

# **BM102 DERSİ DÖNEM SONU PROJESİ RAPORU**

Proje Konusu: Yiyecek ve İçecek Otomatı Uygulaması  
Hazırlayan: ELİF KILIÇ  
Öğrenci Numarası: 21118080005  
Tarih: 05.06.2022

## İÇİNDEKİLER

Bölüm No.	Başlık	Sayfa No.
	<i>Özet</i>	iii
<b>I</b>	<b>Giriş</b>	1
<b>II</b>	<b>Temel Kavramlar</b>	1-3
2.1	Kaynakça	3
<b>III</b>	<b>Yiyecek ve İçecek Otomatı – Genel Bakış</b>	3-14
3.1	Projenin Özellikleri	4-9
3.2	Veri Tabanı	10-13
3.2.1	Veri Tabanına Bağlantı	10-11
3.2.2	Veri Tabanında Bulunan Tablolar	11-13
3.3	Kaynakça	13
<b>IV</b>	<b>Projedeki Hata ve Eksiklikler</b>	13-16
4.1	Yeni Ürün Ekleme Aşamasında Oluşan Hata	13-14
4.2	Yeni Kullanıcı Ekleme Aşamasında Oluşan Hata	14-15
4.3	Uygulamanın Giriş Ekranının Kapatılması Sonucu Oluşan Hata	15-16
<b>V</b>	<b>Sonuç ve Değerlendirme</b>	17-18

## ÖZET

Yiyecek ve içecek otomatı uygulaması günlük hayatta kullandığımız otomat makinelerinin sanal bir sürümüdür. Bazı özellikleri farklılıklar gösterse de temel fonksiyonu bu makinelerle aynıdır: kullanıcıya satın alabileceği yiyecek ve içecek çeşitleri sunmak.

Hazırladığım uygulamada beş farklı ara yüz bulunmaktadır. Bunlar:

- Giriş ekranı,
- Yeni kullanıcı oluşturma ekranı,
- Kullanıcı ara yüzü,
- Yönetici ara yüzüne giriş ekranı,
- Yönetici ara yüzü.

Bu farklı ara yüzlerini kullanarak uygulamanın kolay bir şekilde kullanılabilmesi adına hem kullanıcı hem de yönetici için uygun bir ortam oluşturmayı hedefledim. Hazırladığım ara yüzler oldukça anlaşılırdır.

Uygulama içerisinde kullanıcı kullanmak istediği kullanıcıyı seçerek bu kullanıcının sahip olduğu bakiyeyle istediği ürünlerin satın alım işlemlerini gerçekleştirebilir. Yönetici ise kullanıcının alacağı bu ürünlerin stokunu görüntüleyerek bu stoku güncelleyebilir veya stoka yeni ürün ekleyebilir.

Bu uygulamayı yazarken Visual Studio 22 IDE'ı kullandım ve kodları C# dilinde yazdım. Uygulamanın kullandığı veri tabanı Microsoft SQL Server ve bu veri tabanını kontrol etmek için kullandığım uygulama da Microsoft SQL Server Management Studio 18'dir.

Projem için bu uygulamayı geliştirmeyi seçmemin amacı temel olarak, insanların günlük hayatta sıklıkla kullanabilecekleri ve günlük hayatlarını kolaylaştırabilecek bir sistemin tasarlanmasında kendimi geliştirebilmektir. Bu projenin ilerideki meslek hayatımda hazırlayacağım projeler için bana bir hazırlık olmasını istedim.

Bu projede C# dilinin daha etkin kullanımı için pratik yapmakla birlikte veri tabanlarının nasıl çalıştığı ve bu veri tabanlarını nasıl kullanabileceğim konusunda SQL Server kullanarak birçok bilgi edindim.

## **BÖLÜM I – GİRİŞ**

Yiyecek ve içecek otomatı uygulaması, günlük hayatta birçok yerde gördüğümüz ve kullandığımız otomatların Visual Studio 2022 IDE ortamında C# programlama diliyle yazılmış hâlidir. Windows formu uygulaması şeklindedir. Veri tabanı olarak Microsoft SQL Server ve bu veri tabanının yönetimi için de Microsoft SQL Server Management Studio 18 kullanılmıştır.

Bu projeyi yaparak kazanmayı hedeflediğim beceriler:

- C# dilini kullanarak program geliştirebilmek,
- Bir program oluştururken karşılaşılabilecek problemlere yönelik en uygun algoritmayı geliştirebilmek,
- Microsoft SQL Server kullanımını öğrenmek,
- Veri tabanı kullanımlarıyla ilgili bilgi edinmek,
- Günlük hayat problemlerinden birine çözüm üretmek gelecekte insanlara faydalı olabilecek projeler geliştirme konusundaki becerilerimi geliştirmek.

Günümüzde insanlar çok hızlı bir hayat yaşamaktalar. Bir işten diğerine yetişmek için koştururken insanın temel ihtiyacı olan yemek yeme ihtiyacını karşılayabilmek için hızlı bir çözüm bulması gerekebilir. Bu gibi durumlarda iş, okul, toplu taşıma vb. ortamlarda bulunan yiyecek ve içecek otomatları oldukça kullanışlıdır. Hayatımızı kolaylaştıran birçok makine gibi bu makine de gömülü sistemler sayesinde çalışır. Bir bilgisayar mühendisi olarak insan hayatını kolaylaştırmak en önemli amaçlarımdan biridir. Bu yüzden gelecekte meslek hayatımda gerçekleştireceğim projelere hazırlık olarak bu şekilde bir proje konusu seçtim. Projemi hazırlarken de olabildiğince kullanıcı dostu – kullanımı kolay, basit ve faydalı – bir uygulama oluşturmaya çalıştım.

## **BÖLÜM II – TEMEL KAVRAMLAR**

Bir IDE (integrated development environment) ,yani tümleşik geliştirme ortamı, program geliştiricilerin kullandıkları araçları tek bir görsel ara yüzde toplayan yazılımdır[1]. Birçok aracı bir araya toplaması sayesinde geliştiricileri farklı araçların nasıl kullanılacağını öğrenme yükünden kurtaran bu yazılımlar program geliştirmeyi oldukça kolaylaştırdıkları için geliştiricilerin çoğu

tarafından kullanılır. Ben de bu sebepten ötürü bir IDE olan Visual Studio 2022'yi projemi geliştirirken kullandım.

C# nesne tabanlı bir programlama dilidir. Kökünü C dil ailesinden alan bu dil C diline kıyasla daha güvenlidir. Microsoft tarafından geliştirilen bu dil yine Microsoft'a ait olan .NET isimli açık kaynaklı geliştirici platformunda çalışan programlar yazılabilmesini sağlar. .NET ile birçok dil, editör, kütüphane kullanılarak çeşitli platformlarda çalışan programlar geliştirilebilir[2]. C# sahip olduğu özellikler sayesinde (garbage collection, nullable types, exception handling gibi) güçlü ve güvenli uygulamalar geliştirilmesine yardımcı olur[3].

Veri tabanları, bilgisayar ortamında bilgi veya verileri düzenli bir şekilde depolayabilmemizi sağlarlar. Çoğu veri tabanında işlemler SQL (structured query language), yani yapılandırılmış sorgu dili, kullanılarak gerçekleştirilir[4].

Günümüzde geliştiricilere oldukça kolaylık sağlayan mekanizmalardan biri geliştiricilerin birbirleriyle yazdıkları kodları paylaşabilmesini sağlayan geliştirme platformlarıdır. .NET platformunda bu mekanizma NuGet adı verilen bir araçla sağlanır. Bu araç geliştiricilerin yazdıkları derlenmiş kodları (DLL gibi) ve projeler için gerekli olan diğer içerikleri barındıran paketler oluşturur[5]. Ben de bu projemi geliştirirken Dapper adında bir NuGet paketinden yararlandım.

SQL kullanımı sırasında programla veri tabanı arasındaki bağlantıyı sağlamak oldukça karmaşık bir hâl alabilir. Bu bağlantıyı sağlamak için onlarca satır kod yazmak gerekebilir ve bu da hem performans açısından hem de kodun yazılabilirliği ve okunabilirliği açısından pek verimli olmayacaktır. Dapper aracı, veri tabanı ve program arasında bağlantı kurmak için yazılan satırlarca kodu çok daha kısa olan kodlarla değiştirerek veri tabanına bağlanmayı oldukça kolay bir hâl getirir[6].

## 2.1 Kaynakça

- [1] <https://www.redhat.com/en/topics/middleware/what-is-ide>
- [2] <https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet>
- [3] <https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/>
- [4] <https://www.oracle.com/tr/database/what-is-database/>

[5] <https://docs.microsoft.com/en-us/nuget/what-is-nuget>

[6] <https://www.learnapper.com/>

### **BÖLÜM III – YİYECEK VE İÇECEK OTOMATI – GENEL BAKIŞ**

Yiyecek ve içecek otomati, insanların temel ihtiyaçlarından olan yemek yeme ve su içme ihtiyaçlarına çözüm olarak geliştirilmiş bir makinedir. Günlük hayatta oldukça kullanılan bu makinenin içindeki gömülü sistemin bir benzerini oluşturarak hazırladığım bu uygulama Windows formu uygulamasıdır.

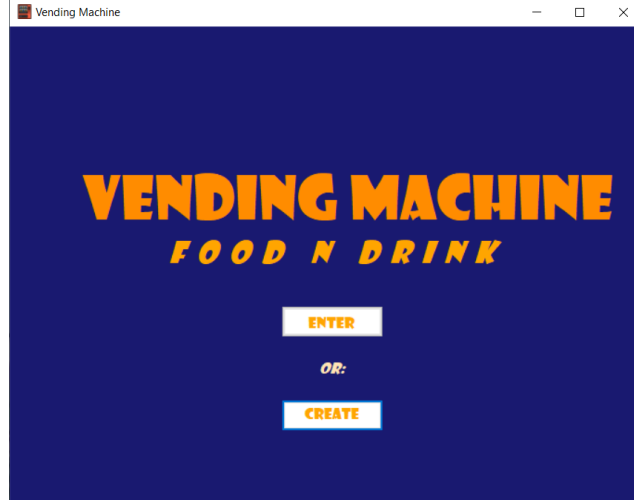
Hazırladığım bu uygulamada bulunan temel özellikler şu şekildedir:

- Kullanıcı oluşturma ve bu kullanıcı için bir bakiye belirleme,
- “Yiyecekler” ve “İçecekler” kategorileri altında otomatta bulunan ürünleri ve bu ürünlerin fiyatlarını kullanıcı ara yüzünde görebilme,
- İstenilen kullanıcıyı seçme ve bu kullanıcının bakiyesini kullanıcı ara yüzünde görebilme,
- İstenilen ürünün, seçilen kullanıcının bakiyesi yeterince, belirli miktarda satın alınması ve bu satın alma işlemi sonucunda kullanıcının bakiyesinin güncellenmesi,
- Kullanıcı ara yüzünde bulunan bir butona basılarak yönetici ara yüzüne giriş yapma sayfasında gidebilme,
- Yönetici ara yüzüne giriş yapma ekranında önceden belirlenmiş kullanıcı adı ve şifrenin girilerek yönetici ara yüzüne erişme,
- Yönetici ara yüzünde, makinede bulunan ürünlerin isimlerinin ve kalan ürün sayısının görüntülenmesi ve kalan ürün sayıları üzerinde değişiklik yapabilmesi,
- Yöneticinin seçtiği kategoriye ait olan yeni bir ürünü; isim, fiyat ve stok bilgisi girerek uygulamaya ekleyebilmesi.

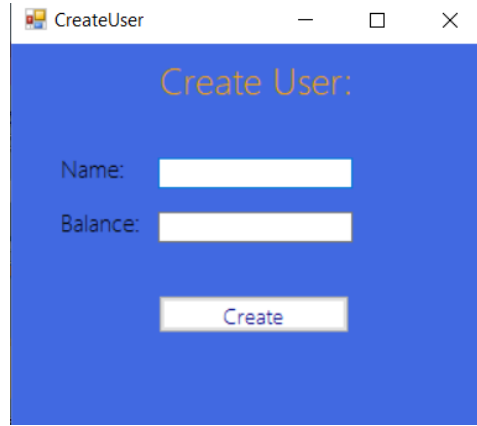
Bir sonraki bölümde yukarıdaki bahsettiğim özelliklerin her biri tek tek incelenecektir.

#### **3.1 Projenin Özellikleri**

Kullanıcı uygulamayı çalıştırdığı zaman Görsel 1’de gösterilen ekranla karşılaşır. Bu ekranda bulunan “CREATE” butonuna basarak Görsel 2’de gösterilen kullanıcı oluşturma ekranını açabilir ya da “ENTER” butonuna basarak uygulamaya giriş yapabilir.

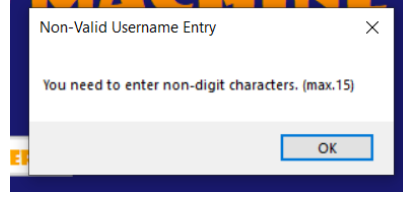


Görsel 1. Giriş ekranı

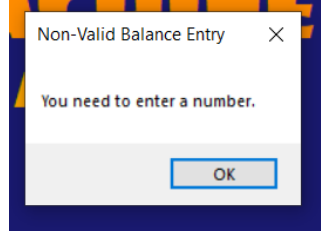


Görsel 2. Kullanıcı oluşturma ekranı

Kullanıcı oluşturma ekranında istenilen bilgilerin uygun formatta girilmemesi durumunda Görsel 3 ve 4’teki hata mesajları gösterilir.



Görsel 3. Kullanıcı adının uygun formatta girilmemesi durumunda gösterilen hata mesajı



Görsel 4. Bakiyenin uygun formatta girilmemesi durumunda gösterilen hata mesajı

“ENTER” butonuna basıldıktan sonra kullanıcı ara yüzü (Görsel 5) açılır. Bu aşamadan sonra giriş ekranına tekrar dönme imkânı bulunmamaktadır yani bu aşamadan sonra yeni kullanıcı oluşturulamaz.



Görsel 5. Kullanıcı ara yüzü

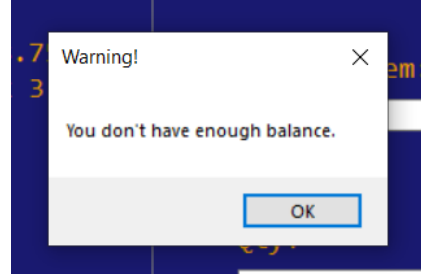
Kullanıcı ara yüzünün sol alt köşesinde bulunan açılır kutu kullanılarak veri tabanına bağlı bulunan “Users” sınıfındaki kullanıcılarının “username”leri görüntülenir. Buradan istenilen kullanıcı



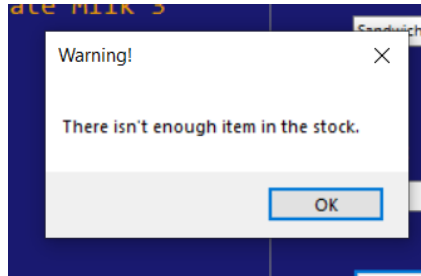
seçildiği zaman bu kutucuğun hemen altında, seçilen kullanıcının “balance” bilgisi görüntülenebilecektir.

Ara yüzün ortasında bulunan “Foods” ve “Drinks” listeleri veri tabanında bulunan aynı isimli sınıftaki ürünlerin “name” ve “price” bilgilerini göstermektedir. Kullanıcı buradan istediği ürünün ne olduğuna karar verdikten sonra ekranın sağ tarafında bulunan birinci açılır kutudan istediği ürünün kategorisini, ikinci açılır kutudan istediği ürünün ismini ve üçüncü açılır kutudan bu üründen kaç tane almak istediğini seçebilmektedir. Bu seçimler yapıldıktan sonra “BUY” butonuna basılarak ürün satın alınır ve ekranın sol alt köşesinde bulunan bakiye bilgisi güncellenir. Bu işlem sonucunda veri tabanında hem seçilmiş olan “username”e ait “balance” bilgisi hem de seçilen “name”e ait ürünün “stock” bilgisi güncellenir.

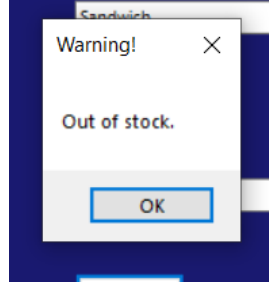
Yeterli ürün veya yeterli bakiye bulunmaması durumlarında ortaya çıkan hata mesajları Görsel 6, 7, 8’de görünmektedir.



Görsel 6. Bakiyenin yeterli olmaması durumunda çıkan hata mesajı

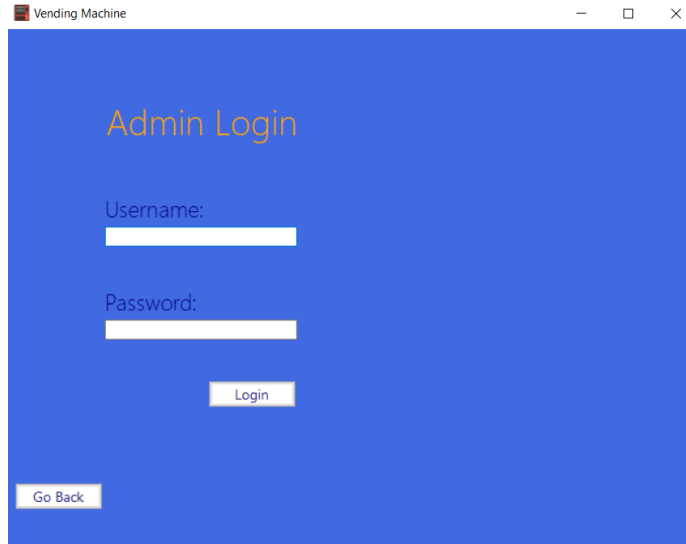


Görsel 7. Stokta istenilen miktarda ürün bulunmaması durumunda çıkan hata mesajı



Görsel 8. Ürün stokunun tamamen tükenmesi durumunda çıkan hata mesajı

Kullanıcı ara yüzünün sağ alt köşesinde bulunan “Admin Login” butonuna basılarak yönetici ekranına giriş için gerekli bilgilerin girileceği ekrana (Görsel 9) gidilebilir.

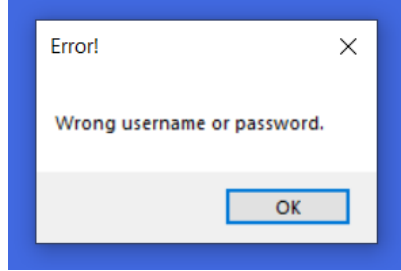


Görsel 9. Yönetici ekranına giriş ekranı

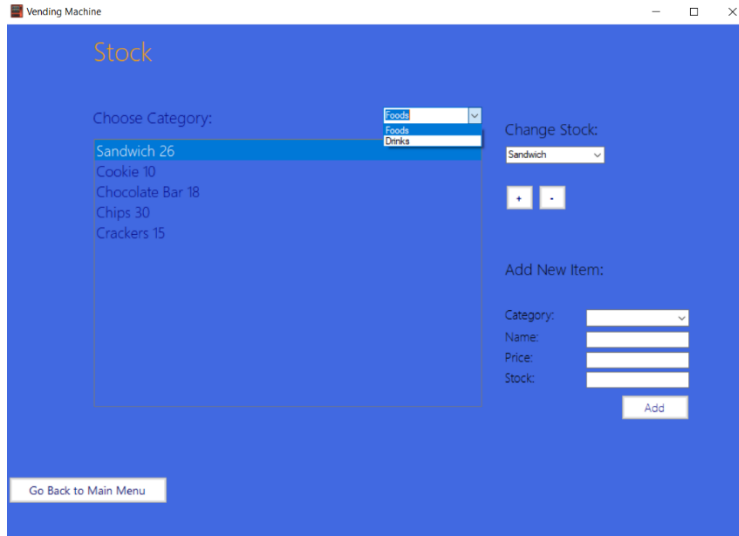
Kullanıcının yanlışlıkla bu ekrana gelmesi veya kullanıcı adı ve şifre bilgilerinin bilinmemesi gibi durumlar için ekranın sol alt köşesinde “Go Back” butonu bulunmaktadır. Bu butona basılarak bir önceki ekrana dönüş yapılabilir.

Yönetici ara yüzüne giriş yapmak için belirlenen “Username” yani “admin” ve belirlenen “Password” yani “1234” bilgiler ekrandaki gerekli kutucuklara yazıldıktan sonra “Login”

butonuna basılarak Görsel 11’da bulunan yönetici ara yüzü ekranına giriş yapılabilir. Hatalı kullanıcı adı veya şifre girilmesi durumunda Görsel 10’daki gibi hata mesajı verilir.



Görsel 10. Hatalı kullanıcı adı veya şifre girilmesi durumunda gösterilen hata mesajı



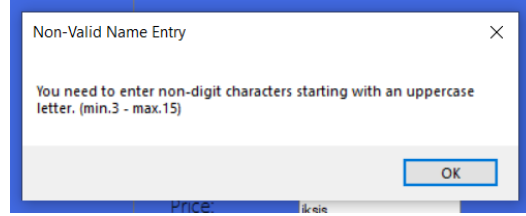
Görsel 11. Yönetici ara yüzü

Yönetici yapacağı işlemleri bitirdikten sonra ekranın sol alt köşesinde bulunan “Go Back to Main Menu” butonuna basarak kullanıcı ara yüzüne dönüş yapabilir.

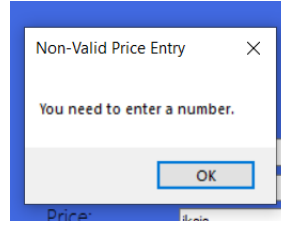
Ekranın ortasında ürün stokunun görüntülenmesini sağlayan bir liste bulunmaktadır. Bu listede gösterilecek olan ürün kategorisi listenin üzerinde bulunan açılır kutudan seçilebilir.

Stoktaki ürün miktarı değiştirilmek istenilen ürünün “name” bilgisi “Change Stock:” yazısının altında bulunan açılır kutudan seçilir. Daha sonra ürün stoku “+” veya “- ” butonlarına basılarak güncellenebilir.

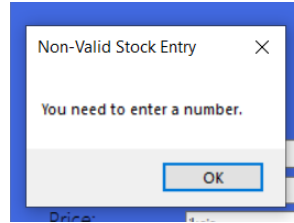
Ekranın sağ orta kısmında bulunan “Add New Item” yazısı altında bulunan açılır kutudan eklenmek istenen ürünün kategorisi seçilir. Daha sonra ürünün “name”, “price” ve “stock” bilgileri girilerek “Add” butonuna basılır ve yeni ürün eklenir. Kutucuklara girilen bilgilerin uygun formatta olup olmadığı uygulama tarafından kontrol edilir. Uygun formatta olmayan bilgiler için Görsel 12, 13 ve 14’teki hata mesajları gösterilir.



Görsel 12. İsim bilgisinin uygun formatta girilmemesi durumunda gösterilen hata mesajı



Görsel 13. Fiyat bilgisinin uygun formatta girilmemesi durumunda gösterilen hata mesajı



Görsel 14. Stok bilgisinin uygun formatta girilmemesi durumunda gösterilen hata mesajı

## 3.2 Veri Tabanı

### 3.2.1 Veri Tabanına Bağlantı

Bu bölümde kullandığım veri tabanına uygulamamı nasıl bağladığımı ve veri tabanında bulunan tabloları inceleyeceğim.

Uygulamamı SQL Server'ına bağlayabilmek için Dapper adı verilen bir NuGet paketi kullandım. Bu paketi Visual Studio'da projemin bulunduğu "Solution Explorer" kısmından projemin üzerine sağ tıklayarak "Manage NuGet Packages..." seçeneğini seçtiğimizde açılan pencereden indirdim. Bu paket sayesinde veri tabanına erişme, veri tabanından bilgi çekme veya veri tabanına yeni bilgi ekleme işlemleri oldukça kolaylaştı. Bu işlemleri nasıl gerçekleştirdiğimi Görsel 15,16 ve 17'de sizinle paylaşıyorum.

```
<connectionStrings>
  <add name="VendingMachineDB" connectionString="Server=.;Database=VendingMachine;Trusted_Connection=True;"
    providerName="System.Data.SqlClient" />
  <add name="VendingMachine2.Properties.Settings.VendingMachineConnectionString" />
</connectionStrings>
```

Görsel 15. Proje dosyalarından olan App.config'de bulunan "Connection String"

"Connection String" veri tabanına bağlanmak için gerekli bilgileri içeren böylece veri tabanı ve uygulama arasındaki bağlantıyı sağlayan bir "string"tir[7].

```
8 namespace VendingMachineLib
9 {
10     public static class Helper
11     {
12         public static string CnnVal(string name)
13         {
14             return ConfigurationManager.ConnectionStrings[name].ConnectionString;
15         }
16     }
17 }
```

Görsel 16. "VendingMachineLib" isimli sınıf kütüphanesinde bulunan "Helper" isimli sınıf

"Helper" adını verdiğim sınıfı kullanarak Görsel 15'te oluşturduğum "Connection String"i uygulamamda istediğim şekilde çağırıp kullanabiliyorum.

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Dapper;
7  using System.Data;
8  using System.Data.SqlClient;
9
10 namespace VendingMachineLib
11 {
12     public class DataAccess
13     {
14
15         5 references
16         public List<Foods> GetFoods()
17         {
18             using (IDbConnection connection = new System.Data.SqlClient.SqlConnection(Helper.CnnVal("VendingMachineDB")))
19             {
20                 return connection.Query<Foods>($"select * from Foods").ToList();
21             }
22         }
23     }
24 }

```

Görsel 17. “VendingMachineLib” isimli sınıf kütüphanesinde bulunan “DataAccess” isimli sınıf

Görsel 17’de ilk 21 satırı görünen “DataAccess” isimli sınıf uygulama kullanılırken istenilen yerde istenilen şekilde bilgilerin ekrana çağırılması, bilgilerin güncellenmesi veya yeni bilgi eklenmesi için kullanılan metotları içermektedir. Bu metotların çalışabilmesi için gerekli olan veri tabanı bağlantısının yapılabilmesi için de Görsel 16’da gösterilen “Helper” sınıfı kullanılmaktadır.

“DataAccess” sınıfındaki metotların kullanımına örnek olarak Görsel 17’de 15-21.satırlarda bulunan “GetFoods()” metodu gösterilebilir. Bu metot kullanılarak uygulamanın istenilen yerinde veri tabanında bulunan “Foods” tablosundaki ürünlerin bilgileri liste olarak çağırılabilir.

### 3.2.2 Veri Tabanında Bulunan Tablolar

Görsel 18, 19 ve 20’de “VendingMachine” isimli veri tabanında bulunan tabloları görebilirsiniz. Bu ekran görüntüleri veri tabanını kontrol edebilmek için kullandığım SQL Server Management Studio 18 uygulamasından alınmıştır.

ELIF-PC.VendingMachine - dbo.Users		ELIF-PC.VendingMachine - dbo.Foods	
Column Name	Data Type	Allow Nulls	
id	int	<input type="checkbox"/>	
username	nvarchar(15)	<input type="checkbox"/>	
balance	money	<input type="checkbox"/>	
		<input type="checkbox"/>	

Görsel 18. “Users” isimli tablo

ELIF-PC.VendingMachine - dbo.Drinks		ELIF-PC.VendingMachine - dbo.Users	
Column Name	Data Type	Allow Nulls	
id	int	<input type="checkbox"/>	
name	nvarchar(15)	<input type="checkbox"/>	
price	money	<input type="checkbox"/>	
stock	int	<input type="checkbox"/>	
		<input type="checkbox"/>	

Görsel 19. “Drinks” isimli tablo

ELIF-PC.VendingMachine - dbo.Users		ELIF-PC.VendingMachine - dbo.Foods	
Column Name	Data Type	Allow Nulls	
id	int	<input type="checkbox"/>	
name	nvarchar(15)	<input type="checkbox"/>	
price	money	<input type="checkbox"/>	
stock	int	<input type="checkbox"/>	
		<input type="checkbox"/>	

Görsel 20. “Foods” isimli tablo

“Users” tablosu uygulamayı kullanan kullanıcıların isim ve bakiye bilgilerini tutmak için kullanılmaktadır. “Foods” ve “Drinks” tabloları uygulamada bulunan ürünlerin isim, fiyat ve stok bilgilerini tutmaktadır.

Bu tablolarda bulunan bilgiler Görsel 21, 22 ve 23’te gösterilmektedir.

ELIF-PC.VendingMachine - dbo.Users			
	id	username	balance
▶	1	Elif	19.0000
	2	Nazli	17.0000
	3	Deniz	25.0000
	4	Ata	42.0000
*	NULL	NULL	NULL

Görsel 21. “Users” tablosu içinde bulunan bilgiler

id	name	price	stock
1	Sandwich	25.0000	17
4	Cookie	5.0000	4
5	Chocolate B...	4.0000	18
6	Chips	6.0000	30
15	Crackers	3.0000	15
35	Potato	5.0000	5
* NULL	NULL	NULL	NULL

Görsel 22. “Foods” tablosu içinde bulunan bilgiler

id	name	price	stock
1	Water	2.0000	7
2	Milk	3.7500	10
3	Coke	5.0000	20
4	Sprite	5.0000	12
5	Orange Juice	8.7500	13
6	Chocolate ...	3.0000	2
7	Apple Juice	5.0000	5
* NULL	NULL	NULL	NULL

Görsel 23. “Drinks” tablosu

içinde bulunan bilgiler

### 3.3 Kaynakça

[7] <http://csharp.net-informations.com/ado.net/csharp-connection-string.htm#:~:text=Connection%20String%20is%20a%20normal,between%20Database%20and%20the%20Application.>

## BÖLÜM IV – PROJEDEKİ HATA VE EKSİKLİKLER

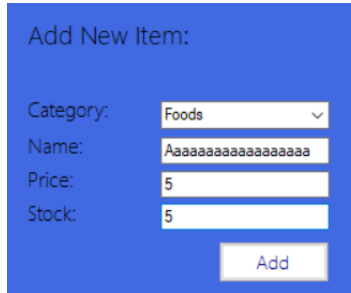
Bu bölümde hazırladığım projede oluşan sebebini bildiğim ve çözüm üretemediğim veya sebebini bilmediğim hataları inceleyeceğim.

### 4.1 Yeni Ürün Ekleme Aşamasında Oluşan Hata

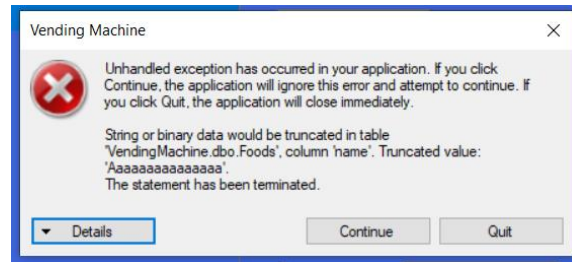
Görsel 19 ve 20’de görüldüğü üzere, uygulamamda bulunan ürünlerin “name” bilgisi “nvarchar(15)” tipindedir. Bu tipteki değişkenler içinde en fazla 15 karakter tutabilirler. Görsel 11’de bulunan yönetici ara yüzündeki yeni ürün ekleme kısmında isim bilgisi olarak 15 karakterden fazla karakter içeren bir “string” girilmesi durumunda uygulama hata vermektedir. Bu hata Görsel 24 ve 25’te gösterilmektedir. Bu hatanın çözümü için isim bilgisinin alınması sırasında



kullanılan “Regex” deseni deđiřtirilmelidir. řu anda kullanılan “Regex” deseni de Grsel 26’da gsterilmektedir.



Grsel 24. Yeni rn ekleme kısmında rn ismi olarak 15 karakterden daha uzun bir “string” girilmiřtir.



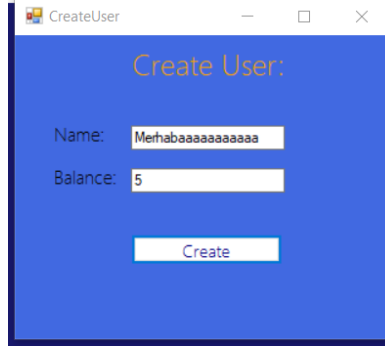
Grsel 25. Fazla karakter girilmesi sonucunda oluřan hata

```
Regex rgxname = new Regex(@"^([A-Z]\w+)$|([A-Z]\w* [A-Z]\w+)$");
```

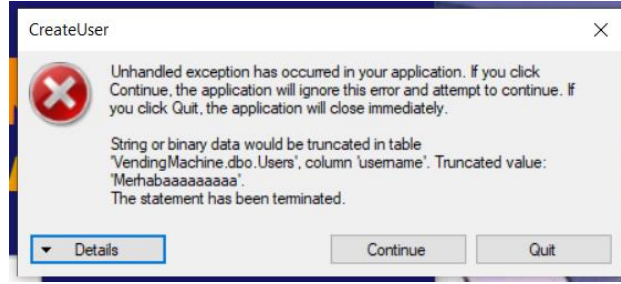
Grsel 26. “Regex” deseni

## 4.2 Yeni Kullanıcı Ekleme Ařamasında Oluřan Hata

Grsel 2’de bulunan yeni kullanıcı ekleme kısmında oluřan bu hata 4.1. blmde bahsedilen hatayla aynı sebepten kaynaklanmaktadır. Hatanın oluřumu Grsel 27 ve 28’de gsterilmektedir.



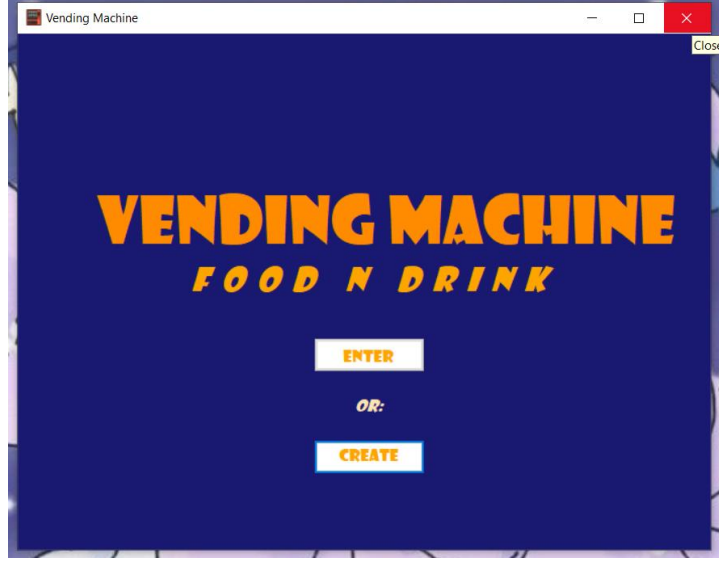
Görsel 27. Yeni kullanıcı ekleme sırasında isim bilgisi için 15 karakterden fazla karakter girilmesi durumu



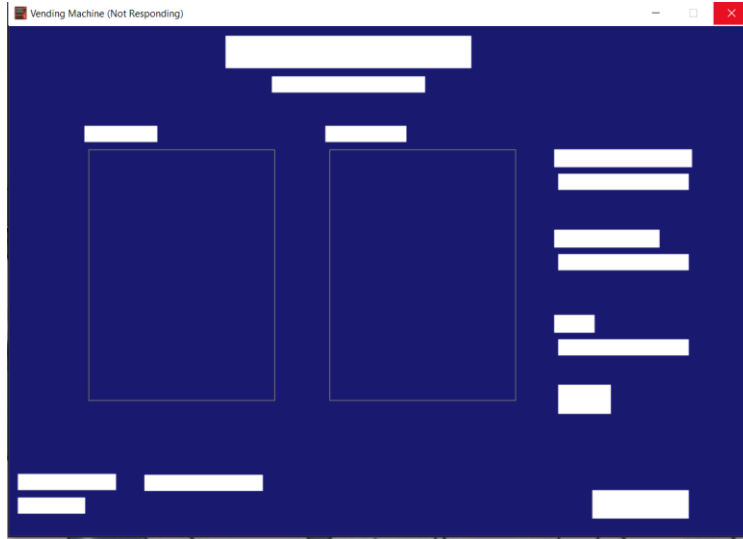
Görsel 28.Fazla karakter girilmesi sonucunda oluşan hata

#### 4.3 Uygulamanın Giriş Ekranının Kapatılması Sonucu Oluşan Hata

Görsel 1’de gösterilen giriş ekranının, ekranın sağ üst köşesinde bulunan çarpı işaretine basılarak kapatılmaya çalışılması sonucu uygulama yanıt vermemektedir. Bu hatanın gerçekleşmesi Görsel 29 ve 30’da gösterilmektedir.



Görsel 29. Giriş ekranının sağ üst köşesinde bulunan çarpı işaretine basılmaktadır.



Görsel 30.

Uygulama yanıt

vermemektedir.

4.1 ve 4.2’de belirttiğim hataların aksine bu hatanın nedenini saptayamadım, bu yüzden bir çözüm yöntemi geliştiremedim.

## BÖLÜM V – SONUÇ VE DEĞERLENDİRME

Bu projenin gerçekleştirilmesi sırasında *Giriş* bölümünde belirttiğim, kazanmayı hedeflediğim beceriler konusunda kendimi geliştirdiğime inanıyorum. Bu proje için hazırladığım uygulamayı yazmaya başlamadan önce veri tabanlarının nasıl çalıştığı konusunda hiçbir bilgi sahibi değildim ama bu uygulamayı yazarken SQL Server Management Studio yardımıyla veri tabanında bulunan bilgileri nasıl kullanabileceğimi öğrendim ve bu süreçte veri tabanı mantığıyla ilgili bilgiler edindim. Ayrıca veri tabanını oluşturmak ve kontrol etmenin yanı sıra, oluşturduğum bu veri tabanını C# dilinde yazılmış bir uygulamaya nasıl bağlayacağımı ve verimli bir uygulama oluşturmak için bu bağlantıyı nasıl kullanabileceğimi kavramış oldum.

Uygulamanın yazımı sırasında C# dilinin kullanımıyla ilgili BM102 dersi sırasında öğrendiğim bilgileri kullanmak ve kod yazmak konusunda pratik yapmış oldum ama bu projenin bana C# kullanımından daha çok veri tabanı kullanımıyla ilgili bilgiler kattığını düşünüyorum.

Yiyecek ve içecek otomatı uygulamam “Foods”, “Drinks” ve “Users” olmak üzere 3 tane veri tabanı tablosundan oluşmaktadır ve bu tablolarla aynı isimdeki sınıflar, uygulamaya ait olan sınıf kütüphanesinde bulunmaktadır. Bu tablolar ve sınıflar arasındaki veri tabanı bağlantısı yine uygulamaya ait olan sınıf kütüphanesindeki “DataAccess” sınıfındaki metotlarla gerçekleştirilmektedir.

Uygulamaya giriş kısmında kullanıcı yeni bir kullanıcı oluşturabilir ya da uygulamaya giriş yaparak var olan kullanıcılardan birini seçerek bu kullanıcının bakiyesi dâhilinde alışveriş yapabilir. Uygulamadaki ürünler veri tabanındaki “Foods” ve “Drinks” tablolarından çağırılarak kullanıcıya gösterilir. Kullanıcının yaptığı satın alma işlemi sonucunda kullanıcının bakiyesi ve ürün stoku gereken şekilde güncellenir. Yönetici ara yüzünden, yönetici isterse bu ürünlerin stokunu yenileyebilir veya stoka yeni ürün ekleyebilir.

Hazırladığım uygulama yiyecek ve içecek otomatlarının bir benzerini gerçekleştirse de bu otomatlarda bulunmayan “Create User” ve “Choose User” gibi farklı kullanıcılar için farklı bakiyeler girilmesi kısımlarını da içermektedir. Bu kısımları eklememin sebebi, uygulamaya bilgisayar ortamında farklı kullanıcılar tarafından kullanılacağı durumlar için kolaylık sağlamak ve de veri tabanına bilgi kaydetmek ve veri tabanından bilgi çekmek konusunda daha çok pratik yapabilmektir.

Uygulamayı hazırlarken yazdığım kodlarda, gereksiz ve fazladan kod kullanımı olduğunu düşünmekteyim ama bu proje esnasında daha çok veri tabanı kullanımını öğrenmeyi hedeflediğim için yazdığım bu fazladan kodların nasıl daha kısa ve kullanışlı bir şekilde yazılabileceği konusunda pek fazla araştırma yapmadım. İlerleyen süreçte bu konuda araştırma yaparak uygulama kodlarını çok daha verimli hâle getirebileceğimi düşünmekteyim.

Uygulamamda bulunan –ve uygulamayı deneme aşamalarında fark edebildiğim– hataları *Bölüm 4’te* açıkladım. İlerleyen süreçte bu hataların giderilmesiyle uygulamanın çok daha kullanışlı olabileceğini düşünmekteyim.