

**T.C**

**KOCAELİ SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ**

**MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR/YAZILIM MÜHENDİSLİĞİ**

**PROJE KONUSU: WEB ARAYÜZÜNDEN OTOMATİK DERS PROGRAMI OLUŞTURMA**

**HAZIRLAYANLAR**

**AHMET EREN ŞENGÜL 220502036**

**BAŞAR BURAK ÜNAL 220501025**

**YUSUF USTAOĞLU 220502003**

**EFE ÖZTAŞ 220501014**

**ONUR KOCA 240502040**

**DERS SORUMLUSU:**

**Dr. Öğr. Üyesi ELİF PINAR HACIBEYOĞLU**

**TARİH: 7.05.2025**

1. **GİRİŞ**
   1. **Projenin amacı**

* Bu projenin amacı, üniversitelerde ve eğitim kurumlarında ders programlarını yönetmek ve optimize etmek için bir yazılım geliştirmektir. Geliştirilecek yazılım, derslerin, öğretim üyelerinin, dersliklerin ve öğrenci gruplarının yönetimini kolaylaştırmayı hedefler. Aynı zamanda, kullanıcıların zaman ve kaynak verimliliğini artırarak eğitim süreçlerinin daha düzenli ve verimli olmasını sağlamayı amaçlar.
* Proje kapsamında geliştirilen web uygulaması, mühendislik fakültesi dersleri için haftalık ders programlarını otomatik olarak oluşturacaktır. Kullanıcılar, web arayüzü üzerinden ders programlarını görüntüleyebilir, ders seçimleri yapabilir, öğretim üyeleri için haftalık programları görüntüleyebilir ve gerekirse manuel düzenlemeler gerçekleştirebilir. Uygulama, veritabanından alınan bilgilerle otomatik olarak ders programını oluşturacak ve bu sayede kullanıcılar tüm dersler hakkında detaylı bilgilere ulaşabilecektir.

**1.2 Projede Gerçekleştirilmesi Beklenenler**

* Haftalık ders programlarının otomatik olarak oluşturulması
* Kullanıcıların ders seçimlerini yapabilmesi
* Öğretim üyelerinin programlarını görüntüleyebilmesi
* Manuel ders programı düzenlemeleri yapılabilmesi
* Ders programındaki çakışmaların tespiti ve yönetilmesi
* Kullanıcı dostu bir web arayüzü tasarlanması
* Ders programının Excel formatında dışa aktarılabilmesi
* Veritabanı entegrasyonu ile ders, öğretim üyesi ve derslik bilgilerine kolay erişim sağlanması

1. **GEREKSİNİM ANALİZİ**
   1. **Arayüz gereksinimleri**

**Kullanıcı Arayüzü Gereksinimleri**

* **Kullanıcı Dostu Tasarım**: Arayüz, her türden kullanıcının rahatlıkla anlayabileceği ve kullanabileceği bir yapıya sahip olmalıdır. Kullanıcılar, sisteme dair temel işlemleri hızlı bir şekilde yapabilmelidir.
* **Duyarlı Tasarım (Responsive Design)**: Arayüz, farklı cihazlar (mobil, tablet, masaüstü) ile uyumlu olmalıdır.
* **Renk ve Kontrast Kullanımı**: Görsel uyum ve okunabilirliği artırmak için uygun renk paletleri kullanılmalıdır.
* **Görsel Takvim**: Ders programının günlere ve saatlere göre tablo formunda görüntülenmesi
* **Ders Seçim ve Kayıt Ekranı**: Öğrenciler, dersleri kolayca seçebilecek, ilgili derslerin ayrıntılarını görüntüleyebilecek ve kaydedebilecektir.
* **Çakışma Tespit ve Yönetim Aracı**: Derslerin zamanları çakıştığında, kullanıcıya uyarı gösterilmelidir.
* **Program Yönetimi Ekranı**: Öğretim üyeleri ve program koordinatörleri, ders programlarını görüntüleyip düzenleyebilecektir. Bu ekranlar, veri girişi için kullanıcı dostu olmalı ve kolayca düzenlenebilir özelliklere sahip olmalıdır.

**Donanım Arayüzü Gereksinimleri**

* **Bilgisayarlar ve Mobil Cihazlar**: Web uygulaması tarayıcı tabanlı olduğu için, kullanıcıların internet tarayıcıları üzerinden erişebilmesi için herhangi bir özel donanım gereksinimi yoktur. Ancak, cihazların internete bağlanabiliyor olması gerekmektedir.
  1. **Fonksiyonel gereksinimler**

**Ders Programı Yönetimi**

* **Otomatik Ders Programı Oluşturma**: Sistem, veritabanında yer alan ders, öğretim üyesi, derslik ve öğrenci bilgilerini kullanarak, çakışmasız haftalık ders programlarını otomatik olarak oluşturmalıdır.
* **Ders Programı Görüntüleme**: Kullanıcılar, kendilerine ait ders programlarını görsel olarak görüntüleyebilmeli, tarih ve saat dilimlerine göre kolayca takip edebilmelidir.
* **Ders Seçme**: Öğrenciler, dersleri seçebilir ve kaydedebilir. Seçilen dersler otomatik olarak öğrencinin programına eklenir.
* **Ders Seçim Filtreleme**: Öğrenciler, dersleri bölüm, öğretim üyesi, ders saati gibi kriterlere göre filtreleyebilir.

**Ders Seçim ve Kayıt**

* **Öğrenci Kayıt İşlemleri**: Öğrenciler, kendilerine uygun dersleri seçip kaydedebilir. Seçim sırasında dersin öğretim üyesi, saat ve derslik bilgisi gibi detaylar gösterilir.
* **Ders Kayıt Çakışması**: Derslerin zamanları çakıştığında sistem, kullanıcıyı uyararak çakışmanın çözülmesi için önerilerde bulunur.
* **Kapasite Kontrolü**: Her dersin öğrenci kapasitesi sınırlandırılmıştır. Öğrenci, kapasiteyi aşan bir ders seçmeye çalıştığında sistem uyarı verir.

**Kullanıcı Yönetimi**

* **Kullanıcı Hesapları**: Öğrenciler, öğretim üyeleri ve program koordinatörleri farklı yetkilerle sisteme giriş yapacaktır. Her kullanıcı tipi sadece yetkili olduğu alanlara erişebilecektir.
* **Hesap Yönetimi**: Kullanıcılar, hesap bilgilerini güncelleyebilir, şifre değiştirebilir ve sistem üzerinden kendi ders programlarını görüntüleyebilirler.

**Derslik ve Zaman Yönetimi**

* **Derslik Yönetimi**: Dersler, dersliklerin kapasitesine göre atanacaktır. Derslik kapasitesine göre, her dersin öğrenci sayısı kontrol edilecektir.
* **Zaman Çakışması Algoritması**: Derslerin zamanları çakıştığında sistem, çakışmayı tespit eder ve kullanıcıya çözüm önerileri sunar.

**Veritabanı Yönetimi**

* **Veri Ekleme ve Güncelleme**: Sistem, dersler, öğretim üyeleri, derslikler ve öğrenciler hakkında verileri güncelleyebilecek ve bu verileri veritabanında tutacaktır.

**Çakışma Yönetimi**

* **Çakışma Tespiti**: Sistem, derslerin öğretim üyeleri ve dersliklerle olan zaman çakışmalarını tespit eder.
* **Çakışma Çözme**: Kullanıcılar, çakışmaların çözülmesi için öneriler alabilir ve manuel olarak düzenlemeler yapabilir.
  1. **Use-Case diyagramı**
* Use-Case diyagramı eklenmesi

1. **TASARIM**
   1. **Mimari tasarım**

* Mimari tasarımın maddeler halinde açıklanması
* Modül diyagramı eklenmesi
  1. **Kullanılacak teknolojiler**
* Yazılımın hangi dilde yazılacağı hakkında bilgi
* Kullanılacak harici kütüphaneler hakkında bilgi
* Varsa diğer teknolojiler ile ilgili açıklama
  1. **Veri tabanı tasarımı**
* Varsa veri tabanı tasarımı hakkında açıklama
* ER Diyagramı eklenmesi
  1. **Kullanıcı arayüzü tasarımı**
* Kullanıcı arayüzü tasarımı ile ilgili açıklama.
* Yazılımdan ekran çıktıları alınarak üzerinden açıklama yapılması
* Uygulamanın nasıl çalıştırılacağı ile ilgili açıklama

1. **UYGULAMA**
   1. **Kodlanan bileşenlerin açıklamaları**

* Bileşenlerin (fonksiyonlar, sınıflar ya da modüller) açıklanması
  1. **Görev dağılımı**
* Bileşenlerinin tasarım ve geliştirme aşamalarındaki görev dağılımının açıklanması
* Raporun hazırlanması sürecindeki görev dağılımının açıklanması
  1. **Karşılaşılan zorluklar ve çözüm yöntemleri**
* Geliştirme sürecinde karşılaşılan problemler ve bunlara getirilen çözümler
  1. **Proje isterlerine göre eksik yönler**
* Proje gerçeklenmesi beklenen görevlerden hiç kodlanamayanlar varsa belirtilmesi

1. **TEST VE DOĞRULAMA**
   1. **Yazılımın test süreci**

* Yazılım için bir test uygulaması (bir sınıf ya da bir fonksiyon halinde olabilir) geliştirilmeli ve bu uygulama ile test edilen bileşenler açıklanmalıdır.
* Test uygulaması yazılımda geliştirilmiş olan bütün bileşenleri test ediyor olmalı.
* Test uygulaması tekrar tekrar test etmeye imkan tanıyacak şekilde geliştirilmiş olmalıdır.
  1. **Yazılımın doğrulanması**
* Test uygulaması ile yazılımın test edilmesi sonucunda elde edilen yazılım doğruluğu açıklanmalıdır.
* Test sonucunda eksik ya da hatalı çalışan bileşenler ile tam ve doğru çalışan bileşenler hakkında maddeler halinde açıklama yapılmalıdır.

1. **KAYNAKÇA**

<https://flask-login.readthedocs.io>

1. **GitHub LİNKLERİ**

Onur Koca <https://github.com/OnurKoca1>

Ahmet Eren Şengül <https://github.com/Eren1213>

Yusuf Ustaoğlu <https://github.com/katlicia>

Efe Öztaş <https://github.com/EOztas>

Başar Burak Ünal <https://github.com/BasarBurakUnal>