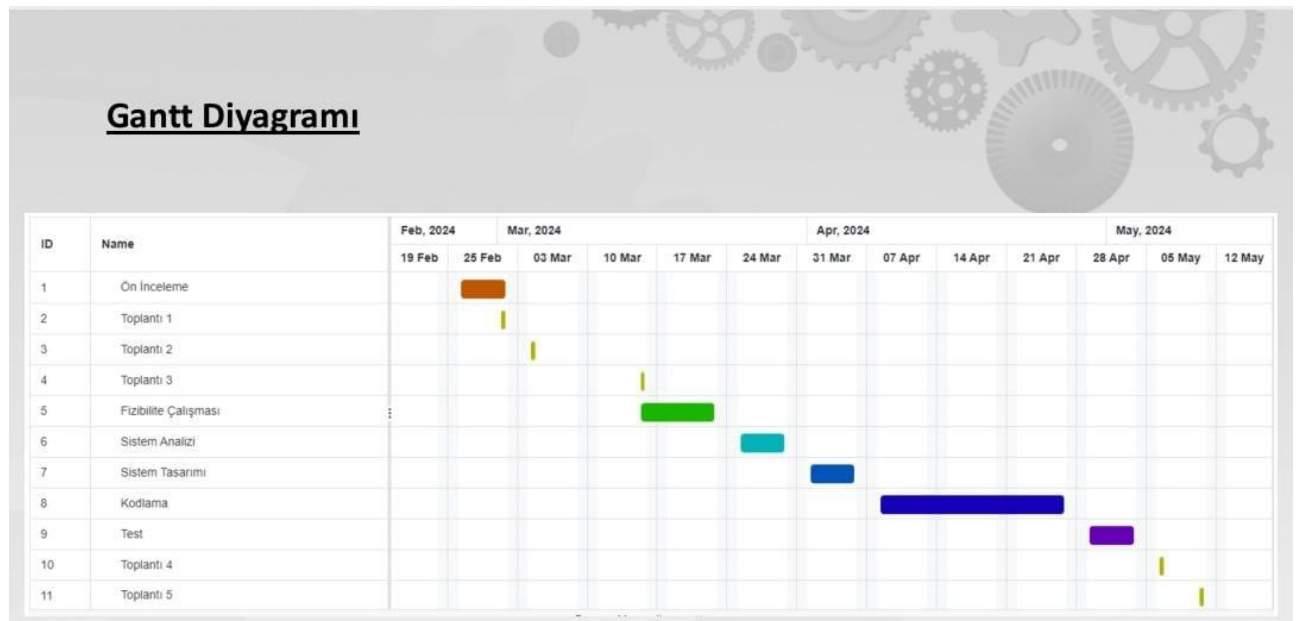
Sistem, satış işlemlerinin tamamını (kursiyer, seçilen kurs, ödenen miktar ve tarih) kaydeder. Bu yazılım sistemi sayesinde, kurs ve kursiyer yönetimi daha hızlı ve düzenli bir şekilde yapılacaktır.

## 2.Kabul ve Kısıtlar

- Bu kurum ihtiyaca göre ve uygun öğretmen bulunması ile beraber yeni el işi kategorileri açılabilir.
- Kursiyer ödemeyi yaparsa, personel ilgili kursa kursiyeri kaydeder.
- Kursiyerler yalnızca uygun zaman ve bütçe kriterlerine göre kendilerine sunulan kurs programlarına kayıt yapabilir.
- Kursların fiyatlandırması, öğretmenlerin saatlik ücretlerine ve talebe göre ayarlanır.
- Personel kursiyerin ve öğretmenin kayıtlarına ulaşabilir.
- Personel kurs programı oluşturma, öğretmen ve kursiyer ekleme gibi işlemleri yapabilir.
- Kurslar el işi derslerinden oluşmaktadır.
- Dersler her gün aynı saatte verilmektedir.
- Kurs ücreti, öğretmenlerin günlük ücretleri temel alınarak hesaplanır.
- Eğitmenler yetkin olmadığı konularda ders veremezler.
- Eğitmenler her gün çalışabilmektedir.
- Eğer bir ders eğitime atanırsa, o dersin günü eğitmenin takviminde dolu gözükür. Bu güne yeni bir ders atanamaz.
- Kurslar var olan el işi derslerinden oluşturulur. Eğer yeni bir el işi eklenmek isteniyorsa, önce bu el işi sistemde tanımlanmalıdır.
- Müşteriler sadece derslere kaydolamazlar. Ders paketlerinden oluşan kurslara kaydolmak zorundadırlar.
- Müşteriler, kurslara personel aracılığıyla kaydolabilir.
- Sistemde ödeme yapılırken sadece nakit veya kredi kartı kabul edilir.
- Kurs rağbet görmezse fiyatı düşer. Buna zamanı, fiyatı etkili olabilir.

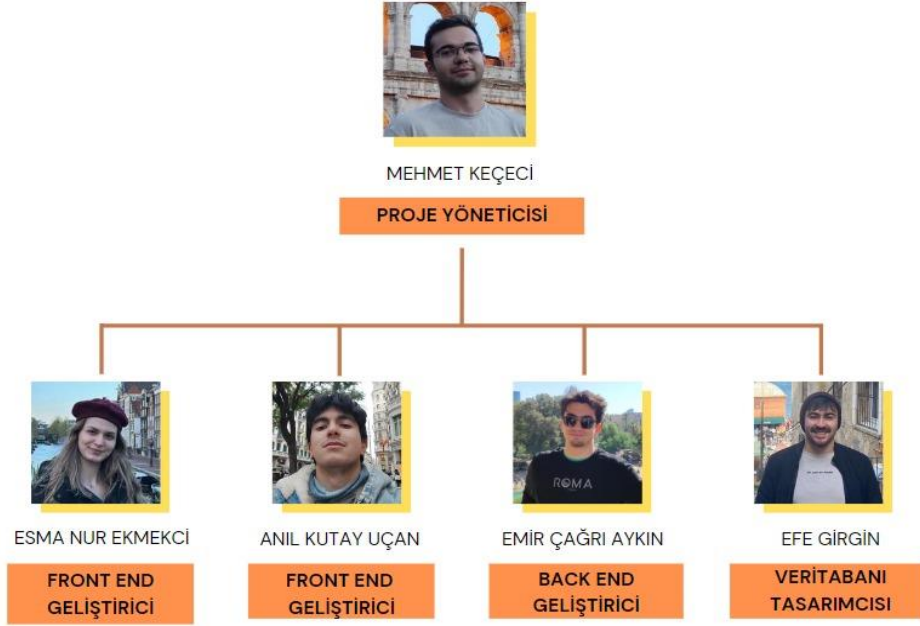
## 3.Proje İş-Zaman Çizelgesi

Bu kısımda projeye dair gantt diyagramı bulunmaktadır.



## 4.Ekip Organizasyon Şeması ve Görev Dağılımları

Bu kısımda ekip organizasyon şeması ve görev dağılımları bulunmaktadır.



## 5.Risk Tablosu

Tehlike	Olası Sonuç	Risk Seviyesi	Önlem
<b>Güvenlik açıkları nedeniyle veri hırsızlığı</b>	Kişisel ve ödeme bilgilerinin çalınması, şirketin itibarının zarar görmesi, müşteri kaybı	Yüksek	Firestore gibi güvenli bir veri tabanı kullanarak, düzenli olarak veri yedeklemesi yaparak ve yazılım güncellemelerini takip ederek risk seviyesi azaltılabilir.
<b>İşlem yaparken sistem kesintisi</b>	Müşterilerin işlemlerinin tamamlanmaması, müşteri kaybı	Orta	Sistemin yüksek kapasitesine dayanabilecek bir altyapı kullanmak ve yedek sistemlerin oluşturulması gibi önlemler alınarak risk seviyesi azaltılabilir
<b>Yanlış bilgi veya işlem hatası</b>	Yanlış fiyat veya yanlış uçuş saatleri gibi yanlış bilgilendirme, müşteri memnuniyetsizliği, mali kayıplar	Orta	Sistemin test edilmesi ve doğru bir şekilde yapılandırılması, müşteri hizmetlerinde eğitimli çalışanların bulunması gibi önlemler alınarak risk seviyesi azaltılabilir.
<b>Sistem yöneticisi hataları</b>	Yanlış veri silme veya yanlışlıkla veri değiştirme, mali kayıplar, şirket itibarının zarar görmesi	Orta	Verilerin düzenli olarak yedeklenmesi, sistem yöneticilerinin eğitilmesi, yetkilendirilme ve erişim haklarının kontrol edilmesi gibi önlemler alınarak risk seviyesi azaltılabilir.
<b>Yazılım hataları</b>	Uygulamanın çökmesi, müşteri memnuniyetsizliği, mali kayıplar	Orta-Yüksek	Yazılımın doğru şekilde test edilmesi, düzenli yazılım güncellemeleri, açık kaynak kod kullanımı ve güvenlik kontrollerinin sıkı bir şekilde uygulanması gibi önlemler alınarak risk seviyesi azaltılabilir

## 6.Analiz

Bu kısımda analize dair diyagramlar ve kullanım senaryoları bulunmaktadır.

### 6.1.Kullanım Senaryosu Diyagramı(Use Case diagram)

Bu kısımda sistemin genel kullanım senaryosu bulunmaktadır.



## 6.2.Kullanım Senaryoları

Bu kısımda kullanım senaryoları bulunmaktadır.

### a.Senaryo-Kursiyer Kaydı

Kullanım Senaryosu No:	1
Senaryo İsmi:	Kursiyer Kaydı
Birincil Aktör:	Kursiyer
İlgililer ve Beklentileri:	Kursiyer: Kendi ilgi alanlarına, müsait olduğu zamana ve bütçesine uygun kurs programlarına katılmak ister.
Ön Koşullar:	Kursiyer daha önce sisteme kaydolmamıştır. Kurs programı oluşturulmuş ve onaylanmış.
Ana Senaryo Adımları:	1. Personel, kursiyerin öğrenmek istediği el sanatlarını ve uygun zaman dilimini (hafta içi/hafta sonu) sisteme girer. 2. Sistem, uygun kurs programlarını listeler. 3. Kursiyer, programlardan birini seçer ve ödemesini (nakit veya kredi kartı) yapar. 4. Personel, kursiyerin kursa kayıt tarihini ve ödediği miktarı girer. 5. Sistem, kursiyer bilgilerini ve kurs kaydını doğrular ve kayıt işlemi tamamlanır.
Alternatif Akışlar:	Adım 2'de uygun program bulunamazsa, sistem kullanıcıyı uyarır ve arama kriterlerinin değiştirilmesini önerir.
Son Koşul:	Kursiyerin kurs kaydı sistemde kayıtlı hale gelir.

### b.Senaryo-Öğretmen Kaydı

Kullanım Senaryosu No:	2
Senaryo İsmi:	Öğretmen Kaydı
Birincil Aktör:	Personel
Ön Koşullar:	Personelin sisteme giriş yapmış olması.
Ana Senaryo Adımları:	1. Personel, "Yeni Öğretmen Ekle" seçeneğine tıklar. 2. Öğretmen bilgilerini (isim, telefon, adres, e-posta, çalışabileceği saatler, verebileceği dersler ve ücretler) girer. 3. Bilgiler doğrulandıktan sonra personel "Kaydet" butonuna tıklar. 4. Sistem, öğretmen kaydını başarılı bir şekilde tamamlar ve onay mesajı gösterir.
Alternatif Akışlar:	Adım 2'de eksik veya hatalı bilgi girilirse, sistem personeli uyarır ve düzeltilmesini ister.
Son Koşul:	Yeni öğretmen bilgileri sistemde kayıtlı hale gelir.



### c.Senaryo-Ders Programı Oluřturma

Kullanım Senaryosu No	3
Senaryo İsmi	Ders Programı Oluřturma
Birincil Aktör	Personel
Ön Kořullar	Personelin sisteme giriş yapmış olması.
Ana Senaryo Adımları	1. Personel, "Yeni Ders Programı Oluřtur" seçeneğine tıklar. 2. Ders türünü (örneğin, ahşap boyama, vitray, vb.) ve Ders hafta içi mi hafta sonu mu olacağını seçer. 3. Sistem, seçilen türe göre uygun el sanatlarını listeleyerek kullanıcıya sunar. 4. Personel, eklenmesini istediğı el sanatlarını seçer. 5. Personel, her ders için uygun öğretmeni seçer. 6. Sistem, seçilen öğretmenlerin uygun olup olmadığını kontrol eder. 7. Sistem, ders programını ve maliyetini hesaplar. 8. Personel onay verirse ders programı oluşturulur ve kaydedilir.
Alternatif Akıřlar	Adım 6'da seçilen öğretmenlerden biri müsait değilse, sistem uyarı verir ve alternatif öğretmen sunar.
Son Kořul	Ders programı oluşturulur ve sistemde kayıtlı hale gelir.

### d.Senaryo-Kurs oluřturma

Kullanım Senaryosu No	4
Senaryo İsmi	Kurs Programı Oluřturma
Birincil Aktör	Personel
Ön Kořullar	Personelin sisteme giriş yapmış olması ve mevcut derslerin belirlenmiş olması.
Ana Senaryo Adımları	1. Personel, "Mevcut Derslerden Kurs Programı Oluřtur" seçeneğine tıklar. 2. Personel, mevcut dersler arasından kursa dahil edilecekleri seçer ve kursun hafta içi mi hafta sonu mu olacağını belirler. 3. Sistem, seçilen derslere göre uygun sınıflama ve zamanlamayı kullanıcıya sunar. 4. Personel, kurs içerisinde yer alacak dersler için zaman çizelgesini onaylar. 5. Personel, her ders için uygun öğretmeni seçer. 6. Sistem, seçilen öğretmenlerin uygun olup olmadığını kontrol eder. 7. Sistem, kurs programını ve maliyetini hesaplar. 8. Personel onay verirse kurs programı oluşturulur ve kaydedilir.
Alternatif Akıřlar	Adım 6'da seçilen öğretmenlerden biri müsait değilse, sistem uyarı verir ve alternatif öğretmen sunar.
Son Kořul	Kurs programı oluşturulur ve sistemde kayıtlı hale gelir.

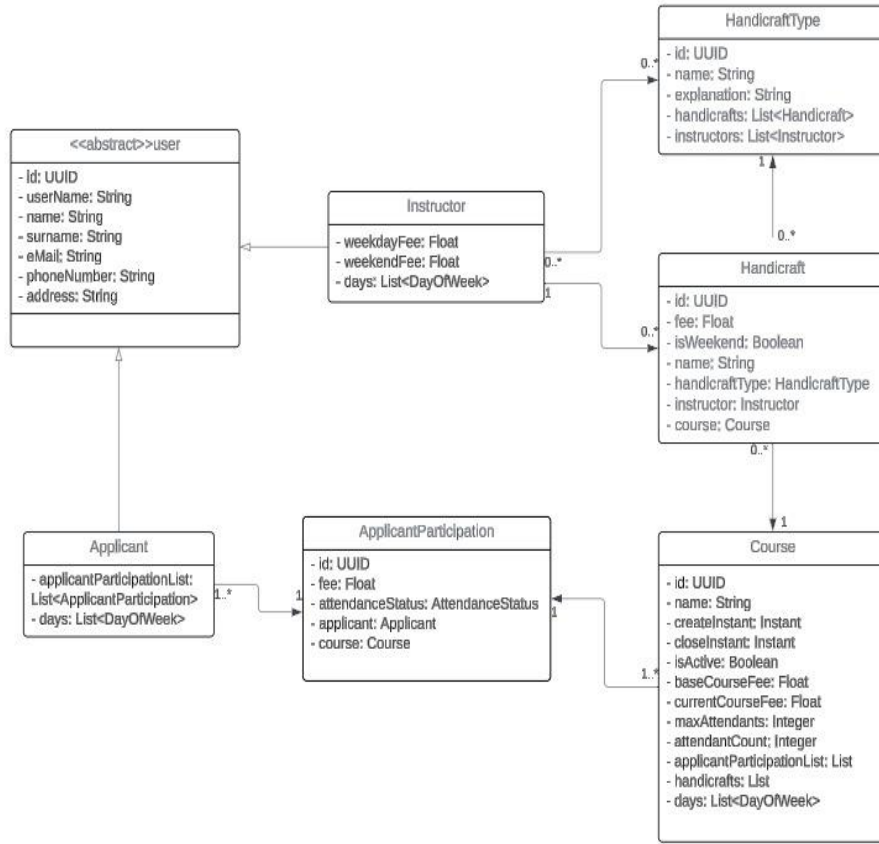
### 6.3.İzlenebilirlik Tablosu

Bu kısımda projeye ait gereksinimlerin ve gereksinimleri sağlayan servis, modül ve sınıfların yer aldığı izlenebilirlik tablosu bulunmaktadır.

İzlenebilirlik tablosu	Personel	Kursiyer	Öğretmen
Öğretmen kaydı yapılabilmelidir	x		
Kursiyer kaydı yapılabilmelidir	x		
Kurs programları hazırlanabilmelidir	x		
Kursiyerler geçmiş kurs bilgileri vb. görülebilmelidir.	x		
Öğretmenlerin çalıştığı saatler ,verebildiği dersler vb. görülebilmelidir.	x		
Kursiyerler, gelecekteki kurslara kayıt Yaptırabilmelidir	x		
Kurs maliyeti hesaplanabilmelidir	x		
Kurs satışlarının kayıtları tutulmalıdır	x		
Sistem, uygun saatlerdeki kursları otomatik olarak göstermelidir	x		
Kursiyer ödemelerini nakit veya kredi kartı ile alma	x		

## 6.4.Kavramsal Sınıf Diyagramı

Bu kısımda sınıflara dair methodların yer almadığı kavramsal sınıf diyagramı bulunur.

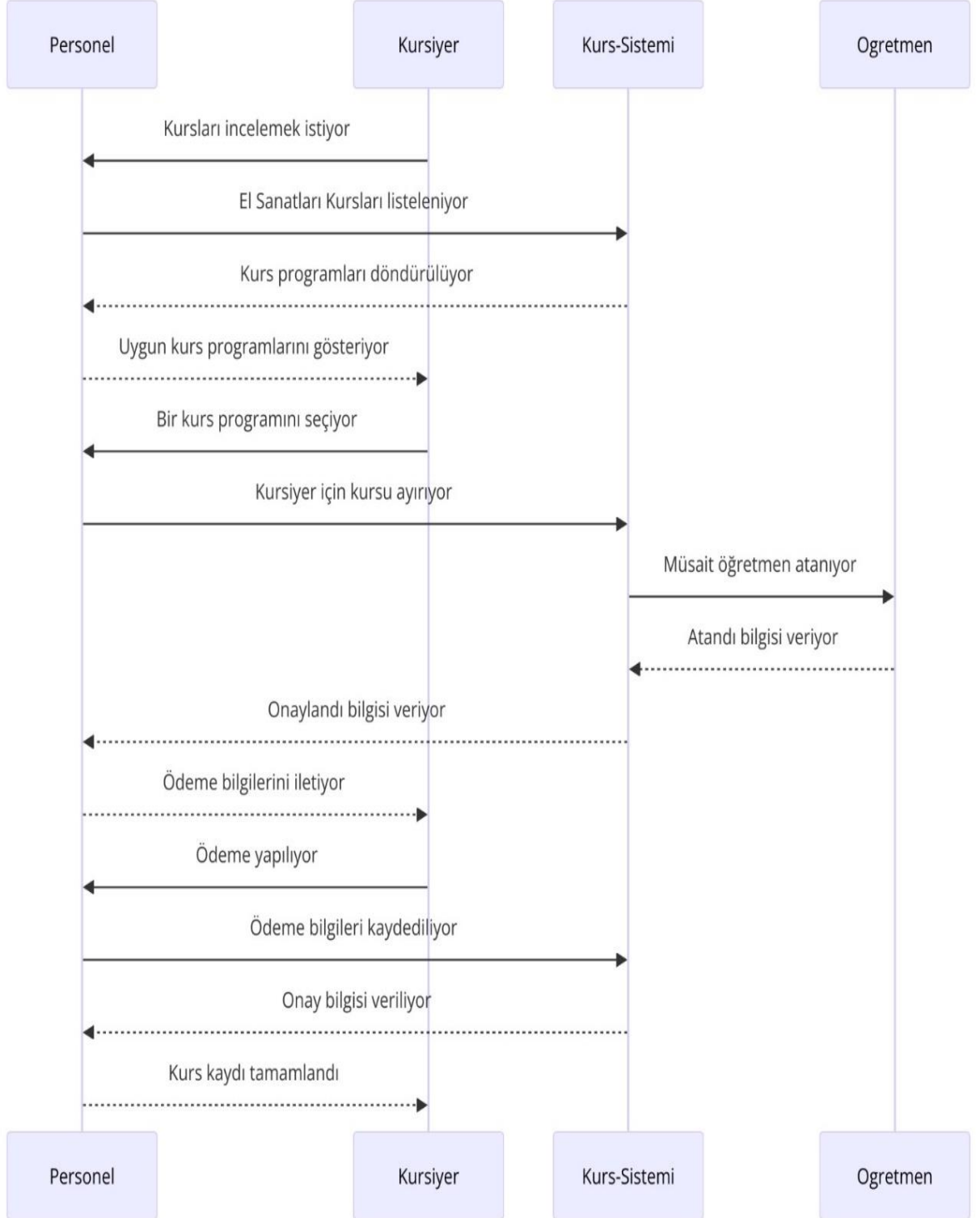


## 7.Tasarım

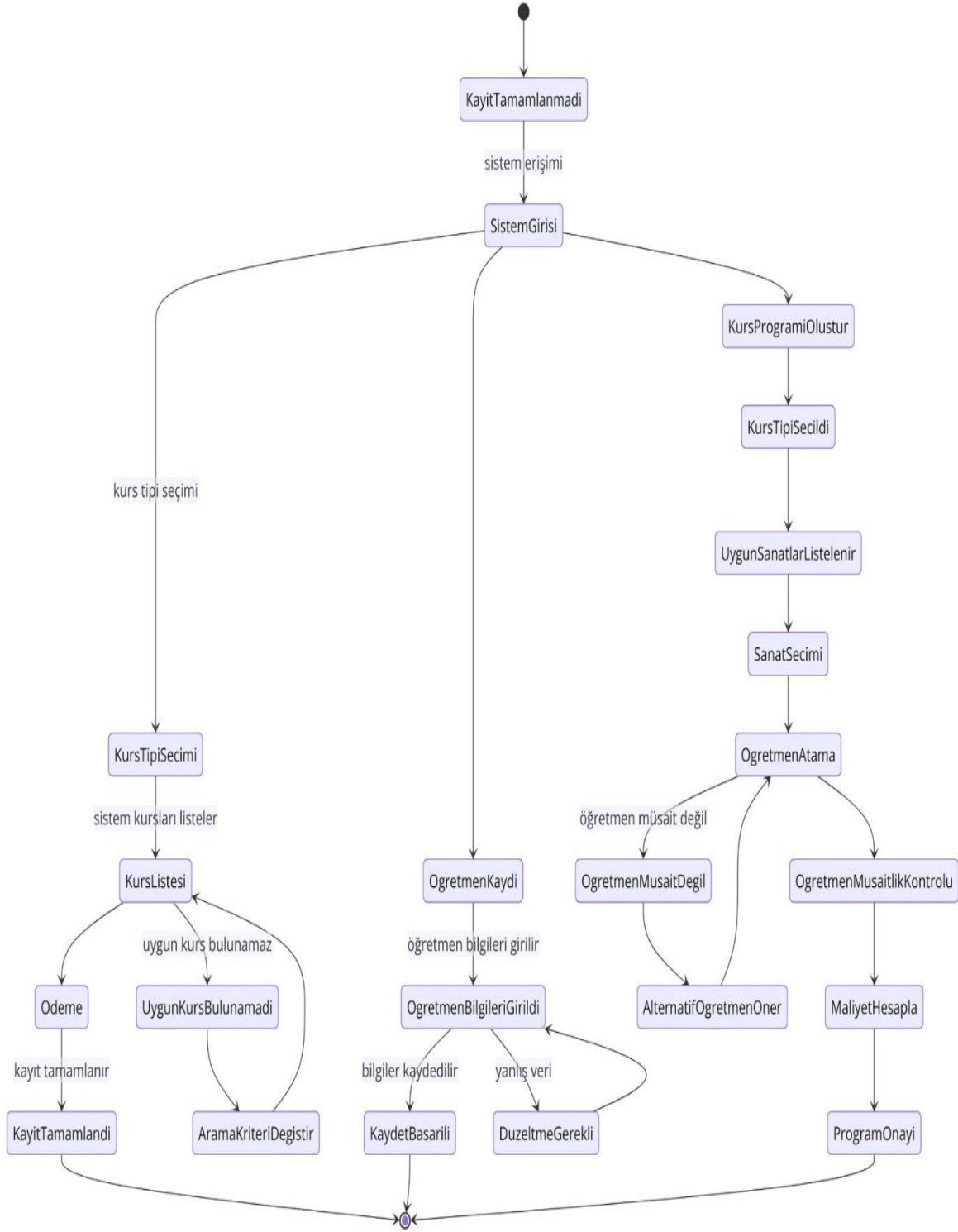
Bu kısımda tasarıma dair diyagramlar bulunmaktadır.

### 7.1.Sıralama(Sequence) Diyagramları

Bu kısımda ardışıl diyagramlar bulunmaktadır

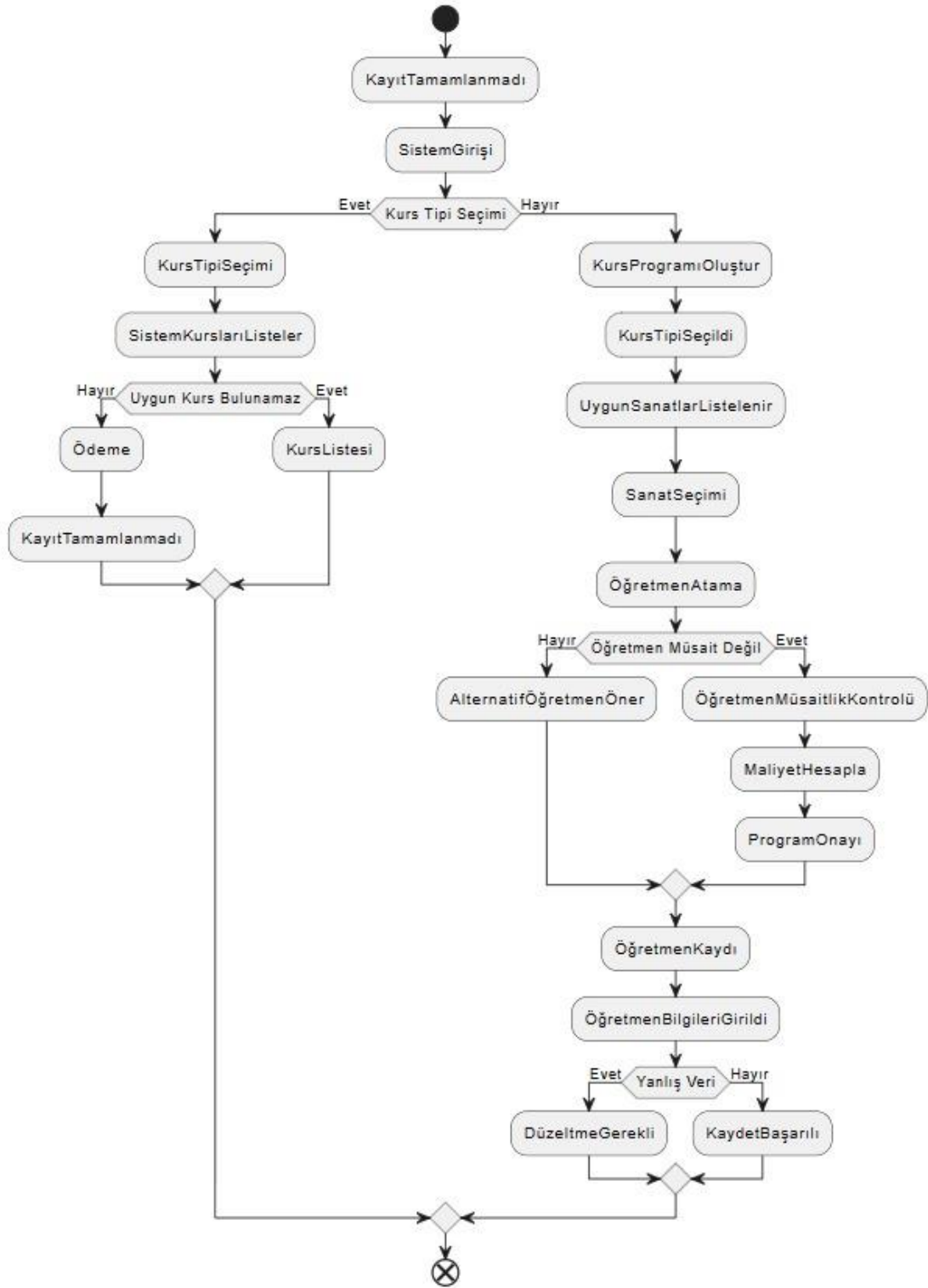


## 7.2. Durum (State) Diyagramları



### 7.3.Etkinlik(Activity)Diyagramları

Bu kısımda etkinlik diyagramları bulunmaktadır.



## 8.Birim Test Sınamaları

### a. Anıl Kutay Uçan

**Eğitmene el işi yeteneği eklerken bilgilerin doğru girilmesi durumunun testi:**

@Test

```
public void testAddHandicraftTypeToInstructor_whenCorrect() {  
  
    Instructor instructor = new Instructor("username", "surname", "name", "surname",  
"123456789", "address", 100F, 150F);  
  
    HandicraftType handicraftType = new HandicraftType("wood paint", "painting woods with  
colors");  
  
    //    instructor.setHandicraftTypeList(new ArrayList<>());  
  
    when(instructorRepository.save(any(Instructor.class))).thenReturn(instructor);  
  
    Instructor updatedInstructor = instructorService.addHandicraftTypeToInstructor(handicraftType,  
instructor);  
  
    assertNotNull(updatedInstructor);  
  
    assertTrue(updatedInstructor.getHandicraftTypeList().contains(handicraftType));  
  
    verify(instructorRepository, times(1)).save(any(Instructor.class));  
  
}
```

**Eğitmene el işi yeteneği eklerken bilgilerin ve eksik girilmesi durumunun testi:**

@Test

```
public void testAddHandicraftTypeToInstructor_whenNullInput() {  
  
    // Test when handicraftType is null  
  
    Instructor instructor = new Instructor("username", "surname", "name", "surname",  
"123456789", "address", 100F, 150F);  
  
    HandicraftType handicraftType = null;  
  
    instructor.setHandicraftTypeList(new ArrayList<>());  
  
    // Expecting RuntimeException when handicraftType is null  
  
    assertThrows(RuntimeException.class, () ->  
instructorService.addHandicraftTypeToInstructor(handicraftType, instructor));  
  
    // Test when instructor is null
```

```

    HandicraftType validHandicraftType = new HandicraftType("wood paint", "painting woods with
colors");

    Instructor nullInstructor = null;

    // Expecting RuntimeException when instructor is null

    assertThrows(RuntimeException.class, () ->
instructorService.addHandicraftTypeToInstructor(validHandicraftType, nullInstructor));

    // Verify that repository.save() is not called

    verify(instructorRepository, never()).save(any(Instructor.class));

}

```

## **b. Efe Girgin**

**Eğitmene el işi yeteneği eklerken zaten eğitimde olan bir yetenek eklenmeye çalışılması durumu testi:**

```

@Test

public void testAddHandicraftTypeToInstructor_whenAlreadyExist() {

    // Create an instructor

    Instructor instructor = new Instructor("username", "surname", "name", "surname",
"123456789", "address", 100F, 150F);

    // Create a handicraft type

    HandicraftType handicraftType = new HandicraftType("wood paint", "painting woods with
colors");

    // Set instructor's handicraft type list

    List<HandicraftType> handicraftTypeList = new ArrayList<>();

    handicraftTypeList.add(handicraftType);

    instructor.setHandicraftTypeList(handicraftTypeList);

    // Expecting RuntimeException when attempting to add an already existing handicraft type

    assertThrows(RuntimeException.class, () ->
instructorService.addHandicraftTypeToInstructor(handicraftType, instructor));

    // Verify that repository.save() is not called

    verify(instructorRepository, never()).save(any(Instructor.class));

}

```



### Eğitmen mevcutken ID'si ile döndüren metodların testi:

```
@Test

public void testGetInstructorById_whenExists() {

    // Create a mock instructor and its ID

    UUID instructorId = java.util.UUID.randomUUID();

    Instructor mockInstructor = new Instructor("username", "surname", "name", "surname",
"123456789", "address", 100F, 150F);

    mockInstructor.setId(instructorId);

    // Stub the findById method of the repository to return the mock instructor

    when(instructorRepository.findById(instructorId)).thenReturn(Optional.of(mockInstructor));

    // Call the service method

    Instructor retrievedInstructor = instructorService.getInstructorById(instructorId);

    // Verify that the repository's findById method was called once with the correct ID

    verify(instructorRepository, times(1)).findById(instructorId);

    // Check that the returned instructor is not null and has the correct ID

    assertNotNull(retrievedInstructor);

    assertEquals(instructorId, retrievedInstructor.getId());

}
```

### c. Emir Çağrı Aykın

### Olmayan bir eğitmenin girilen ID ile çağırılması durumu testi:

```
@Test

public void testGetInstructorById_whenNotExists() {

    // Create a non-existent ID

    UUID nonExistentId = java.util.UUID.randomUUID();

    // Stub the findById method of the repository to return an empty Optional

    when(instructorRepository.findById(nonExistentId)).thenReturn(Optional.empty());

    // Call the service method and expect a RuntimeException

    assertThrows(RuntimeException.class, () -> instructorService.getInstructorById(nonExistentId));

}
```

```
// Verify that the repository's findById method was called once with the correct ID
verify(instructorRepository, times(1)).findById(nonExistentId);
}
```

**Sisteme yeni bir kursiyerin başarılı şekilde eklenmesi durumunun testi:**

```
@Test
public void testAddApplicant_whenCorrect() {
    // Create a mock request
    CreateApplicantRequest request = new CreateApplicantRequest("username", "surname",
"name", "surname", "123456789", "address");

    // Mock the repository save method
    when(applicantRepository.save(any(Applicant.class))).thenReturn(new Applicant());

    // Call the service method
    Applicant savedApplicant = applicantService.addApplicant(request);

    // Verify that repository.save() is called once
    verify(applicantRepository, times(1)).save(any(Applicant.class));

    // Check that the saved applicant is not null
    assertNotNull(savedApplicant);
}
```

**d. Esmâ Nur Ekmekci**

**Sistemdeki kursiyerin başarılı şekilde güncellenmesi durumunun testi:**

```
@Test
public void testUpdateApplicant_whenCorrect() {
    // Create a mock request
    UpdateApplicantRequest request = new UpdateApplicantRequest("username", "name",
"surname", "test@gmail.com", "123456789", "address", "email");

    // Mock the repository findById method
    UUID applicantId = UUID.randomUUID();

    Applicant mockApplicant = new Applicant();
```

```

when(applicantRepository.findById(applicantId)).thenReturn(java.util.Optional.of(mockApplicant));

// Mock the repository save method

when(applicantRepository.save(any(Applicant.class))).thenReturn(new Applicant());

// Call the service method

Applicant updatedApplicant = applicantService.updateApplicant(request, applicantId);

// Verify that repository.findById() is called once with the correct ID

verify(applicantRepository, times(1)).findById(applicantId);

// Verify that repository.save() is called once

verify(applicantRepository, times(1)).save(any(Applicant.class));

// Check that the updated applicant is not null

assertNotNull(updatedApplicant);

}

```

#### **Kursiyerin girilen ID ile başarılı şekilde elde edildiği durumun testi:**

```

@Test

public void testGetApplicantById_whenExists() {

    // Create a mock applicant and its ID

    UUID applicantId = UUID.randomUUID();

    Applicant mockApplicant = new Applicant();

    mockApplicant.setId(applicantId);

    // Stub the findById method of the repository to return the mock applicant

when(applicantRepository.findById(applicantId)).thenReturn(java.util.Optional.of(mockApplicant));

// Call the service method

Applicant retrievedApplicant = applicantService.getApplicantById(applicantId);

// Verify that the repository's findById method was called once with the correct ID

verify(applicantRepository, times(1)).findById(applicantId);

// Check that the returned applicant is not null and has the correct ID

```

```
assertNotNull(retrievedApplicant);

assertEquals(applicantId, retrievedApplicant.getId());

}
```

#### **e. Mehmet Keçeci**

**Hiçbir kursiyere ait olmayan ID girilmesi durumunun testi:**

```
@Test

public void testGetApplicantById_whenNotExists() {

    // Create a non-existent ID

    UUID nonExistentId = UUID.randomUUID();

    // Stub the findById method of the repository to return an empty Optional

    when(applicantRepository.findById(nonExistentId)).thenReturn(java.util.Optional.empty());

    // Call the service method and expect a RuntimeException

    assertThrows(RuntimeException.class, () -> applicantService.getApplicantById(nonExistentId));

    // Verify that the repository's findById method was called once with the correct ID

    verify(applicantRepository, times(1)).findById(nonExistentId);

}
```

**Kursiyerin zaten kayıtlı olduğu kursa tekrar eklenmeye çalışılması durumunun testi:**

```
@Test

public void testAddCourseToApplicant_whenAlreadyAttending() {

    // Create a mock applicant and course

    Applicant applicant = new Applicant();

    Course course = new Course();

    course.setDays(List.of(DayOfWeek.MONDAY, DayOfWeek.WEDNESDAY)); // Assume the course
    is on Monday and Wednesday

    // Stub the repository methods to simulate the applicant is already attending the course

    when(applicantParticipationRepository.existsByApplicantAndCourse(applicant,
    course)).thenReturn(true);

    // Call the service method and expect a RuntimeException
```

```

    assertThrows(RuntimeException.class, () -> applicantService.addCourseToApplicant(applicant,
course));

    // Verify that the repository methods are not called

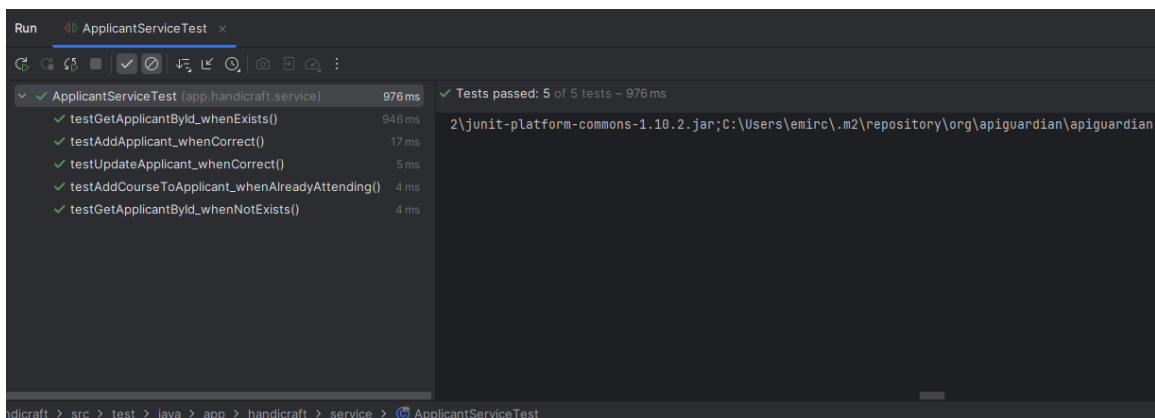
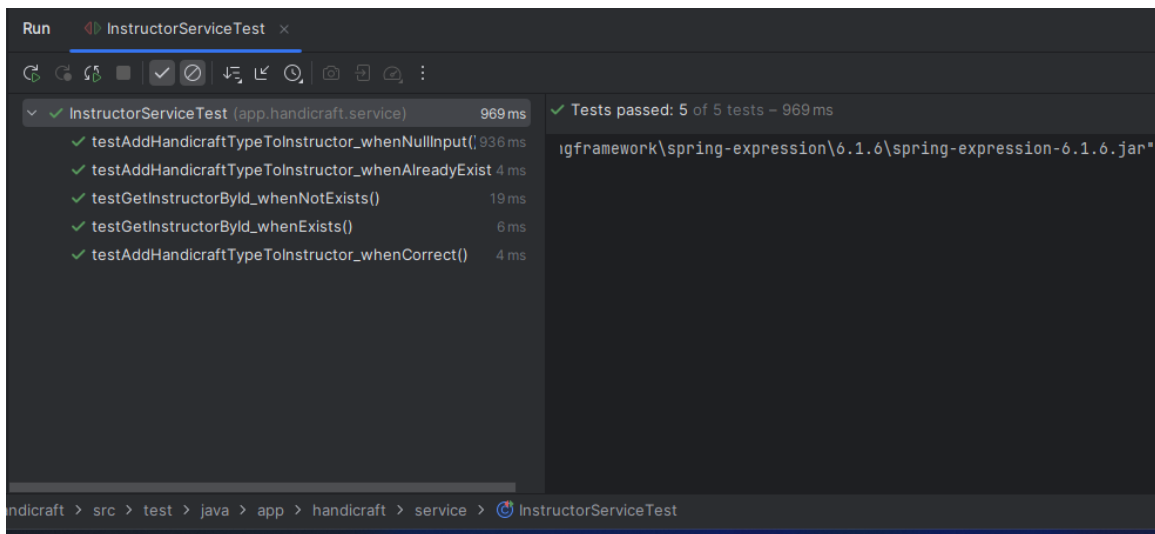
    verify(applicantRepository, never()).save(any(Applicant.class));

    verify(applicantParticipationRepository, never()).save(any(ApplicantParticipation.class));

}


```

## Test Sonuçları



## 9.Uygulama Görüntüleri

### a.Uygulama-Giriş:



Anasayfa Hakkımızda Kurslar İşlemler Kullanıcı Bilgileri Kurs Satın Al


Email

Email Adresinizi Girin


Şifre

Şifrenizi Girin

Giriş Yap



### b.Uygulama-Hakkımızda



Anasayfa Hakkımızda Kurslar İşlemler Kullanıcı Bilgileri Kurs Satın Al

## EMEEK

Emeek olarak, çeşitli el sanatları kursları ile hayalinizdeki becerilere ulaşmanıza yardımcı oluyoruz. Yeni başlayanlardan deneyimli sanatçılara kadar herkese hitap eden geniş bir kurs yelpazesi sunuyoruz.

#### Neden Bizi Seçmelisiniz?

- Geniş Kurs Yelpazesi:** Ahşap Boyama, Kumaş Boyama, Vitray, Tahta Oymacılık, Rölyef ve daha fazlasını içeren geniş bir yelpazede kurslar sunuyoruz.
- Deneyimli Eğitmenler:** Alanında uzman ve tutkulu eğitmenlerimizle becerilerinizi geliştirmenize ve yeni teknikler öğrenmenize yardımcı oluyoruz.
- Esnek Programlar:** Hafta içi ve hafta sonu kurslarımızla size en uygun zamanı seçme imkanı veriyoruz.

#### İletişim


05355555555 numaralı telefondan bize ulaşarak da bilgi alabilirsiniz.

#### Adres


Davutpaşa Kampüsü

El Sanatları Kursu ile hayalinizdeki becerilere ulaşın!

localhost:3000




El Sanatları Kursu ile hayalinizdeki becerilere ulaşın!



### Yazılım Ekibimiz

<b>Esmâ Nur Ekmekci</b> Frontend Developer Öğrenci No: 20011620	<b>Anıl Kutay Uçan</b> Frontend Developer Öğrenci No: 123-456-7890	<b>Mehmet Keçeci</b> Backend Developer Öğrenci No: 123-456-7890	<b>Emir Çağrı Aykın</b> Backend Developer Öğrenci No: 123-456-7890	<b>Efe Girgin</b> Database Manager Öğrenci No: 123-456-7890
-----------------------------------------------------------------------	--------------------------------------------------------------------------	-----------------------------------------------------------------------	--------------------------------------------------------------------------	-------------------------------------------------------------------


## c.Uygulama-Kurslar

 <a href="#">Anasayfa</a> <a href="#">Hakkımızda</a> <a href="#">Kurslar</a> <a href="#">İşlemler</a> <a href="#">Kullanıcı Bilgileri</a> <a href="#">Kurs Satın Al</a>		
El Sanatları Paketi		
<b>Kurs Adı</b> Ahşap Boyama	<b>Eğitmen</b> Meryem	<b>Zaman</b> Pazartesi 12.00
Parmak Boyama	Ali	Çarşamba 13.00
Kapasite: 9/25 Fiyat: 1500 TL		
Resim Sanatı Paketi		
<b>Kurs Adı</b> Karakalem Çizimi	<b>Eğitmen</b> Ali	<b>Zaman</b> Cumartesi 13.00
Yağlı Boya	Yeşim	Çarşamba 13.00
Kapasite: 12/30 Fiyat: 2000 TL		

## d.Uygulama-işlemler

 <a href="#">Anasayfa</a> <a href="#">Hakkımızda</a> <a href="#">Kurslar</a> <a href="#">İşlemler</a> <a href="#">Kullanıcı Bilgileri</a> <a href="#">Kurs Satın Al</a>	
Personel İşlemleri	
Ders Ekle	Kurs Paketi Oluştur
Kursiyer Ekle	Eğitmen Ekle

## e.Uygulama-Ders Ekleme



[Anasayfa](#) [Hakkımızda](#) [Kurslar](#) [İşlemler](#) [Kullanıcı Bilgileri](#) [Kurs Satın Al](#)

### Ders Ekleme Sayfası

Ders Türü:  
Dekoratif El Ürünleri Yapımı

Ders Adı:  
Deneme


Gün:  
Pazartesi

Eğitmen:  
Mehmet Keçeci

Ekle

Eklenen Dersler

## f.Uygulama-Kurs Ekle



[Anasayfa](#) [Hakkımızda](#) [Kurslar](#) [İşlemler](#) [Kullanıcı Bilgileri](#) [Kurs Satın Al](#)

### Kurs Paketi Oluştur

Paket Adı  
Deneme

Kapsite  
23

Seçilen Kurslar:

Kurs Türü  
Tekstil Tasarımı

Kurs Seçin  
Keçe Aksesuarları Yapımı

Kursu Ekle

Oluştur



## g.Uygulama-Kursiyer Ekle

[Anasayfa](#) [Hakkımızda](#) [Kurslar](#) [İşlemler](#) [Kullanıcı Bilgileri](#) [Kurs Satın Al](#)

### Kursiyer Ekle

Ad

Cep Telefonu

Adres

Email

Ekle

## h.Uygulama-Eğitmen Ekle

### Eğitmen Ekle

Ad

Cep Telefonu

Adres

Email

Çalışma Saatleri

Verdiği Dersler

Hafta İçi Ücreti

Hafta Sonu Ücreti

Ekle

EEMK

AnasayfaHakkımızdaKurşurİçerimlerKullanıcı BilgileriKurs Satın Al

Müşteri Bilgileri

NameAra

Name: John Doe

User Name: deneme

Phone Number: +1234567890

Address: 123 Main St, Cityville

Email: john@example.com

Taken Courses:

Introduction to Programming

Web Development Basics

Courses Information:

Course: Introduction to Programming, Date: 2024-06-15,  
Payment Amount: \$100

Course: Web Development Basics, Date: 2024-07-10,  
Payment Amount: \$150

Antrenör Bilgileri

NameAra

Name: Alice Trainer

Phone Number: +1234567890

House Number: 789

Address: 789 Oak St, Villagetown

Email: alice@example.com

Working Hours:

Monday: 09:00 - 17:00

Tuesday: 10:00 - 18:00

Wednesday: 09:30 - 16:30

Courses Information:

Course: Yoga Basics, Details: Learn foundational yoga poses  
and breathing techniques. Payment Amount: \$50

Course: Pilates for Beginners, Details: Introduction to core  
strengthening and flexibility exercises. Payment Amount: \$60

EMEEK

AnasayfaHakkımızdaKurslarİçerilerKullanıcı BilgileriKurs Satın Al

Kurs Bilgisi Filtrele

BütçeZaman Seçiniz

Ahşap Boyama  
Kumaş Boyama

Filtrele

Müşteri Bilgisi Filtrele

NameAra

İsim: Basket Weaving Workshop

Kurs Ücreti: 200

Kalan Kontenjan: 10

Kurs Zamanı: MONDAY, WEDNESDAY

El İşleri:

El İşi Türü: Basket Weaving

Eğitmen: John Doe

Gün: MONDAY

Ücret: 50

Name: John Doe

User Name: deneme

Phone Number: +1234567890

Address: 123 Main St, Cityville

Email: john@example.com

Taken Courses:

Introduction to Programming

Web Development Basics

Courses Information:

Course Introduction to Programming, Date: 2024-05-15