T.C
KOCAELİ SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR / YAZILIM MÜHENDİSLİĞİ

PROJE KONUSU:
PROGRAM DERS İLİŞKİ MATRİSİ

ÖĞRENCİ ADI:
ÖĞRENCİ NUMARASI:
Sema Su YILMAZ - 220502016
Zehra YARDIMCI - 220502038
Senem ADALAN - 220502045
Şevval ÖZEREN - 220501028
Hilal AYDIN - 220501030

DERS SORUMLUSU:
PROF. DR. TARIK DURU / DR. ÖĞR. ÜYESİ ELİF PINAR HACIBEYOĞLU

TARİH:
30 ARALIK 2024

İÇİNDEKİLER

1. GİRİŞ

1.1 Projenin Amacı

Bu projenin amacı bir eğitim programının ders içeriklerini, değerlendirme kriterlerini ve öğrenci başarılarını sistematik bir şekilde analiz ederek öğrencilerin ders ve program çıktıları üzerindeki başarı oranlarını hesaplamaktır. Projede yazılım geliştirme kısmında Python programlama dili kullanılmış, tablo verileri SQL Server veri tabanında saklanmış ve Excel tablolarıyla tablolar görselleştirilmiştir.

2. GEREKSİNİM ANALİZİ

2.1 Fonksiyonel Gereksinimler

Ders Çıktıları ve Değerlendirme Kriterleri İlişki Matrisi Oluşturulması (İG-1)

- Sistem, ders çıktıları ile değerlendirme kriterleri arasındaki ilişkiyi gösterecek bir matris oluşturmalıdır.
- Matris, her bir ders çıktısı ile ilgili değerlendirme kriterlerinin etkisini 0-1 aralığında ifade etmelidir.
- Değerlendirme kriterlerinin ağırlıkları matrise entegre edilecektir.

Program Çıktıları ve Ders Çıktıları İlişki Matrisi (İG-2)

- Sistem, program çıktıları ile ders çıktıları arasındaki ilişkiyi gösteren bir matris oluşturmalıdır.
- Matris, her dersin program çıktılarıyla ne kadar ilişkili olduğunu 0-1 aralığında hesaplamalıdır.
- Her ders çıktısı için toplam ilişkinin değeri hesaplanarak program hedeflerine katkı oranları belirlenmelidir.

Öğrenci Notları Tablosu (İG-3)

- Her öğrencinin aldığı değerlendirme notlarını (ödev, vize, final) içeren bir tablo oluşturmalıdır.
- Öğrenci numarası ve ilgili değerlendirme kriterlerine göre alınan notlar doğru şekilde yerleştirilmelidir.

Ağırlıklı Değerlendirme Tablosu (İG-4)

- Değerlendirme kriterlerinin ağırlıkları ile ders çıktıları arasındaki ilişki hesaplanmalı ve bu sonuçlar bir ağırlıklı değerlendirme tablosunda sunulmalıdır.
- Tablo, her değerlendirme kriterinin ağırlığı ile ilgili ders çıktılarının ilişkisini göstermelidir.

Ders Çıktıları Başarı Oranı Hesaplama (İG-5)

- Her öğrenci için aldığı notlar ile ders çıktılarının başarı oranı hesaplanmalıdır.
- Başarı oranları, ağırlıklı değerlendirme tablosu ve öğrenci notlarının çarpımıyla oluşturulmalıdır.

Program Çıktıları Başarı Oranı Hesaplama (İG-6)

- Her öğrenci için program çıktıları başarı oranı hesaplanmalıdır.
- Bu hesaplama ders çıktılarının başarı oranlarının program çıktıları ile ilişkilendirilmesiyle yapılmalıdır.

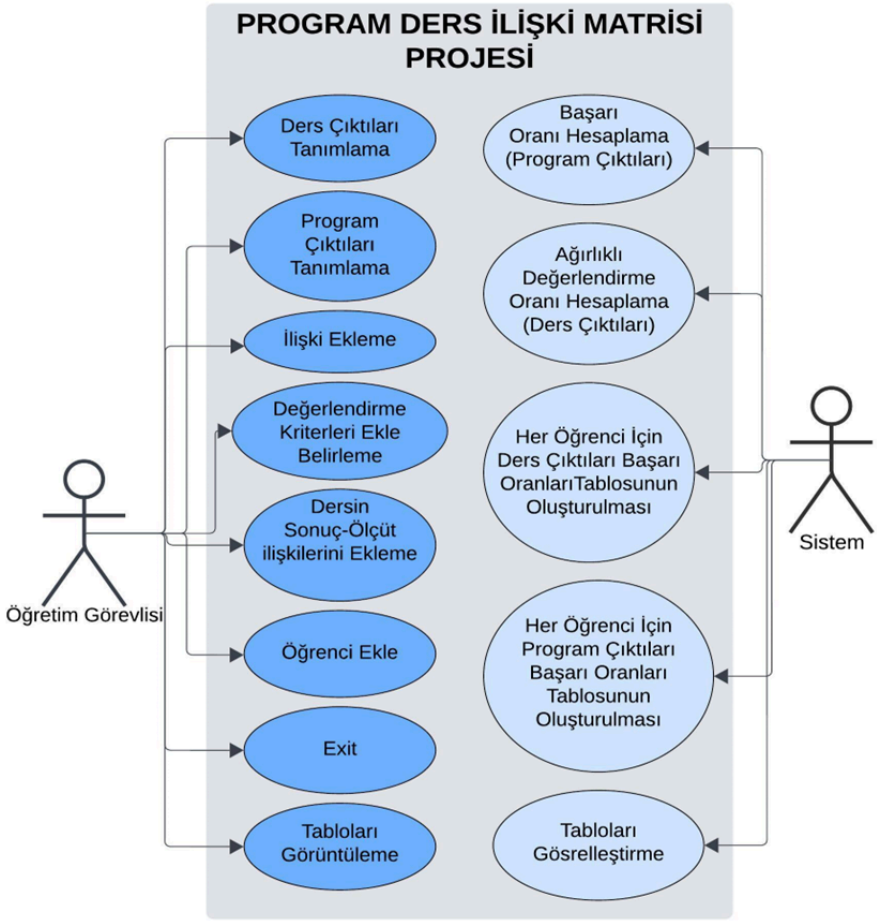
Tabloların Excel Üzerinde Görselleştirilmesi (İG-7)

- Sistemde yapılan tüm hesaplamalar ve analizler, kullanıcı dostu bir şekilde Excel dosyasına aktarılmalıdır.
- Excel dosyasında, her bir öğrenci için ders çıktıları ve program çıktıları başarı oranlarını gösteren tablolar oluşturulmalıdır.

Veri ve Hesaplama Doğruluğu (İG-8)

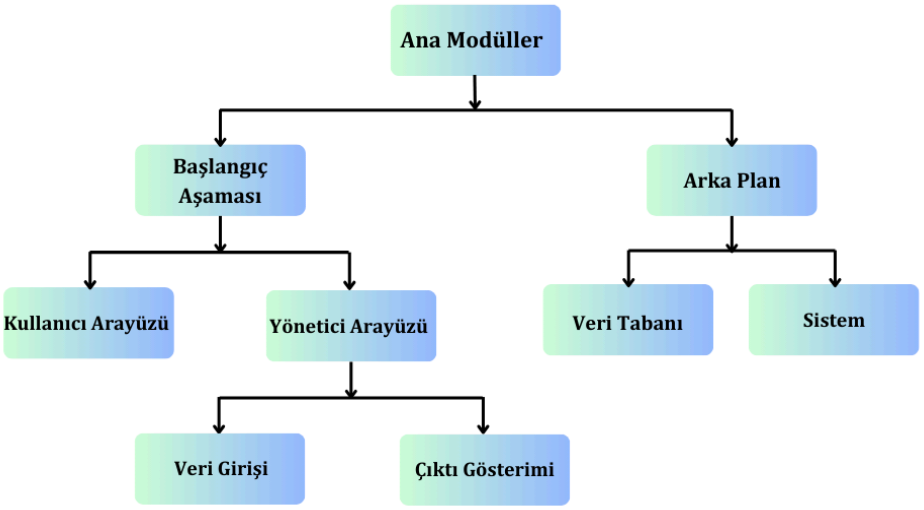
- Tüm veriler doğru bir şekilde işlenmeli ve hesaplamalar doğru sonuçlar üretmelidir.
- Sistemde hata oranı düşük olmalı ve kullanıcılar, verileri doğru bir şekilde görüntülemelidir.

2.2 Use-Case Diyagramı



3. TASARIM
3.1 Mimari Tasarım
- Veri Tabanı Katmanı
  - Veri Tabanı Bağlantısı: ODBC üzerinden yapılır ve her işlem için ayrı bir bağlantı kullanılır.
  - Veri ekleme, veri çekme, veri tabanı ilişkileri, ilişkisel tablolarda tutulur ve ağırlıklı değerlendirme hesaplamaları bu verilere dayanır.
  - Excel Dosyaları: OpenPyXL kütüphanesi ile veriler Excel dosyalarına yazılır ve düzenlenir.
- İşlem Katmanı
  - Veri tabanındaki veriler işlenir ve hesaplamalar yapılır.
- Kullanıcı Katmanı
  - Kullanıcı, konsol üzerinden veri girişi yapar ve çıktı alır.

MODÜL DİYAGRAMI



3.2 Kullanılacak Teknolojiler
- Bu projede Python programlama dili kullanılmıştır.
- Pyodbc kütüphanesi, Open Database Connectivity (ODBC) protokolünü kullanarak veri tabanlarına bağlanmak için kullanılan bir kütüphanedir. Bu kütüphane sayesinde Python uygulamalarında SQL tabanlı Microsoft SQL Server ile kolayca iletişim kurulabilir.
- Openpyxl kütüphanesi kullanılmıştır. Microsoft Excel dosyalarıyla çalışmak için kullanılan bir kütüphanedir. Bu kütüphane sayesinde Excel dosyaları okunabilir, oluşturulabilir ve düzenlenebilir.

3.3 Veri Tabanı Tasarımı

RelationMatrix Veri Tabanı
- SQL veri tabanında RelationMatrix adlı bir veri tabanı oluşturuldu.
- CourseOutcomes, ProgramOutcomes, ProgramCourseRelations, EvaluationCriteria, CourseEvaluationRelations, Students, Table3 ve Table4 olmak üzere 8 tablo oluşturulmuştur.

CourseOutcomes (Ders Çıktıları) Tablosu
- Ders çıktılarının metinsel bir açıklamasını içerir.
- Her ders çıktısına bir id atanır, bu id otomatik olarak artar ve birincil anahtar olarak kullanılır.

ProgramOutcomes (Program Çıktıları) Tablosu
- Program çıktılarının metinsel bir açıklamasını içerir.
- Her program çıktısına bir id atanır, bu id otomatik olarak artar ve birincil anahtar olarak kullanılır.

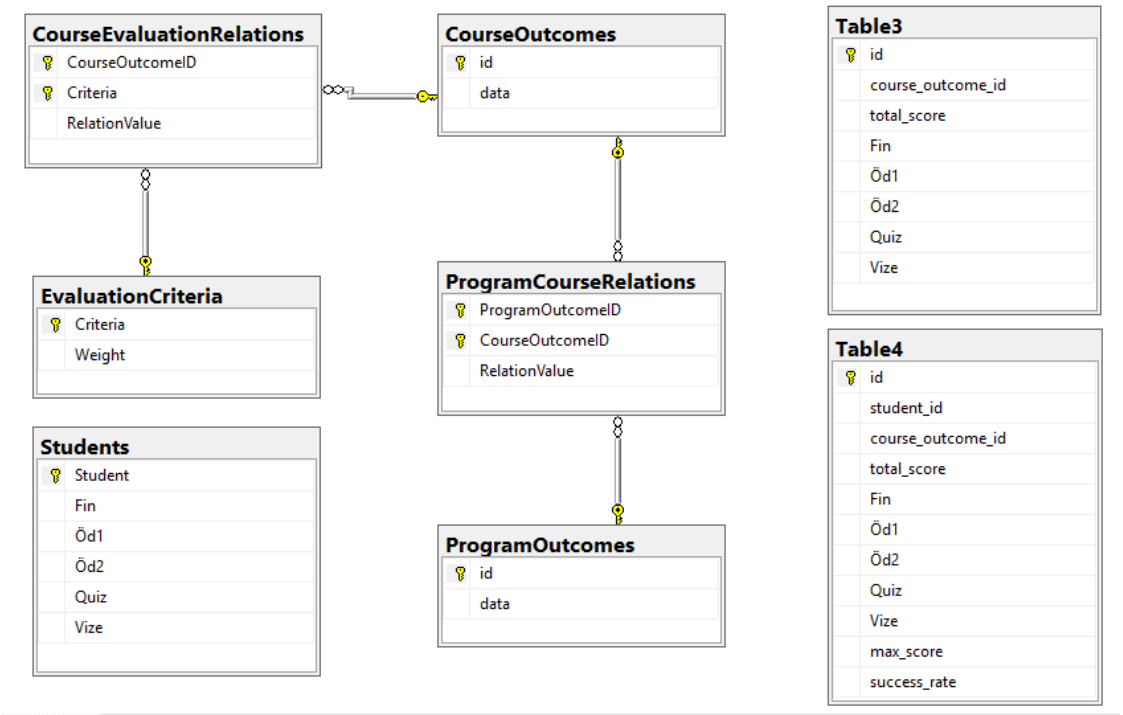
ProgramCourseRelations (Program ve Ders Çıktıları İlişkisi) Tablosu
- Program çıktıları ile ders çıktıları arasındaki ilişkiyi tanımlar.
- Program çıktısının kimliği, ProgramOutcomes.id ile ilişkilidir.
- Ders çıktısının kimliği, CourseOutcomes.id ile ilişkilidir.
- RelationValue, 0 ile 1 arasında olmalıdır.

EvaluationCriteria (Değerlendirme Kriterleri) Tablosu
- Ders veya program çıktılarının değerlendirilmesi için kullanılan kriterleri tanımlar.
- Kriterin adı ve kriterin ağırlığını tutar.

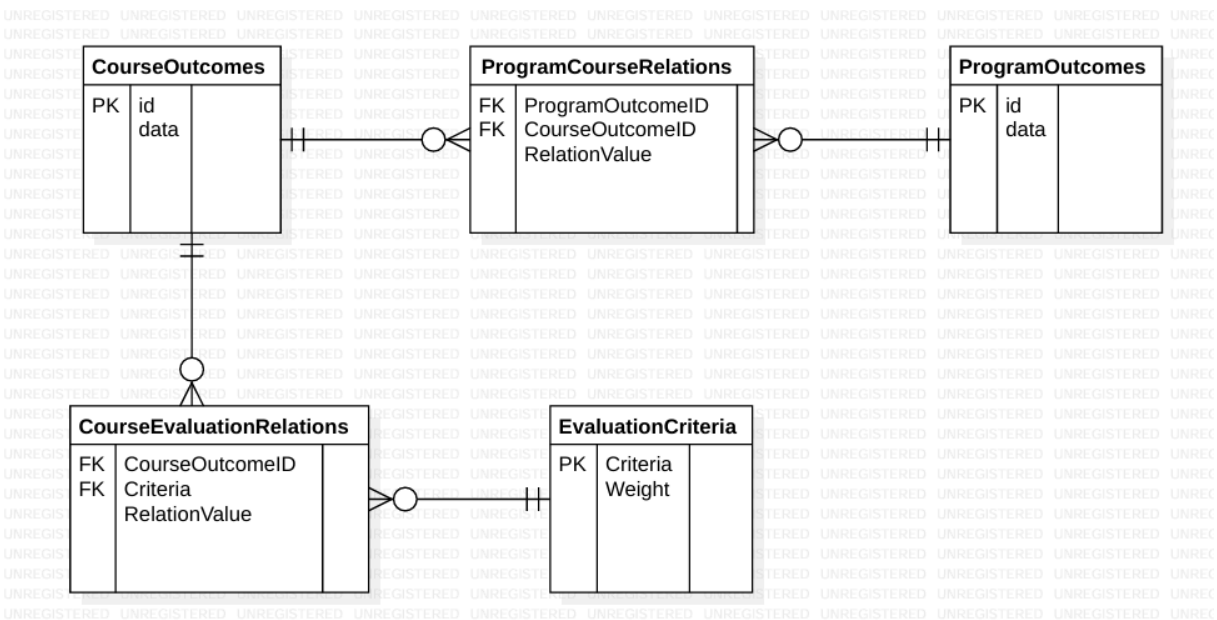
CourseEvaluationRelations (Ders Çıktıları ve Değerlendirme Kriterleri İlişkisi) Tablosu
- Ders çıktıları ile değerlendirme kriterleri arasındaki ilişkiyi tanımlar.
- Ders çıktısının kimliği, CourseOutcomes.id ile ilişkilidir.
- Değerlendirme kriterinin adı, EvaluationCriteria.Criteria ile ilişkilidir.
- RelationValue, ilişkinin değerini tutar.

Students Tablosu
- Öğrenci bilgilerini içerir.
- Öğrenci numaraları ve her bir bölümden (Fin, Quiz, Vize) aldığı notlar tutulmaktadır

Table3 ve Table4 Tablosu
- table3.xlsx ve table4.xlsx dosyalarındaki tablolar tutulmaktadır.



Veri Tabanı Tasarımının Diyagramlar ile Gösterimi



ER Diyagramı

4. UYGULAMA
4.1 Kodlanan Bileşenlerin Açıklamaları

- **get_connection()**: Bu fonksiyon, bir SQL Server veri tabanına bağlanır ve ODBC sürücüsünü kullanarak bir bağlantı nesnesi oluşturur.
- **check_database()**: Bu fonksiyon, RelationMatrix adlı veri tabanının olup olmadığını kontrol eder yoksa oluşturur.
- **check_tables()**: Bu fonksiyon, RelationMatrix veri tabanında gerekli tabloların mevcut olup olmadığını kontrol eder ve eksik olanları oluşturur.
- **insert_data_into_table()**: Bu fonksiyon, CourseOutcomes ya da ProgramOutcomes tablolarına veri ekler.
- **fetch_relations()**, **fetch_evaluation_relations()**, **fetch_table_data()**, **fetch_evaluation_data()**, **fetch_student_data()**, **fetch_success_rate()**: Bu fetch fonksiyonlarıyla ilgili tablolardaki veriler çekilir. Uygulamanın ihtiyaç duyduğu ilişkiler ve bilgiler işlenir.
- **create_table1()**: Bu fonksiyon, Table1 adlı bir Excel çalışma sayfası oluşturur ve program çıktıları ile ders çıktıları arasındaki ilişkiyi gösteren bir tabloyu yapılandırır. Program çıktıları id'leri satırlara, ders çıktılarının id'leri sütunlara eklenir. Bunların string karşılıkları comment olarak eklenir. Kullanıcının girdiği ilişkiler ilgili hücrelere yazılır.
- **create_table2()**: Bu fonksiyon, Table2 adlı bir Excel çalışma sayfası oluşturur ve CourseOutcomes ile EvaluationCriteria arasındaki ilişkileri gösteren bir tabloyu yapılandırır. Ders çıktısı id'leri satırlara, değerlendirme kriterleri ve bunların ağırlıkları sütunlara yazılır. Kullanıcının girdiği ilişkiler ilgili sütunlara yazılır.
- **create_table3()**: Bu fonksiyon, Table3 adlı bir Excel çalışma sayfası oluşturur ve CourseOutcomes ile EvaluationCriteria arasındaki ağırlıklı değerlendirme ilişkilerini gösteren bir tabloyu yapılandırır. Satırlarda ders çıktı id'leri, sütunlarda değerlendirme kriterleri bulunur. Her ağırlıklı değeri şu formülle hesaplanarak ilgili hücreye yazılır: (ilişki değeri x ağırlık) / 100.
- **save_table3_to_database()**: Bu fonksiyon oluşturulan Table3'ü veri tabanına kaydeder.
- **create_notes()**: Bu fonksiyon, Students tablosundaki verilerle bir notlar tablosu oluşturur ve Excel dosyasına kaydeder. Her öğrencinin değerlendirme kriterlerinden aldığı notlar ve ortalaması ilgili hücrelere eklenir.
- **create_table4()**: Bu fonksiyon, Students, CourseOutcomes ve CourseEvaluationRelations tablolarındaki veriler ile her öğrencinin ders çıktılarına dayalı ağırlıklı başarı notlarını hesaplar ve table4.xlsx adlı Excel dosyasında her öğrenci için ayrı ayrı raporlar. Table3'teki değerlerin öğrenci notlarıyla çarpılmasıyla oluşturulur. Her öğrencinin başarı oranı şu formülle hesaplanır: (toplam puan / max puan) x 100.
- **save_table4_to_database()**: Bu fonksiyon oluşturulan Table4'ü veri tabanına kaydeder.
- **create_table5()**: Bu fonksiyon her öğrencinin başarı oranları ve program çıktılarına olan katkılarıyla ilgili bir Excel raporu oluşturur.

- **get_input_and_insert_relations()**, **get_input_and_insert_evaluation_relations()**, **get_input_and_insert_table()**, **get_evaluation_criteria_and_insert()**, **insert_relation_value()**, **insert_evaluation_relation_value()**: Bu fonksiyonlar kullanıcıdan alınan girdileri ilgili tablolara ekler.
- **create_students_table()**, **add_student()**: Bu iki fonksiyon öğrenci verilerini veri tabanına eklemek ve Students tablosunu oluşturmak için kullanılır.
- **menu()**: Bu fonksiyon kullanıcıya belirli işlemleri seçme fırsatı sunan bir metin tabanlı menü sağlar. Kullanıcı menü seçeneklerinden birini girdiğinde fonksiyon ilgili işlem için doğru işlevi çağırır.
  1. Program çıktısı ekler.
  2. Ders çıktısı ekler.
  3. Program çıktısı - ders çıktısı ilişkisi ekler.
  4. Değerlendirme kriteri ekler.
  5. Ders çıktısı - değerlendirme kriteri ilişkisi ekler.
  6. Öğrenci ekler.
  7. Çıkış yapar.

4.2 Görev Dağılımı
Tüm aşamalar birlikte gerçekleştirilmiştir.

4.3 Karşılaşılan Zorluklar ve Çözüm Yöntemleri
Geliştirme sürecinde herhangi bir problemle karşılaşılmamıştır.

4.4 Proje İsterlerine Göre Eksik Yönler
Projede gerçeklenmesi beklenen görevlerin tümü tamamlanmıştır.

5. TEST VE DOĞRULAMA
5.1 Yazılımın Test Süreci

Bu birim testi, veri tabanı bağlantısının, tablo oluşturma işlemlerinin ve veri ekleme işlevlerinin doğru bir şekilde çalışıp çalışmadığını kontrol etmek için kullanılır. İlk olarak veri tabanı bağlantısı get_connection fonksiyonu kullanılarak oluşturulur. Ardından check_database ve check_tables fonksiyonları çalıştırılarak veri tabanının ve tabloların varlığı doğrulanır. Daha sonra test amaçlı bir veri, insert_data_into_table fonksiyonu ile tabloya eklenir ve bu veri fetch_relations fonksiyonu kullanılarak kontrol edilir. Her bir adım assert ifadeleri ile beklenen sonuçlarla karşılaştırılarak test edilir. Eğer bir test başarısız olursa bir hata mesajı görüntülenir. Tüm bu işlemler test_functional_integrity() fonksiyonu çağrılarak bir bütün halinde test edilir. Bu sayede projenin temel altyapısının beklenen şekilde çalışıp çalışmadığı doğrulanır.

```python
def test_functional_integrity():
    """Veritabanı bağlantısı, tablo oluşturma ve veri ekleme işlevlerini test eder."""
    # Veritabanı bağlantısını test et
    conn = get_connection()
    assert conn is not None, "Veritabanı bağlantısı kurulamadı."
    conn.close()

    # Veritabanı oluşturmayı test et
    try:
        check_database()
    except Exception as e:
        pytest.fail(f"Veritabanı oluşturulamadı: {e}")

    # Tablo oluşturmayı test et
    try:
        check_tables()
    except Exception as e:
        pytest.fail(f"Tablolar oluşturulamadı: {e}")

    # Veri eklemeyi test et
    try:
        insert_data_into_table('CourseOutcomes', 'Test Course Outcome')
        data = fetch_relations()
        assert len(data) >= 0, "Veriler eklenemedi."
    except Exception as e:
        pytest.fail(f"Veri ekleme başarısız: {e}")
```

Tablo 1, Tablo 2 ve Tablo 3'ün mantıksal hesaplama işlevlerini doğrulamak için testler içerir. Tablo 1'de, program çıktıları ile ders çıktıları arasındaki ilişkilerin 0 ile 1 arasında değer alıp almadığı kontrol edilir. İlişki matrisindeki her bir değerin bu aralığa uygunluğu doğrulanarak matrisin geçerliliği test edilir. Tablo 2'de, ders çıktıları ile değerlendirme kriterleri arasındaki ilişkilerin yalnızca 0 veya 1 değerine sahip olup olmadığı kontrol edilir. Ayrıca değerlendirme kriterlerinin ağırlıkları toplanarak toplamın 100'ü geçip geçmediği doğrulanır. Tablo 3'te, değerlendirme kriterlerinin ağırlıkları ile ilişki değerlerinin çarpımı sonucu oluşan ağırlıklı değerlere bakılır ve bu değerlerin toplamının 0 ile 1 arasında olması gerektiği test edilir. Her bir test, ilgili verilerin tutarlılığını ve doğru hesaplandığını kontrol ederek sistemin mantıksal bütünlüğünü sağlar. Bu testler, test_logical_integrity fonksiyonu altında çağrılarak sırasıyla çalıştırılır.

```python
def test_logical_integrity():
    """Tüm mantıksal hesaplama işlevlerini test eder."""

    def test_table1_logic():
        """Program çıktıları ve ders çıktıları ilişki matrisini test eder."""
        relations = fetch_relations()
        assert relations, "Table 1 için ilişki verisi bulunamadı."
        for program_outcome_id, course_outcome_id, relation_value in relations:
            assert 0 <= relation_value <= 1, f"İlişki değeri 0 ile 1 arasında olmalıdır. Ancak {relation_value} bulundu."

    def test_table2_logic():
        """Ders çıktıları ve değerlendirme kriterleri ilişki matrisini test eder."""
        evaluation_relations = fetch_evaluation_relations()
        assert evaluation_relations, "Table 2 için ilişki verisi bulunamadı."
        for course_outcome_id, criteria, relation_value in evaluation_relations:
            assert relation_value in [0, 1], f"İlişki değeri yalnızca 0 veya 1 olabilir. Ancak {relation_value} bulundu."
        evaluation_data = fetch_evaluation_data()
        assert evaluation_data, "Değerlendirme kriterleri bulunamadı."
        total_weight = sum(weight for _, weight in evaluation_data)
        assert total_weight <= 100, "Değerlendirme kriterlerinin toplam ağırlığı 100'ü geçmemelidir."
```

```python
def test_table3_logic():
    """Ağırlıklı değerlendirme tablosunu test eder."""
    evaluation_data = fetch_evaluation_data()
    criteria_weights = {criteria: weight for criteria, weight in evaluation_data}
    evaluation_relations = fetch_evaluation_relations()
    course_outcomes = fetch_table_data("CourseOutcomes")

    for course_outcome_id, _ in course_outcomes:
        total_weighted_value = 0
        for criteria, weight in criteria_weights.items():
            relation_value = next((rel[2] for rel in evaluation_relations if rel[0] == course_outcome_id and rel[1] == criteria), 0)
            total_weighted_value += (relation_value * weight) / 100
        assert 0 <= total_weighted_value <= 1, f"Toplam ağırlıklı değer 0 ile 1 arasında olmalıdır.
         Ancak {total_weighted_value} bulundu."

def test_table4_logic():
    """Ders çıktıları başarı oranlarını test eder."""
    success_rates = fetch_success_rate()
    assert success_rates, "Table 4 için başarı oranları bulunamadı."
    for student_id, success_rate in success_rates:
        assert 0 <= success_rate <= 100, f"Başarı oranı 0 ile 100 arasında olmalıdır. Ancak {success_rate} bulundu."
```

```python
def test_table5_logic():
    """Program çıktıları başarı oranlarını test eder."""
    relations = fetch_relations()
    success_rates = fetch_success_rate()
    assert success_rates, "Table 5 için başarı oranları bulunamadı."
    for program_outcome_id, course_outcome_id, relation_value in relations:
        related_success = [rate for student_id, rate in success_rates if student_id == course_outcome_id]
        if related_success:
            avg_success = sum(related_success) / len(related_success)
            expected_success = avg_success / relation_value if relation_value > 0 else 0
            assert 0 <= expected_success <= 100, f"Hesaplanan başarı oranı 0 ile 100 arasında olmalıdır.
            Ancak {expected_success} bulundu."
```

Excel tablolarının doğru oluşturulup oluşturulmadığı test edilir. Her bir tablo için belirtilen dosyanın oluşturulup oluşturulmadığı, başlıkların doğruluğu ve içerikte eksik veri olup olmadığı kontrol edilir. İlk olarak dosyanın varlığı doğrulanır. Daha sonra tablonun başlıklarının doğru şekilde oluşturulduğu ve içerik satırlarında eksik veri bulunmadığı test edilir. Testler, test_excel_content_integrity fonksiyonu ile çalıştırılır ve her tablo sırasıyla kontrol edilir. Testler tamamlandıktan sonra oluşturulan dosyalar silinerek geçici dosyalar temizlenir. Bu süreç Excel tablolarının eksiksiz ve doğru bir şekilde oluşturulduğunu garanti eder.

```python
def test_excel_content_integrity():
    """Excel tablolarının doğru oluşturulup oluşturulmadığını test eder."""
    tables = [
        (create_table1, "table1.xlsx", "Table 1", "Program Outcomes", "Course Outcomes"),
        (create_table2, "table2.xlsx", "Table 2", "Course Outcomes", None),
        (create_table3, "table3.xlsx", "Table 3", None, "Weighted Evaluation"),
        (create_table4, "table4.xlsx", "Table 4", None, "Student"),
        (create_table5, "table5.xlsx", "Table 5", None, None)
    ]

    for create_func, file_name, table_title, col1_title, col2_title in tables:
        try:
            create_func()
            assert os.path.exists(file_name), f"{file_name} dosyası oluşturulamadı."

            wb = load_workbook(file_name)
            sheet = wb.active

            if table_title:
                assert sheet['A1'].value == table_title, f"{table_title} başlığı hatalı."
            if col1_title:
                assert sheet['A2'].value == col1_title, f"{col1_title} başlığı eksik."
            if col2_title:
                assert col2_title in (sheet['C1'].value or ""), f"{col2_title} başlığı eksik."

            # Örnek içerik doğrulama
            for row in sheet.iter_rows(min_row=3, max_col=sheet.max_column, values_only=True):
                assert row[0] is not None, f"{file_name} içeriğinde eksik veri var."
        finally:
            if os.path.exists(file_name):
                os.remove(file_name)
```

Sistemin tüm işlevlerinin bir arada çalışması test edilir. test_full_integration fonksiyonu, sırasıyla test_functional_integrity, test_logical_integrity ve test_excel_content_integrity testlerini çağırır. Bu testler, sistemin fonksiyonel bütünlüğünü, mantıksal hesaplama işlevlerini ve Excel tablolarının doğruluğunu kontrol ederek sistemin bir bütün olarak düzgün çalıştığını doğrular.

5.2 Yazılımın Doğrulanması

- **Fonksiyonel Testler:** Veri tabanı bağlantısının doğru bir şekilde kurulup kurulmadığını, gerekli tabloların oluşturulup oluşturulmadığını ve veri ekleme işlemlerinin doğru çalışıp çalışmadığını kontrol eder. Bu testlerde bağlantı hataları, eksik tablolar veya veri ekleme sırasında oluşabilecek sorunlar belirlenir.
- **Mantıksal Testler:** Hesaplama fonksiyonlarının doğru çalıştığını kontrol eder. Ağırlıklı değerlendirme ve başarı oranı hesaplamalarının belirlenen kurallara uygun sonuçlar üretip üretmediği test edilir.
- **Sonuç Doğrulama Testleri:** Oluşturulan Excel tablolarının doğru içerik ve formatta olup olmadığını kontrol eder. Her bir tablo için başlıkların doğruluğu, içeriklerin eksiksizliği ve sütunların uygun şekilde oluşturulup oluşturulmadığı test edilir.

Bu testler, her bir işlemin doğru şekilde gerçekleştirildiğini doğrular. Her bir test, ilgili işlemleri başarıyla tamamladığında bir başarı mesajı yazdırır, aksi takdirde bir hata mesajı yazdırır. Bu şekilde yazılımın beklenen davranışlarını sergileyip sergilemediği test edilir.

6. GİTHUB BAĞLANTILARI

https://github.com/SemaSuYILMAZ/ProgramDersIliskiMatrisi

http://github.com/Zehrayardimci/Program-Ders-iliski-Matrisi

https://github.com/senemadalan/Iliski_Matrisi

https://github.com/sevvalozrn/PROGRAM-DERS-ILISKI-MATRISI

https://github.com/HilallAydinn