**T.C.**

**AKSARAY ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**STAJ DEFTERİ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ö**  **Ğ**  **R**  **E**  **N**  **C**  **İ**  **N**  **İ**  **N** | Adı, Soyadı | Ali Mert ALADAĞ | FOTOĞRAF  (Üzeri  imzalanıp mühürlenecek) |
| Numarası | 230211027 |
| Bölümü | Yazılım Mühendisliği |
| Staj Devresi | 1. Devre ☑ 2. Devre □ |
| Staja Başladığı Tarih | 04.08.2025 |
| Stajı Bitirdiği Tarih |  |
| Kaç İş Günü Çalışma Yaptığı | 30 |

|  |  |
| --- | --- |
| **STAJ ADI ve KONUSU** |  |

|  |  |  |
| --- | --- | --- |
| **İ**  **Ş**  **Y**  **E**  **R**  **İ**  **N**  **İ**  **N** | İşyerinin / Firmanın Adı ve Adresi |  |
| İşyeri veya firma adına defteri tasdik eden işyeri amirinin  Adı-Soyadı:  ..............................................  Unvanı:  ..............................................  Diploma No:  ..............................................  Oda Sicil No:  .............................................. | Yukarıda ismi ve fotoğrafı bulunan öğrencinin işyerimizde 30 iş günü staj yaptığını ve bu defteri kendisinin tanzim ettiğini beyan ve tasdik ederim.  İmza ve Mühür    Tarih: |

|  |
| --- |
| **STAJ KOMİSYON ve STAJ DEĞERLENDİRME SONUCU** |
| Yapılan staj kabul edilmemiştir. □ |
| Yapılan staj kabul edilmiştir. 🗆 |

|  |  |  |  |
| --- | --- | --- | --- |
| **STAJ KOMİSYONU** | **Tarih:**  **İmza**  **BAŞKAN** | **Tarih:**  **İmza**  **ÜYE** | **Tarih:**  **İmza**  **ÜYE** |

**STAJIN GÜNLERE GÖRE DAĞILIM ÇİZELGESİ**

|  |  |  |  |
| --- | --- | --- | --- |
| Tarih | Günlük Çalışma (Saat) | ÖĞRENCİNİN ÇALIŞTIĞI KONULAR | |
| **04.08.2025** | **8** | **Oryantasyon Günü ve Sistem Kurulumu** | |
| **05.08.2025** | **8** | **Windows Forms Uygulamasında Giriş ve Kimlik Doğrulama Modülü Tasarımı** | |
| **06.08.2025** | **8** | **Personel ve Yönetici Ana Menü Formlarının Tasarımı ve Kodlanması** | |
| **07.08.2025** | **8** | **Personel ve Yönetici Ana Menü Formları Tasarımının Güncellenmesi** | |
| **08.08.2025** | **8** | **Proje\_1 için MySQL’de Veritabanı ve Tablo Oluşturma** | |
| **11.08.2025** | **8** | **Giriş Formu ile MySQL Veritabanı Arasında Bağlantının Kurulması** | |
| **12.08.2025** | **8** | **Kişisel Bilgi Formu ve Duyurular/Bildirimler Ekranı Geliştirilmesi** | |
| **13.08.2025** | **8** | **İzin Talebi Ekranı Geliştirilmesi** | |
| **14.08.2025** | **8** | **İzin Durumu Ekranı Geliştirilmesi** | |
| **15.08.2025** | **8** | **Yardım Ekranı Tasarımı ve Geliştirmesi** | |
| **18.08.2025** | **8** | **Form9 (Şifre Değiştirme) Kod Analizi, Testler ve Güvenlik İyileştirme Önerileri** | |
| **19.08.2025** | **8** | **Yönetici ve Personel Ortak Formlarının Entegrasyonu ve Test Çalışmaları** | |
| **20.08.2025** | **8** | **İzin Yönetimi Formu Tasarımı ve Geliştirilmesi** | |
| **21.08.2025** | **8** | **İzin Yönetimi Formu Geliştirilmesi ve Kodun Yeniden Yapılandırılması** | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
| TOPLAM İŞ GÜNÜ | TOPLAM SAAT | İŞYERİ AMİRİNİN İMZASI | ÖĞRENCİNİN İMZASI |
| **30** | **240** |  |  |

**Bugün, Adana Büyükşehir Belediyesi Bilgi İşlem Daire Başkanlığı’nda stajın ilk gününe resmi olarak başlandı. Gerekli evraklarımı teslim edildi .**

**Staj sürecinin ilk günü kapsamında, birim çalışanlarıyla yaklaşık iki saat boyunca kurumun işleyişi, ekip yapısı ve proje kültürü hakkında sohbet ederek tanışma fırsatı edinildi. Bu etkileşim, ekip içi iletişim ve çalışma ortamını daha iyi anlaşılması açısından faydalı oldu.**

**Stajın ilk gününde, Visual Studio 2022 kullanarak **.NET Framework** tabanlı bir **Form Uygulaması** projesi oluşturuldu. Projenin adı “Proje\_1” olarak belirlendi. Yazılım geliştirme süreci boyunca sürüm takibi yapabilmek ve çalışmaları güvenli bir şekilde yedeklemek amacıyla projeyi **GitHub** üzerinde yeni bir repository’ye bağlandı.**

**Yapılan işlemler:**

**-Visual Studio 2022 ortamında .Net Framework tabanlı **Windows Forms Application** projesi oluşturuldu.**

**-Proje\_1.sln çözüm dosyası ve bağlı kaynak dosyaları yapılandırıldı.**

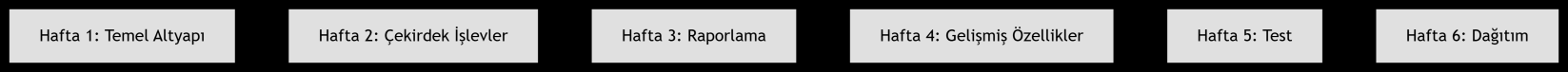
**-Git ile yerel repository oluşturuldu.**

**-GitHub üzerinde yeni bir uzak repository (remote repo) açıldı ve projeye bağlandı.**

**-Initial commit işlemi yapılarak temel proje dosyaları versiyon kontrolüne dahil edildi.**

**Bu işlem sayesinde, staj boyunca geliştirilecek kodlar Git ile takip edilecek ve GitHub üzerinden güvenli bir şekilde yedeklenecektir. Ayrıca sürüm geçmişi üzerinden ilerlemeler takip edilebilecek ve gerektiğinde proje önceki sürümlerine geri döndürülebilecektir.**

**Staj süresince geliştirilecek olan .NET tabanlı Windows Forms uygulamasına yönelik 6 haftalık bir geliştirme planı hazırlanmıştır.Bu plan doğrultusunda proje adım adım ilerletilerek sürdürülebilir ve versiyonlanabilir bir yapıda tamamlanması hedeflenmektedir**.



**Şekil 1.1.1:** Projenin haftalık geliştirme planı.

**Stajın ikinci gününde, bir önceki gün temelleri atılan .NET Framework tabanlı Windows Forms projesi kapsamında kullanıcı giriş sisteminin (Login Form) tasarımı ve temel işlevselliği üzerinde çalışılmıştır. Bu modül, kullanıcıların kimlik doğrulama işlemlerini gerçekleştirecek şekilde yapılandırılmıştır.**

**Yapılan işlemler:**

**\*Form1 adında giriş ekranı tasarlandı. Bu form üzerinde kullanıcı adı ve şifre alanları, “Giriş” ve “Çıkış” butonları eklendi.**

**\*Kullanıcı deneyimini artırmak adına buton ve arka plan düzenlemeleri yapılmış, font ve renk ayarları ile görsel iyileştirmeler sağlanmıştır.**

**\*Kullanıcı adı ve şifre kontrolleri için temel mantık yazılmıştır. Eğer kullanıcı adı ve şifre doğru girilirse Form2 ekranına yönlendirilmekte, aksi takdirde kullanıcıya hata mesajı gösterilmektedir.**

**\*Hatalı girişlerin ayrımını detaylandırmak için, sadece kullanıcı adı ya da sadece şifre hatalıysa ayrı ayrı bilgilendirme mesajları gösterilecek şekilde geliştirme yapılmıştır.**

**\*Kodlama aşamasında if-else yapıları, MessageBox, Hide() ve Show() metotları etkin biçimde kullanılmıştır.**

**Bu çalışmalar sayesinde, kullanıcı doğrulama sisteminin temel yapısı oluşturulmuş ve projenin ilk işlevsel bileşenlerinden biri tamamlanmıştır. Kodlama süreci sürüm kontrol sistemiyle desteklenerek düzenli ve güvenli bir geliştirme ortamı sağlanmıştır.**

**İlk aşamada if koşulları içinde hem kullanıcı adı hem şifre kontrolünü aynı anda değerlendirmek yerine ayrı ayrı kontrol yapısı kurulmadan else bloğuna yönlendirilmişti. Bu durum, kullanıcı adı doğru fakat şifresi yanlış olan kullanıcıya yetersiz bilgi veren bir hata mesajı gösterilmesine neden oldu.Koşullu ifadelerin kullanıcı deneyimine doğrudan etkisi olduğu gözlemlendi. Hataların daha açıklayıcı şekilde ayrıştırılması gerektiği öğrenildi.**



**Şekil 1.1.2:** Projenin giriş kontrol kodlamaları

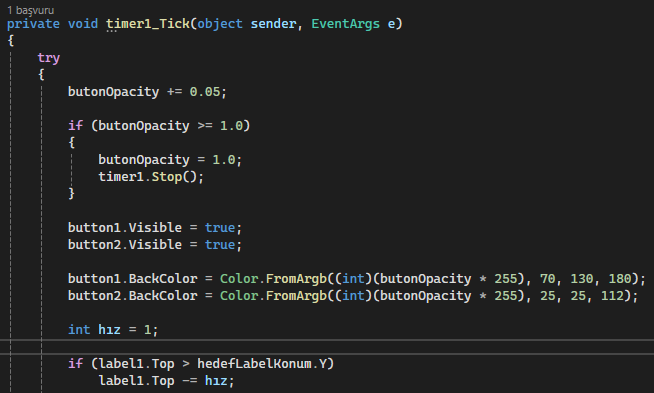
**Personel izin takip sistemi kapsamında iki farklı kullanıcı rolüne (Personel ve Yönetici) ait **ana menü arayüzleri** oluşturuldu. kullanıcılarının sistemden faydalanabileceği işlevler düşünülerek aşağıdaki menü butonları tasarlandı.Proje, Visual Studio 2022 geliştirme ortamında C# dili ve WinForms teknolojisi ile yürütülmektedir.**

**Kullanıcıların kendi rollerine uygun ekranlara yönlendirilebilmesi için iki farklı form oluşturulmuştur. **Personel ve yönetici menüleri büyük ölçüde benzerlik göstermektedir.** Her iki menüde de kullanıcıların kişisel bilgilerine ulaşabileceği, şifrelerini değiştirebileceği ve yardım alabileceği seçenekler bulunmaktadır ve rollerine özel işlemler için farklı butonlar da yer almaktadır .**

**Kullanıcı arayüzünde sade ve anlaşılır bir görünüm oluşturmak amacıyla buton boyutları, yazı fontları ve hizalamalar özenle düzenlenmiştir. Menülerin ekranda ortalanarak açılması için StartPosition özelliği CenterScreen olarak ayarlanmıştır.**

**Görsel estetik açısından form arka planına görsel eklenmiş, BackgroundImage ve BackgroundImageLayout özellikleri kullanılarak uyumlu bir görünüm elde edilmeye çalışılmıştır. Bu süreçte, eklenen görsellerin form boyutuna göre bozulmaması için uygun konumlandırma ve yeniden boyutlandırma ayarları yapılmıştır.**

**Yapılan çalışmalar kapsamında, ana giriş formlarındaki yüksek sadelik kullanıcı deneyimini olumsuz etkileyebileceği fark edilmiştir. Kullanıcı deneyimini artırmak amacıyla ana menü formlarındaki butonlar ve etiketler üzerine animasyon efektleri Timer bileşeni kullanılarak uygulanmıştır. Bu süreçte, animasyonların sistem kaynaklarını aşırı tüketmemesi adına her bir adım için makul bir Interval süresi belirlenmiş ve UI performansı gözlemlenmiştir. Aynı zamanda, animasyon işlemleri sırasında buton konumlarının form dışına taşmaması için sınır kontrolleri uygulanmış ve görsel tutarlılık korunmuştur.**

****

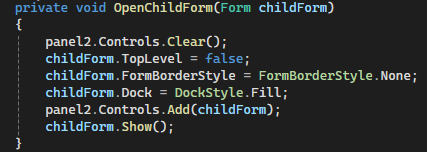
**Şekil 1.1.3:Timer tabanlı animasyon kodlaması**

**Stajın dördüncü gününde, Visual Studio 2022 geliştirme ortamı kullanılarak yürütülen masaüstü .NET Framework projesinde, kullanıcı arayüzü tasarımı üzerine odaklanılmıştır. Bir önceki gün tasarlanmış olan ana menü yapısının, profesyonel bir uygulama standartlarını karşılamadığı tespit edilmiş; bu durum ilgili birim sorumluları ile yapılan teknik görüşmeler sonucu detaylı şekilde değerlendirilmiştir.**

**Gerçekleştirilen görüşmelerde, menü yapısının hem görsel bütünlük hem de işlevsel ölçeklenebilirlik açısından yetersiz olduğu ortaya konmuştur. Kullanıcı arayüzünde bölgesel ayrım, içerik yönetimi ve dinamik yükleme gibi ihtiyaçların artması nedeniyle, Panel kontrolüne dayalı daha kurumsal bir arayüz tasarımına geçilmesi gerektiği kararlaştırılmıştır.**

**Bu doğrultuda panel1 aracı, formun sol bölümünde sabitlenen bir ana menü paneli olarak yapılandırılmış; içerisinde btnKisiselBilgiler, btnIzinDurumu, btnIzinTalepleri, btnSifreDegistir ve btnYardim gibi Button kontrolleri düzenli bir şekilde konumlandırılmıştır. Bu butonlara tıklandığında açılacak olan içerik formları ise panel2 adlı ikinci bir Panel aracılığıyla yönetilmiştir. panel2, formun içerik bölgesi olarak tanımlanmış ve tüm alt formların bu panel içinde açılması sağlanmıştır.**

**Alt formların dinamik olarak panel2 içinde yüklenebilmesi için aşağıdaki C# metodu kullanılmıştır:**



**Şekil 1.1.4:Panel tabanlı menü yapısının genel görünümü**

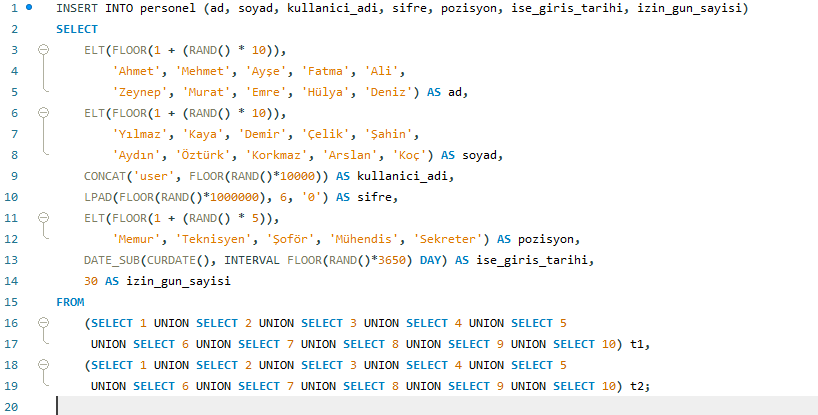
**Bu metodun uygulanmasıyla birlikte, farklı alt formların (FormKisiselBilgiler, FormIzinDurumu vb.) ana formun içinde, ayrı pencereler açılmadan tek bir panel üzerinde gösterilmesi mümkün kılınmıştır. Böylece uygulamada hem kullanıcı deneyimi geliştirilmiş hem de form geçişleri daha kontrollü ve tutarlı hale getirilmiştir.**

**Sonuç olarak, arayüz tasarımında yalnızca işlevsellik değil, aynı zamanda bileşenlerin düzenli hiyerarşisi ve görsel entegrasyonu da ön plana alınmış; Panel, Button, Dock, FormBorderStyle, Controls.Clear() gibi araçların bilinçli kullanımı sayesinde daha ölçeklenebilir ve sürdürülebilir bir mimari yapı kurulmuştur. Bu süreçte, dinamik form yükleme yöntemleri ve kapsayıcı kontrollerin profesyonel yazılım geliştirme süreçlerindeki yeri kavranmıştır.**

**Personel izin takip sistemi için MySQL üzerinde veritabanı ve tablo oluşturma çalışmaları gerçekleştiridi. Öncelikle “personel\_izin\_takip” isimli veritabanı oluşturulmuş, ardından içerisinde ad, soyad, kullanici\_adi, sifre, pozisyon, ise\_giris\_tarihi ve izin\_gun\_sayisi gibi sütunlara sahip “personel” tablosu tasarlanmıştır.**

**Sistemde yüzlerce personel kaydı olması gerektiğinden, veri girişini manuel olarak yapmak yerine toplu veri ekleme yöntemi tercih edildi. Bunun için RAND() ve ELT() fonksiyonları kullanılarak rastgele ad, soyad ve meslek atamaları yapan SQL komutları hazırlandı, böylece tek bir sorgu ile yüzlerce kayıt otomatik olarak eklendi.**

**Veri ekleme sırasında kullanici\_adi alanının UNIQUE olarak tanımlı olması nedeniyle, aynı kullanıcı adının tekrar eklenmesi durumunda “Error Code: 1062 – Duplicate entry” hatası ile karşılaşıldı. Bu hatanın tekrarlamaması için rastgele sayı aralığı genişletildi, ekleme öncesinde benzersizlik kontrolü yapılabileceği ve gerekirse INSERT IGNORE veya ON DUPLICATE KEY UPDATE komutlarının kullanılabileceği öğrenildi.**



**Şekil 1.1.5:MySQL tablosuna rastgele kayıtlar eklenmesi**

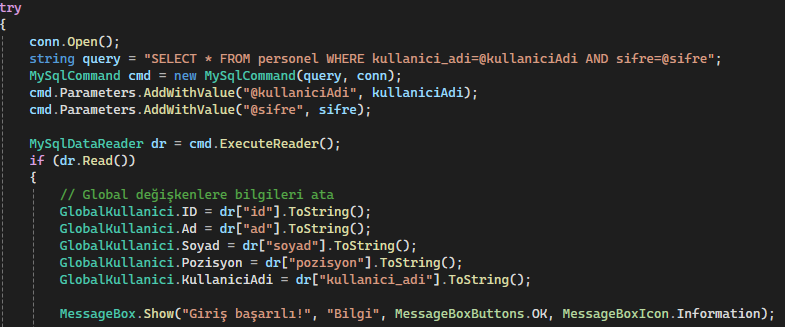
**Veritabanı oluşturma ve veri ekleme işlemleri tamamlandıktan sonra, Visual Studio 2022 ile MySQL veritabanı bağlantı kontürolü gerçekleştirildi. Bunun için MySQL Connector for .NET kütüphanesi yüklenmiş ve MySql.Data.MySqlClient isimli namespace projeye eklenmiştir. Bağlantı denemeleri sırasında bağlantı dizesinde (connection string) sunucu adresi, port numarası, veritabanı adı, kullanıcı adı ve şifre bilgileri girildi ve MessageBox aracıyla veritabanı bağlantısı kontrol edildi.**

**Bugün **Personel Sistemi** projesinde personel giriş formu üzerinde çalışmalara odaklanıldı.Kullanıcıların sisteme güvenli bir şekilde giriş yapabilmesi için MySQL veritabanı ile bağlantının kurulması ve kimlik doğrulama mekanizmasının geliştirildi.**

**Şifre kutusunda güvenlik amacıyla PasswordChar özelliği ayarlanarak girilen karakterlerin gizlenmesi sağlandı. Tasarım düzenlemeleri **Properties** penceresi üzerinden yapıldı; yazı tipleri, renkler ve hizalamalar düzenlendi.**

**Veritabanı bağlantısı için **MySQL Connector/NET** kütüphanesi kullanıldı. **Solution Explorer** üzerinden projeye MySql.Data.dll referansı eklendi ve gerekli using MySql.Data.MySqlClient; bildirimi kodun başında tanımlandı. Bağlantı cümlesi (connection string) oluşturulurken, sunucu adresi, veritabanı adı, kullanıcı adı ve şifre bilgileri değişkenler halinde düzenlenerek daha yönetilebilir bir yapı oluşturuldu.**

**Kodlama sürecinde **try-catch** blokları ile olası bağlantı hataları yakalandı ve kullanıcıya anlaşılır hata mesajları gösterildi. Bağlantı testleri sırasında **Debug** modu kullanılarak, bağlantı açma-kapama ve sorgu çalışma süreleri gözlemlendi.**

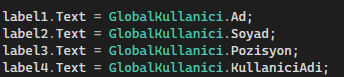
****

**Şekil 1.2.1: **Personel giriş formuna MySQL veritabanına bağlayan sistem****

**Bu çalışmalar sonucunda, uygulama ile MySQL veritabanı arasında güvenli ve kararlı bir bağlantı sağlandı. Kullanıcı doğrulama süreci hem tasarım hem de kodlama açısından tamamlanarak sistemin temel işlevlerinden biri başarıyla hayata geçirildi. Böylece, ilerleyen günlerde yönetici ve diğer kullanıcı rollerinin giriş modüllerine geçiş için sağlam bir altyapı oluşturulmuş oldu.**

**Bugün **Personel Sistemi** projesinde sistem içerisinde kullanıcıların kendi bilgilerini görüntüleyip güncelleyebileceği **Kişisel Bilgi Formu** tasarımı tamamlandı.Label (Etiketler), her bir veri alanını tanımlamak ve kullanıcıların hangi TextBox’a hangi bilgiyi gireceklerini net bir şekilde anlamalarını sağlamak amacıyla kullanıldı. TextBox (Metin Kutuları) ise kullanıcıların sistemde kayıtlı mevcut bilgilerini görüntüleyebilmesi ve gerekirse değiştirebilmesi için düzenlenebilir şekilde oluşturuldu ve varsayılan olarak veritabanından gelen bilgilerle dolduruldu. Button (“Değiştir” Butonu) ise kullanıcı tarafından yapılan değişiklikleri MySQL veritabanına aktarmak amacıyla eklendi; butona tıklandığında ilgili güncelleme sorgusu çalıştırılarak değişikliklerin sisteme yansıması sağlandı.**

**Bileşenlerin konumlandırılmasında TableLayoutPanel ve GroupBox gibi düzen araçları kullanılarak düzenli ve kullanıcı dostu bir arayüz elde edildi. Kodlama aşamasında ise **C#** dilinde Click Event metotları yazılarak butonların işlevleri tanımlandı.**

****

**Şekil 1.2.2: **Personel giriş formuna Global sınıfını bağlayan sistem****

**Ayrıca, sistemin bilgilendirme ve güncel gelişmeleri yansıtma amacıyla **Duyurular/Bildirimler Formu** oluşturuldu. Bu formda, **WebBrowser** (projede kullanılan .NET sürümünde WebView2 bileşeni yerine mevcut **WebBrowser** kontrolü) kullanılarak **Adana Büyükşehir Belediyesi** resmi internet sitesinin canlı olarak görüntülenmesi sağlandı. Bu sayede kullanıcılar uygulama içerisinden çıkmadan belediye duyurularını ve haberlerini takip edebilmektedir.**

****WebBrowser bileşeninin avantajları:****

**+Harici bir tarayıcı açmadan web içeriğinin uygulama içerisinde gösterilmesi.**

**+HTML, CSS ve JavaScript desteği ile sayfanın tarayıcıda olduğu gibi görüntülenebilmesi.**

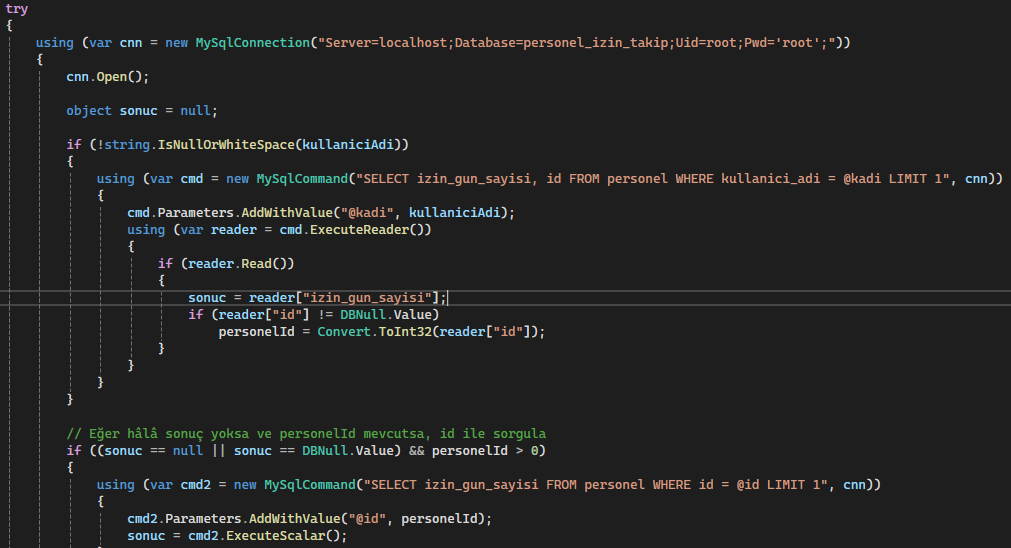
**+Kullanıcı deneyimini artırarak bilgilere hızlı erişim sağlanması.**

**Geliştirme süreci sonunda, hem **Kişisel Bilgi Formu** hem de **Duyurular/Bildirimler Ekranı** sorunsuz bir şekilde çalışır hale getirildi. Bu formlar, kullanıcı deneyimini güçlendiren, hem bilgi yönetimini hem de anlık duyuru takibini kolaylaştıran önemli modüller olarak sistemde yer aldı.**

**Bugün, WinForms üzerinde kullanıcı dostu bir arayüz tasarlarken **Label**, **TextBox** ve **Button** gibi temel bileşenlerin etkin kullanımını pekiştirildi. Ayrıca, veritabanından gelen verilerin form bileşenlerine nasıl aktarılacağını ve kullanıcı tarafından yapılan değişikliklerin MySQL veritabanına güncelleme sorguları ile nasıl yansıtılacağını öğrenildi. Bunun yanında, **WebBrowser** kontrolünü kullanarak uygulama içine web sayfası entegre etme ve bu sayede harici tarayıcıya ihtiyaç duymadan anlık bilgi akışı sağlama deneyimi edinildi.**

**Bugün, **“İzin Durumu”** adındaki Form7 üzerinde çalışma yapıldı. Bu form, sisteme giriş yapan personelin **kalan izin hakkını**, **geçmiş izinlerini** ve **bugünkü çalışma/izin durumunu** otomatik olarak gösterecek şekilde geliştirildi.**

**Form yüklendiğinde (Form7\_Load), üç ana metod sırasıyla çalıştırıldı:IzinHakkiGetir() → Personelin kalan izin gün sayısı veritabanından çekilerek label1 üzerinde gösterildi.Öncelikle giriş yapan personelin kullanıcı adını bulmak için TryResolveKullaniciAdiFromParent() metodu çalıştırıldı.Kullanıcı adı bulunamadığında, alternatif olarak bilgisayarın Windows kullanıcı adı veya daha önce set edilen personelId bilgisi kullanıldı.MySQL üzerinden personel tablosuna sorgu atılarak izin\_gun\_sayisi bilgisi elde edildi.Veri bulunamadığında, tabloda ilk kayıt alınarak (fallback yöntemi) kullanıcıya bilgilendirme yapıldı.GecmisIzinleriGetir() → Personelin daha önce aldığı izinler listelendi.izinler tablosundan baslangic\_tarihi, bitis\_tarihi ve aciklama alanları çekildi.Çekilen veriler DataTable içerisine dolduruldu ve dataGridView1 üzerinde kullanıcıya gösterildi.BugunIzinDurumu() → Personelin bugün izinli olup olmadığı tespit edildi.izinler tablosundan, bugünün tarihi (CURDATE()) izin başlangıç ve bitiş tarihleri arasındaysa kayıt sayısı kontrol edildi.Sonuç **1 veya daha fazla** ise label2 üzerinde “Bugün izinlisiniz”, aksi durumda “Bugün çalışıyorsunuz” ifadesi gösterildi.**

****

**Şekil 1.2.3:Personelin kalan izin gün sayısını ve ID’sini sorgulayan SQL komutu**

**Veritabanı bağlantısı MySqlConnection sınıfı ile kuruldu.SQL sorgularında parametre kullanılarak güvenli veri çekme işlemi sağlandı (@id, @kadi).Kod içerisinde olası veri bulunamama durumlarına karşı **hata önleme** ve **fallback** mekanizmaları eklendi.Form tamamen otomatik çalışacak şekilde yapılandırıldı, kullanıcı herhangi bir veri girişi yapmadan izin bilgilerini görebildi.**

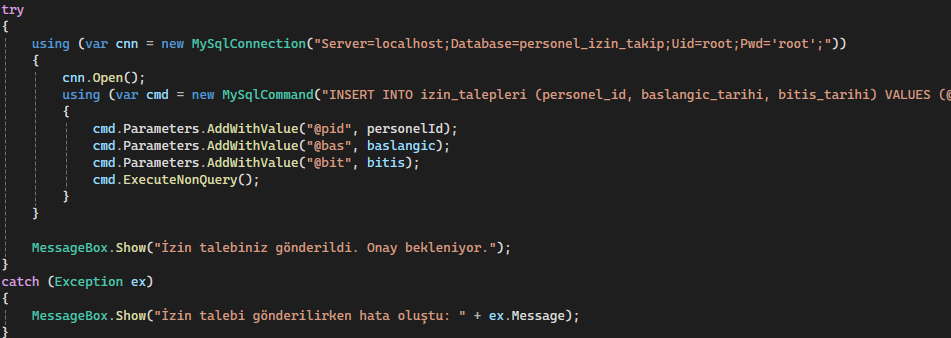
**Bu çalışmada, **formlar arası veri aktarımı**, **MySQL veritabanı sorguları**, **DataGridView veri gösterimi** ve **koşullu label güncelleme** konularında deneyim kazanıldı. Ayrıca, kullanıcı bilgilerini dinamik olarak çözümleyen bir metod (TryResolveKullaniciAdiFromParent) geliştirildi.**

**Bugün, **“İzin Talebi”** adındaki Form8 üzerinde çalışma yapıldı. Bu form, sisteme giriş yapan personelin izin hakkını görüntüleyip yeni izin talebi oluşturmasına olanak sağlayacak şekilde geliştirildi.Form yüklendiğinde (Form8\_Load), **IzinHakkiGetir()** metodu çalıştırılarak personelin kalan izin hakkı **label1** üzerinde gösterildi.**

****IzinHakkiGetir()** →Öncelikle giriş yapan personelin kullanıcı adını bulmak için **TryResolveKullaniciAdiFromParent()** metodu çalıştırıldı.Kullanıcı adı bulunamadığında alternatif olarak **Windows kullanıcı adı** veya daha önce set edilen **personelId** bilgisi kullanıldı.MySQL üzerinden **personel** tablosuna sorgu atılarak **izin\_gun\_sayisi** bilgisi elde edildi.Veri bulunamazsa tabloda ilk kayıt alınarak (fallback yöntemi) kullanıcıya bilgilendirme yapıldı.Sonuç, izin hakkı gün sayısı olarak **label1** üzerinde gösterildi.**

**İzin\_Talep\_Et butonuna basılmadan önce kullanıcı **başlangıç** ve **bitiş** tarihlerini dateTimePicker1 ve dateTimePicker2 kontrollerinden seçti.Bitiş tarihinin, başlangıç tarihinden önce olup olmadığı kontrol edildi. Hatalı seçimde kullanıcı uyarıldı.personelId mevcut değilse, veritabanından kullanıcı adına göre çekildi.Gerekli bilgiler tamamlandıktan sonra İzin\_Talep\_Et butonuna basıldığında MySQL veritabanındaki **izin\_talepleri** tablosuna INSERT sorgusu ile yeni izin kaydı eklendi.İşlem başarıyla tamamlandığında kullanıcıya **“İzin talebiniz gönderildi. Onay bekleniyor.”** mesajı gösterildi.**

**Veritabanı bağlantısı **MySqlConnection** sınıfı ile kuruldu.SQL sorgularında **parametre** kullanılarak güvenli veri ekleme sağlandı (@pid, @bas, @bit, @kadi).Kod içerisinde veri bulunamama ve bağlantı hatalarına karşı **try-catch** blokları ile hata yönetimi uygulandı.Form, otomatik olarak kullanıcı bilgilerini çözüp, izin hakkını gösterecek ve izin talebini veritabanına kaydedecek şekilde yapılandırıldı.**

****

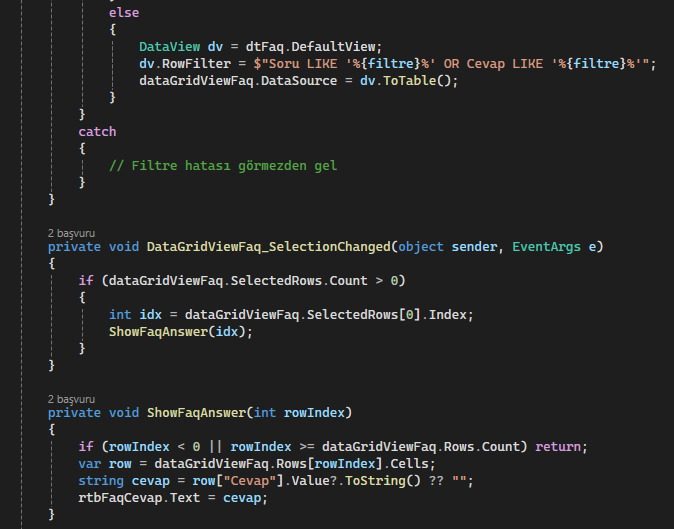
**Şekil 1.2.4: İzin talebini veritabanına ekleyen SQL INSERT komutu**

**Bu çalışmada, V**eritabanına veri ekleme (INSERT)**, **kullanıcı bilgilerini otomatik çözümleme**, **formlar arası veri aktarımı**, **tarih doğrulama işlemleri** ve **hata yönetimi** konularında deneyim kazanıldı. Ayrıca, parametreli SQL sorgularının güvenlik avantajları ve MySQL bağlantı yönetimi üzerine pratik yapıldı.**

**Bugün, "FormYardim" isimli yardım ve destek formu üzerinde geliştirme çalışmaları yapıldı. Bu form, kullanıcıların sık sorulan sorulara (SSS) erişebilmesini, kısa bir kullanım rehberini görüntülemesini ve iletişim/geri bildirim işlemlerini gerçekleştirmesini sağlamak amacıyla tasarlandı.**

**Formun yüklenmesi sırasında (FormYardim\_Load) versiyon bilgisi gösterildi, SSS listesi oluşturularak DataGridView bileşenine bağlandı ve kullanım rehberi RichTextBox’a eklendi. Ayrıca, iletişim panelinde destek e-posta adresi ve telefon numarası görüntülendi. Ayrıca, form içeriğini farklı sekmelerde düzenlemek için TabControl bileşeninin kullanımı konusunda pratik yapıldı.**

**SSS (FAQ) bölümünde, soru ve cevaplar DataTable kullanılarak tanımlandı, kullanıcı arama kutusuna yazdıkça filtreleme yapıldı (TxtFaqArama\_TextChanged). Seçilen sorunun cevabı, ayrı bir RichTextBox alanında gösterildi (DataGridViewFaq\_SelectionChanged).**

****

**Şekil 1.2.5:SSS arama filtresi ve seçilen sorunun cevabını gösteren yöntemler**

**İletişim ve geri bildirim kısmında, e-posta adresine tıklandığında varsayılan e-posta istemcisinin açılması veya adresin panoya kopyalanması sağlandı (LblDestekEmail\_LinkClicked). Telefon numarası için de kopyalama işlevi eklendi (BtnKopyalaTel\_Click). Kullanıcılar konu ve mesaj yazarak geri bildirim gönderebilecek, bu veriler MySQL veritabanındaki yardim\_talepleri tablosuna kayıt edilecek (BtnGeriBildirimGonder\_Click).**

**Bu çalışmada, WinForms’ta DataTable ve DataGridView kullanarak veri gösterimi ve filtreleme yöntemleri, TextBox TextChanged olayı ile gerçek zamanlı arama mantığı, RichTextBox üzerinden metinlerin programatik olarak doldurulması, LinkLabel ile e-posta istemcisi açma veya panoya kopyalama işlemleri, MySQL veritabanına form üzerinden veri ekleme (INSERT) işlemleri ile kullanıcı girişlerini doğrulama ve boş alan kontrolü konularında deneyim kazanıldı.**

**Bugün, WinForms tabanlı personel izin takip uygulaması kapsamında yer alan **Form9 (Şifre Değiştirme)** üzerinde detaylı geliştirme ve test çalışmaları gerçekleştirildi.Formun açılış akışı, kullanıcı girdisi doğrulamaları, veritabanı etkileşimleri ve hata yönetimi senaryoları test edildi.**

**Form9, uygulamanın oturum yönetiminden gelen GlobalKullanici.ID değeri ile aktif kullanıcının kimliğini belirleyerek çalışmaktadır. Form yüklenmesi (Form9\_Load) sırasında gelen bu ID int.TryParse ile sayısal bir değere dönüştürülmekte; dönüşüm başarısız olursa form kapatılarak kullanıcıdan yeniden giriş yapması beklenmektedir. Bu mekanizma, yanlış veya bozuk oturum verilerinin uygulama akışını bozmasını önlemek için basit ama etkili bir önlemdir. Formun arayüzünde üç adet şifre alanı bulunmaktadır: mevcut şifre, yeni şifre ve yeni şifre tekrar. Bu alanlar UseSystemPasswordChar = true ile gizlenmekte ve her biri için MaxLength = 6 olarak sınırlandırılmıştır. Bu kullanıcı arayüzü tercihleri, kısa vadede kullanıcıya konfor sağlasa da güvenlik politikaları açısından yetersiz kabul edildi.**

**Yapılan testlerde ilk olarak formun alan doğrulamaları kontrol edildi: herhangi bir alan boş bırakıldığında kullanıcıya uygun uyarı mesajı gösteriliyor; yeni şifre ile tekrarının uyuşmaması durumunda kullanıcı bilgilendiriliyor. Buton tıklaması ile tetiklenen işlem akışı şu sırayla ilerliyor: boş alan kontrolü, yeni şifre eşleşme kontrolü, veritabanı bağlantısı açılarak mevcut şifrenin (SELECT sifre FROM personel WHERE id = @id) doğrulanması ve eşleşme halinde UPDATE personel SET sifre = @yeniSifre WHERE id = @id sorgusu ile şifrenin güncellenmesi. ExecuteScalar ve ExecuteNonQuery çağrılarının döndürdüğü değerler kullanıcıya bilgi mesajı olarak yansıtılmaktadır. Bu akış, temel işlevsellik açısından doğru çalışmakla birlikte pratik testlerde bazı güvenlik ve kullanılabilirlik eksikleri gözlemlendi.**



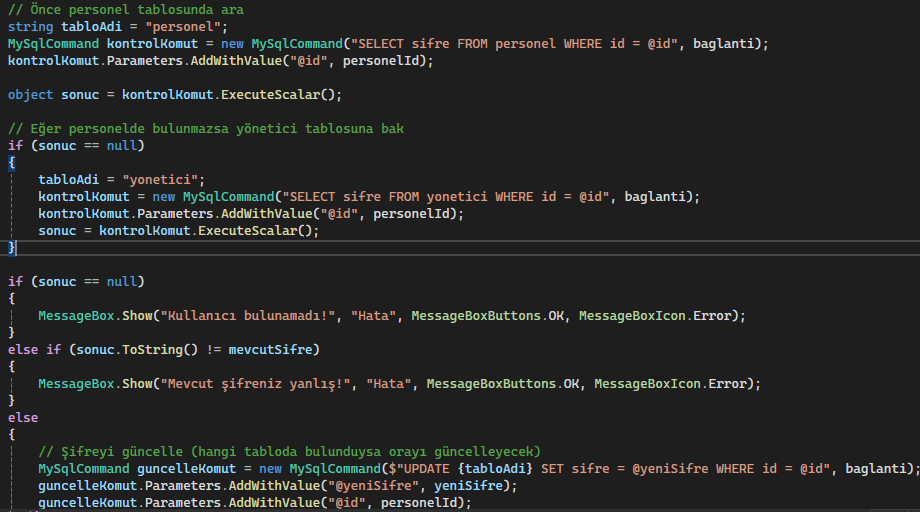
**Şekil 1.3.1:Şifre güncelleme işlemini gerçekleştiren SQL komutu ve hata yakalama yapısı**

**Bu çalışma, hem WinForms ile form akışı ve kullanıcı girdisi doğrulama konularında hem de veritabanı etkileşimlerinde pratik deneyim kazanmamı sağladı. Özellikle oturum bilgisinin form akışına entegrasyonu, ExecuteScalar/ExecuteNonQuery kullanımının davranışları ve try-catch-finally ile hata yönetimi konularında uygulamalı dersler alındı. En önemlisi, yazılım güvenliğinin (şifre yönetimi, bağlantı bilgileri, audit log) uygulama yaşam döngüsünün bir parçası olması gerektiği daha iyi anlaşıldı.**

**Bugün yapılan çalışmalarda, Yönetici Ana Menü formunda bulunan butonlardan, Personel Ana Menü formu ile ortak olan **Kişisel Bilgiler**, **Şifre Değiştir** ve **Yardım/Bilgi** formlarının bağlamaları gerçekleştirildi. Bu sayede, yöneticilerin de personeller ile aynı formlara erişebilmesi sağlandı. Bağlantıların tamamlanmasının ardından gerekli testler yapıldı ve sistemin çalışması incelendi.**

**Testler sırasında **Şifre Değiştirme Formu**nun yalnızca personel kullanıcıları için çalıştığı, ancak yönetici kullanıcıları için işlev göstermediği gözlemlendi. İlk aşamada bu sorun, Global sınıfı üzerinden **true/false mantığı** ile çözülmeye çalışıldı. Fakat bu yöntemin yalnızca şifre değiştirme formunu değil, diğer dört formda da fazladan değişiklik gerektirdiği anlaşıldı. Bu durum kodun karmaşıklaşmasına yol açacağından, yalnızca ilgili form üzerinde yapılabilecek alternatif çözümler tartışıldı.**

**Ekip içinde yapılan değerlendirmeler sonucunda, mevcut kodlara müdahale edilmeden yalnızca şifre değiştirme formunun ara kısmına ek bir kontrol yapısının eklenmesine karar verildi. Bu kapsamda, personel tablosunda kullanıcı bulunamadığı durumda sistemin yönetici tablosunu da kontrol etmesi sağlandı. Aşağıdaki kod bloğu, ilgili kısma eklenerek çözüm uygulandı:**

****

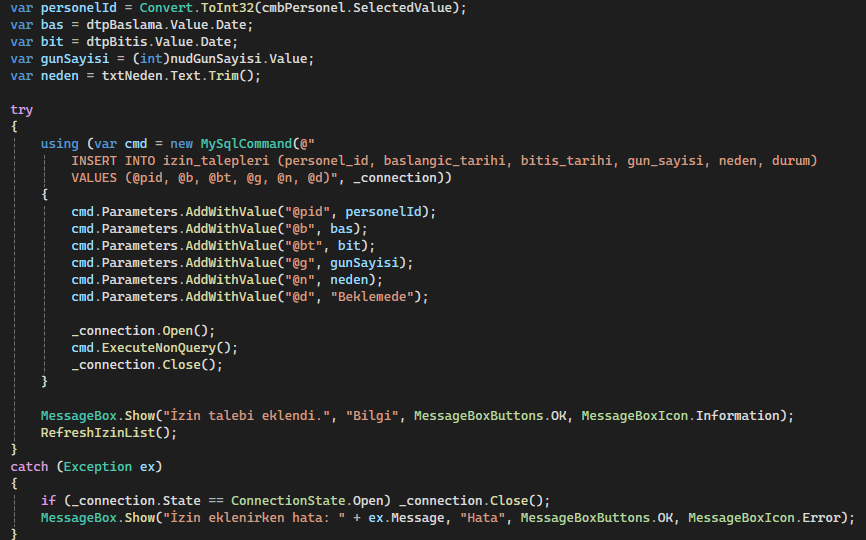
****Şekil 1.3.2:K**ayıt bulunamadığında yönetici tablosundan şifre kontrolü yapan SQL komutu**

**Bu ekleme ile birlikte, sistem önce personel tablosunu kontrol etmekte, eğer sonuç alınamazsa yönetici tablosuna yönelmektedir. Yapılan testler sonucunda, bu çözümün başarılı olduğu ve artık hem personel hem de yönetici kullanıcılarının aynı şifre değiştirme formunu kullanabildiği gözlemlendi.**

**Sonuç olarak, bugünkü çalışmada sistemin hem personel hem de yönetici tarafında ortak kullanılan formların bağlantıları sağlanmış, ayrıca şifre değiştirme formunda tespit edilen hata çözülmüştür. Böylece uygulamanın işlevselliği artırılmış ve kullanıcı deneyimi geliştirilmiştir. Ayrıca ekip içinde yapılan tartışmalar sonucunda, en az kod değişikliği ile en verimli çözümün uygulanabileceği görülmüştür.**

**Bugün kullanılan MySQL veritabanında ciddi bir hata ile karşılaşıldı. Sistem kararlılığını sağlamak amacıyla veritabanı tamamen kaldırılıp sıfırdan yeniden kuruldu. Bu süreçte, önceki günlerde oluşturulan tablolar daha profesyonel bir bakış açısıyla yeniden tasarlandı. **Personel**, **İzinler** ve **İzin Talepleri** tabloları ilişkisel bütünlük gözetilerek yapılandırıldı ve yabancı anahtar kısıtlamaları eklenerek veri tutarlılığı güvence altına alındı. Ayrıca sütun tipleri, indeksler ve varsayılan değerler, uzun vadeli performans ve sürdürülebilirlik için optimize edildi.**

**Uygulama tarafında, yönetici ekranında kullanılacak olan **“İzin Yönetimi” formu** üzerinde çalışıldı. Form tasarımında kullanıcı dostu bir arayüz hedeflenerek **DataGridView** (izin taleplerinin listelenmesi), **ComboBox** (izin türü ve durum seçimleri), **DateTimePicker** (başlangıç ve bitiş tarihleri), **TextBox** (açıklamalar) ve **Button** (Kaydet, Güncelle, Onayla, Reddet, Filtrele) gibi temel araçlar kullanıldı. Butonlara bağlı olaylar, veritabanı ile etkileşim kuracak şekilde kodlandı ve hata yönetimi için **try-catch** blokları uygulandı. Böylece izin taleplerinin eklenmesi, güncellenmesi, onaylanması ve reddedilmesi işlemleri başarıyla gerçekleştirildi.**

****

****Şekil 1.3.3:**Parametreli SQL sorgusu ile izin talebinin veritabanına eklenmesi**

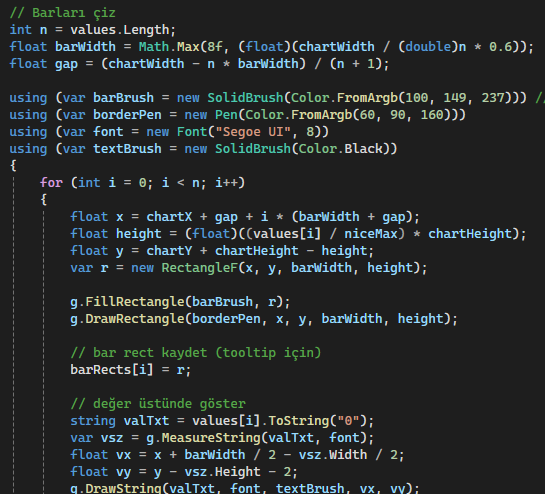
**Veritabanının sıfırdan kurulumu sırasında profesyonel bir şema tasarımının önemi pekiştirildi. İlişkisel bütünlüğün sağlanmasında **foreign key** kısıtlamalarının etkin kullanımı öğrenildi. **WinForms araçlarının** (DataGridView, ComboBox, DateTimePicker, Button vb.) form üzerinde koordineli şekilde nasıl kullanılacağı gözlemlendi. Kullanıcı arayüzü ile arka plan veri yönetiminin eş zamanlı geliştirilmesi gerektiği deneyimlendi.**

**Ek olarak, formun geliştirilmesi sırasında **kullanıcı geri bildirimlerinin simülasyonu** yapıldı ve olası senaryolar test edildi. Yanlış tarih aralıklarının seçilmesi, eksik bilgi girişi veya geçersiz izin türü seçimi gibi durumlarda sistemin doğru hata mesajlarını üretmesi sağlandı. Bu sayede hem kullanıcı deneyiminin iyileştirilmesi hem de yazılımın hataya dayanıklı hale getirilmesi konusunda önemli kazanımlar elde edildi.**

**Bugün, **İzin İstatistikleri** formu geliştirildi. Arayüz,üst kısımda bilgi etiketi (**Label**), sağ üstte veri tazeleme amacıyla **Yenile** butonu (**Button**), ana alan olarak da grafiğin çizildiği **Panel** konumlandırıldı. Grafik için hazır kütüphane kullanılmadı; bunun yerine Panel’in **Paint** olayı üzerinde **GDI+** ile özel çubuk grafik (bar chart) çizimi tasarlandı.**

**Form **Load** anında ve **Yenile** butonuna basıldığında aynı akış çalıştırılacak şekilde bir işlev kurgulandı. Bu akış, veri erişimi sonrası çizim için gerekli ham dizileri üretir (etiket ve değer dizileri), ardından Panel’i **Invalidate** ederek yeniden boyamayı tetikler. Çizim tarafında, Panel’in **ClientRectangle** değerinden hareketle grafik alanı tanımlanır; sol/sağ/üst/alt boşluklar (**margins**) hesaplanır ve çizim koordinatları bu alan üzerinde yürütülür. Y ekseni ölçekleme için dizi içindeki **maksimum değer** bulunur, daha okunabilir bir skala elde etmek amacıyla “**niceMax**” yaklaşımı uygulanır: maksimum değer 10’un katlarına yuvarlanır, böylece ızgara çizgileri (**grid**) 5 eşit aralıkla ve bütün sayılarla görüntülenir. Bu sayede eksen okumaları düzenli ve tutarlı hale getirilir.**

**Kullanıcı deneyimini artırmak için Panel’in **MouseMove** olayı yakalanarak basit bir **tooltip** davranışı eklendi. Her barın ekran koordinatları çizim sırasında bir **RectangleF** dizisinde saklandı; fare imleci bu dikdörtgenlerden birinin üzerine geldiğinde ilgili ay ve değer küçük bir baloncukta gösterildi.**

****

****Şekil 1.3.4:** İzin istatistiklerinde çubuk grafiklerin çizimini sağlayan C# kodu**

**Bugün yapılan çalışmalar sayesinde WinForms uygulamalarında istatistiksel verilerin görsel olarak kullanıcıya sunulabilmesi için grafik kütüphanelerinin nasıl kullanılacağını öğrenildi. Form üzerine eklenen grafik bileşeninin (ScottPlot kütüphanesi) programatik olarak nasıl doldurulduğunu, veritabanından alınan değerlerin dizi haline getirilerek grafiğe aktarıldığını ve eksen ayarlarının nasıl yapıldığını deneyimlendi. Bu süreçte, kullanıcıya yalnızca tablo değil aynı zamanda grafiksel bir görünüm sunmanın karar alma ve izleme süreçlerinde ne kadar faydalı olduğunu gözlemlendi.**

**AKSARAY ÜNİVERSİTESİ**

YÜKSEK ÖĞRENİMLERİ SIRASINDAN ZORUNLU STAJA TABİ

TUTULAN ÖĞRENCİLER HAKKINDA UYGULAMA SÜRECİ

Bilindiği üzere ‘‘Aksaray Üniversitesi Ön Lisans ve Lisans Eğitim Öğretim ve Yüksekokul Yönetmeliğinin 2. Bölüm 12. maddesinin 3. bendi gereği, her fakülte ve Yüksekokul kurulunca belirlenen ‘‘Staj Esasları’’ çerçevesinde öğrencilerimiz zorunlu staj yapmaktadır.

01 Ekim 2008 tarihinde uygulanmasına başlanılan 5510 Sayılı Sosyal Sigortalar ve Genel Sağlık Sigortası Kanunu’nun 4.a maddesinde;

1- Hizmet akdi ile bir veya birden fazla işveren tarafından çalıştırılanların, bu kanunun kısa ve uzun vadeli sigorta kolları uygulaması bakımından sigortalı sayılacağı belirtilmiştir.

2- Bazı sigorta kollarının uygulanacağı sigortalılar başlıklı aynı kanunun 5/b maddesinde, yükseköğrenimleri sırasında zorunlu staja tabi tutulan öğrenciler hakkında iş kazası ve meslek hastalığı sigortası uygulanacağı ve bu bentte sayılanların, 4. maddenin 1. Fıkrası (a) bendi kapsamında sigortalı sayılacağı belirtilmiştir.

3- Yine aynı kanunun 87/e maddesinde, yükseköğrenim sırasında zorunlu staja tabi tutulan öğrenciler için öğrenim gördükleri yükseköğretim kurumu, pirim ödeme yükümlüsüdür, denilmiştir.

4- Sosyal Sigorta İşlemleri Yönetmeliğinin Sigortalılığın Başlangıcı ve Bildirim Yükümlülüğü başlıklı 11/6 maddesinde, yüksek öğrenimleri sırasında zorunlu staja tabi tutulan öğrencilerin bildirimlerinin öğrenim gördükleri yüksek öğretim kurumlarınca yapılacağı belirtilmiştir.

Staj uygulaması çerçevesinde birimlerimizin ve öğrencilerimiz yasal yükümlülüklerinin yerine getirilebilmesi için aşağıda belirtilen işlemleri sırasıyla gerçekleştirilmesi gerekmektedir.

**ÖĞRENCİ SORUMLULUKLARI**

1- Staj yapacak öğrenci, staj başvuru dilekçesi (Ek-1) ve iş yeri staj kabul formunu (Ek-2) kurum onayı kısmı hariç tüm bilgi, belge ve imzaların tamamlanmış olarak staja başlayacakları günden en az 10 iş günü önce eksiksiz olarak nüfuz cüzdanı fotokopisi ile birlikte öğrenim gördükleri birimin öğrenci işlerine teslim edeceklerdir. Sigorta girişlerinin yasal olarak yapılabilmesi *için bu bilgilerin zamanında ve doğru olarak verilmesi önemlidir.* Bildirilen staj başlangıç ve bitiş tarihleri üzerinden e-bildirge sistemi kullanılarak sigorta giriş ve çıkışları yapılacağından daha sonra bu tarihleri değiştirmesi mümkün değildir.

2- İş kazaları ve meslek hastalıkları sigorta priminin zamanında tahakkuk ettirilerek SGK’ya yatırılabilmesi için öğrenciler staj yaptıkları kurumdan staj yaptıkları ay sonları itibariyle onaylı puantaj formlarını (Ek-3) okul öğrenci işleri birimine teslim edeceklerdir.

**BİRİM (FAKÜLTE / YÜKSEKOKUL) SORUMLULUKLARI**

**Dekanlık / Müdürlük**

Staj yapacak öğrencilerin işe girişleri e-bildirge kullanılarak yapılacak olup, birimler mevcut işleri sicil numaralarını kullanacaklar, Sosyal Güvenlik Kurumuna kayıtları bulunmayan birimler ise ilgili kuruma başvurarak işyeri tescilli yaptıracak ve kurumdan;

- İş yeri sicil numarası

- Kullanıcı adı

- Sistem şifresi

- İş yeri şifresi alacaktır.

**Fakülte / Yüksekokul**

**Öğrenci İşleri Birimi**

Öğrenci İşleri Birimince Fakülte/Yüksekokul Staj esasları çerçevesinde yapılacak rutin işlemlerin yanı sıra öncelikle staj yapacak öğrencinin sigortalı ilk işe giriş bildirgesi düzenlenecektir. Sigorta işe giriş ve çıkış bildirgeleri Sosyal Güvenlik Kurumunun (SGK) web sayfasında yer alan e-bildirge sayfasından yapılmaktadır. (<http://ebildirge.ssk.gov.tr/WEB/amp/login1dap>)

İşe giriş bildirgeleri stajyer öğrencilerin T.C. kimlik numarası üzerinden yapılacaktır. İşe giriş bildirgesi mutlaka işe başlanılacak tarihten en az 1(bir) gün önce düzenlenmelidir. Herhangi bir cezai duruma düşülmemesi için bu hususa dikkat edilmelidir. Unutulmamalıdır karşılıklı olarak yapılacak hatalarda ortaya çıkacak idari para cezası kuruma değil işlemi yapan kişilere çıkmaktadır.

Hazırlanan işe giriş bildirgelerinin 2 adet çıktı alınarak ilgili imzalar tamamlandıktan sonra bir nüshası öğrenciye verilecek, diğer nüsha ise birim dosyasına muhafaza edilecektir.

Zorunlu staja tabi öğrencilerin staj süresince ödenecek prim oranları %1’dir. Bu asgari ücretin 1 günlük tutarının gün sayısı ile çarpımı, aylık tutarı oluşturur. ***Örnek:*** 20 gün staj yapan bit öğrenci için asgari ücretin 1 günlük tutarı 22, 20x20 =444 TL olup, bu 20 günlük sigorta matrahıdır. 444x0,01=666 TL. 20 günlük prim tutarıdır. 30 gün staj yapmışsa 22.20x30=666 TL. Bu da asgari ücretin aylık tutarı. 666x0,01=6,66TL. Şu an ki asgari ücret aylık prim tutarıdır. Bu tutar aynı zamanda bize maliyeti ifade eder.

Staj yapan öğrencilerin staj yaptıkları kurumdan onaylı olarak gönderecekleri puantajlar esas alınarak yukarıdaki örnek doğrultusunda ‘‘*Zorunlu Staj Prim Bildirge Bordrosu’’*(Ek-4) düzenlenecek ve tahakkuk ettirilen bu pirim tutarları e-bildirge üzerinden her öğrencinin tek tek sigorta kaydına işlenecektir. Daha sonra e-bildirge üzerinden toplu olarak pirim tutarları onaylanıp, onaylı (barkodlu) tahakkuk fişi çıktısı alınarak Puantaj, Zorunlu Staj Prim Bildirge Bordrosu ve onaylı tahakkuk fişi bir yazı ekinde ilgili Dekanlık/Müdürlük tarafından Strateji Dairesi Başkanlığına ödeme işlemlerinin gerçekleştirilebilmesi için gönderilecektir.