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

PROJE KONUSU: HAFTALIK DERS PROGRAMI UYGULAMASI

HAZIRLAYANLAR

220502021 NURDAN BULUT

220501012 ZEYNEP KEDİKLİ

220501026 ZEHRA KANDAZ

DERS SORUMLUSU:
DR. ÖĞR. ÜYESİ ELİF PINAR HACIBEYOĞLU

TARİH:22.03.2025

1. GİRİŞ

1.1 Projenin amacı

Projenin temel amacı, Mühendislik Fakültesi'ne ait bölümler için kullanıcı dostu, dinamik ve düzenlenebilir bir haftalık ders programı sistemi oluşturmaktır. Öğrenciler, öğretim üyeleri ve yöneticiler gibi farklı kullanıcı tiplerinin sistem üzerinde işlem yapabilmesi, bölüm ve ders tanımlamalarının kolaylıkla eklenip çıkarılabilmesi hedeflenmiştir. Proje, farklı bölümlere ait derslerin çakışmadan ve çeşitli kısıtlamalara uyarak programa yerleştirilmesini sağlamayı amaçlamaktadır. Aynı zamanda derslerin öğretim üyelerinin uygunluklarına göre planlanması ve derslik kapasitesine göre atanması da sistemin kritik fonksiyonlarındandır.

Bu sistem, ileride bir web servisine dönüştürülmeye uygun altyapı ile geliştirilmiştir ve sonraki aşamada tüm okulun ders programlarını kapsayacak şekilde genişletilebilir yapıda tasarlanmıştır.

1.2 Projede beklenenler

- Öğretim üyesi, öğrenci ve yönetici rollerinin sisteme tanımlanabilmesi
- Yeni bölüm ekleme ve çıkarma işlemlerinin yapılabilmesi
- Her bölüm için 8 yarıyıllık derslerin tanımlanabilmesi
- Derslerin sisteme eklenip çıkarılabilmesi ve her ders için gerekli bilgilerin (kod, saat, öğretim üyesi vb.) girilebilmesi
- Dersliklerin sisteme tanımlanabilmesi ve düzenlenebilmesi
- Günlük ve saatlik zaman dilimlerine uygun haftalık ders programı oluşturulabilmesi
- Ortak dersler ve öğretim üyeleri arasında çakışmaların önlenmesi
- Derslik kapasitesine göre atama yapılması
- Bazı dersler için zorunlu saatlerin belirtilebilmesi (İngilizce, Atatürk İlkeleri ve İnkılap Tarihi gibi)
- Excel çıktısı ile ders programının verilebilmesi
- Dinamik olarak her sınıfın ders programının görüntülenip düzenlenebilmesi

2. GEREKSİNİM ANALİZİ

2.1 Arayüz gereksinimleri

Uygulamanın, öğrenciler, öğretim üyeleri ve yöneticiler tarafından rahat kullanılabilmesi için sade bir tasarım tercih ettik. Menü, buton ve formları kullanıcıların kolayca anlayabileceği şekilde düzenledik.

1. **Kolay Navigasyon:**
   Ana sayfadan, öğrenci, öğretim üyesi, bölüm, ders ve derslik gibi modüllere hızlı erişim sağladık. Hepsi basit ve kolay anlaşılır bir şekilde gözükmektedir.
2. **Veri Girişi ve Form Tasarımı:**
   Ders ekleme, bilgi güncelleme gibi işlemler için form alanlarını düzenledik. Giriş sırasında oluşabilecek hatalar için net hata mesajları ekledik.
3. **Görsel Tasarım:**
   Uygulamanın modern ve temiz görünmesi için renk, yazı tipi ve şablon kısmına özen gösterdik.
4. **Kullanıcı Girişi ve Yetkilendirme:**
   Kullanıcı adı ve şifre ile güvenli giriş yapılabilmesi için gerekli kontrolleri ekledik.

Farklı kullanıcı rolleri (öğrenci, öğretim üyesi, yönetici) için yetkilendirme sistemimizi oluşturduk.

5. **Donanım Arayüzü Gereksinimleri**

• Giriş Aygıtları:

Fare, klavye gibi standart giriş araçları ile uyumlu çalışacak şekilde uygulamayı geliştirdik.

2.2 Fonksiyonel gereksinimler

1) Kullanıcı Yönetimi

- Öğrenci, öğretim üyesi ve yönetici ekleme, silme ve güncelleme işlemleri.

2) Ders Yönetimi

- Ders ekleme, düzenleme ve silme fonksiyonları.

3) Bölüm Yönetimi

- Bölüm ekleme ve silme işlemleri.

4) Ders Programı Oluşturma

- Haftalık ders programı oluşturma, güncelleme ve indirme işlemleri.

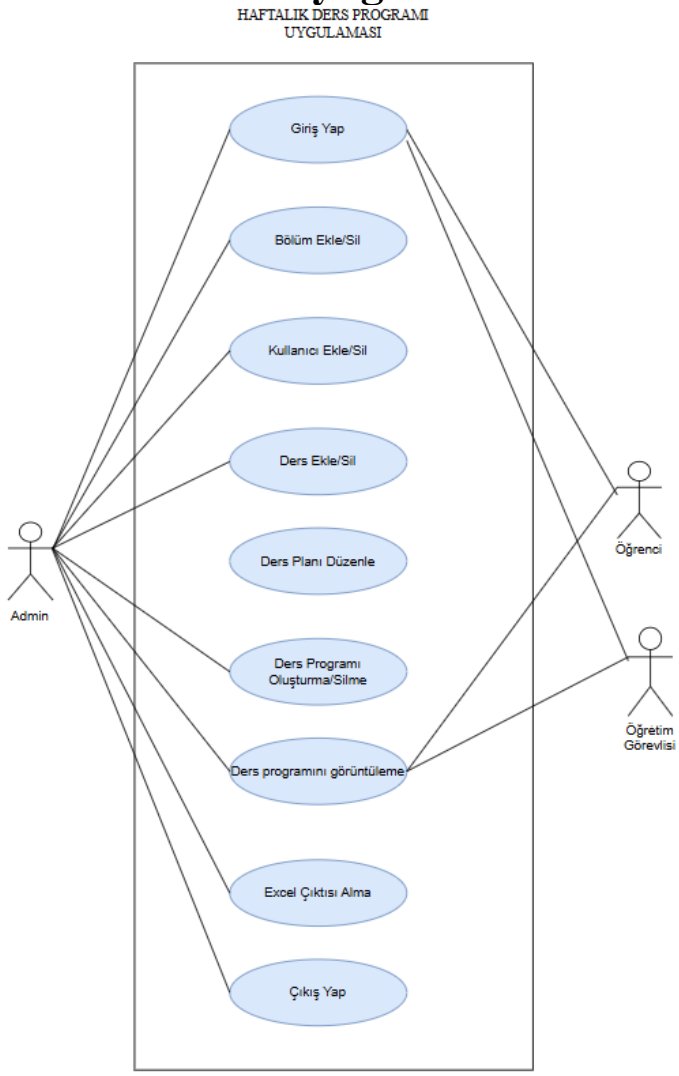
5) Çakışma Kontrolü

- Aynı dersin veya ortak verilen derslerin zaman çakışmalarını önleme.

6) Kullanıcı Girişi ve Yetkilendirme

- Güvenli oturum açma, rol bazlı erişim kontrolü.

2.3 Use-Case diyagramı



3. TASARIM

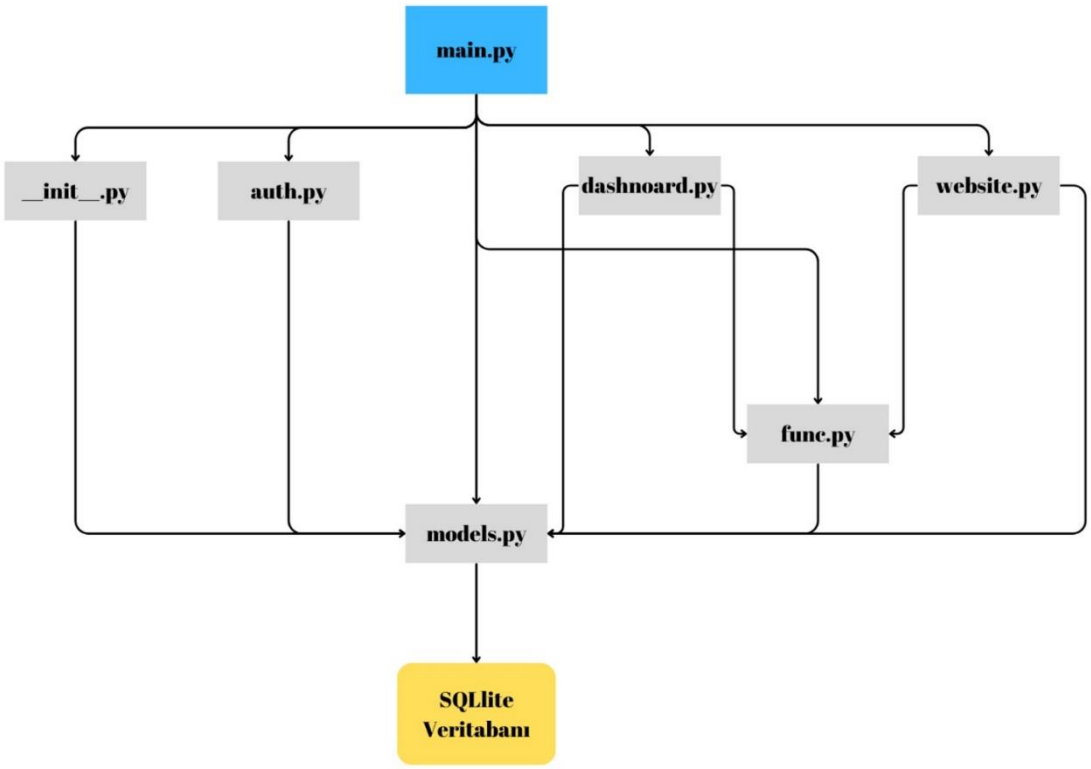
3.1 Mimari tasarım

1) **Model Katmanı:**
   - BaseModel: Diğer modellerin türetildiği temel modeldir.
   - Department: Bölüm bilgilerini içermektedir.
   - User: Kullanıcı bilgilerini (Ad, bölüm, şifre, rol vb.) saklamaktadır.
   - LessonType: Ders türlerini içermektedir. (Online veya yüz yüze olup olmadığı gibi bilgileri içerir.)
   - SyllabusOptions: Ders programı seçeneklerini tutmaktadır.
   - Syllabus: Ders programlarını saklamaktadır.
   - Excel: Excel dosyalarına ait bilgileri içermektedir.
   - Lesson: Ders bilgilerini ve ders programlarını içermektedir.

2) **Flask Uygulama Yapısı:**
   - create_app() fonksiyonu: Flask uygulamasını oluşturur ve aşağıdaki bileşenleri yüklemektedir:
     - app.secret_key tanımlanır.
     - ROOT_URL belirlenir ve proje dizini terminale yazdırılmaktadır.
     - Kullanıcı giriş gereksinimini kontrol eden before_request middleware fonksiyonu eklenmektedir.
     - auth, dashboard ve website blueprintleri kayıt edilmektedir.
     - Uygulama 0.0.0.0:80 adresinde çalıştırılmaktadır.
3) **Blueprintler:**
   - Auth (auth):
     - Kullanıcıların giriş yapmasını sağlamaktadır.
     - Yetkilendirme kontrollerini yönetmektedir.
   - Dashboard (dashboard):
     - Yönetici ve öğretim görevlisi panelini içermektedir.
   - Website (website):
     - Genel kullanıcılar için ana sayfa içeriklerini sağlamaktadır.
     - Öğrenciler için ders programı görüntüleme imkanı sunmaktadır.
     - Öğretim görevlileri için ilgili dersleri listelemektedir.
4) **Güvenlik ve Yetkilendirme:**
   - session kullanılarak oturum yönetimi yapılmaktadır.
   - before_request fonksiyonu, giriş yapmayan kullanıcıları oturum açmaya yönlendirmektedir.
   - Yönetici yetkisi gerektiren sayfalara erişim kısıtlanmıştır.
5) **Şablonlar ve Kullanıcı Arayüzü:**
   - Jinja2 şablon motoru kullanılarak dinamik HTML sayfaları oluşturulmaktadır.
   - student/index.html: Öğrenciler için ders programı görünümü sağlamaktadır.
   - teacher/index.html: Öğretim görevlileri için ders listesini sunmaktadır.
   - Flash mesajlar ile kullanıcı bilgilendirmesi yapılmaktadır.

6) **Genel İşleyiş:**
   - Kullanıcı giriş yaptığında session bilgileri saklanmaktadır.
   - Öğrenciler ders programlarını görüntüleyebilmektedir.
   - Öğretim görevlileri, verdikleri dersleri listeleyebilmektedir.
   - Yönetici kullanıcılar, ders programlarını düzenleyebilmektedir.

3.2 Modül Diyagramı



3.3 Kullanılacak teknolojiler

Proje, Python programlama dili kullanılarak geliştirilmiştir.

Kullanılan Kütüphaneler

Proje içerisinde kullanılan Python kütüphaneleri aşağıda listelenmiştir:

Flask:

- Web uygulaması geliştirmek için kullanılmıştır.
- Yönlendirme, şablon kullanımı ve oturum yönetimi gibi işlevleri sağlamaktadır.
- Blueprint yapısı ile modüler bir geliştirme yaklaşımı sunmaktadır.

Peewee:

- Veritabanı yönetimi için kullanılan bir ORM (Object Relational Mapping) kütüphanesidir.
- SQLite veritabanı ile veri işlemlerini kolaylaştırmaktadır.
- Tabloların modellenmesini ve veritabanı sorgularının daha okunabilir hale gelmesini sağlamaktadır.

Werkzeug:
- Kullanıcı kimlik doğrulama ve şifre güvenliği için kullanılmıştır.
- Kullanıcı şifrelerinin güvenli bir şekilde saklanmasını sağlamaktadır.

OpenPyXL:
- Excel dosyaları ile çalışma ve veri manipülasyonu yapmak amacıyla kullanılmıştır.
- Ders programlarının okunması, işlenmesi ve yazılması sağlanmaktadır.

UUID:
- Sistem içinde benzersiz kimlikler oluşturmak için kullanılmaktadır.
- Özellikle kullanıcılar ve dosyalar için eşsiz ID'ler üretmektedir.

Pathlib:
- Dosya ve dizin işlemlerini kolaylaştırmak için kullanılmıştır.
- Dinamik dosya yolları oluşturulmasını sağlamaktadır.

Datetime:
- Tarih ve zaman işlemlerinin yönetilmesi amacıyla kullanılmıştır.
- Ders programlarının belirli tarihlere göre düzenlenmesini mümkün kılmaktadır.

OS:
- Dosya işlemleri ve dizin yönetimi için kullanılmıştır.
- Gerekli klasörlerin otomatik olarak oluşturulmasını sağlamaktadır.

Shutil:
- Dosya taşıma ve kopyalama işlemlerinde kullanılmıştır.
- Verilerin yedeklenmesine ve düzenlenmesine yardımcı olmaktadır.

Random & String:
- Rastgele değerler oluşturmak amacıyla kullanılmıştır.
- Şifre oluşturma ve benzersiz anahtar üretme işlemlerinde kullanılmaktadır.

Diğer Teknolojiler

SQLite:
- Hafif ve gömülü bir veritabanı yönetim sistemidir.
- Kullanıcı, ders ve program verilerinin saklanmasını sağlamaktadır.

Jinja2:
- Flask tarafından kullanılan şablon motorudur.
- HTML sayfalarında dinamik içerik oluşturulmasını sağlamaktadır.

HTML, CSS ve JavaScript:
- Kullanıcı arayüzünün oluşturulmasında kullanılmıştır.
- Etkileşimli ve kullanıcı dostu bir deneyim sunulmasını sağlamaktadır.

Excel Dosyaları:
- Ders programlarının dışa aktarılması ve içe alınması için kullanılmaktadır.

3.4 Veri tabanı tasarımı

Haftalık ders programını oluşturmak ve yönetmek için **ilişkisel bir veri tabanı modeli** tasarlanmıştır. Veri tabanı, öğrenciler, öğretim üyeleri, bölümler, dersler ve ders atama süreçlerini kapsayan tablolar içermektedir. Tasarım sürecinde, **ders çakışmalarını önlemek**, **ortak dersleri yönetmek**, **derslik kapasitesine uygun atamalar yapmak** ve **dinamik bir program oluşturmak** gibi temel gereksinimler göz önünde bulundurulmuştur.Veri tabanı yönetim sistemi olarak **SQLite** tercih edilmiştir.

1. **User (Kullanıcılar)**

- Tüm kullanıcı türlerini (öğrenci, öğretim üyesi ve yönetici) tek bir tabloda toplar.
- Kullanıcıların **bölüm bilgileri**, **öğrenciyse sınıfı**, **kullanıcı türü (öğrenci, öğretim üyesi, yönetici)** gibi bilgileri içerir.

2. **Department (Bölümler)**

- Üniversitenin bölümlerini saklar.
- **Bölüm adı, kısa adı (YZM, BLM gibi)** ve **haftalık maksimum ders saati** gibi bilgiler içerir.

3. **LessonType (Ders Türleri)**

- Üniversitede verilen dersleri tanımlar.
- **Ders adı, ders kodu, öğretim üyesi ve online olup olmadığı** gibi bilgileri içerir.

4. **SyllabusOptions (Ders Programı Seçenekleri)**

- Derslerin hangi bölümlerde okutulacağını ve haftalık saatlerini belirler.
- **Hangi bölümde okutulacağı, dersin haftalık saati (hpw)** ve **dersin hangi sınıfta okutulacağı** gibi bilgileri tutar.

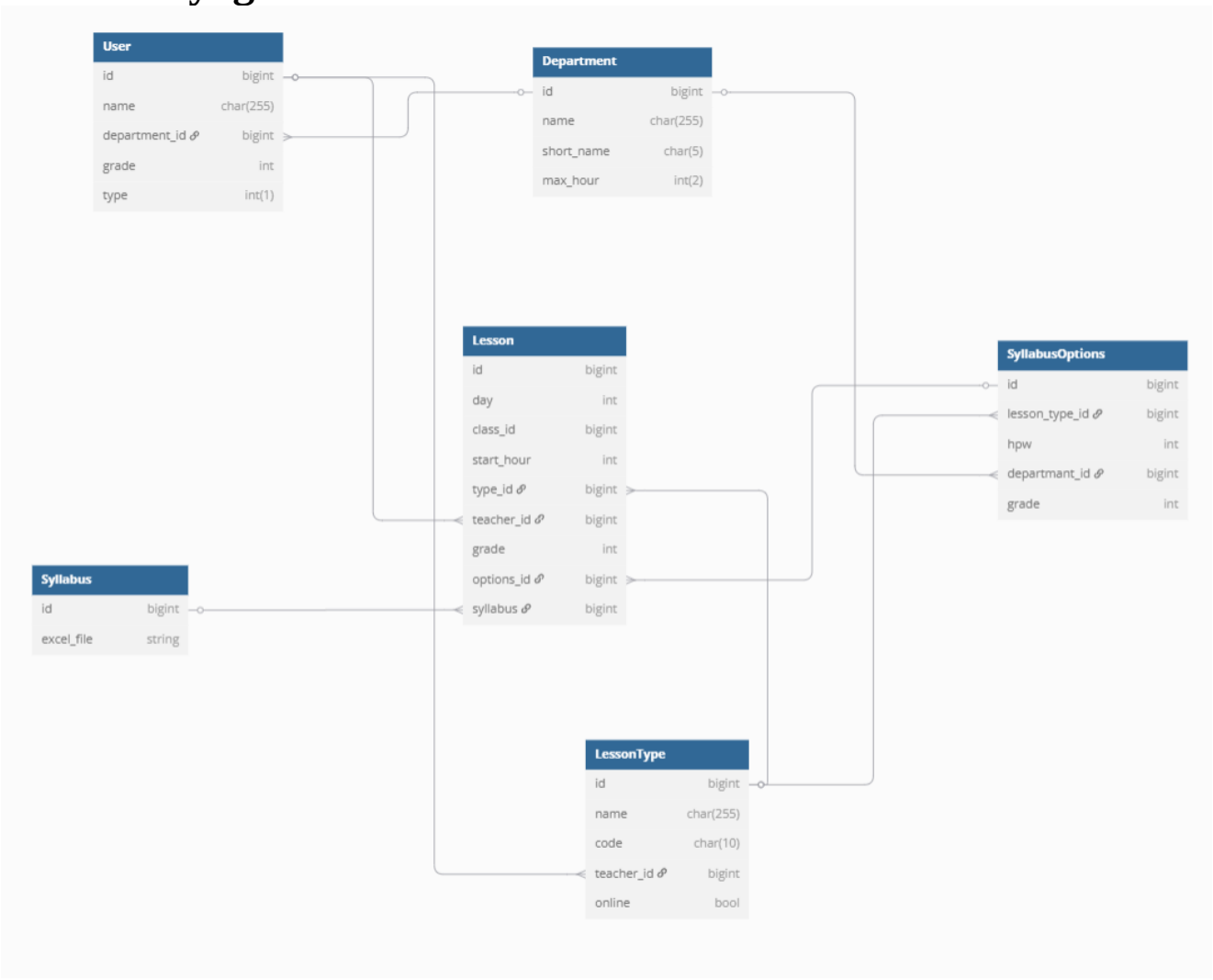
5. **Lesson (Ders Atama Tablosu)**

- Derslerin **hangi gün, hangi sınıfta ve hangi saat** gerçekleştirileceğini belirler.
- **Ders başlangıç saati, öğretmen ataması, sınıf bilgisi, gün bilgisi** gibi detayları içerir.

6. **Syllabus (Ders Programı Çıktısı)**

- Haftalık programın **Excel formatında oluşturulmuş halini** saklar.

3.5 ER Diyagramı



4. UYGULAMA

4.1 Kodlanan bileşenlerin açıklamaları

1) Main.py

Bu dosya Flask uygulamasını başlatır ve create_app() fonksiyonunu çağırarak uygulamanın yapılandırmasını gerçekleştirir. Sunucu 0.0.0.0 adresinde ve 80 portunda çalışacak şekilde ayarlanmıştır.

2) __init__.py

create_app() fonksiyonu ile Flask uygulaması yapılandırılır. Blueprint yapısı kullanılarak auth, dashboard ve website modülleri uygulamaya entegre edilir. Aynı zamanda kullanıcı oturum kontrolü için before_request fonksiyonu tanımlanmıştır. Böylece kullanıcı giriş yapmadan yetkili sayfalara erişemez.

3) auth.py

Kullanıcı kimlik doğrulama işlemleri bu dosyada yapılır. Giriş yapma, çıkış yapma ve şifre değiştirme işlemleri burada yürütülür. Kullanıcı oturumları Flask session yapısıyla yönetilir.

login():
- GET isteği: Giriş ekranını gösterir.
- POST isteği: Kullanıcının ID ve şifresini kontrol eder, doğruysa session bilgilerini ayarlar.
-Kullanıcı yetkisine göre ilgili sayfaya yönlendirilir

logout():
Kullanıcının oturumunu temizler (session.clear()) ve giriş ekranına yönlendirir.

change_password():
- Giriş yapılmamışsa kullanıcı giriş sayfasına yönlendirilir.
- GET: Şifre değiştirme formunu gösterir.
- POST: Eski şifreyi kontrol eder, yeni şifreler uyuşuyorsa günceller ve oturumu kapatır.

4) Website.py

Bu dosya, öğrenci ve öğretim üyelerinin kullanıcı arayüzünü yönetir. Her kullanıcı tipi için ders programı görüntüleme işlevi sağlar.
@website.before_request:
Yönetici oturum açmışsa bu arayüze erişimi engeller ve dashboard sayfasına yönlendirir.

index():
- Öğrenci (type == 0): Bölüm ve sınıfa ait en güncel ders programı getirilir ve tablo halinde gösterilir.
- Öğretim Görevlisi (type == 1): Verdiği derslerin tablosu hazırlanır.
- Giriş yapılmamışsa login sayfasına yönlendirilir.

5) Dashboard.py

Bu dosya, sadece yönetici rolündeki kullanıcıların erişebildiği admin paneli bileşenlerini içerir. Flask Blueprint kullanılarak oluşturulmuş dashboard modülü; bölüm, öğrenci, öğretim görevlisi, ders tipi yönetimi, ders saatlerinin tanımı ve haftalık ders programının otomatik oluşturulması gibi işlemleri kapsar.

@dashboard.before_request:
Her istekten önce çalışır. Kullanıcının yönetici olup olmadığını kontrol eder, değilse giriş sayfasına yönlendirir.

index(): Admin panelinin ana sayfasını oluşturur. Bölümler ve ilgili sınıflar için ders tanımı kontrol edilir. Ayrıca tüm SyllabusOptions ve Syllabus kayıtları yüklenir.

syllabus_options_add(department, grade):
Belirli bir bölüm ve sınıf için haftalık ders tanımı yapılmasını sağlar. Formdan alınan veriler parse edilir, online ders saati kontrol edilir ve ayarlar kaydedilir.

syllabus_create():
Tanımlı tüm ders ayarlarına göre otomatik ders programı oluşturur. Ortak dersler önceliklidir. Tüm sınıflar için çakışmasız dersler yerleştirilir ve Excel çıktıları oluşturulur.

syllabus_view(id):
Belirtilen ID'ye ait programı tablo olarak görüntüler. Tüm bölümler ve sınıflar için haftalık ders çizelgesi hazırlanır.

syllabus_delete(id): Verilen ID'ye ait ders programını, ilişkili dersleri ve Excel çıktısını sistemden tamamen siler.

get_excel(syllabus_id, department_id): Belirli bir bölüm ve programa ait oluşturulmuş Excel dosyasını istemciye gönderir.

teacher_page():Tüm öğretim üyelerini listeler.

teacher_add(): Yeni öğretim üyesi ekler. Şifre otomatik ya da kullanıcı tarafından belirlenebilir.

teacher_delete(id):Öğretim üyesiyle ilişkili tüm veriler silinir, ardından kullanıcı kaydı kaldırılır.

student_page():Tüm öğrencileri listeler.

student_add(): Yeni öğrenci kaydı oluşturur. Bölüm ve sınıf bilgileri atanır.

student_delete(id): Belirtilen öğrenciyi sistemden siler.

department_page():Tüm bölümleri listeler, öğrenci sayılarıyla birlikte.

department_add(): Yeni bölüm ekler. Kısa kod benzersiz olmalıdır.

department_delete(id):Bölüm ve ilişkili tüm verileri siler.

lesson_type_page():Tüm ders tiplerini listeler.

lesson_type_add(): Yeni ders tipi tanımlar. Aynı kodla farklı isim girişine izin verilmez.

lesson_type_delete(id): Ders tipini ve ilişkili ayarları sistemden siler.

6) Models.py

Bu dosya, uygulamanın veritabanı modellerini tanımlar. Peewee ORM kütüphanesi kullanılarak ilişkisel SQLite veritabanı yapısı kurulmuştur. Her tablo bir sınıf ile temsil edilir ve tablolar arası ilişkiler ForeignKeyField ile modellenmiştir.

db = SqliteDatabase("database/db.sqlite"):
Uygulamanın SQLite veritabanı bağlantısı burada kurulur. Tüm modeller bu veritabanını kullanır.

class BaseModel(Model):
Tüm modellerin miras aldığı temel sınıftır. İçerisindeki Meta sınıfı ile kullanılan veritabanı belirtilir.

class Department(BaseModel):
Bölüm bilgilerini saklayan tablodur.
- name: Bölüm adı
- short_name: Kısa kod (ör. BLM)
- max_hour: Haftalık maksimum ders saati

class User(BaseModel):
Tüm kullanıcıları (öğrenci, öğretim görevlisi, yönetici) temsil eder.
  - name: Kullanıcı adı
  - department: Ait olduğu bölüm (nullable)
- grade: Sınıfı (nullable)
- type: Kullanıcı tipi (0: Öğrenci, 1: Öğretim Görevlisi, 2: Yönetici)
- password: Şifre (hashlenmiş olarak saklanır)

class LessonType(BaseModel):
Verilebilecek dersleri tanımlar.
  - name: Ders adı
  - code: Ders kodu
  - teacher: Dersi veren öğretim üyesi
  - online: Çevrimiçi olup olmadığını belirtir

class SyllabusOptions(BaseModel):
Belirli bir dersin, hangi bölümde ve sınıfta, kaç saat verileceğini belirtir.
  - lesson_type
  - hpw: Haftalık ders saati
  - department

7) func.py

Func.py, projedeki işlevsel (fonksiyon bazlı) yardımcı bileşenleri içerir. Verilerin işlenmesi, ders yerleştirme algoritması, program tablosu oluşturma ve Excel çıktısı alma gibi birçok temel görevi üstlenir.
generate_password(length: int = 12): Belirtilen uzunlukta rastgele bir şifre üretir. Büyük-küçük harf, rakam ve özel karakterler içerir.

get_student_count_of_department(department: Department) -> int: Bir bölüme kayıtlı öğrenci sayısını döndürür. Kullanıcı türü öğrenci (type=0) olanları filtreler.

parse_options(request: request) -> list: Formdan alınan ders tipi, saat ve paylaşım bilgilerini parse eder. Dönen yapı [(lesson_type_id, hpw, shareable), ...] şeklindedir.

format_options(department, grade) -> list: Daha önce kaydedilmiş ders seçeneklerini saat ve paylaşım bilgileriyle birlikte döndürür.

get_shared_lessons() -> list[SyllabusOptions]: Birden fazla bölüm tarafından paylaşılan dersleri sorgular. Join işlemleriyle ortak ders tipi eşleştirilir.

place_lesson(syllabus_options, syllabus, shared=False): Dersin öğretmen ve sınıf uygunluğuna göre programa otomatik yerleştirilmesini sağlar. Online dersler 17:00–18:00 arası, yüzyüze dersler 09:00–16:00 arası saatlerde atanır.

get_lessons(syllabus, department, grade): Belirtilen bölüm ve sınıfa ait haftalık ders çizelgesini tablo şeklinde döndürür.

get_lessons_for_teacher(syllabus, teacher): Öğretim üyesinin haftalık ders çizelgesini döndürür. Hücrelerde birden fazla ders olabilir

get_day = lambda day: days[day - 1]: 1–5 arası sayıları gün isimlerine çevirir. (1 → Pazartesi vb.)

excel_locations: Excel çıktısında hücrelerin konumlarını belirten koordinat yapısıdır.
create_excel(excel_folder: Path, folder_name, syllabus, department): Verilen bölüm için şablon Excel dosyasına programı yazar ve sistemde kaydeder. Dosya ismi UUID ile belirlenir ve veritabanına kaydedilir.

4.2 Görev dağılımı

1. Bileşenlerinin Tasarım ve Geliştirme Aşamalarındaki Görev Dağılımı

Proje kapsamında haftalık ders programı sisteminin geliştirilmesi için aşağıdaki bileşenler belirlenmiş ve ekip üyeleri arasında görev paylaşımı yapılmıştır:

a) Veritabanı Tasarımı ve Yönetimi

- Kullanıcı, bölüm, ders ve derslik tablolarının oluşturulması
- Veri tabanı ilişkilerinin belirlenmesi ve normalizasyon işlemlerinin yapılması

Sorumlu Kişi(ler): [tüm ekip üyeleri]

b) Kullanıcı Arayüzü (Frontend) Geliştirme

- Kullanıcı giriş ve kayıt ekranlarının oluşturulması
- Öğrenci, öğretim üyesi ve yönetici için farklı arayüzlerin tasarlanması
- Ders ekleme, çıkarma, program görüntüleme gibi fonksiyonların frontend ile entegrasyonu
- Responsive tasarım ve kullanıcı deneyimi iyileştirmeleri

Sorumlu Kişi(ler): [tüm ekip üyeleri]

c) İş Mantığı ve Backend Geliştirme

- Kullanıcı yetkilendirme ve doğrulama sisteminin oluşturulması
- Ders programı oluşturma algoritmasının geliştirilmesi
- Öğretim üyeleri ve öğrenciler için program oluşturma kısıtlarının işlenmesi
- Veri tabanı ile arayüz arasındaki bağlantıların sağlanması

Sorumlu Kişi(ler): [tüm ekip üyeleri]

d) Otomatik Ders Programı Planlama ve Çakışma Kontrolleri

- Ders saatleri ve öğretim üyelerinin uygunluk durumlarına göre ders atamalarının yapılması
- Aynı sınıfa ait derslerin çakışmasını önleyen algoritmaların geliştirilmesi
- Derslik kapasitesine göre derslerin uygun şekilde atanması

Sorumlu Kişi(ler): [tüm ekip üyeleri]

e) Çıktı Alma ve Raporlama Modülü

- Ders programlarının Excel formatında çıktısının alınması
- Kullanıcıların ders programlarını görüntüleyebileceği raporlama ekranlarının oluşturulması
- Sistem performans analizlerinin yapılması

Sorumlu Kişi(ler): [tüm ekip üyeleri]

2. Raporun Hazırlanması Sürecindeki Görev Dağılımı

Proje raporunun hazırlanması sürecinde her ekip üyesi belirli bölümlerden sorumlu olmuştur. Görev dağılımı şu şekilde yapılmıştır:

Raporun Hazırlanması Sürecindeki Görev Dağılımı

Proje raporunun hazırlanması sürecinde her ekip üyesi belirli bölümlerden sorumlu olmuştur. Görev dağılımı şu şekilde gerçekleştirilmiştir:

- **1. Başlık:** Nurdan Bulut tarafından hazırlanmıştır.
- **2. Başlık:** Zeynep Kedikli tarafından yazılmıştır.
- **2.3 Use Case Diyagramı ve ER Diyagramı:** Zeynep Kedikli ve Nurdan Bulut tarafından oluşturulmuştur.
- **3. Başlık:** Zehra Kandaz tarafından yazılmıştır.
- **3. Başlıktaki Modül Diyagramı:** Zehra Kandaz tarafından hazırlanmıştır.
- **4.1. Başlık:** Nurdan Bulut tarafından hazırlanmıştır.
- **4.2, 4.3 ve 4.4 . Başlık:** Zeynep Kedikli tarafından yazılmıştır.

Bu görev paylaşımı ile proje raporu ekip üyelerinin katkılarıyla tamamlanmıştır.

4.3 Karşılaşılan zorluklar ve çözüm yöntemleri

1. Ders Çakışmalarını Önleme Problemi

Zorluk:

Ders programı oluşturulurken, aynı anda birden fazla dersin aynı sınıfta veya aynı öğretim üyesiyle çakışmasını önlemek zor oldu. Ayrıca, ortak derslerin uygun zamanlara yerleştirilmesi de dikkat edilmesi gereken bir noktaydı.

Çözüm:

Bu problemi çözmek için derslerin ve öğretim üyelerinin uygunluk durumlarını dikkate alan bir algoritma geliştirildi. Program, öğretim üyelerinin uygun saatlerini kontrol ederek ve dersliklerin uygun kapasiteye sahip olup olmadığını doğrulayarak atamaları yaptı.

2. Kullanıcı Arayüzü Tasarımındaki Zorluklar

Zorluk:

Öğrenci, öğretim üyesi ve yöneticilerin sisteme giriş yapabilmesi ve dersleri yönetebilmesi için kullanıcı dostu bir arayüz oluşturmak zaman aldı.

Çözüm:

Öncelikle temel bir prototip oluşturuldu ve ekip içinde test edilerek eksiklikler belirlendi.

3. Ders Programının Excel Formatında Çıktı Olarak Alınması

Zorluk:

Ders programının sistem tarafından otomatik olarak belirlenen formatta Excel çıktısı alması gerektiğinden, veri tabanındaki bilgilerin Excel formatına uygun şekilde aktarılması teknik bir zorluk oluşturdu.

Çözüm:

Excel kütüphaneleri kullanılarak, ders programındaki bilgilerin istenen formatta oluşturulmasını sağlayan bir fonksiyon geliştirildi.

4.4 Proje isterlerine göre eksik yönler

Projenin geliştirilme sürecinde belirlenen bazı işlevler, çeşitli nedenlerden dolayı tam olarak kodlanmamış veya proje kapsamında çıkarılmıştır.

- **Derslik (Sınıf) Kapasitesi Kısıtlaması Kaldırıldı:** Başlangıçta derslerin belirli kapasitedeki sınıflara atanması planlanmıştı. Ancak bu kısıt, derslerin yerleşimini zorlaştırdığı için sistemden çıkarılmıştır. Böylece, sınıf kapasitesi kontrolü olmadan daha esnek bir ders atama süreci sağlanmıştır.
- **Öğretmenlerin Uygunluk Durumu Kaldırıldı:** Derslerin öğretmenlerin müsait olduğu saatlere göre planlanması düşünülmüştü. Ancak, bu süreç çok fazla veri girişi gerektirdiğinden ve sistemin yönetimini zorlaştırdığından dolayı kaldırılmıştır. Öğretmenlerin uygunluk durumu, ders dağılımını daha iyi yönetmek için faydalı olabilirdi, ancak fazla zaman alacağı ve süreci uzatacağı için proje kapsamında uygulanmamıştır.

Projemiz genel işleyiş açısından sistemin temel fonksiyonları korunmuş ve proje isterlerine mümkün olduğunca uygun şekilde geliştirilmiştir.

5. TEST VE DOĞRULAMA

5.1 Yazılımın test süreci

Bu projede herhangi bir otomatik test kodu yazılmadığı için testler **manuel** olarak gerçekleştirilmiştir. Test sürecinde sistemin doğru çalıştığını kontrol etmek amacıyla belirli

senaryolar uygulanmıştır.

Test Aşamaları

1. **Giriş ve Yetkilendirme Testi**
    a. Tarayıcıdan http://127.0.0.1/ adresine giriş yapıldı.
    b. Kullanıcı adı ve şifre girilerek sisteme giriş denendi.
    c. Başarılı giriş yapıldıktan sonra kullanıcı yetkilerine göre ilgili sayfalara yönlendirildiği gözlemlendi.
2. **Ders ve Kullanıcı Yönetimi Testleri**
    a. Yeni bir ders eklenerek, eklenen dersin listede görünüp görünmediği kontrol edildi.
    b. Var olan bir ders düzenlendi ve değişikliğin kaydedilip kaydedilmediği doğrulandı.
    c. Kullanıcı ekleme ve silme işlemleri yapılarak sistemin doğru çalıştığı gözlemlendi.
3. **Ders Programı Oluşturma Testleri**
    a. Farklı dersler için haftalık program oluşturuldu.
    b. Aynı saatte birden fazla ders atanıp atanamayacağı test edildi.
    c. Ders çakışmalarının sistem tarafından engellenip engellenmediği kontrol edildi.
4. **Veri Kaydı ve Veri Çekme Testleri**
    a. Eklenen derslerin veri tabanında doğru şekilde saklanıp saklanmadığı kontrol edildi.
    b. Ders programının doğru şekilde görüntülenip görüntülenmediği incelendi.
5. **Hata Senaryoları ve Geri Bildirim Testleri**
    a. Eksik veya hatalı giriş yapıldığında sistemin nasıl tepki verdiği test edildi.

Test Sonuçları

Testlerin sonucunda uygulamanın temel fonksiyonlarının çalıştığı görülmüştür. Ancak, bazı işlemlerde geliştirmeye açık noktalar tespit edilmiştir.

- **Eksiklikler:**
    - Otomatik test kodlarının olmaması nedeniyle tüm testler manuel yapıldı.

5.2 Yazılımın doğrulanması

1. Tam ve Doğru Çalışan Bileşenler

- **Kullanıcı Girişi ve Yetkilendirme**
  Kullanıcı adı ve şifre ile giriş yapma işlemi başarılı çalışmaktadır. Yanlış giriş denemelerinde sistem hata mesajı göstermektedir.
- **Ders ve Kullanıcı Yönetimi**
  Ders ekleme, düzenleme ve silme işlevleri beklendiği gibi çalışmaktadır. Kullanıcı (öğrenci, öğretim üyesi) ekleme ve silme işlemleri başarılıdır.
- **Ders Programı Oluşturma ve Görüntüleme**
  Haftalık ders programı başarıyla oluşturulabilmektedir. Kullanıcılar kendi ders programlarını sorunsuz bir şekilde görüntüleyebilmektedir.
- **Veri tabanı İşlemleri**
  Eklenen dersler ve kullanıcı bilgileri doğru şekilde veritabanına kaydedilmektedir. Sistemde yapılan değişiklikler veri tabanına işlenmekte ve güncellenmektedir.

2. Eksik veya Hatalı Çalışan Bileşenler
Derslik Bilgisi Eksikliği
Ders programında hangi dersin hangi derslikte yapılacağı bilgisi bulunmamaktadır. Bu nedenle derslerin kapasiteye uygun şekilde atanıp atanmadığı doğrulanamamaktadır.
Sonuç:
Yazılım, temel işlevleri yerine getirmektedir ancak bazı bileşenlerde hata ve eksiklikler bulunmaktadır.

6 GitHub Bağlantıları

• Zehra KANDAZ: https://github.com/ZehraKandaz
• Nurdan BULUT: https://github.com/nurdanbulut
• Zeynep KEDİKLİ: https://github.com/Zeynepkedikli