Club Starboys

ScheduleX Final Document

Ali Berkay Görgülü, Hüseyin Özgür Kamalı, Barış Güngör, Umut Çağatay Tapur, Görkem Mert, Mehmet Zahid Kaya

08.12.2024

ScheduleX

1. Giriş

1.1. Amaç ve Kapsam

1.1.1. Amaç

ScheduleX, üniversitelerde ders programlarının düzenlenmesini kolaylaştırarak, sınıf kapasitesi, engelli erişimi, öğretim görevlisi talepleri ve bölüm öncelikleri gibi kriterleri göz önünde bulunduran akıllı bir sistem geliştirmeyi amaçlamaktadır. Bu sistem sayesinde, eğitim kurumları kaynaklarını verimli bir şekilde kullanabilirken, öğrenci ve öğretim görevlilerinin ihtiyaçlarına uygun, esnek ve optimize edilmiş ders programları oluşturulması hedeflenmektedir. Proje, üniversitedeki tüm paydaşlara düzenli ve erişilebilir bir planlama deneyimi sunarak, eğitim süreçlerini daha organize ve verimli bir hale getirmeyi amaçlamaktadır.

1.1.2. Kapsam

ScheduleX, üniversitelerde ders programı düzenleme süreçlerini kolaylaştırmak ve verimliliği artırmak amacıyla geliştirilmiş yenilikçi bir sistemdir. Hem öğretim üyeleri hem de yöneticiler için farklı kullanıcı girişleri sunarak eğitim yönetimini daha etkili hale getirir.

ScheduleX'in Gerçekleştirebileceği Özellikler:

- **Ders Programı Oluşturma:** Sınıf kapasiteleri ve öğrenci sayılarına göre otomatik ders programları hazırlar.
- **Akıllı Yerleşim Algoritması:** Her ders için en uygun sınıf ve zaman yerleşimini belirler, engelli öğrenciler için uygun düzenlemeler yapar.
- **Öğretim Üyeleri Taleplerine Göre Düzenleme:** Öğretim üyelerinin özel taleplerini dikkate alarak ders programlarını oluşturur.
- **Raporlama ve Geri Bildirim:** Kullanıcılara programlama nedenlerini bildirir ve açıklayıcı geri bildirim sağlar.
- **Ders Süreleri ve Araları Yönetimi:** Ders sürelerini ve geçiş sürelerini dikkate alarak zamanlama yapar.
- **Engelli Öğrenci Erişimi:** Engelli öğrencilerin erişim durumunu kontrol eder ve uygun sınıfları belirler.

ScheduleX'in Gerçekleştiremeyeceği Özellikler:

- **Fiziksel Sınıf Düzenlemesi:** Altyapı değişiklikleri yapmaz; yalnızca programlama sağlar.
- **Öğrenci Devamsızlık Takibi:** Devamsızlık durumu öğretim üyeleri tarafından manuel olarak takip edilir.
- **Ders İçeriği Yönetimi:** Ders içeriğiyle ilgili işlevsellik sağlamaz.
- **Gerçek Zamanlı Güncellemeler:** Ders programı değişiklikleri anlık olarak yansıtılmaz.
- **Öğrenci İletişimi:** Öğrenciler arasında doğrudan iletişim sağlamaz; iletişim başka platformlarla yapılmalıdır.

ScheduleX, bu özellikleriyle eğitim süreçlerini daha düzenli ve verimli hale getirmeyi hedefler.

1.2. Paydaşlar

Bu bölüm Potansiyel Müşteriler, Geliştirme Ekibi, ScheduleX Kullanıcılar olarak üç kısma ayrılır.

1.2.1. Potansiyel Müşteriler

ScheduleX projesi derslerin program ve saatlerini; sınıf kapasiteleri, laboratuvar uygunlukları, öğretim üyelerinin uygunluk saatleri ve verimli eğitim-öğretim saatleri gibi parametreleri ele alarak en optimize şekilde ayarlamaya yönelik bir üründür. Aşağıda, ScheduleX'in bu amaç ve kapsam doğrultusunda özneler farklı olmak üzere farklı koşullarda da kullanılabilir:

1. **Eğitim ve Öğretim Kurumları:** Okullar, kolejler ve üniversiteler genellikle yeni eğitim-öğretim yılında öğrenci ve öğretim üyesi ilişiği içerisinde oluşturulan ders programını manuel olarak oluştururlar. ScheduleX, bu bağlamda en optimize ve otomasyon temelinde bir araç deneyimi işlevi görür.
2. **Şirketler ve İşyerleri:** İşletmeler, personelin ve şirket organizasyonlarının saat takibini yapabilmek için bir takvime ihtiyaç duyar. Bu bağlamda takvimde organizasyonlar ve iş saatlerinin çakışmaması büyük önem arz eder. ScheduleX, alternatif olarak bu problemlere de çözüm önerisi sunar.
3. **Etkinlik ve konferans organizatörleri:** Etkinlik ve konferans organizatörleri, oluşturacakları organizasyonlar ve saatleri için belirli bir planlama sistemine ihtiyaç duyabilirler. ScheduleX, bu konuda onlara yardımcı olabilecek bir alternatif içerir.
4. **Çeşitli Organizasyonlar ve Topluluklar:** Çeşitli organizasyonlar ve topluluklar, üyelerin toplantılara ve etkinliklere katılımını düzenlemek ve

organizasyon tarihlerini ve saatlerini en optimize düzeyde tutabilmek için ScheduleX ürünün sağladığı avantajlardan faydalanabilirler.

1.2.2. Geliştirme Ekibi

ScheduleX sistemi, Çevik (Agile) Yazılım Geliştirme Modeli kullanılarak geliştirilecektir. Bu proje yönetim sisteminde, çeşitli paydaşlar bulunmaktadır ve her biri belirli roller ve sorumluluklar üstlenmektedir. Aşağıda paydaşların tanımları ve sorumlulukları daha detaylı bir şekilde ele alınmıştır:

1. **Geliştiriciler:** ScheduleX projesinde geliştiriciler, yazılımın tasarımından başlayarak, geliştirilmesine ve test edilmesine kadar olan süreçte görev alırlar. Bu süreçte, belirlenen gereksinimlere uygun olarak teknolojilerin uygulanması, sistem bileşenlerinin entegrasyonu ve hata ayıklama gibi görevleri üstlenirler. Ayrıca, proje sürecinde ortaya çıkan teknik zorlukları aşmak ve verimli çözümler üretmek de geliştiricilerin sorumlulukları arasındadır.
2. **İş/Sistem Analistleri:** ScheduleX projesinde iş/sistem analistleri, kullanıcı ihtiyaçlarını belirlemek ve sistemin gereksinimlerini belirlemekle sorumludurlar. Bu kapsamda, kullanıcı ihtiyaçlarını anlamak, gereksinimleri belgelemek, tasarlamak ve geliştiricilerle iş birliği yaparak çözümler üretmek gibi görevleri üstlenirler. Ayrıca, proje sürecinde gereksinim değişikliklerini yönetmek ve kullanıcılarla etkileşim halinde olmak da iş/sistem analistlerinin sorumlulukları arasındadır.
3. **Proje Yöneticileri:** ScheduleX projesinde proje yöneticileri, projeyi planlamak, koordine etmek ve spiral metodolojisini uygulamakla görevlidirler. Bu kapsamda, proje kaynaklarını yönetmek, zaman çizelgelerini oluşturmak, risk yönetimini gerçekleştirmek ve ekip ile iletişimi sağlamak gibi işlerle ilgilenirler. Ayrıca, proje ilerlemesini düzenli olarak izlemek ve gerekli düzeltici önlemleri almak da proje yöneticilerinin sorumlulukları arasındadır.
4. **Kullanıcılar:** ScheduleX projesinde kullanıcılar, sistemi doğrudan kullanan ve belirli işlevleri gerçekleştiren kişilerdir. Öğretim üyeleri, editörler ve asistanlar gibi farklı kullanıcı profilleri bu kategoriye girer. Kullanıcılar, sistemi etkin bir şekilde kullanarak ders yönetimi, yoklama alma ve kullanıcı hesaplarını yönetme gibi işleri yaparlar. Ayrıca, sistemin kullanımı sırasında geri bildirim sağlamak da kullanıcıların sorumlulukları arasındadır.

1.2.3. ScheduleX Kullanıcılar

ScheduleX Projesinde kullanıcılar istemi doğrudan kullanan ve belirli işlevleri gerçekleştiren kişilerdir. Bu kullanıcılar işlevleri bakımından 3 kısımda incelenebilir: Öğretim üyeleri, editörler, asistanlar olmak üzere. Her kullanıcının birbirinden farklı olarak ekstra rolleri vardır, onlar da şu şekildedir:

Editörler:

1. Sisteme yeni öğretim üyesi kaydedebilirler.
2. Sisteme yeni bir bölüm ekleyebilirler ve bu bölümle alakalı görevli asistan atayabilirler.
3. Sisteme ders ekleyebilirler, ekledikleri bu dersleri dönemsel olarak görebilirler ve bu derslere öğretim üyesi atayabilirler.
4. Bölümlerin öğrenci sayısına, belli dönemlerin öğrenci sayısına erişebilirler.
5. Öğretim üyesi kısıtlarını görüntüleyebilir ve düzenleyebilirler.
6. Sınıfların niteliklerini, ait olduğu blok bilgilerini görebilirler.
7. Ders önceliklerini belirleyebilir ve düzenleyebilirler.
8. Ders niteliklerini öğrenebilir.
9. Okul laboratuvar bilgilerine erişebilir.
10. Sınıf mevcudu için bir hata payı belirleyebilir.

Öğretim Üyeleri:

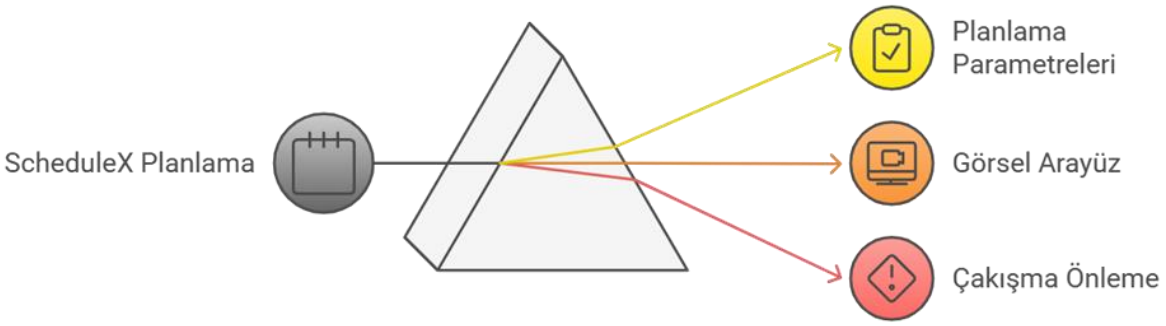
1. Atandıkları dersleri görebilirler.
2. Ders saati kısıtı belirleyebilirler.
3. Ders programı görüntüleyebilirler.
4. Bölümlerin öğrenci sayısına, belli dönemlerin öğrenci sayısına erişebilirler.
5. Ders niteliklerini öğrenebilir.

Asistanlar:

1. Ders programı görüntüleyebilirler.
2. Bölümlerin öğrenci sayısına, belli dönemlerin öğrenci sayısına erişebilirler.
3. Ders niteliklerini öğrenebilir.
4. Öğretim üyesi kısıtlarını ve ders önceliklerini belirleyip sunma hakkına sahiptir.

1.3. Problem Durumu

ScheduleX kullanıcılarına kesin ve eksiksiz planlama imkânı sunar. Günümüzde üniversiteler, okullarda kullanılan planlama aşamalarını otomatize bir hale getirmeyi hedefler. Bu kuruluşlarda yürütülen planlamalar insanlar tarafından yönetilmekte olup bazı kısımları es geçtikleri görülmüştür. Bir ders planı oluştururken öğretim üyesinin uygunluğu, ders çakışması, sınıf çakışması, kalan öğrencilerin derslerinin çakışması, planın görselleştirilmesi, planın hatalı olma şansı vb. parametrelerin gözetilmesi gerekir. ScheduleX planlama ve yönetme safhasına gerekli kriterler eşliğinde buna kesin ve net bir çözüm getirmeyi amaçlar. Yapılacak uygulama görsel bir arayüz ile ders programları geliştirmeyi ve belirli algoritmalar ile bu parametreleri gözetip çakışmaları önlemeyi hedefler.



Şekil 1: ScheduleX İçeriği

ScheduleX esnek bir rol yapısı ve görsel çeşitliliği ile yaygın bir etki hedefler. Günümüzde kullanılan planlama teknikleri öğretim üyelerinin zamanlarını etkili kullanamaması, öğrencilerin ders aralarındaki vakitlerini verimli geçirememelerine sebep olarak verimliliği düşürmektedir. Geleneksel yöntemlerde öğrenci sayısı, derslik kapasitesi, öğretim üyelerinin uygunluk saatleri ve farklı derslerin çakışmaması gibi birçok parametrenin göz önünde bulundurulması gerekmektedir. Bu parametrelerin manuel olarak yönetilmesi zaman kaybına, hatalara yol açmakta, ayrıca her dönem revize gerektirmekte.

1.4. Problem Analizi

Raporun bu bölümünde, bir önceki 1.3. Problem Durumu bölümünde açıklanan problemlerin ScheduleX uygulamasında gerektirdiği ihtiyaçların analizi yapılacaktır.

Web uygulaması olarak geliştirilecek olan ScheduleX, ders programı hazırlama işlemini otomatize ederek derslik ve sınıfları optimum şekilde kullanmayı ve bu derslik-sınıf yönetimi sürecini hızlandırmayı amaçlar. Bu süreçte bölüm sınıfı öncelikleri, kapasiteye göre yerleştirme, öğretim üyesi müsaitlik durumu gibi birçok problemin çözülmesi gerekmektedir. Aşağıda bu problemler için farklı açılardan yapılan analizler maddeler halinde açıklanmıştır.

1.4.1. Derslikler Bakımından Analiz

İnsan eliyle yapılan ders programlarında programı hazırlayan kişinin, dersleri yerleştirebileceği derslikler hakkında bilgi sahibi olmak için tüm derslikleri gezerek tek tek bilgileri toplaması gerekmektedir. ScheduleX, bu bilgileri kayıt altına alarak hazırlama işine bakan kişi değişse bile tekrar sınıf keşfi yapma sürecini ortadan kaldırmış olur.

Bir derslik için toplanması gereken temel bilgiler ise şunlardır:

Kapasite Bilgisi: Her dersliğin kaç öğrenci kapasiteli olduğu bilindiğinde, dersler uygun kapasiteli dersliklere yerleştirilebilir ve sandalye arama sorunu ortadan kalkar.

Bölüm/Blok Bilgisi: Dersler bölümler tarafından verildiğinden bölüme ait derslerin bölüm dersliklerinde veya mümkünse bölüme en yakın dersliklerde olması gerekir, dersler için mesafe kat etmek yorucu olabilir.

Teknolojik İmkanlar: Uygulamalı derslerin daha efektif işlenebilmesi için uygulama araç-gereçlerinin bulunduğu dersliklerde dersler yapılmalıdır.

Bu özellikler istenirse çoğaltılabilir. Örneğin engelli öğrencilerin rahat ulaşabileceği derslikler, dersliğin ısıtma/soğutma ve havalandırma durumu, ışıklandırma durumu gibi özellikler ihtiyaca göre değerlendirilebilir. Bunun dışında sistem, dersliklere ait bir ders programı oluşturarak dersliklerin müsaitlik durumunun daha kolay izlenebilmesini sağlamalıdır.

1.4.2. Dersler Bakımından Analiz

Ders programının ana ögesi olan derslerin birçok temel özelliği vardır.

Ders Saati: Her dersin belirli bir süresi vardır. Belirlenen sürelere göre dersler programda yerlerini alır. Derslikler ve öğretim üyelerinin zaman yönetimleri için gereklidir.

Verildiği Bölüm: Derslik seçiminde öncelik belirleyici etkenlerden biridir. Dersler bölümler tarafından açıldığı için bu bilgi gereklidir.

Dersi Alan Öğrenci Sayısı: Derslerin dersliklere yerleştirilebilmesi için dersi alan öğrenci sayısının bilinmesi gereklidir.

Ders Kategorileri: Teori, uygulama gibi ayrımların yapılması dersler için önemlidir. Derse uygun araç gereçlerin bulunduğu dersliklere atama yapılmasını sağlar.

Dersi Veren Öğretim Üyesi: Zaman çizelgesinde yerleştirme yapılabilmesi için gerekli bilgidir.

Derslik programlarının ayarlanması, öğretim üyesi programlarının ayarlanması, derslik doluluk takibi gibi temel özelliklerin kullanılabilmesi için veri tabanı tasarımındaki dersler tablosu sütun atlanmadan eksiksiz biçimde tasarlanmalıdır. Ders bilgileri ekranlarının tasarımı da ScheduleX sisteminin kullanım kolaylığı için kritik önemdedir.

1.4.3. Eşleştirme İşlemleri Bakımından Analiz

Ders programı oluşturmak için yapılması gereken derslerin dersliklere yerleştirilerek ders zamanlarının gün ve saat olarak belirlenmesi gerekmektedir.

Bu işlemler yapılırken göz önüne alınması gereken birçok kriter vardır. Bunlardan bazıları şu şekildedir:

Zaman Çakışmaları

- Aynı öğretim üyesinin aynı saatte birden farklı ders veremeyeceği göz önünde bulundurulmalıdır.
- Aynı derslikte birden fazla ders aynı anda olamaz.
- Öğretim üyelerinin kesin olarak okulda bulunamadığı zamanlarda dersi olmaması gerekir.

Ders-Derslik Uyumu

- Uygulama dersleri uygun ekipmanların bulunduğu dersliklerde yapılmalıdır.
- Derslik kapasitesi dersi alan öğrenci sayısı ile uyumlu olmalıdır.
- Dersler verildiği bölümlerdeki veya bölümüne en yakın dersliklere verilmelidir.

Kriterler özel ihtiyaçlara göre çoğaltılabilir. Bu kriterleri karşılamak için çeşitli algoritmalar ScheduleX uygulamasında kullanılabilir. Öncelikler için kısıt programlama algoritmaları, acil ihtiyaçları önce karşılamak için greedy algoritmalar gibi seçenekler mevcuttur.

1.4.4. Öğretim Üyeleri Bakımından Analiz

Öğretim üyelerinin sorunlarını analiz ettiğimizde en öne çıkan başlık müsaitlik durumudur. Öğretim üyelerinin okul dışında başka görevleri de vardır. Halihazırda yürüttüğü bir araştırma projesi olabilir, idari bakımdan başka görevleri olabilir şeklinde örnekler çoğaltılabilir. Öğretim üyelerine ders atanırken bu zamanların sistemde işaretlenerek o saatlerde ders vermemesi sağlanmalıdır. Zorunlu olmayan öğretim üyesi istekleri de öncelik sistemine tabi tutulmalıdır.

1.4.5. Ders Programı Hazırlayanlar Bakımından Analiz

ScheduleX sisteminin temel kullanıcıları program hazırlamakla görevli kişilerdir. Bu kişilerin sistemdeki yetkilerinin belirlenmesi sistem güvenliğini sağlamak için önemlidir. Eğer program hazırlayan kişiler sistem ayarlarına karışırsa sistemde beklenmedik sıkıntılara yok açabilir. Bu sebeple sistem ayarlarına bakan bir yönetici grubunun oluşturulması uygun görünüyor. Ayrıca programın neden oluşturulamadığı, hangi kısıt ve önceliklerin ders yerleşimine engel olduğunun geri bildirimi program hazırlayan kişinin önüne düşmelidir. Bu sayede programda gerekli değişiklikler yapılabilir.

1.4.6. Öncelik ve Kısıtlamalar Bakımından Analiz

ScheduleX uygulaması içerisinde önceki alt başlıklarda da yer yer belirtildiği gibi birçok önceliklendirme ve kısıtlama işlemi bulunmaktadır. Optimum bir program oluşturulabilmesi için her bir öncelik ve kısıtın uygunluğu kritik önem taşır. Bu uygunlukların tanımlanması, olası çözümlerin test edilmesi, çelişkiye sebep olan durumların tespitini ve çözümlerini bulabilmek için kısıt programlama algoritmalarının kullanılması gerekli gözüküyor. Bu algoritmalar aynı zamanda esnek çözümlere de sahiptir.

1.5. Çözüm Önerisi

1.3. Problem Durumu başlığında özetlenen sorunları çözmek amacıyla geliştirilen ScheduleX, ders programı oluşturma sürecini insan müdahalesinden alarak süreci tamamen optimize etmeyi ve daha verimli hale getirmeyi hedeflemektedir. ScheduleX; tüm ders, sınıf ve kaynak bilgilerini dikkate alarak en uygun ders-sınıf eşleşmelerini, kapasite sınırlamalarını, bölüm önceliklerini, hoca kısıtlamalarını ve diğer özel koşulları göz önünde bulundurur. Böylece, kullanıcı dostu bir deneyim sunarak hataları azaltır ve daha etkili bir ders programı süreci sağlar. Aşağıda çözümde öngörülen ana işlevler ve bu işlevlerin nasıl işleyeceği açıklanmıştır.

1.5.1. Sınıf Bilgilerinin Tanımlanması ve Kayıt Altına Alınması

Sınıfların doğru şekilde tanımlanması ve kayıt altına alınması, kullanıcıların ders programı sürecinde uygulama ile etkileşime girdikleri ilk noktadır. ScheduleX, her bir sınıfın bölüm bilgisini, kapasitesini ve fiziksel özelliklerini sisteme kaydederek, sınıfların uygun şekilde yönetilmesini sağlar. Bu kapsamda, her sınıf, benzersiz bir kod ile (örneğin M101) tanımlanır ve hangi bölüme ait olduğu bilgisiyle sistemde yer alır. Sınıf kodları ve ait oldukları bölümler sayesinde, ders atamalarında belirli bölümlere öncelik verilmesi gerektiğinde sınıflar hızlıca filtrelenebilir. Bu özellik, ders programı oluşturma sürecinde büyük bir kolaylık sağlarken, hata oranını da minimize eder.

Sınıfların kapasite bilgisi, dersin öğrenci sayısına göre en uygun sınıfın seçilmesinde kilit rol oynar. ScheduleX, her sınıf için öğrenci kapasitesini kaydederek, dersin öğrenci sayısına göre yeterli büyüklükte sınıf ataması yapılmasına olanak tanır. Bu, özellikle kalabalık dersler için doğru sınıfın atanmasını sağlamak adına oldukça önemlidir. Sistem, büyük kapasiteli sınıfları büyük öğrenci gruplarına atayarak, öğrencilerin rahatça eğitim alabilecekleri bir ortam sunmayı amaçlar. Bu ayrıca, küçük sınıfların gereksiz yere büyük ders gruplarına tahsis edilmesini de önler.

Ek olarak, sınıfların blok bilgileri de sistemde kayıt altına alınır. Bu bilgi, ders yerleşiminde coğrafi yakınlık faktörünü dikkate alarak, öğrenciler ve öğretim görevlileri için daha erişilebilir bir ders programı oluşturmayı sağlar. ScheduleX, ana blokta uygun sınıf bulunamadığında komşu bloklara geçiş yaparak sınıf arama sürecini hızlandırır. Ayrıca, engelli öğrenci erişimi gerektiren durumlarda, sınıfların erişilebilirlik bilgisi de kayıt altına alınır. Böylece, erişim gereksinimi olan öğrenciler için uygun sınıflar öncelikli olarak seçilir, erişilebilirlik ihtiyacı olan öğrencilere uygun bir öğrenim ortamı sunulur.

1.5.2. Ders Bilgilerinin Tanımlanması ve Eşleştirilmesi

ScheduleX, her dersin detaylı olarak tanımlanmasını ve doğru sınıflarla eşleştirilmesini sağlayarak etkin bir ders programı oluşturur. Bu süreçte, dersler benzersiz bir kod ile sisteme kaydedilir ve ait oldukları bölüm bilgisiyle birlikte tanımlanır. Ders kodları ve bölüm bilgileri, ders programı oluşturulurken derslerin hangi bölüme ait olduğunu hızlıca görmeyi sağlar. Ayrıca, dersler arasındaki öncelik sıralaması da dikkate alınarak, bölümler arası ders atamalarında bölüm önceliklerine göre yerleştirme yapılır. Böylelikle, bölüm dersleri öncelikli olarak kendi bölümlerine yerleştirilir ve bölüm öğrencileri için kolaylık sağlanır.

Dersleri alan öğrenci sayıları da sisteme kaydedilerek sınıf seçimi sırasında kapasite uyumu göz önünde bulundurulur. Öğrenci sayısının fazla olduğu dersler için geniş kapasiteli sınıflar tercih edilerek, öğrencilerin rahatça ders görebileceği bir ortam sağlanır. Ayrıca, her ders için belirli bir öncelik sırası tanımlanır. Örneğin, zorunlu bölüm dersleri yüksek öncelikli olarak sınıflandırılırken, seçmeli dersler daha düşük öncelik taşıyabilir. Bu yapı, sınırlı sayıdaki büyük sınıfların öncelikli derslere tahsis edilmesini sağlar ve kapasiteye uygun bir yerleştirme yapılmasını kolaylaştırır.

Ayrıca, her dersin günü ve saatleri de sistemde tanımlanır. Ders saatleri tanımlanırken, diğer derslerle çakışmayı önlemek amacıyla ders başlangıç ve bitiş saatleri dikkatle belirlenir. Ardışık dersler arasında yeterli süre bırakılarak öğrenci ve öğretim görevlilerinin bir ders bitiminden diğerine geçiş yapması kolaylaştırılır. Bu bilgiler, derslerin yerleştirilmesi sırasında çakışma riskini ortadan kaldırarak daha düzenli bir program oluşturulmasına olanak tanır.

1.5.3. Yerleşim Kuralları ve Öncelik Yapısı

ScheduleX, sınıf atama sürecinde yerleşim kurallarını ve öncelik yapılarını göz önünde bulundurarak daha etkin bir program sunar. Bu yerleşim kuralları, öğrenci kapasitesi ile sınıf kapasitesi uyumuna, bölüm önceliklerine ve bloklar arası yakınlığa göre belirlenir. Sistemin temel amacı, öğrenci sayısı ile sınıf

kapasitesini en uygun şekilde eşleştirerek, derslerin uygun sınıflarda yapılmasını sağlamaktır. Örneğin, büyük öğrenci grupları için geniş kapasiteli sınıflar atanarak, dar sınıflarda öğrenci yoğunluğu oluşması engellenir. Aynı zamanda, az sayıda öğrenciye sahip dersler için küçük kapasiteli sınıfların tercih edilmesi, kaynak israfını önler.

Bölüm öncelikleri de yerleşim sürecinde dikkate alınan önemli bir faktördür. ScheduleX, bölüm derslerini öncelikli olarak kendi bölümlerine tahsis eder, böylece bölüm öğrencilerinin kendi binalarında eğitim alması sağlanır. Bu yöntem, özellikle kampüs içinde birden fazla bina veya blok bulunması durumunda büyük bir kolaylık sağlar. Eğer dersin yapılacağı blokta uygun sınıf bulunamazsa, sistem yakın blokları otomatik olarak değerlendirir. Örneğin, ana blok olarak belirlenen bir bina uygun sınıf sağlayamıyorsa, komşu bloklarda yer alan sınıflar değerlendirmeye alınır.

Laboratuvar gerektiren derslerde, dersin uygulamalı olup olmadığı bilgisi dikkate alınarak uygun sınıflar seçilir. Örneğin, bilgisayar laboratuvarı veya kimya laboratuvarı gibi özel alanlara ihtiyaç duyulan derslerde, sistem bu tür laboratuvarlara öncelik tanır. Online dersler için fiziksel bir sınıfa ihtiyaç duyulmadığından, online dersler programda ayrı bir kategoriye alınır ve fiziksel sınıf gerektiren dersler için yer açılır. Bu esneklik, programın hem ihtiyaçlara uygun hem de kaynak kullanımını en verimli hale getirir.

1.5.4. Rol Yönetimi ve Kullanıcı Yetkilendirmesi

Ders programı hazırlama sürecine dahil olan kullanıcıların rol ve yetkilerinin doğru belirlenmesi, sürecin güvenli ve düzenli bir şekilde yönetilmesini sağlar. ScheduleX, farklı kullanıcılar için çeşitli rol tanımları yaparak yetkili kişilerin sisteme erişimini sınırlar. Örneğin, editör rolündeki kullanıcılar ders programını düzenleme yetkisine sahiptir ve sınıf-ders eşleştirmelerini yapabilir. Bu rol, derslerin sınıflara atanması sürecinde belirli kişilerin yetki sahibi olmasını sağlar ve sürecin karışıklığını önler.

Yönetici rolü ise sistemin genel ayarlarını ve kullanıcı yönetimini yapar. Yönetici, sistemdeki tüm verileri düzenleme, yeni kullanıcılar ekleme veya mevcut kullanıcıları silme yetkisine sahiptir. Bu sayede, sistemin genel düzeni korunur ve yalnızca yetkili kişilerin sisteme erişimi sağlanır. Kullanıcı rolleri arasındaki bu yetki farkları, ders programı sürecinin kontrollü bir şekilde işlemesine katkı sağlar. Örneğin, yalnızca editör rolüne sahip bir kullanıcı, sınıf bilgilerini düzenleyebilir ve dersleri sınıflara atayabilirken, diğer kullanıcılar bu verilere yalnızca göz atabilir.

1.5.5. Kısıtlama ve Talep Yönetimi

Derslerin yerleştirilmesinde öğretim görevlilerinin özel taleplerinin ve kısıtlamalarının dikkate alınması önemlidir. Örneğin, belirli bir gün veya saat diliminde ders veremeyecek olan bir öğretim görevlisi, bu bilgiyi sisteme ileterek o gün ve saat için ders atanmasını engelleyebilir. ScheduleX, bu tür talepleri sisteme kaydederek, öğretim görevlilerinin uygun zaman dilimlerine göre ders planlaması yapar. Bu özellik, özellikle birden fazla kuruma bağlı olarak çalışan veya kampüs dışında görevleri bulunan öğretim görevlileri için büyük bir kolaylık sağlar.

Uygulamalı derslerde laboratuvar gibi özel alan gereksinimleri de kısıtlama olarak belirlenir. Bu durumda, dersin gerçekleştirilebilmesi için uygun laboratuvar bulunana kadar dersin yerleştirilmesi ertelenir veya başka bir sınıf seçeneği değerlendirilir. Örneğin, kimya veya bilgisayar derslerinde özel laboratuvar gereksinimi, yalnızca uygun laboratuvarların yerleşimde değerlendirilmesini sağlar. Bu kısıtlamalar, hem öğretim görevlilerinin taleplerine uygun ders programı oluşturulmasını hem de derslerin doğru alanlarda yapılmasını sağlar.

1.5.6. Zamanlama, Ders Süreleri ve Aralıklar

ScheduleX, derslerin gün boyunca düzenli ve akışkan bir şekilde planlanmasını sağlamak için, ders sürelerini, başlangıç ve bitiş saatlerini özenle düzenler. Sistem, ardışık dersler arasında yeterli geçiş süreleri bırakarak öğrencilerin ve öğretim görevlilerinin derslikler arasında rahatça geçiş yapmalarını sağlar. Bu geçiş süreleri, dersin gerçekleştirildiği binaya ve iki ders arasındaki mesafeye göre ayarlanabilir; böylece kampüs içinde farklı bloklarda gerçekleşen dersler arasında bile zaman baskısı olmadan geçiş yapılabilir. Örneğin, M Blok'ta gerçekleştirilen bir dersin ardından öğrencinin L Blok'a gitmesi gerektiğinde, sistem bu mesafeyi dikkate alarak yeterli geçiş süresi bırakır. Bu durum, özellikle büyük kampüslerde veya farklı binalarda yapılan derslerde öğrencilerin ve hocaların günlük programlarını daha düzenli yönetmelerine katkıda bulunur.

ScheduleX, aynı zamanda derslerin süresini ders türüne ve içeriğine göre ayarlayarak uygun aralıklarla mola verilmesine olanak tanır. Örneğin, üç saat sürecek bir uygulamalı ders için belirli aralıklarla mola planlanabilir, bu da öğrencilerin dikkat sürelerini korumalarını sağlar. Bu tür düzenlemeler, sadece öğrencilerin derslerdeki verimliliğini artırmakla kalmaz, aynı zamanda öğretim görevlilerinin de dersleri daha verimli bir şekilde işleyebilmelerine yardımcı olur.

1.5.7. Geri Bildirim Mekanizması

ScheduleX, kullanıcıya en uygun ders programını oluşturmada yol gösterici bir geri bildirim mekanizması sunar. Sistem, ders yerleştirilemediğinde veya belirli sınıf-ders eşleştirmelerinde sorun yaşandığında, bunun nedenini kullanıcıya ayrıntılı olarak bildirir. Bu geri bildirimler, dersin yerleştirilememesine sebep olan etkenleri belirtir; örneğin, sınıf kapasitesinin yetersiz olması, dersin zaman aralığı ile çakışan başka bir ders olması veya öğretim görevlisinin kısıtlamaları. Böylece kullanıcı, hangi dersin hangi nedenle programa eklenemediğini net olarak görebilir ve gerekli düzenlemeleri yapabilir.

Geri bildirim mekanizması yalnızca sınıf uygunluğu veya kapasite gibi temel unsurlar hakkında bilgi vermekle kalmaz; aynı zamanda öğretim görevlilerinin talepleri veya uygulamalı derslerin laboratuvar gereksinimleri gibi daha özel durumlar hakkında da ayrıntılı bilgi sağlar. Örneğin, sistem bir dersin laboratuvar gereksinimi nedeniyle uygun bir sınıfa atanamadığını belirttiğinde, kullanıcı laboratuvar yerleşiminde değişiklik yapabilir veya dersin gün ve saatinde esnekliğe giderek sorunu çözebilir. Bu ayrıntılı bildirimler sayesinde, kullanıcı programı gözden geçirerek gerekli uyarlamaları hızlıca yapabilir.

Sistem ayrıca, kullanıcıları belirli sınıfların erişilebilirlik durumu hakkında bilgilendirir. Özellikle engelli öğrenciler için uygun sınıf veya binaların bulunamaması durumunda, programlama sürecine bu geri bildirimler dahil edilerek öğrencilerin gereksinimleri göz önünde bulundurulur. Böylece, öğrenciler için erişilebilir bir eğitim ortamı sağlanır ve kullanıcı, sınıflar arasında engelli erişim bilgisine göre yer değişikliği yapabilir.

1.6. Kullanılacak Teknolojiler

Raporun bu kısmında ScheduleX ders programı hazırlama sisteminin geliştirme aşamalarında kullanılacak yazılım dilleri, kütüphanelere, yazılım çerçeveleri ve araçlara açıklamaları ile yer verilmiştir.

Yazılım Dilleri:

1) **JavaScript:** ScheduleX uygulamasınındaki fonksiyonel gereksinimleri gerçekleştirmek için kullanılacaktır.
2) **CSS:** ScheduleX uygulamasındaki görsel bileşenleri stillendirmek için uygulanacaktır.
3) **HTML:** ScheduleX uygulamasındaki görsel bileşenleri oluşturmak için kullanılacatır.

Kütüphaneler:

1) **React:** ScheduleX uygulamasının görsel arayüzünün geliştirilmesinde kullanılacaktır, kullanıcı arayüzü bileşenlerinin oluşturulması için kullanılacaktır.
2) **Tailwind CSS:** ScheduleX uygulamasında, kullanıcı arayüzünü tasarlamak için kullanılacaktır.
3) **Sequelize:** ScheduleX uygulamasında veritabanı işlemlerinin yönetilmesi ve nesne ilişkilendirmesi için kullanılacaktır.

Diğer Araçlar

1) **Sqlite:** ScheduleX uygulamasında veritabanı yönetimi için kullanılacaktır; verilerin kullanıcının cihazında depolanması ve işlenmesi için kullanılacaktır. Sqlite veritabanının en önemli özelliklerinden birisi kullanıcıların dışarından bir veritabanı kurmasına gerek kalmadan uygulama ile paket bir şekilde gelmesidir.
2) **npm**: ScheduleX projesinin yapılandırılması ve bağımlılıklarının yönetimi için kullanılacaktır; proje yapılandırması ve derleme işlemlerinde kullanılacaktır.
3) **GitHub:** ScheduleX projesinin kod yönetimi ve sürüm kontrolü için kullanılacaktır; kod paylaşımı, iş birliği ve değişiklik takibi için kullanılacaktır.
4) **Jira:** ScheduleX projesinin proje yönetimi ve görev takibi için kullanılacaktır; proje planlaması, görev atama ve takvimleme için kullanılacaktır.

2. Veri Tabanı Gereksinimleri

2.1. Fonksiyonel Gereksinimler

Gereksinim	Öncelik (1-5)	Açıklama
DB-F1	4	Veri tabanındaki her kullanıcı, ders, derslik, blok, bölüm ve özel durum benzersiz bir numaraya sahip olmalıdır.
DB-F2	4	Veri tabanı kullanıcıların giriş bilgilerini depolamalıdır.
DB-F3	4	Veri tabanı mevcut bloğun derslik kapasitelerini kayıt altında tutmalıdır.
DB-F4	4	Veri tabanı her derse ait öncelik numarasını (1-5) tutmalıdır.
DB-F5	2	Veri tabanı derslerin fiziksel veya online olduğu bilgisini tutmalıdır.
DB-F6	2	Veri tabanı derslerin teori mi laboratuvar mı olduğunun bilgisini tutmalıdır.
DB-F7	2	Veri tabanı derslerin ne zaman oluşturulduğunun bilgisini tutmalıdır.
DB-F8	4	Veri tabanı derslerin hangi kullanıcı tarafından oluşturulduğu bilgisini tutmalıdır.

DB-F9	5	Veri tabanı derslerin saat aralıklarının bilgisini tutmalıdır.
DB-F10	1	Veri tabanı dersliklerin engelli öğrencilere uygun olup olmadığı bilgisini tutmalıdır.
DB-F11	5	Veri tabanı dersliklerin hangi bloklarda bulunduğunun verisini tutmalıdır.
DB-F12	4	Veri tabanı dersliklerin laboratuvar uygulamalarına uygun olup olmadığını tutmalıdır.
DB-F13	5	Veri tabanı dersin öğretim üyesi bilgisini tutmalıdır.
DB-F14	5	Veri tabanı dersi alan öğrenci sayısını tutmalıdır.
DB-F15	3	Veri tabanı, blokların diğer bloklara göre öncelik sıralamasını tutmalıdır.
DB-F16	5	Veri tabanı her dersliğin öğrenci kapasitesini tutmalıdır.
DB-F17	5	Veri tabanı, dersliklerin kapasitesinde ±5 öğrenciye kadar hata payını desteklemelidir.
DB-F18	3	Veri tabanı, öğretim üyelerinin kişisel bilgilerini tutmalıdır.
DB-F19	5	Veri tabanı, öğretim üyelerinin hangi dersleri verdiğinin bilgisini tutmalıdır.
DB-F20	3	Veri tabanı, öğretim üyelerinin özel durumlarını (örneğin, belirli günlerde ders verememe gibi kısıtlamalar) saklamalıdır.
DB-F21	4	Veri tabanı bölümlerin açtığı derslerin bilgisini tutmalıdır.
DB-F22	2	Veri tabanı bölümlerin yerleşik olduğu blokların bilgisini tutmalıdır.

Tablo 1: Fonksiyonel Gereksinimler Tablosu

2.2. Fonksiyonel Olmayan Gereksinimler

Gereksinim	Öncelik	Açıklama
DB-NF1	4	Veri tabanı tablolarından veri okunurken gecikme <= 1 milisaniye olmalıdır.
DB-NF2	4	Veri tabanına veri kaydedilirken gecikme <= 1 milisaniye olmalıdır.
DB-NF3	5	Veri tabanının çökmesi durumunda çökme nedenleri bir hata defterine (error log) kaydedilmelidir.
DB-NF4	4	Veri tabanı eşzamanlı olarak en az 50 kullanıcıya hizmet verebilmelidir.
DB-NF5	4	Sistem maksimum 10 saniye içerisinde hazırlanmış ders programını sunabilmelidir.

DB-NF6	5	Veri tabanındaki tüm veriler TDE teknolojisi ile şifrelenmelidir.
DB-NF7	5	Veri tabanında SQL injection ve XSS saldırılarını engellemek için hazırlanmış sorgular kullanılmalıdır.
DB-NF8	4	Veri tabanı en az 10.000 öğrenci, 1000 ders, 1000 derslik, 100 blok ve 50 bölüm kaydını yönetebilecek kapasitede olmalıdır.
DB-NF9	3	Beklenmeyen hatalar ile karşılaşıldığında sistem minimum 5 saniye içerisinde tekrar kullanıma açık olmalı, olamıyorsa kullanıcı bilgilendirilmelidir.
DB-NF10	2	Sistem ders planlarken ülke, bölge ve üniversite genelindeki tatilleri dikkate almalıdır.
DB-NF11	5	Veri tabanı 24 saatte bir olarak yedeklenmelidir.
DB-NF12	5	Veri tabanı ileride yeni eklenebilecek kullanıcı, derslik, blok, ders ve bölümler için genişletilebilir tasarlanmalıdır.
DB-NF13	5	Sistem kritik durumlarla karşılaşıldığında bir önceki tutarlı konumuna geri dönebilmelidir.
DB-NF14	2	Veri tabanı farklı işletim sistemleri (cross-platform) ile uyumlu olarak çalışabilmelidir.
DB-NF15	3	Kullanıcı dostu arayüz geliştirilmeli, sisteme ilk defa giriş yapacak kullanıcının fonksiyonları araçları ilk kullanımda tanıması sağlanmalıdır.
DB-NF16	5	Sistem, yeni özelliklerin eklenmesini kolaylaştıracak şekilde modüler olmalıdır.
DB-NF17	1	Sistem dokümantasyonu, kullanıcılara ve teknik ekiplere hitap edilecek şekilde oluşturulmalıdır.
DB-NF18	5	Veri tabanı kayıt ve işlemleri KVKK kanununa uygun olarak düzenlenmeli ve şifrelenmelidir.

Tablo 2: Fonksiyonel Olmayan Gereksinimler Tablosu

3. Veri Tabanı Tasarımı

3.1. Kavramsal Tasarım

3.1.1. Varlık Tanımı

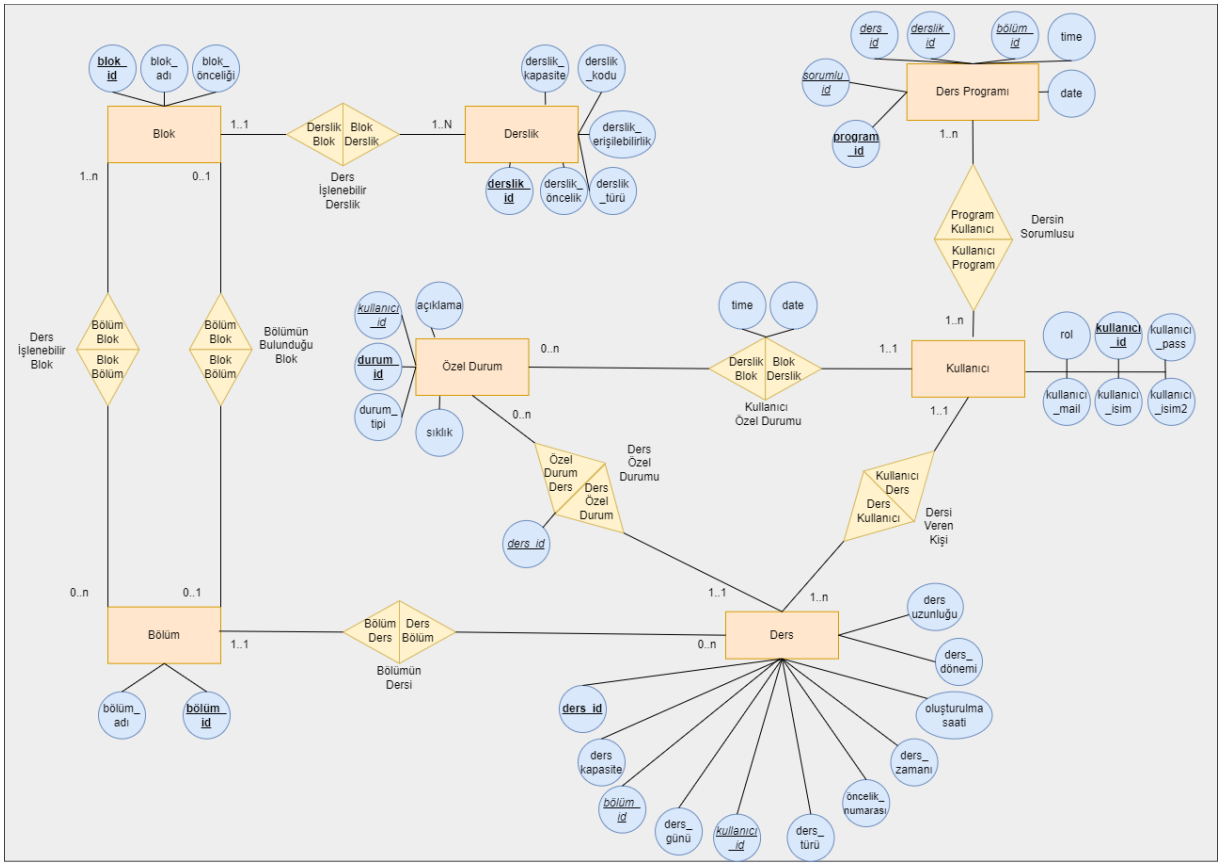
Varlık	Gereksinimler	Açıklama
Derslik	DB-F1, DB-F3, DB-F10, DB-F11, DB-F12, DB-F16, DB-F17, DB-NF12	Bu varlık, öğretim üyelerinin derslerini gerçekleştirdiği fiziksel alanları ifade etmektedir.

Ders	DB-F1, DB-F4, DB-F5, DB-F6, DB-F7, DB-F9, DB-F13, DB-F14, DB-F19, DB-F21, DB-NF12	Bu varlık, öğrenciler ve öğretim üyeleri tarafından belirli bir takvim dahilinde işlenen akademik içerikleri ifade etmektedir.
Blok	DB-F1, DB-F3, DB-F11, DB-F15, DB-F22, DB-NF12	Bu varlık, dersliklerin ve bölümlerin yer aldığı fiziksel binaları temsil etmektedir.
Kullanıcı	DB-F1, DB-F2, DB-F8, DB-F13, DB-F18, DB-F19, DB-NF4, DB-NF12, DB-NF15	Bu varlık sisteme giriş yapan, ders programını görüntüleyebilen ve/veya değiştirebilen kullanıcı türlerini kapsar.
Ders Programı	DB-F1, DB-F5, DB-F6, DB-F9, DB-F11, DB-F13, DB-F19	Bu varlık; dersliklerin, derslerin ve kullanıcıların zaman planlamasını içermektedir.
Bölüm	DB-F1, DB-NF12	Bu varlık, fakülte içerisinde dersleri, öğretim üyeleri ve asistanları bulunan akademik birimleri temsil etmektedir.
Özel Durum	DB-F9, DB-F10, DB-F13, DB-F19, DB-F20	Bu varlık, öğretim üyesi rolüne sahip kullanıcıların özel durumlarını temsil etmektedir.

Tablo 3: Varlık Türleri Tablosu

3.1.2. Varlık-İlişki (Entity-Relationship) Diyagramı

Bu bölümde Şekil 2'de ScheduleX ders programı hazırlama uygulamasına ait ER diyagramına yer verilmiştir.



Şekil 2: ScheduleX Veri Tabanı ER Diyagramı

3.1.3. Varlık Nitelik Açıklamaları

Bu bölümde Şekil 2'de verilen ER diyagramında bulunan varlıkların nitelikleri anlatılacaktır.

3.1.3.1. Derslik Varlığı

Derslik	
Nitelik	**Açıklama**
derslik_id	Bu nitelik her Derslik varlığına özel bir ID tutar.
derslik_kodu	Bu nitelik dersliğin kodunu tutar.
derslik_kapasite	Bu nitelik dersliğin kapasitesini tutar.
derslik_erişilebilirlik	Bu nitelik dersliğin erişebilirlik durumunu tutar.
derslik_öncelik	Bu nitelik dersliğin diğer dersliklere göre öncelik değerini tutar.
derslik_türü	Bu nitelik dersliğin ne tür bir derslik olduğunu tutar.

Tablo 4: Derslik Varlığı Nitelikleri

3.1.3.2. Ders Varlığı

Ders	
Nitelik	**Açıklama**
ders_id	Bu nitelik her Ders varlığına özel bir ID tutar.
ders_türü	Bu nitelik dersin türünü belirtir.
ders_kapasite	Bu nitelik derse kayıtlı öğrenci sayısını tutar.
bölüm_id	Bu nitelik dersi veren bölümün ID numarasını tutar.
kullanıcı_id	Bu nitelik dersi veren öğretim üyesinin ID numarasını tutar.
öncelik_numarası	Bu nitelik dersin zamanlama sırasındaki öncelik değerini tutar.
oluşturulma_saati	Bu nitelik dersin sistemde oluşturulduğu tarihi tutar.
ders_zamanı	Bu nitelik dersin zaman değerini tutar.
ders_uzunluğu	Bu nitelik uzunluğunu tutar.

Tablo 5: Ders Varlığı Nitelikleri

3.1.3.3. Blok Varlığı

Blok	
Nitelik	**Açıklama**
blok_id	Bu nitelik her Blok varlığına özel bir numarayı tutar.
blok_adı	Bu nitelik blok adını tutar.
blok_önceliği	Bu nitelik blokun diğer bloklara göre öncelik değerini tutar.

Tablo 6: Blok Varlığı Nitelikleri

3.1.3.4. Bölüm Varlığı

Bölüm	
Nitelik	**Açıklama**
bölüm_id	Bu nitelik her Bölüm varlığına özel bir ID tutar.
bölüm_adı	Bu nitelik bölüm adını tutar.

Tablo 7: Bölüm Varlığı Nitelikleri

3.1.3.5. Kullanıcı Varlığı

Kullanıcı Hesabı	
Nitelik	**Açıklama**
kullanıcı_id	Bu nitelik her Kullanıcı varlığına özel bir ID tutar
rol	Bu nitelik kullanıcının sistemdeki rolünü tutar.
kullanıcı_pass	Bu nitelik kullanıcı hesabının parolasını tutar.
kullanıcı_mail	Bu nitelik kullanıcı hesabına ait e-posta adresini tutar.
kullanıcı_isim	Bu nitelik kullanıcının ilk ismini tutar.
kullanıcı_isim2	Bu nitelik kullanıcının ikinci ismini tutar.

Tablo 8: Kullanıcı Varlığı Nitelikleri

3.1.3.6. Özel Durum Varlığı

Özel Durum	
Nitelik	**Açıklama**
durum_id	Bu nitelik her Özel Durum varlığına özel bir ID tutar.
durum_tipi	Bu nitelik özel durumun tipini tutar.
kullanıcı_id	Bu nitelik hangi kullanıcının bir özel durumu olduğunu tutar.
açıklama	Bu nitelik özel durumun açıklamasını tutar.
sıklık	Bu nitelik özel durumun gerçekleşme sıklığını tutar.

Tablo 9: Özel Durum Varlığı Nitelikleri

3.1.3.7. Ders Programı Varlığı

Ders Programı	
Nitelik	**Açıklama**
program_id	Bu nitelik her Ders Programı varlığına özel bir ID tutar.
ders_id	Bu nitelik ders programındaki derse ait ID numarasını tutar.
derslik_id	Bu nitelik ders programındaki dersin dersliğine ait ID numarasını tutar.
bölüm_id	Bu nitelik ders programındaki dersin bölümüne ait ID numarasını tutar.
sorumlu_id	Bu nitelik ders programındaki dersin öğretim üyesine ait ID numarasını tutar.
time	Bu nitelik ders programındaki dersin zamanını tutar.
date	Bu nitelik ders programında dersin gününü tutar.

Tablo 10: Ders Programı Varlığı Nitelikleri

3.2. Mantıksal Tasarım

3.2.1. Haritalama (Mapping)

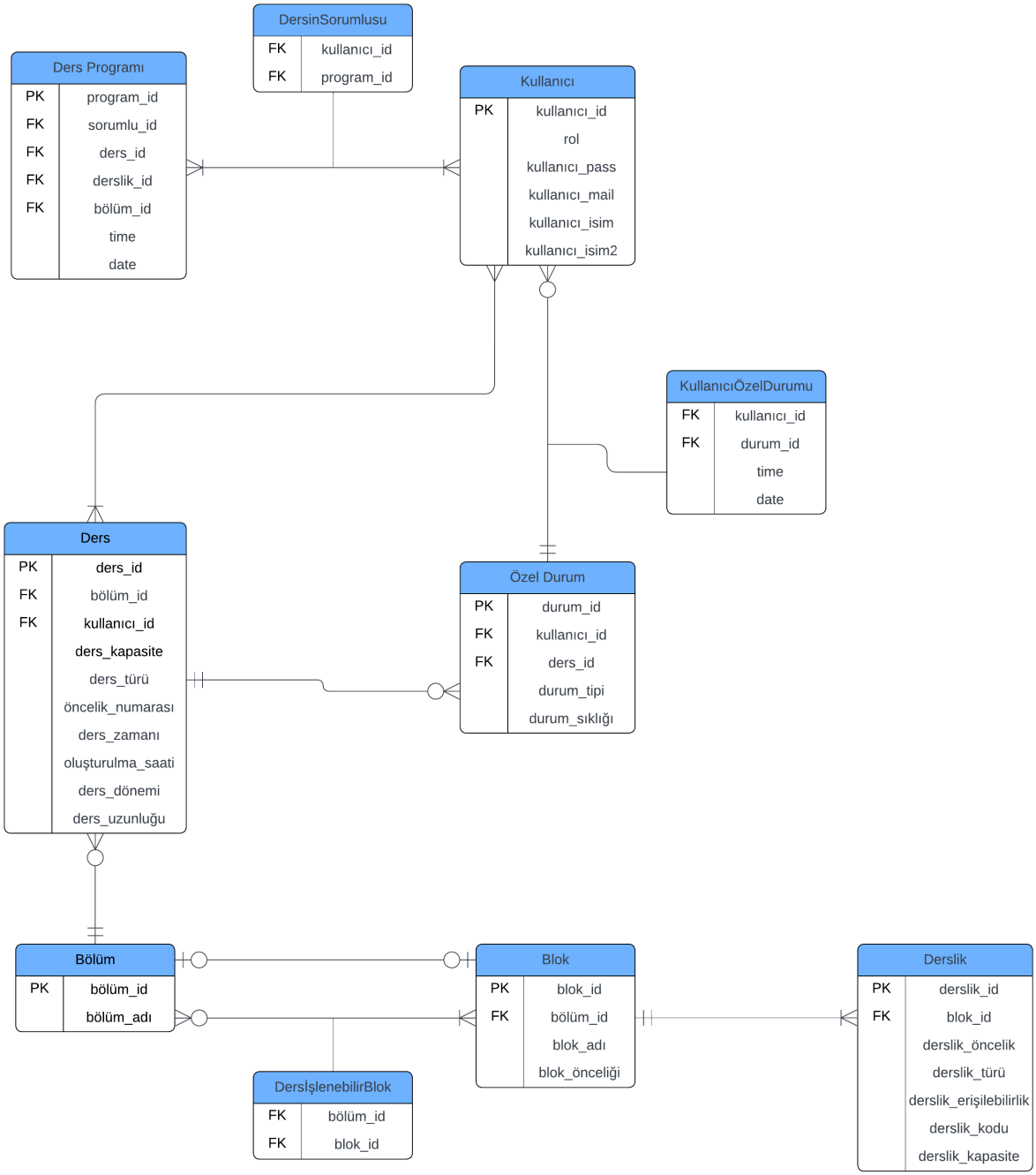
Bu bölümde ScheduleX sisteminin veri tabanı için haritalama (mapping) yapılacaktır.

- Kullanıcı(**PK** kullanıcı_id, rol, kullanıcı_pass, kullanıcı_mail, kullanıcı_isim, kullanıcı_isim2)
- Ders(**PK** ders_id, ders_kapasite, ***FK*** bölüm_id **REFERENCES Bölüm(**bölüm_id), ders_günü, ***FK*** kullanıcı_id **REFERENCES Kullanıcı**(kullanıcı_id), ders_türü, öncelik_numarası, ders_zamanı, oluşturulma_saati, ders_dönemi, ders_uzunluğu)
- Bölüm(**PK** bölüm_id, bölüm_adı)
- Blok(**PK** blok_id,***FK*** bölüm_id **REFERENCES Bölüm**(bölüm_id) blok_adı, blok_önceliği)
- Derslik(**PK** derslik_id,***FK*** blok_id **REFERENCES Blok**(blok_id), derslik_öncelik, derslik_türü, derslik_erişilebilirlik, derslik_kodu, derslik_kapasite)

- Özel Durum(**PK** durum_id, ***FK*** kullanıcı_id **REFERENCES Kullanıcı**(kullanıcı_id),***FK*** ders_id **REFERENCES Ders**(ders_id), durum_tipi, sıklık)
- Ders Programı(**PK** program_id, ***FK*** sorumlu_id **REFERENCES Kullanıcı**(kullanıcı_id), ***FK*** ders_id **REFERENCES Ders**(ders_id), ***FK*** derslik_id **REFERENCES Derslik**(derslik_id), ***FK*** bölüm_id **REFERENCES Bölüm**(bölüm_id), time, date)
- DersinSorumlusu(**COMPOSITE PK**(**PK** kullanici_id, **PK** program_id)), ***FK*** kullanici_id **REFERENCES** Kullanıcı(kullanıcı_id), ***FK*** program_id **REFERENCES** Ders Programı(program_id))
- DersİşlenebilirBlok(**COMPOSITE PK** (**PK** bölüm_id, **PK** blok_id)), ***FK*** bölüm_id **REFERENCES** Bölüm(bölüm_id), ***FK*** blok_id **REFERENCES** Blok(blok_id))
- KullanıcıÖzelDurumu(**COMPOSITE PK** (**PK** kullanici_id, **PK** durum_id)), ***FK*** kullanici_id **REFERENCES Kullanıcı**(kullanici_id), ***FK*** durum_id **REFERENCES Özel Durum**(durum_id), time, date)

3.2.2. Mantıksal İlişkiler

Bu bölümde Şekil 3'te ScheduleX ders programı hazırlama uygulamasına ait Mantıksal İlişkiler diyagramına yer verilmiştir.



Şekil 3: ScheduleX Veri Tabanı Mantıksal İlişkiler Diyagramı

3.3. Fiziksel Tasarım

3.3.1. Normalizasyon

Bu kısımda veri tabanı bileşenlerinin 1NF (1. Normal Form), 2NF (2. Normal Form) ve 3NF (3. Normal Form) uyumluluğu test edilecek ve uymayanlar gerekli forma uyarlanacaktır.

3.3.1.1. 1. Normal Form

Veri tabanı tablolarının 1.normal forma uyması için gereken düzenlemeler şu şekildedir:

- Aynı niteliğe sahip tüm değerler, tutarlı bir şekilde aynı veri tipinde tanımlanmalıdır.
- Tablodaki nitelik alanları, benzersiz ve ayırt edici özelliklere sahip olacak şekilde ayarlanmalıdır.
- Tüm nitelik değerleri, atomik bir yapıda olacak şekilde düzenlenmelidir.

Mapping üzerinden çıkarılan tabloların ilk halleri 1. normal forma uygundur.

3.3.1.2. 2. Normal Form

Veri tabanının 2. normal forma uyması için gerekenler aşağıdadır:

- Tüm tabloların 1. Normal Forma uyuyor olması gerekir.
- Veriler arasında kısmi fonksiyonel bağımlılık olmamalıdır. Örn: {x, y, z} **->** {t}

Özel Durum tablosundaki izin_sayısı niteliği, aynı tablodaki birincil anahtar olan durum_id anahtarıyla bir ilişki içerisinde değildi. **Kullanıcı İzin** adında yeni bir tablo oluşturuldu, böylece kullanıcı_id'nin(***Kullanıcı*** *tablosundaki* ***kullanıcı_id*** *birincil anahtarına referans eder)* birincil anahtar, izin_sayisi niteliğinin ise birincil olmayan anahtar niteliğinde bulunduğu bir tablo tanımlandı. Bu işlem, kısmi bağımlılıkları ortadan kaldırmak için yapıldı.

Özel Durum tablosundan kullanici_id *(**Kullanıcı** tablosundaki **kullanıcı_id** birincil anahtarına referans eder)* yabancı anahtarı çıkarıldı ve kısmi bağımlılıkları gidermek için yeni oluşturulan **Kullanıcı Özel Durum** tablosuna eklendi.

Ders tablosunda yer alan birincil anahtar olmayan nitelik özelliğine sahip ders_öncelik niteliği, aynı tabloda bulunan birleşik birincil anahtarlara kısmi bağımlıydı.

{ders_id, ~~bölüm_id~~, ~~ders_türü~~} -> {ders_öncelik}

Bu sorunu giderebilmek adına, ders_öncelik niteliğinin bağımlılığı olduğu tek birincil anahtar olan ders_id'yi yeni oluşturulan **Ders Öncelik** tablosunda birincil anahtar olarak, ders_öncelik niteliğini ise birincil olmayan anahtar olarak belirledik. Bu şekilde kısmi ilişkisi bulunan nitelikler bir tabloda birleştirildi ve bağlantı karışıklığı giderilmiş oldu.

Bu yapılan adımlarla tablolar, 2. normal forma uygun hale getirilmiştir.

3.3.1.3. 3. Normal Form

Veri tabanının 3. normal forma uyması için gerekenler aşağıdadır:

- Tüm tabloların 2. Normal Forma uyuyor olması gerekir.
- Tablodaki nitelikler arasında geçişli fonksiyonel bağımlılık olmamalıdır.
  (Örn. {**x**} -> {y} -> {z})

Bu koşullara göre veri tabanı tabloları karşılaştırıldığında **Ders Programı** tablosundaki sorumlu_id *(**Kullanıcı** tablosundaki **kullanıcı_id** birincil anahtarına referans eder)* yabancı anahtarının program_id anahtarı ile bağımlı ilişki içinde olduğu görülmektedir.

{composite} ->{program_id} -> {sorumlu_id}

Bu durumun önüne geçebilmek için Dersin Sorumlusu adında yeni bir tablo oluşturuldu ve program_id *(**Ders Programı** tablosundaki **program_id** birincil anahtarına referans etmektedir)* yabancı anahtarı ile kullanıcı_id *(**Kullanıcı** tablosundaki **kullanıcı_id** birincil anahtarına referans etmektedir)* yabancı anahtarı bir birleşik birincil anahtar oluşturacak şekilde birleştirildi. Bu durumda birbirine bağımlı ilişkiler tabloda gösterildi, nitelik fazlalığı engellendi.

Bu yapılan adımlarla tablolar, 3. normal forma uygun hale getirilmiştir.

Bütün bu normalizasyon aşamaları sonucunda oluşan tablolar Şekil 4'te gösterildiği gibidir.

KULLANICI

kullanici_id	rol	kullanici_pass	kullanici_mail	kullanici_isim	kullanici_soyisim
1	Öğretim Üyesi	********	example@gmail.com	Murat	Karakuş
2	Öğretim Üyesi	********	example@gmail.com	Rukiye	Kızıltepe
3	Asistan	********	example@gmail.com	Enes	Asana
4	Editör	********	example@gmail.com	Barış	Gecegör
5	Öğretim Üyesi	********	example@gmail.com	Eren	Akça

DERSLİK

derslik_kodu	blok_id	derslik_öncelik	derslik_türü	derslik_erişilebilirlik	derslik_kapasite
M101	BLK1	1	Sınıf	Uygun	40
D7	BLK2	1	Sınıf	Uygun Değil	45
LAB1	BLK3	2	Laboratuvar	Uygun Değil	55
LAB1	BLK4	2	Laboratuvar	Uygun Değil	40
H1	BLK5	4	Sınıf	Uygun	100

BLOK

blok_id	bölüm_id	blok_adı	blok_önceliği
BLK1	YMH	M	1
BLK2	YMH	K	2
BLK3	EEMH	J	3
BLK4	BLMH	L	4
BLK5	BLMH	G	5

DERS

ders_id	ders_türü	bölüm_id	kullanıcı_id	ders_kapasite	ders_zamanı	oluşturulma_tarihi	ders_uzunluğu
YMH339	Sınıf	YMH	1	60	40dk	12.02.2025	1
YMH339	Laboratuvar	YMH	1	80	80dk	13.03.2026	2
MAT101	Sınıf	EEMH	4	110	120dk	14.04.2027	3
MAT101	Sınıf	EEMH	4	110	120dk	14.04.2028	3
YMH219	Laboratuvar	BLMH	5	40	80dk	16.06.2029	2

DERS ÖNCELİK

ders_id	öncelik_numarası
YMH339	1
YMH345	2
YMH351	3
YMH337	4
YMH219	5

BÖLÜM

bölüm_id	bölüm_adı
YMH	Yazılım Mühendisliği
YZMH	Yapay Zeka ve Veri Mühendisliği
EEMH	Elektrik-Elektronik Mühendisliği
BLMH	Bilgisayar Mühendisliği
BIMH	Biyomedikal Mühendisliği

DERS PROGRAMI

program_id	ders_id	derslik_id	time	day
PROG1	YMH339	1	13.30	Pazartesi
PROG1	YMH345	2	15.00	Salı
PROG1	YMH351	3	09.00	Salı
PROG1	YMH337	4	13.30	Salı
PROG1	YMH219	5	09.30	Salı

ÖZEL DURUM

durum_id	durum_tipi	durum_açıklama
1	Acil	iş
2	Özel	staj
3	Özel	okul
4	Acil	iş
5	Özel	özel_durum

KULLANICI İZİN

kullanici_id	kullanıcı_izin_sayısı
1	1
2	3
3	2
4	6
5	4

DERSİN SORUMLUSU

kullanıcı_id	program_id
1	PROG1
1	PROG1
3	PROG2
4	PROG2
5	PROG2

KULLANICI ÖZEL DURUMU

durum_id	kullanıcı_id	start_time	end_time	day
1	1	13.30	15.00	pzt
2	2	10.00	12.30	salı
3	3	15.00	18.30	çarşamba
4	4	11.00	17.00	pzt
5	5	13.30	15.30	perşembe

DERS ÖZEL DURUMU

durum_id	ders_id	start_time	end_time	day
1	1	13.30	15.00	pzt
2	2	10.00	12.30	salı
3	3	15.00	18.30	çarşamba
4	4	11.00	17.00	pzt
5	5	13.30	15.30	perşembe

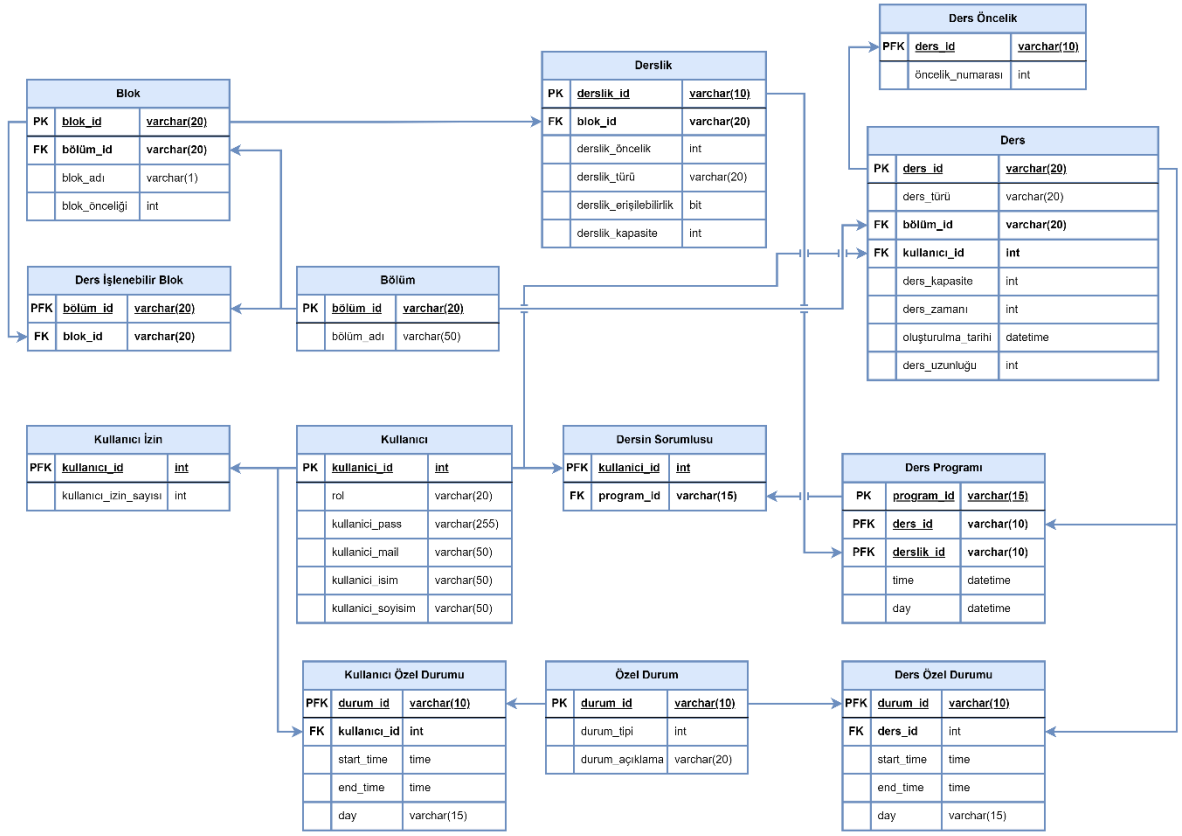
DERS İŞLENEBİLİR BLOK

bölüm_id	blok_id
YMH	BLK1
YMH	BLK1
BLMH	BLK2
EEMH	BLK2
EEMH	BLK3

Şekil 4: Normalizasyon Sonrası ScheduleX Tabloları

3.3.2. Fiziksel Veri Tabanı Tabloları

Bu bölümde projenin fiziksel tasarımı içerisinde fiziksel veri tabanı tabloları anlatılacaktır. Şekil 5'te tablolarda primary key ve foreign key ilişkileri görülebilir.



Şekil 5: Fiziksel Veri Tabanı Tabloları

3.3.2.1. Blok

- **blok_id (PK):** Blokların eşsiz olarak tanımlanabilmesi için primary key olarak tanımlanmıştır. Blokları ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi tanımlanmıştır.
- **bölüm_id (FK):** Blokların hangi bölüme ait olduğunu belirtmek ve bölüm varlığıyla ilişki kurabilmek için foreign key olarak tanımlanmıştır. Bölümleri ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi tanımlanmıştır.
- blok_adı: Blok isimlerinin kayıt edilmesi için tanımlanmıştır. Blok isimlerinin tek harfli olmasından dolayı varchar(1) veri tipi tanımlanmıştır.
- blok_önceliği: Dersleri öncelik sırasına göre belirleyebilmek için tanımlanmıştır. Sadece büyüklük kıyası yapılacağı için integer veri tipi tanımlanmıştır.

3.3.2.2. Derslik

- **derslik_id (PK):** Dersliklerin eşsiz olarak tanımlanabilmesi için primary key olarak tanımlanmıştır. Derslikleri ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi tanımlanmıştır.
- **blok_id (FK):** Dersliklerin hangi bloğa ait olduğunu belirtmek ve blok varlığıyla ilişki kurmak için foreign key olarak tanımlanmıştır. Blokları ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi tanımlanmıştır.
- derslik_öncelik: Dersliğin kullanım önceliğini belirtmek için tanımlanmıştır. Örneğin, dersliklerin rezervasyon sırasına göre önceliklendirilmesi sağlanabilir. Yalnızca sayısal değerler içereceği için int veri tipi kullanılmıştır.
- derslik_türü: Dersliğin türünü belirtmek için tanımlanmıştır. Tür bilgisinin metin olarak saklanabilmesi için varchar(20) veri tipi seçilmiştir. Maksimum 20 karakter uzunluğunda tanımlanmıştır.
- derslik_erişilebilirlik: Dersliğin erişilebilir olup olmadığını belirtmek için tanımlanmıştır. Bu alan, iki olası değer (0 veya 1) içerecek şekilde bit veri tipi kullanılarak tanımlanmıştır.
- derslik_kapasite: Dersliğin maksimum kapasitesini belirtmek için tanımlanmıştır. Kapasite sayısal bir değer olacağından int veri tipi kullanılmıştır.

3.3.2.3. Bölüm

- **bölüm_id (PK):** Bölümlerin eşsiz olarak tanımlanabilmesi için primary key olarak tanımlanmıştır. Aynı zamanda ders işlenebilir blok, blok ve ders varlıklarıyla arasında ilişki kurulmuştur. Bölümleri ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi tanımlanmıştır.
- bölüm_adı: Bölümün ismini belirtmek için tanımlanmıştır. Bölümün adının metin olarak saklanabilmesi için varchar veri tipi kullanılmıştır.

3.3.2.4. Ders

- **ders_id (PK):** Derslerin benzersiz olarak tanımlanabilmesi için primary key olarak tanımlanmıştır. Dersleri ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi kullanılmıştır.

- ders_türü: Dersin türünü belirtmek için tanımlanmıştır. Bu bilgi, dersin yapısına göre organizasyon ve planlama yapılmasına olanak tanır. Veri tipi varchar olarak seçilmiştir.
- **bölüm_id (FK):** Dersin hangi bölüme ait olduğunu göstermek için foreign key olarak tanımlanmıştır. Bölüm tablosuyla ilişki kurulmuştur. Derslerin bağlı olduğu bölümü ayırt edebilmek için varchar veri tipi kullanılmıştır.
- **kullanıcı_id (FK):** Dersi veren öğretim elemanı veya dersi alan öğrenci gibi kullanıcıların kimliklerini belirtmek için foreign key olarak tanımlanmıştır. Kullanıcı tablosuyla ilişkilidir. Bu bilgiyi tanımlamak için int veri tipi kullanılmıştır.
- ders_kapasite: Derse katılabilecek maksimum kişi sayısını belirtmek için tanımlanmıştır. Planlama ve yerleşim düzenlemeleri için önemlidir. Veri tipi int olarak belirlenmiştir.
- ders_zamanı: Dersin saat veya zaman aralığını belirtmek için tanımlanmıştır. Zaman yönetimi ve takvim planlaması yapılabilmesi için kullanılır. Veri tipi int olarak belirlenmiştir.
- oluşturulma_tarihi: Dersin sisteme eklendiği veya tanımlandığı tarihi belirtmek için tanımlanmıştır. Veri tipi datetime olarak seçilmiş olup, tarih ve saat bilgisini içerecek şekilde tasarlanmıştır.
- ders_uzunluğu: Dersin ne kadar süreceğini (dakika veya saat cinsinden) belirtmek için tanımlanmıştır. Programlama ve zaman yönetimi için kullanılır. Veri tipi int olarak belirlenmiştir.

3.3.2.5. Ders Programı

- **program_id(PK):** Programların benzersiz olarak tanımlanabilmesi için primary key olarak tanımlanmıştır. Programları ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi kullanılmıştır.
- **ders_id(PFK):** Programın hangi derse ait olduğunu belirtmek için primary ve foreign key olarak tanımlanmıştır. Ders tablosuyla ilişkilidir. Bu sayede program ile ders arasında birebir bağlantı sağlanır. Dersleri ayırt edebilmek için varchar veri tipi kullanılmıştır.
- **derslik_id(PFK):** Programın hangi derslikte işleneceğini belirtmek için primary ve foreign key olarak tanımlanmıştır.

Derslik tablosuyla ilişkilidir. Derslikleri ayırt edebilmek için varchar veri tipi kullanılmıştır.

- time: Dersin saat aralığını göstermek için tanımlanmıştır. Zaman yönetimi yapılabilmesi için datetime veri tipi kullanılmıştır.
- day: Dersin hangi gün işleneceğini belirtmek için tanımlanmıştır. Programın takvim düzenine uygun planlanabilmesi için datetime veri tipi kullanılmıştır.

3.3.2.6. Özel Durum

- **durum_id(PK):** Durumların benzersiz olarak tanımlanabilmesi için primary key olarak tanımlanmıştır. Durumları ayırt edebilmek ve kolay işleyebilmek için varchar veri tipi kullanılmıştır.
- durum_tipi: Durumun türünü veya kategorisini belirtmek için tanımlanmıştır. Sınırlı ve sayısal değerler içereceği için int veri tipi kullanılmıştır.
- durum_açıklama: Durumla ilgili ek bilgi veya açıklama sağlamak için tanımlanmıştır. Açıklama, durumun daha iyi anlaşılmasını sağlar. Metin tabanlı bilgiler içereceği için varchar veri tipi kullanılmıştır.

3.3.2.7. Ders İşlenebilir Blok

- **bölüm_id(PFK):** Bölüm varlığı ile ilişkiyi sağlamak için primary ve foreign key olarak tanımlanmıştır. Bölüm tablosundaki bölüm_id alanıyla ilişkilidir. Bölümleri benzersiz olarak tanımlamak ve ayırt edebilmek için varchar veri tipi kullanılmıştır.
- **blok_id(FK):** Blok varlığı ile ilişkiyi sağlamak için foreign key (yabancı anahtar) olarak tanımlanmıştır. Blok tablosundaki blok_id alanıyla ilişkilidir. Blokları ayırt edebilmek ve yönetebilmek için varchar veri tipi seçilmiştir.

3.3.2.8. Ders Öncelik

- **ders_id(PFK):** Ders varlığı ile ilişkiyi sağlamak için primary ve foreign key olarak tanımlanmıştır. Ders tablosundaki ders_id alanıyla ilişkilidir. Dersleri benzersiz olarak tanımlamak ve ayırt edebilmek için varchar veri tipi kullanılmıştır.
- öncelik_numarası: Dersin öncelik sırasını belirtmek için tanımlanmıştır. Örneğin, derslerin planlanmasında veya

sıralamasında öncelikli dersleri belirlemek amacıyla kullanılır. Sadece sayısal değerler içereceği için int veri tipi seçilmiştir.

3.3.2.9. Kullanıcı İzin

- **kullanıcı_id(PFK):** Kullanıcı varlığı ile ilişkiyi sağlamak için primary ve foreign key olarak tanımlanmıştır. Kullanıcı tablosundaki kullanıcı_id alanıyla ilişkilidir. Kullanıcıları benzersiz olarak tanımlamak ve ayırt edebilmek için veri tipi int olarak seçilmiştir.
- kullanıcı_izin_sayısı: Kullanıcının sahip olduğu izin sayısını belirtmek için tanımlanmıştır. Yalnızca sayısal değerler içereceği için int veri tipi seçilmiştir.

3.3.2.10. Dersin Sorumlusu

- **kullanici_id (PFK):** Kullanıcı varlığı ile ilişkiyi sağlamak için primary ve foreign key olarak tanımlanmıştır. Kullanıcı tablosundaki kullanici_id alanıyla ilişkilidir. Kullanıcıları benzersiz olarak tanımlamak ve ayırt edebilmek için int veri tipi kullanılmıştır.
- **program_id (FK):** Program varlığı ile ilişkiyi sağlamak için foreign key olarak tanımlanmıştır. Program tablosundaki program_id alanıyla ilişkilidir. Programları ayırt edebilmek ve ilişkilendirebilmek için varchar veri tipi kullanılmıştır.

3.3.2.11. Kullanıcı Özel Durumu

- **durum_id (PFK):** Özel Durum varlığı ile ilişkiyi sağlamak için primary ve foreign key olarak tanımlanmıştır. Durum tablosundaki durum_id alanıyla ilişkilidir. Durumları benzersiz olarak tanımlamak ve ayırt edebilmek için varchar veri tipi kullanılmıştır.
- **kullanici_id (FK):** Kullanıcı varlığı ile ilişkiyi sağlamak için foreign key olarak tanımlanmıştır. Kullanıcı tablosundaki kullanici_id alanıyla ilişkilidir. Kullanıcıları benzersiz olarak tanımlamak için int veri tipi kullanılmıştır.
- start_time: Özel Durumun başlangıç saatini belirtmek amacıyla tanımlanmıştır. Zaman formatını içerebilmesi için time veri tipi kullanılmıştır.

- end_time: Özel Durumun bitiş saatini belirtmek amacıyla tanımlanmıştır. Zaman formatını içerebilmesi için time veri tipi kullanılmıştır.
- day: Özel Durumun hangi gün geçerli olduğunu belirtmek için tanımlanmıştır. Gün bilgisini saklamak amacıyla varchar veri tipi kullanılmıştır.

3.3.2.12. Ders Özel Durumu

- **durum_id (PFK):** Durum varlığı ile ilişkiyi sağlamak için primary ve foreign key olarak tanımlanmıştır. Durum tablosundaki durum_id alanıyla ilişkilidir. Durumları benzersiz olarak tanımlamak ve ayırt edebilmek için varchar veri tipi kullanılmıştır.
- **ders_id (FK):** Ders varlığı ile ilişkiyi sağlamak için foreign key olarak tanımlanmıştır. Ders tablosundaki ders_id alanıyla ilişkilidir. Dersleri ayırt edebilmek ve ilişkilendirebilmek için varchar veri tipi kullanılmıştır.
- start_time: Özel Durumun başlangıç saatini belirtmek amacıyla tanımlanmıştır. Zaman formatını içerebilmesi için time veri tipi kullanılmıştır.
- end_time: Özel Durumun bitiş saatini belirtmek amacıyla tanımlanmıştır. Zaman formatını içerebilmesi için time veri tipi kullanılmıştır.
- day: Özel Durumun hangi gün geçerli olduğunu belirtmek için tanımlanmıştır. Gün bilgisini saklamak amacıyla varchar veri tipi kullanılmıştır.

4. Veri Tabanı Sorguları

Bu bölümde, projenin fiziksel tasarım aşamasında tanımlanan veri tabanı ilişkilerini ve yapısını detaylı bir şekilde ortaya koymak amacıyla örnek SQL sorgularına ve bu sorguların oluşturduğu sonuçlara yer verilecektir. Bu kapsamda, farklı sorgu türlerinin kullanımıyla veri tabanı tasarımının işlevselliği gösterilecektir. Aşağıda, örnek olarak dahil edilecek sorgu türleri sıralanmıştır:

1. **İç içe (nested) sorgular** içeren bir sorgu ile alt sorguların kullanımına yönelik bir örnek,

2. **Group by** ve **having** ifadelerinin kullanıldığı bir sorgu ile gruplama ve filtreleme işlemlerine ilişkin bir örnek,
3. **Join işlemi** içeren bir sorgu ile birden fazla tablo arasında ilişkilendirme işleminin örneklendirilmesi,
4. Proje kapsamına en uygun ve en gerekli olduğu düşünülen **2 adet view (görünüm)** sorgusu ile veri tabanındaki verilerin düzenli ve kolay erişilebilir bir şekilde sunumu.

4.1. Veri Tabanı Sorguları

4.1.1. İç İçe Nested Sorgular

Sorgu 1'de yazılmış SQL sorgusu, üniversite derslikleri ve blokları arasındaki ilişkileri kullanarak her blokta bulunan en yüksek kapasiteye sahip derslikleri belirlemeyi amaçlamaktadır. İlgili bloklarda verilen derslerin adlarını da listeleyerek, derslik kapasitesine göre sıralama yapılmasını sağlar. Bu sorgu, dersliklerin etkin kullanımını ve blokların kapasite yönetimini incelemek amacıyla kullanılır. Ayrıca, en fazla kapasiteye sahip dersliklerin hangi derslere ev sahipliği yaptığını göstererek, üniversitenin ders programının optimize edilmesine yönelik analiz yapılmasına olanak tanır. Sonuçlarına Tablo 11'de yer verilmiştir.

```
SELECT d.ders_id, b.blok_adı
FROM Blok b
JOIN Derslik ds ON b.blok_id = ds.blok_id
JOIN DersProgramı dp ON ds.derslik_id = dp.derslik_id
JOIN Ders d ON dp.ders_id = d.ders_id
WHERE ds.derslik_kapasite = (
  SELECT MAX(derslik_kapasite)
  FROM Derslik
  WHERE blok_id = b.blok_id
)
ORDER BY d.ders_id, d.ders_türü;
```

Sorgu 1

ders_id	blok_adı
A	DRS01
B	DRS02
C	DRS03
D	DRS04
E	DRS05

F	DRS06
G	DRS07
H	DRS08

Tablo 11: Sorgu 1 Çıktıları

4.1.2. Group by ve Having İçeren Sorgular

Sorgu 2, bloklardaki dersliklerin toplam kapasitesinin 50'den fazla olan blokları listelemeyi amaçlamaktadır. Bu sorgu, Blok ve Derslik tablolarını JOIN kullanarak birleştirir. GROUP BY ifadesiyle her bir blok için bir grup oluşturulurken, HAVING koşulu sayesinde yalnızca toplam kapasitesi 50'yi aşan bloklar sonuçta yer alır. Bu sorgu, üniversitenin derslik kapasite yönetimi açısından hangi blokların daha büyük kapasiteye sahip olduğunu analiz etmek ve bu bloklar arasındaki farkları görmek için kullanılabilir.

```
SELECT b.blok_adı, SUM(d.derslik_kapasite) AS toplam_kapasite
FROM Blok b
JOIN Derslik d ON b.blok_id = d.blok_id
GROUP BY b.blok_adı
HAVING SUM(d.derslik_kapasite) > 50;
```

Sorgu 2

blok_adı	*toplam_kapasite*
A	*100*
E	*120*

Tablo 12: Sorgu 2 Çıktıları

4.1.3. Join İçeren Sorgular

Sorgu 3'te yazılmış SQL sorgusu, Ders ve Derslik tablolarındaki verileri birleştirerek her dersin hangi derslikte verildiğini ve dersliklerin kapasite bilgilerini sunar. JOIN işlemi ile DersProgramı ve Derslik tabloları arasında bağlantı kurularak, her dersin hangi derslikte yapıldığını ve ilgili dersliklerin kapasite bilgileri elde edilir. Bu sorgu, derslerin adını, türünü, kapasitesini ve dersliklerin adını ve kapasitesini listeleyerek, dersliklerin etkin kullanımını ve ders programının düzenlenmesi için önemli veriler sağlar. Ayrıca, farklı dersliklerdeki derslerin karşılaştırılması, üniversitenin ders programının kapasiteye dayalı olarak optimize edilmesine yardımcı olur. Sorgu sonuçları, her dersin derslik kapasitesine göre sıralanmış

olarak sunulur ve bu da kapasite yönetimini ve kullanım analizlerini kolaylaştırır. Sonuçlar, Tablo 13'te gösterilmektedir.

```
SELECT Ders.ders_id, Ders.ders_türü, Ders.ders_kapasite, Derslik.blok_id
,Derslik.derslik_id, Derslik.derslik_kapasite
FROM Ders
JOIN DersProgramı ON Ders.ders_id = DersProgramı.ders_id
JOIN Derslik ON DersProgramı.derslik_id = Derslik.derslik_id;
```

Sorgu 3

ders_id	*ders_türü*	*ders_kapasite*	*blok_id*	*derslik_id*	*derslik_kapasite*
DRS01	*Teorik*	*50*	*BL01*	*D01*	*100*
DRS02	*Uygulama*	*30*	*BL02*	*D02*	*50*
DRS03	*Teorik*	*60*	*BL03*	*D03*	*40*
DRS04	*Seminer*	*20*	*BL04*	*D04*	*30*
DRS05	*Laboratuvar*	*40*	*BL05*	*D05*	*120*
DRS06	*Teorik*	*25*	*BL06*	*D06*	*35*
DRS07	*Uygulama*	*35*	*BL07*	*D07*	*45*
DRS08	*Teorik*	*40*	*BL08*	*D08*	*20*

Tablo 13: Sorgu 3 Çıktıları

4.1.4. View'lar ve View İçeren Sorgular

4.1.4.1. Ders-Bölüm Görüntüsü

DersinVerildigiBolum adlı view, her bir dersin hangi bölümde verildiğini gösteren bir görünüm sağlamaktadır. Bu view, Ders ve Bölüm tablolarının birleşimiyle oluşturulmuştur ve her dersin ders_id, ders_türü ve verildiği bölüm_adı bilgilerini içerir. Görünüm 1'de görülebileceği gibi, bu view üzerinden derslerin hangi bölümler için verildiği hızlı bir şekilde sorgulanabilir. Ayrıca, Sorgu 4'te örnek kullanımda gösterildiği gibi, bu görünüm derslerin bölümlere göre çekilebilmesini sağlar ve üniversite ders programlarının yönetimi için önemli bir araçtır. Bu örnek gösterimde Endüstri Mühendisliği Bölümünde yer alan derslere erişilmiştir. Sonuçlara Tablo 14'te yer verilmiştir.

Bu view, aynı zamanda bölüm bazında ders planlaması, kaynak yönetimi ve bölüm bazında ders yükü analizleri gibi süreçlerde kullanıma uygundur. DersinVerildigiBolum view'ı, derslerin hangi bölümler tarafından sunulduğunu hızlıca sorgulamak isteyen kullanıcılar için etkili bir araçtır. Sonuçlar bölüm adına göre sıralanmış ve ders türüne göre gruplandırılmıştır.

```
CREATE VIEW DersinVerildigiBolum AS
SELECT
  D.ders_id,
  D.ders_türü,
  B.bölüm_adı
FROM
  Ders D
JOIN
  Bölüm B ON D.bölüm_id = B.bölüm_id;
```

Görünüm 1: Ders-Bölüm Görünümü

```
SELECT
  ders_id,
  ders_türü,
  bölüm_adı
FROM
  DersinVerildigiBolum
WHERE bölüm_adı = 'Endüstri Mühendisliği'
```

Sorgu 4: Ders-Bölüm Görünümü Sorgusu

ders_id	ders_türü	bölüm_id
DRS_06	Teorik	Endüstri Mühendisliği

Tablo 14: Ders-Bölüm Görünümü Sorgusu Sonuçları

4.1.4.2. Ders-Blok Görüntüsü

DersBloklar adlı view, her bir dersin hangi blokta verildiğini ve o blok hakkında önemli bilgileri gösteren bir görünüm sağlamaktadır. Bu view, Ders, DersProgramı, Derslik ve Blok tablolarının birleşimiyle oluşturulmuştur ve her dersin ders_id, ders_adı, blok_id, blok_adı ve blok_önceliği bilgilerini içerir. Görünüm 2'de görülebileceği gibi, bu view üzerinden derslerin hangi bloklarda verildiği hızlı bir şekilde sorgulanabilir. Ayrıca, Sorgu 6'da örnek kullanımda gösterildiği gibi, bu görünüm derslerin hangi bloklarda yer aldığına dair analizler yapılmasını sağlar ve üniversite ders programlarının yönetimi için önemli bir araçtır. Bu örnek gösterimde Endüstri Mühendisliği derslerinin hangi bloklarda yer aldığına erişilmiştir. Sonuçlara Tablo 15'te yer verilmiştir.

Bu view, aynı zamanda blok bazında ders planlaması, blok yönetimi ve derslik kapasitesinin analiz edilmesi gibi süreçlerde kullanıma uygundur. DersBloklar view'ı, derslerin hangi bloklarda yer aldığını hızlıca sorgulamak isteyen kullanıcılar için etkili bir araçtır. Sonuçlar blok adına göre sıralanmış ve blok önceliğine göre gruplandırılmıştır.

```
CREATE VIEW DersBloklar AS
SELECT
   D.ders_id,
   B.blok_adı
FROM
   Ders D
JOIN
   DersProgramı DP ON D.ders_id = DP.ders_id
JOIN
   Derslik DL ON DP.derslik_id = DL.derslik_id
JOIN
   Blok B ON DL.blok_id = B.blok_id;
```

Görünüm 2: Ders-BlokGörünümü

```
SELECT *
FROM DersBloklar
WHERE blok_adı = 'A';
```

Sorgu 5: Ders-BlokGörünümü A bloğunda alınan dersler sorgusu

ders_id	*blok_adı*
DRS_01	*A*

Tablo 15: Ders-BlokGörünümü A bloğunda alınan dersler sorgusu sonuçları