



FACULTY OF ENGINEERING
COMPUTER SCIENCE AND ENGINEERING
DEPARTMENT

CSE 492
SENIOR DESIGN PROJECT
FINAL REPORT

SUPERVISOR
Prof. Dr. MELİH GÜNAY

Enes DEMİRTAŞ
20190808018

Süleyman Türker GÜNER
20200808073

June 6, 2023

Contents

1	Introduction	3
1.1	Purpose	3
1.2	Product Scope	3
1.3	References	3
2	Overall Descriptions	3
2.1	Product Perspective	3
2.2	Product Functions	3
2.3	User Types and Characteristics	4
2.4	Operating Environment	4
2.5	Design and Implementation Constraints	4
2.6	User Documentation	4
2.7	Assumptions and Dependencies	4
2.8	Technology Selection	4
2.8.1	PHP versus ASP.NET	5
2.8.2	MySQL	5
2.8.3	HTML, CSS, and JavaScript	5
2.8.4	Laravel	6
3	External Interface Requirements	7
3.1	User Interfaces	7
4	System Features	21
4.1	User Class 1 - Administrator	21
4.2	User Class 2 - Customer	22
4.3	User Class 3 - Field Worker	22
4.4	Electronic Signature	23
5	Nonfunctional System Requirements	24
5.1	Performance Requirements	24
5.2	Safety Requirements	26
5.3	Security Requirements	26
5.4	Organizational Requirements	26
5.5	Usability Requirements	26
6	Use Cases	26
6.1	Creating a New Account	26
6.2	Login	26
6.3	Add a Customer	27
6.4	Add an Employee	27
6.5	Delete an Employee	27
6.6	Delete a Customer	28
6.7	Edit Customer Information	28
6.8	Edit Employee Information	28
6.9	View an Offer	28
6.10	Delete an Offer	29
6.11	Edit an Offer	29
6.12	Send an Offer	29
6.13	Edit Profile Information	30
6.14	Fill Form	30
7	Interaction Models	31
7.1	Use Case Models	31
7.2	Sequence Diagrams	32
7.3	Class Diagrams	33
8	Further Development	34

1 Introduction

1.1 Purpose

This document shows a detailed planning and description of the requirements for the “Customer Management System” application. It will illustrate the aimed and fully declaration for the development of the software. It will explain system constraints, interface, and interactions. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

1.2 Product Scope

The customer management system is a conventional concept. This project is an application based on this concept but specially prepared for SBE Engineering. Basically, it is planned to have 3 roles in practice. These are admin, customer, field worker. The Admin role is considered as a superior role that can use all the features in the system. As can be seen from this, all roles will have different features available to them. For example, the customer will not be able to delete another customer. Field worker is employee role in the customer management system. The main purpose of the customer management system is to manage the offers over the system, to handle the work that would normally be done on paper by the employee who goes to the field work. In addition, being able to view these processes through the system will be a great convenience for the owner of the customer management system(i.e. the admin user). It is also an advantage for the company that the data will be recorded in a database rather than being on paper.

1.3 References

[CRM Demo - CRM Software Demo - Free CRM System Demo](#) : To understand how should a Customer Management System be designed, we are inspired by this demo. Although this software does not exactly fit to our desired system, we understood the basic requirements of a Customer Management System. For example, it should be easy to use, contains related features together, looks user-friendly, etc.

[CRM Software — Agile CRM](#) : This software dashboard inspires us. Its dashboard contains essential information in good form. Graphs and tables are used to show the numbers that are so important that the users check frequently. Showing these numbers in the dashboard is a best practice of Customer Management Systems.

2 Overall Descriptions

2.1 Product Perspective

Our platform is a self-contained product so that we are not using any external API except MERNIS. This software is designed to be a Customer Management System and will be a Web application. There will be 2 different login types and 3 different user roles. The roles are Manager/Admin, field worker, and customer. The first 2 user types login using their TCKN and password while a customer logins using his Tax Number/User ID and password. From the view of the system owners, they will manage and view customer records, periodic maintenance of customers’ machines, and make offers to the customers. Besides that, the general manager/admin can keep track of his employees. Although an admin has full access to system. When field worker login to the system, he/she can see all reservations from offers of customers. They select the reservation and fill the necessary forms to complete periodic maintenance then they send this form. So the process will be completed.

2.2 Product Functions

- Briefly, owner of the Customer Management System can manage and view their customer records while customers can only view their own transaction/operation histories, debit balances, payment details, invoices, next periodic maintenance date, information card, etc.
- A field worker can fill a maintenance form, create a report according to maintenance results, and view the offers without seeing the offer price, because that’s not a necessary information for a field worker.

- A general manager or admin, has full access to all these features. Additionally, he can manage the company's employees.

2.3 User Types and Characteristics

There are three types of users for our system: admin, customer, and field worker. Each has a different use of the system. Since they have different uses of the system they have their own responsibilities and requirements. Our project, Customer Management System, appeals to a wide range of the sector. For example, a propane cylinder shop can use this system to keep track of the sales as well as its customers can view their purchasing and payment histories. For the first version, we are developing the application for a machine engineering company. They do periodic maintenance of machines in factories and warehouses. However, application will be scalable and customizable at some point. The user defined as an admin in the system has unlimited operation authority. In short, the admin user is the owner of the customer management system. Customers are the type of users who are not allowed to make changes in the system. Customers usually consist of sole proprietorships or corporate companies. The field worker is the role that provides periodic maintenance to customers. He/she reports what he/she sees in the field. Fills out the necessary forms and record them in the system.

2.4 Operating Environment

The first version of this Customer Management System Web application will only be for Web and mobile browsers. All user types can log into the system and use it on all modern Web and mobile browsers. Due to simplicity of our application, it will support almost all of the devices and browsers.

2.5 Design and Implementation Constraints

The Internet connection is a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function. For example, add/view an offer, general CRUD operations, etc. The system will be constrained by the capacity of the database. it may be forced to queue incoming requests and therefore increase the time it takes to fetch data.

2.6 User Documentation

Our main goal is to develop a user interface that is clear and easy so that every person who knows how to use mouse and keyboard can understand the system by reading the texts on the screen. Nevertheless, we planned to add question marks that show what kind of features are available on each page. Thanks to these question marks, a new user can realize the possible actions on the page.

2.7 Assumptions and Dependencies

One and most important assumption about the product is that it can always be used where there is sufficient internet. When there is no internet connection, it is not possible to log in to the system and no operation can be performed. Another assumption is that since the project is a web application, the user has a mobile device or computer with the appropriate hardware, capable of opening a web browser.

2.8 Technology Selection

Technology selection is an important part of a project from the very beginning. To get the best results at the end of the project, we must choose the proper technologies at the beginning of the project. From this point of view, we considered some crucial points under this phase of the project. Some of them are:

- Project size and complexity
- Budget
- Size of the development team

- Project requirements
- Customer expectations
- Skills of the development team
- Production environment
- Performances of the considered technologies

We reviewed all these points and decided to use proper technologies for the Customer Management System.

2.8.1 PHP versus ASP.NET

ASP.NET and PHP are two of the internet's most used technologies and there exist very important differences between them. The most important difference is that one of them has developed by Microsoft. Besides that, PHP was created by Ramus Lerdorf, and it is a client-side scripting language. Using ASP.NET, people can develop websites, Web applications, and Web services. It provides people to develop scalable Web applications. On the other side, PHP is an open-source scripting language that runs on server-side. It makes development of dynamic Web applications easy.

ASP.NET runs on the Microsoft servers better than Linux-based servers because it is developed by the Microsoft. On the other hand, PHP can run on all kinds of servers even IIS which is a Microsoft product. PHP can be used all the major operating systems i.e., Microsoft Windows, Open BSD, and MacOS. PHP source code provides faster script engine. Also, code optimization tools may increase the compiled code's performance. Hosting prices of the PHP is lower than the ASP.NET.

We took the Web Development course at the second year of the university. We learned how to use HTML, CSS, JS, and PHP thanks to that course given by the Prof. Dr. Melih Gunay. Therefore, we found it appropriate to use PHP as the main language of the project.

2.8.2 MySQL

As the almost all Web applications need a database, our Customer Management System also needs a database. As has been told in the previous part, we took Web Development course in the university and we learned how to use MySQL as a database in our projects. For this reason, we are familiar with the MySQL.

MySQL was released in 1995 as a RDBMS, Relational Database Management System. MySQL is the most popular open-source RDBMS in the internet. As its name suggests, its core is the basic SQL, Structured Query Language. MySQL is free and open source, that's why it is still preferred by most people. It is way simpler than the other databases. It works with all programming languages. We can get high performance even with big data.

2.8.3 HTML, CSS, and JavaScript

For the UI part of Customer Management System, we decided to use the simplest tech stack, HTML, CSS, and JavaScript. We are familiar with them because of the Web Development course. We used these technologies on our old projects. Besides that, our Product Owner, Mehmet Bey, also wanted us to use these Frontend technologies because of their simpleness by their natures. For these reasons, we decided to use them.

HTML, HyperText Markup Language, is the most popular and basic markup language. Technically, it is not a programming language because there cannot exist a program that is written with HTML and working standalone. HTML is used to create Web pages and browsers handle it and transform the HTML content into the visual interfaces for Web users. HTML files are stored with the extension .html or .htm. HTML is standardized by the W3C. HTML language basically performs operations such as placing and positioning items such as text, images, and videos on the page, displaying these pages properly in web browsers, and providing information about the page to search engines.

CSS, Cascading Style Sheets, like HTML and JavaScript, is one of the 3 main technologies used in coding web pages. CSS describes how to display an HTML page. In this recipe, besides the visual properties of the elements on the page such as title, text, image, video, there is also information about the page layout and how this layout will change on different screens, devices,

screen, paper or other media. CSS implies these styling properties to the HTML elements and changes their visual representations.

JavaScript is a scripting language that is developed by Brendan Eich in 1995. It is designed to run on the browsers, add dynamism into the web pages, and increase the interaction with the user. JavaScript codes are interpreted by an interpreter. There are 3 basic reasons that make JavaScript precious:

- Working in harmony with HTML and CSS
- It comes as a default in all the most known browsers and works without any problems
- Enabling you to do easy tasks in an easy way

Although it was originally a language designed to work only in web browsers, it can now run in many places such as client-side and native mobile applications. It has no equal due to its seamless compatibility with HTML and CSS. Although it was seen as an incompetent sub-language of the Java language at first, it is a very advanced programming language today and has nothing to do with Java.

All these reasons make us to tend to use the JavaScript for our Customer Management System. We are familiar with the JavaScript already. Therefore, we used its power in our project.

2.8.4 Laravel

Laravel is an awesome Web framework of the PHP language. As a developer, we do not want to code every requirements of a Web application because some of the basic parts are common in almost all Web applications, i.e., authentication, authorization, catching requests and sending responses, etc. To avoid manually typing all these required common piece of codes over and over, we use Web frameworks to handle that kinds of features. For PHP, Laravel is the best and most popular option as a Web framework. Even though people say “PHP is a dead language”, Laravel still shows that the PHP is alive and well. A few of the great features of the Laravel are as follows:

- Laravel uses Composer as its Dependency Manager. Composer is a tool that contains and manages packages called classes, components or plugins while developing applications.
- Laravel has an ORM in it. ORM or Object Relational Mapping is the creation of a model against each table in your database. You do not need to write long queries with ORM, you can handle object and structure with ORM.
- You can make a pagination system with very short piece of codes.
- Authentication system comes ready with Laravel. You can add the authentication system to your own project with simple configurations.
- Routing system is also available in Laravel. With the routing structure, you can present a more performance and scalable project to the user.

3 External Interface Requirements

3.1 User Interfaces

- Login

HOŞGELDİNİZ

Burada yeni misiniz? [Yeni bir hesap oluşturun](#)
Hizmetlerimizden yararlanmak mı istiyorsunuz? [Tıkla! Oluşturun](#)

Sifreyi Göster ☐

☐ Oturumu açık tut

[Sifrenizi mi unuttunuz?](#)

© 2022 Tüm hakları saklıdır. [DAKİK](#)



Figure 1: Login Page

- Signup

Anında yeni hesabınızı oluşturun

Zaten bir hesabınız var mı? [Giriş yapın](#)

☐ Üyelik koşullarını kabul ediyorum.

Sifreyi göster ☐

© 2022 Tüm hakları saklıdır. [DAKİK](#)



Figure 2: Signup Page

- Forgot Password

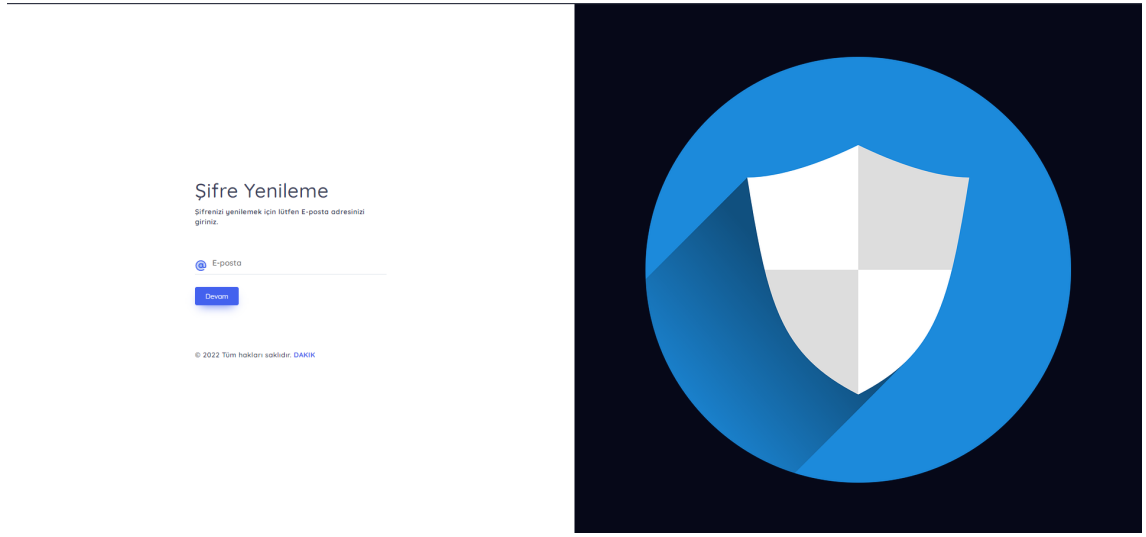


Figure 3: Forgot Password Page

- Homepage

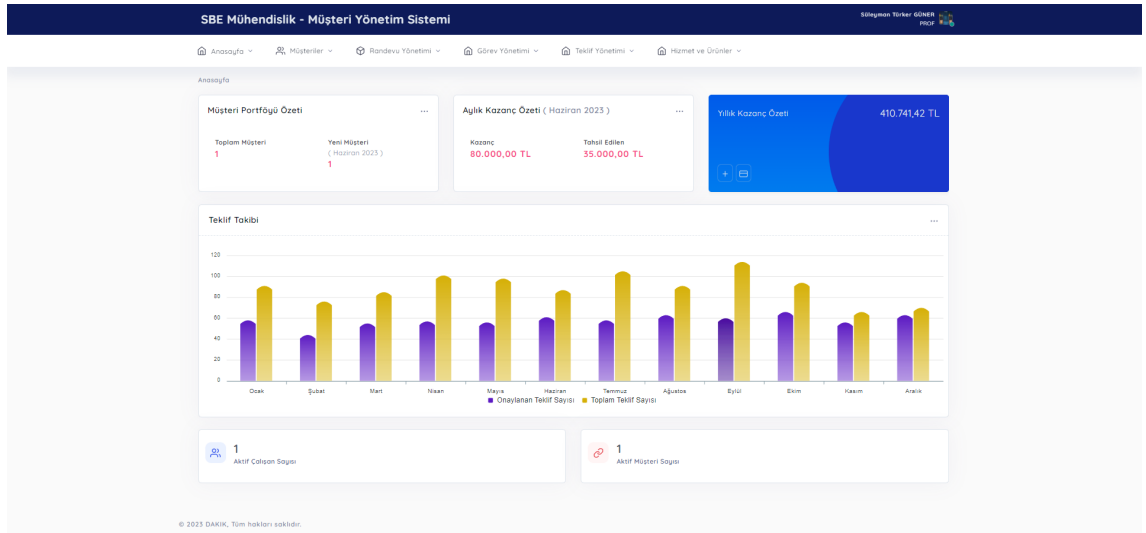


Figure 4: Homepage

- Customers

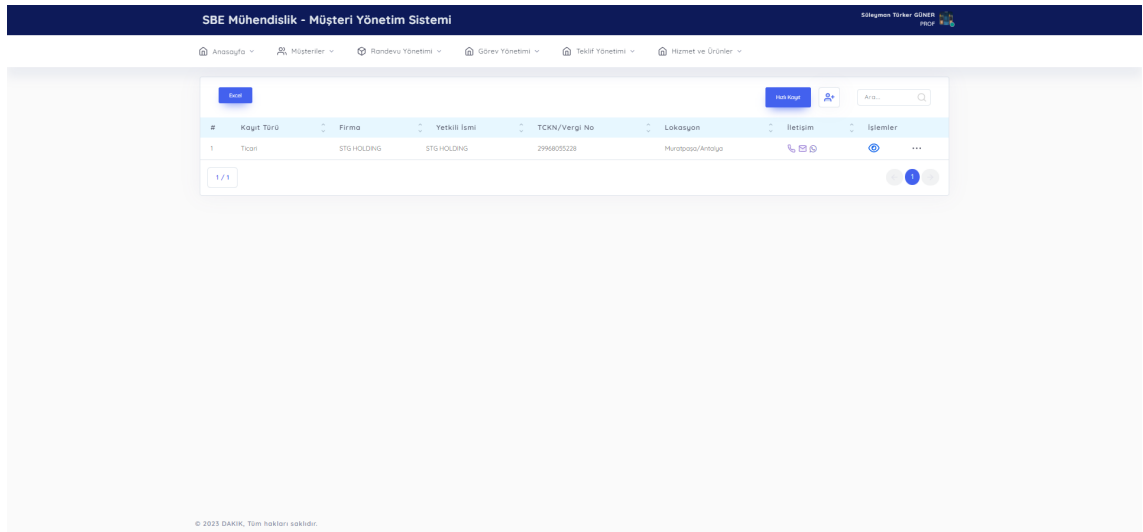


Figure 5: Customers Page

- Customer's Information - Model

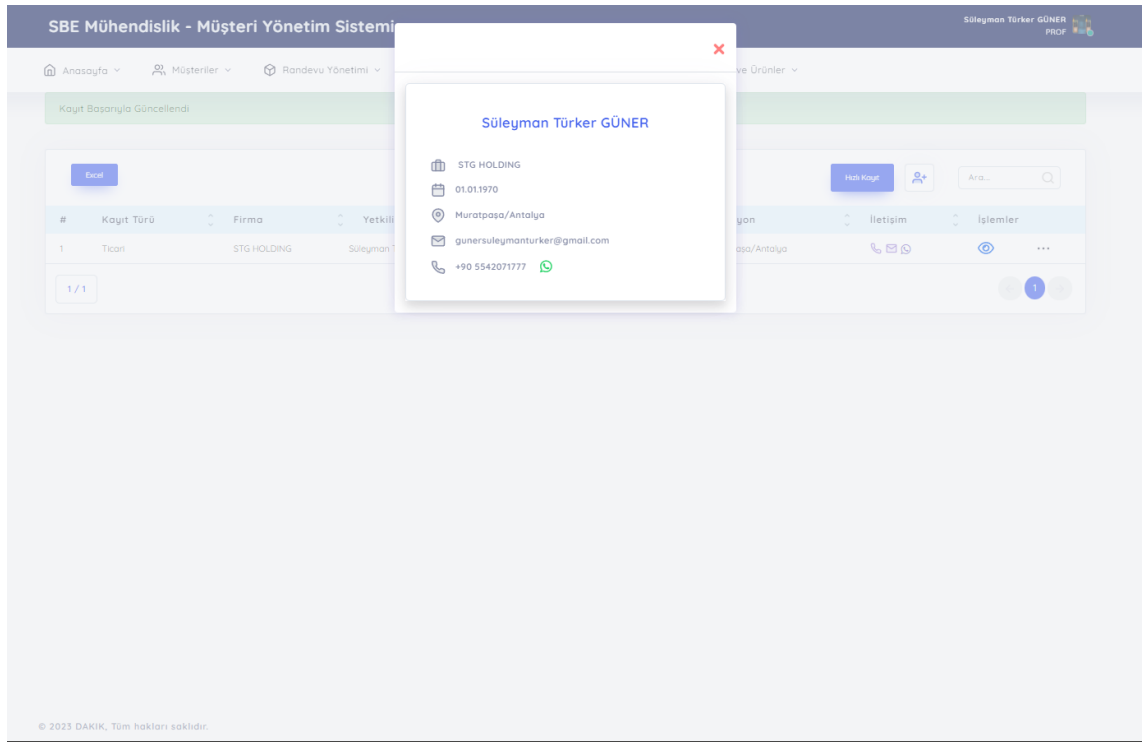


Figure 6: Customer's Information - Model

- Edit Customer Information - 1

SBE Mühendislik - Müşteri Yönetim Sistemi

Süleyman Türker GÜNER PROF

Anasayfa Müşteriler Randevu Yönetimi Görev Yönetimi Teklif Yönetimi Hizmet ve Ürünler

1 Ön Bilgiler 2 Banka ve İletişim Bilgileri 3 Adres Bilgileri

Ticari Tüzel Kişi STG HOLDING ÇALIŞAN

29968055228 29968055228 Süleyman Türker GÜNER

STG HOLDING BİLGİSAYAR MÜHENDİSİ 02.02.2000

Önceki Sonraki

Figure 7: Edit Customer Information First Page

- Edit Customer Information - 2

SBE Mühendislik - Müşteri Yönetim Sistemi

Süleyman Türker GÜNER PROF

Anasayfa Müşteriler Randevu Yönetimi Görev Yönetimi Teklif Yönetimi Hizmet ve Ürünler

1 Ön Bilgiler 2 Banka ve İletişim Bilgileri 3 Adres Bilgileri

YAPIKREDİ IBAN +90 5542071777 5542071777

gunersuleymanturker@gmail.com Web Site Faks

Önceki Sonraki

Figure 8: Edit Customer Information Second Page

- Edit Customer Information - 3

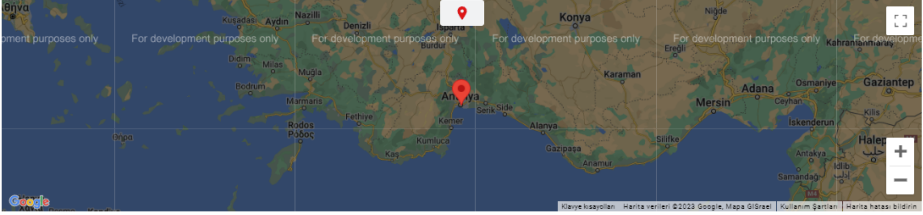
SBE Mühendislik - Müşteri Yönetim Sistemi Süleyman Türker GÜNER PROF

Anasayfa Müşteriler Randevu Yönetimi Görev Yönetimi Teklif Yönetimi Hizmet ve Ürünler

1 Ön Bilgiler 2 Banka ve İletişim Bilgileri 3 Adres Bilgileri

ÇAYBAŞI Notlar

AKDENİZ Antalya Muratpaşa



Önceki Güncelle

Figure 9: Edit Customer Information Third Page

- Add Customer - 1

SBE Mühendislik - Müşteri Yönetim Sistemi Süleyman Türker GÜNER PROF

Anasayfa Müşteriler Randevu Yönetimi Görev Yönetimi Teklif Yönetimi Hizmet ve Ürünler

1 Ön Bilgiler 2 Ödeme Bilgileri 3 Adres Bilgileri

Ticari Müşteri Türü Marka Adı Şube Adı

Vergi Dairesi Vergi No Kısaltma Firma Tam Unvan

Unvan Devamı Ticaret Sicil No Oda Sicil No Mersis No

Telefon E-Posta Web Site Faks

+90 Cep Telefonu

Önceki Sonraki

Figure 10: Add Customer First Page

- Add Customer - 2

SBE Mühendislik - Müşteri Yönetim Sistemi

Süleyman Türker GÜNER PROF

Anasayfa Müşteriler Randevu Yönetimi Görev Yönetimi Teklif Yönetimi Hizmet ve Ürünler

1 Ön Bilgiler 2 Ödeme Bilgileri 3 Adres Bilgileri

Banka Adı IBAN Notlar

Önceki Sonraki

Figure 11: Add Customer Second Page

- Add Customer - 3

SBE Mühendislik - Müşteri Yönetim Sistemi

Süleyman Türker GÜNER PROF

Anasayfa Müşteriler Randevu Yönetimi Görev Yönetimi Teklif Yönetimi Hizmet ve Ürünler

1 Ön Bilgiler 2 Ödeme Bilgileri 3 Adres Bilgileri

Adres Bölge Lütfen Bir İl Seçin Lütfen Bir İlçe Seç

Önceki Müşteri Ekle

Figure 12: Add Customer Third Page

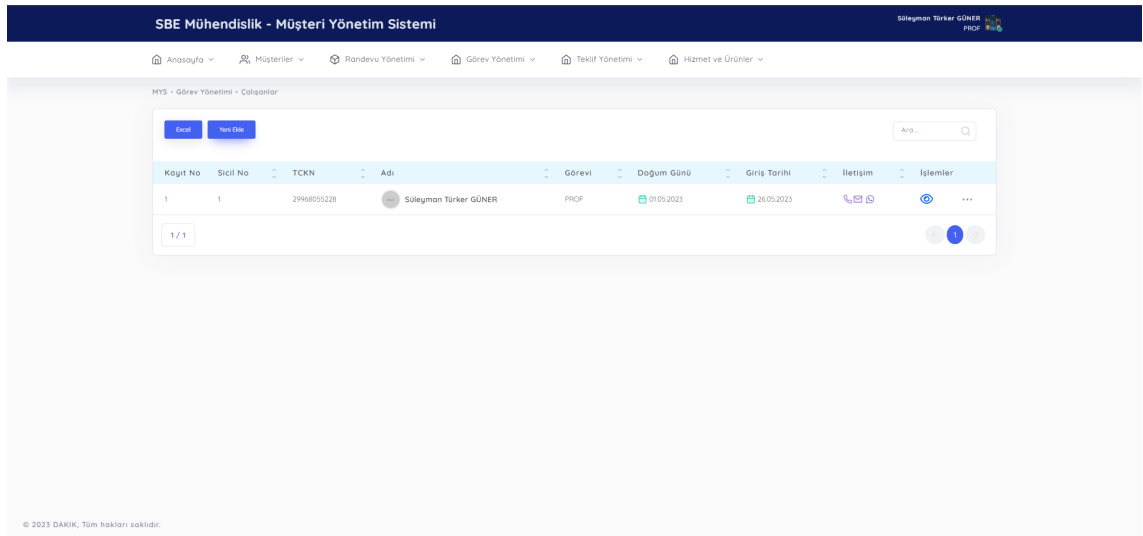
- Quick Customer Registration - 1

Figure 13: Quick Customer Registration First Page

- Quick Customer Registration - 2

Figure 14: Quick Customer Registration Second Page

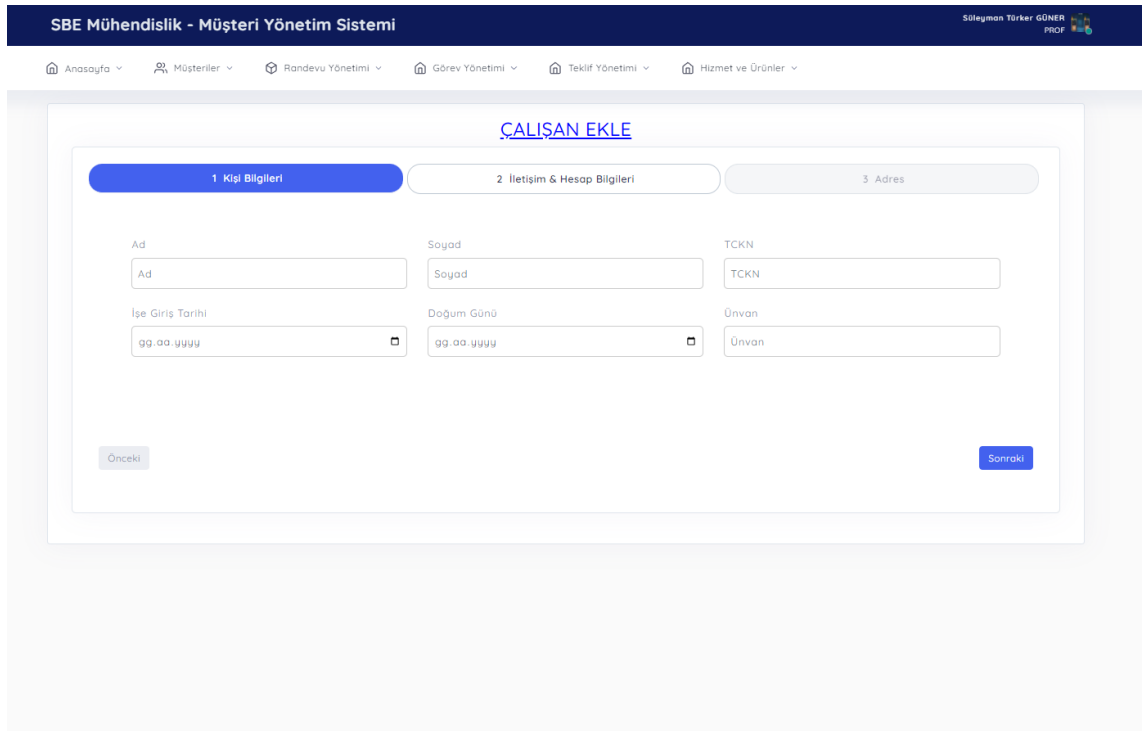
- Employees



Kayıt No	Sicil No	TCKN	Adı	Görevi	Doğum Günü	Giriş Tarihi	İletişim	İşlemler
1	1	2996805228	Süleyman Türker GÜNER	PROF	01.05.2023	26.05.2023	📞 📧 📱	🔍 ⚙️

Figure 15: Employees

- Add Employee - 1



ÇALIŞAN EKLE

1 Kişi Bilgileri | 2 İletişim & Hesap Bilgileri | 3 Adres

Ad:

Soyad:

TCKN:

İşe Giriş Tarihi:

Doğum Günü:

Ünvan:

Önceki | Sonraki

Figure 16: Add Employee First Page

- Add Employee - 2

Figure 17: Add Employee Second Page

- Add Employee - 3

Figure 18: Add Employee Third Page

- Employee's Information - Model

The screenshot displays the 'SBE Mühendislik - Müşteri Yönetim Sistemi' interface. A modal window titled 'Süleyman Türker GÜNER' is open, showing the following details:

- PROF**
- 01.05.2023**
- ANTALYA/MURATPAŞA**
- gunersuleymanturker@gmail.com**
- 90 554 207 17 77**

The background shows a table with columns: Kağıt No, Sicil No, TCKN, and Adı. The first row contains the values: 1, 1, 29968055228, and Süleyman.

Figure 19: Employee's Information - Model

- Edit Employee's Information

The screenshot displays the 'SBE Mühendislik - Müşteri Yönetim Sistemi' interface. The 'ÇALIŞAN BİLGİLERİNİ GÜNCELLE' (Update Employee Information) form is open, showing the following details:

- Kişisel Bilgiler**
 - Ad: Süleyman Türker
 - Soyad: GÜNER
 - TCKN: 29968055228
 - Ünvan: PROF
 - Doğum Günü: 01.05.2023
- İletişim Bilgileri**
 - Ülke Kodu: 90
 - Telefon Numarası: 5542071777
 - Eposta: gunersuleymanturker@gmail.com
- Adres Bilgileri**
 - İl: ANTALYA
 - İlçe: MURATPAŞA
 - Adres: henüz yokk anladın
- Hesap Bilgileri**
 - Banka Adı: Yapı kredi
 - IBAN: IBAN
 - Hesap No: Hesap Numarası

The form includes a 'Güncelle' (Update) button at the bottom.

Figure 20: Edit Employee's Information

- Offers

SBE Mühendislik - Müşteri Yönetim Sistemi

Süleyman Türker GÖNER
PROF

Anasayfa

Müşteriler

Randevu Yönetimi

Görev Yönetimi

Teklif Yönetimi

Hizmet ve Ürünler

MYS » Teklif » Teklifler

Sonuçlar : 7Yeni Ekle

Ara...

Teklif Numarası	Adı	Eposta	Tarih	Miktar	Durum	İşlemler
3	asdasd	alma.clarke@gmail.com	10 Feb 2021	\$23440	Ödendi	...
4	asdasd	alma.clarke@gmail.com	10 Feb 2021	\$23440	Ödendi	...
2	son kez	alma.clarke@gmail.com	10 Feb 2021	\$23440	Ödendi	...
1	Süleyman Türker	alma.clarke@gmail.com	10 Feb 2021	\$23440	Ödendi	...

1 / 1

1

Figure 21: Offers Page

- View Offer

SBE Mühendislik - Müşteri Yönetim Sistemi

Söyleman Törker GÜNER
PROF

Anasayfa

Müşteriler

Randevu Yönetimi

Görev Yönetimi

Teklif Yönetimi

Hizmet ve Ürünler

MYS • Teklif • Önizle

Submenu 1

Submenu 2

Submenu 3

comp

XYZ Delta Street
Info@company.com
(120) 456 789

Teklif : #0001

Teklif Tarihi : 20 Aug 2020
Bitiş Tarihi : 26 Aug 2020

Teklif Veren

asd
asd
stg@gmail.com
asdfsasdfsdf

Ödeme Bilgileri:

Banka Adı: Bank of America
Hesap Numarası: 1234567890
SWIFT code: V570134
Ülke: United States

S.NO	ITEMS	QTY	PRICE	AMOUNT
1	Calendar App Customization	1	\$120	\$120
2	Chat App Customization	1	\$230	\$230
3	Laravel Integration	1	\$405	\$405
4	Backend UI Design	1	\$2500	\$2500

Ara Toplam:

İndirim : 5%

Net Toplam :

\$10

\$3845

Bizimle çalıştığınız için teşekkürler.

Yazdır

Figure 22: View Offer Page

- Appointment Management

SBE Mühendislik - Müşteri Yönetim Sistemi

MYS - Randevu Yönetimi - Randevular

Sonuçlar: 7 Yeni Ekle

Ara...

Teklif Numarası	Adı	Eposta	Başlangıç - Bitiş Tarihi	Miktar	Durum	İşlemler
1	Süleyman Türker Güner	alma.clarke@gmail.com	10.06.2023 - 16.06.2023	\$254.40	İşleniyor	...

1 / 1

© 2023 DAKIK, Tüm hakları saklıdır.

Figure 25: Appointment Management Page

- Periodic Maintenance Form Example

SBE MÜHENDİSLİK

Periyodik Bakım Formu
kalerifer_kazani

Tarih
04/06/2023

Genel Bilgiler

Kurum Adı

Faaliyet Alanı

Adresi

Telefon

E-Posta

İş Ekipmanına Ait Bilgiler

Özel Bilgiler

Yapımcı firma

Markası

Modeli/Tipi

Üretim tarihi

Seri no

Teknik Bilgiler

Hacmi

Isıtma yüzeyi

Isıtma kapasitesi

İşletme basıncı

Test basıncı

Periyodik Kontrol Metodu

Kullanılan Metod

Ölçüm Cihazı

Marka-Model

Seri No

Figure 26: Periodic Maintenance Form Example Page - 1

KONTROL	UYGUN	UYGUN DEĞİL	ONARILDI	YENİLENDİ	AÇIKLAMALAR
Yarın nasıldın?	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="text"/>
dün nasıldın?	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="text"/>
akşam nasıldın?	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="text"/>

İkaz ve Öneriler

Sonuç ve Kanaat

Yukarıda özellikleri yazılı kalorifer kazanının fenni muayenesi, kriterlere uygun olarak tarafımdan yapılmış, işçi sağlığı ve iş güvenliği mevzuatına uygun olup olmadığı tespit edilmiş olup **04/06/2024** tarihinde periyodik kontrolünün tekrar yapılması ve yukarıda zikredilen önerilerin yerine getirilmesi şartıyla BİR YIL boyunca emniyetli bir şekilde kullanılmasında bir sakınca olmadığına dair işbu rapor tanzim edilmiştir.

Kontrolü Yapanın

T.C. Kimlik No

Adı Soyadı

Mesleği

Diploma Tarihi ve No

Kurum Yetkilisinin

T.C. Kimlik No

Adı Soyadı

Unvanı

Bildir

Figure 27: Periodic Maintenance Form Example Page - 2

- Profile

SBE Mühendislik - Müşteri Yönetim Sistemi
Süleyman Türker GÜNER PROF

Anasayfa
Müşteriler
Randevu Yönetimi
Görev Yönetimi
Teklif Yönetimi
Hizmet ve Ürünler

Profil Bilgileri

Your Deck

Kişisel Bilgiler

Ad
Süleyman Türker
Soyad
GÜNER
TCKN
29968055228
Ünvan
PROF
Doğum Günü
01.05.2023

İletişim Bilgileri

Ülke Kodu
90
Telefon Numarası
5542071777
Eposta
gunersuleymanturker@gmail.com

Adres Bilgileri

İl
ANTALYA
İlçe
MURATPAŞA
Adres
henüz yok anladın

Hesap Bilgileri

Banka Adı
Yapı kredi
IBAN
IBAN
Hesap No
Hesap Numarası

Figure 28: Edit Profile Information

4 System Features

4.1 User Class 1 - Administrator

ID: FR1

TITLE: Create an account

DESC: The user must first have an account in order to log in to the system. Otherwise, they cannot log in to the system. This applies to both employees and customers.

DEP: None

ID: FR2

TITLE: Login

DESC: The system admin can enter the system as an administrator using its ID and password.

DEP: FR1

ID: FR3

TITLE: Add Employee

DESC: The system admin can add a new employee.

DEP: FR2

ID: FR4

TITLE: Delete Employee

DESC: The system admin can delete any employee from the system.

DEP: FR2

ID: FR5

TITLE: Edit Employee Information

DESC: The system admin can edit the information of an employee in the system.

DEP: FR2

ID: FR6

TITLE: Add Customer

DESC: The system admin can add a new customer.

DEP: FR2

ID: FR7

TITLE: Delete Customer

DESC: The system admin can delete any customer from the system.

DEP: FR2

ID: FR8

TITLE: Edit Customer Information

DESC: The system admin can edit a customer's information in the system.

DEP: FR2

ID: FR9

TITLE: Add Offer

DESC: The system admin can send a new offer to the customers in the system from the add offer page.

DEP: FR2

ID: FR10

TITLE: Delete Offer

DESC: The system admin can delete an offer in the system.

DEP: FR2

ID: FR11

TITLE: View Offer

DESC: The system admin can review any offer in the system.

DEP: FR2

ID: FR12
TITLE: Edit Offer
DESC: The system admin can edit any offer in the system.
DEP: FR2

ID: FR13
TITLE: View Employees
DESC: The system admin can view all employees in the system.
DEP: FR2

ID: FR14
TITLE: View Customers
DESC: The system admin can view all customers in the system.
DEP: FR2

ID: FR15
TITLE: Logout
DESC: An admin can log out of the system.
DEP: FR2

4.2 User Class 2 - Customer

ID: FR16
TITLE: Create an account
DESC: The customer must first have an account in order to log in to the system. Otherwise, they cannot log in to the system. This applies to both employees and customers.
DEP: None

ID: FR17
TITLE: Login
DESC: The customer can enter the system as an customer using its ID and password.
DEP: FR16

ID: FR18
TITLE: View Informations
DESC: The customer can view the previous period reports, transactions, payment details, debit balances, invoices, the counter for the next service purchase periodically, and information about themselves, of her company and, if any, branches.
DEP: FR17

ID: FR19
TITLE: Add offer
DESC: A customer can create an offer form and send it to the Customer Management System through the system.
DEP: FR17

ID: FR20
TITLE: Logout
DESC: A customer can log out of the system.
DEP: FR17

4.3 User Class 3 - Field Worker

ID: FR21
TITLE: Create an account
DESC: The field worker must first have an account in order to log in to the system. Otherwise, they cannot log in to the system. This applies to both employees and customers.
DEP: None

ID: FR22
TITLE: Login
DESC: The field worker can enter the system as an field worker using its ID and password.
DEP: FR21

ID: FR23
TITLE: View Reservations
DESC: The field worker can view all reservations that customers offer before.
DEP: FR22

ID: FR24
TITLE: Fill Forms
DESC: The field worker can fill form which is related to reservation.
DEP: FR22

ID: FR25
TITLE: Logout
DESC: A field worker can log out of the system.
DEP: FR22

4.4 Electronic Signature

E-Signature (electronic signature) is a legal authentication system that can be used instead of signature in electronic environments. E-Signature, which has the same legal characteristics as a wet signature, proves the accuracy of the identity information of the person in electronic transactions. e-Signature is used instead of wet signature in electronic environment. Signing is done by using certificates distributed by organizations authorized to issue e-Signature certificates. According to Electronic Signature Law No. 5070, e-Signature is the digital equivalent of a handwritten signature with the same legal validity. In Turkey, e-Signature (electronic signature) is offered by Electronic Certificate Service Providers approved by the Information Technologies and Communications Authority (BTK).

Due to we have no such an access to get an e-signature, **we implemented our own electronic signature mechanism**. We use electronic signatures to authenticate the origin and integrity of digital documents or transactions. We typically rely on cryptographic methods to ensure the security and non-repudiation of the signed content. Here's an overview of how our electronic signatures work from a technical perspective step by step:

1. Key Pair Generation: We begin by generating a key pair. We used the Laravel's built-in encryption features to generate a private key and a corresponding public key.
2. Signature Creation: When the field worker submits the periodic maintenance form, we create a digital signature for the form using our private key. We used the Laravel's encryption functions to generate the signature.
3. Attaching the Signature: We store the digital signature along with the submitted form data in our database. We created a separate column in the periodic maintenance form table to store the signature.
4. Signature Verification: To verify the signature later, we retrieve the form data along with the signature from the database. We use our public key to verify the signature's authenticity.
5. Displaying the Signature Details: Once the verification is complete, we display the signature details on the periodic maintenance form. This includes information such as the name of the signer, the timestamp of the signature, and any other relevant details.
6. Optional Timestamping: To provide an extra layer of trust, we considered integrating a trusted timestamping service. This service will provide a timestamp for the signature, proving the exact time of the signing process. This step was optional but we thought that this enhance the validity of the signature.
7. User Interface Design: We designed the user interface to allow the field worker to sign the form electronically.

8. QR Code Generation: After the verification process is complete and the signature is deemed valid, we generate a QR code containing the verification status or a link to verify the document. We used `simplesoftwareio/simple-qrcode` library to generate QR code.
9. Attaching the QR Code: We store the generated QR code data in our database, associating it with the periodic maintenance form. We added a column to the periodic maintenance form table to store the QR code data.
10. Displaying the QR Code: When displaying the verified form, we include the QR code image on the document. We are embedding the QR code image that includes the related data into the document.
11. Handling the Verification Requests: We set up a verification endpoint in our Laravel application that can be accessed via the URL or data embedded in the QR code. This endpoint handles the verification request and displays the verification status of the submitted form. We use the stored signature data in the database to verify the signature's authenticity again and respond accordingly.

5 Nonfunctional System Requirements

5.1 Performance Requirements

ID: NF1

TAG: Response Time < ID: FR1 TITLE: Create an account >

GIST: The fastness of creating an account.

SCALE: The response time of creating an account.

METER: Measurements obtained from 10 "create an account" scenarios during testing.

MUST: No more than 2 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

RES: No more than 1 second.

ID: NF2

TAG: Response Time < ID: FR2 TITLE: Login >

GIST: The fastness of logging in.

SCALE: The response time of logging in.

METER: Measurements obtained from 10 'login' scenarios during testing.

MUST: No more than 2 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

RES: No more than 1 second.

ID: NF3

TAG: Response Time < ID: FR13 TITLE: Logout >

GIST: The fastness of logout.

SCALE: The response time of logout.

METER: Measurements obtained from 10 'logout' scenarios during testing.

MUST: No more than 2 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

RES: No more than 1 second.

ID: NF4

TAG: Response Time < ID: FR3 TITLE: Add Employee >

GIST: The fastness of adding an employee.

SCALE: The response time of adding an employee.

METER: Measurements obtained from 10 'add employee' scenarios during testing.

MUST: No more than 2 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

RES: No more than 1 second.

ID: NF5

TAG: Response Time < ID: FR4 TITLE: Delete Employee >

GIST: The fastness of deleting an employee.

SCALE: The response time of deleting an employee.

METER: Measurements obtained from 10 'add employee' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

ID: NF6

TAG: Response Time < ID: FR5 TITLE: Edit Employee Information >
GIST: The fastness of updating employee information.
SCALE: The response time of updating employee information.
METER: Measurements obtained from 10 'edit employee information' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

ID: NF7

TAG: Response Time < ID: FR6 TITLE: Add Customer >
GIST: The fastness of adding a customer.
SCALE: The response time of adding a customer.
METER: Measurements obtained from 10 'add customer' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

ID: NF8

TAG: Response Time < ID: FR7 TITLE: Delete Customer >
GIST: The fastness of deleting a customer.
SCALE: The response time of deleting a customer.
METER: Measurements obtained from 10 'delete customer' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

ID: NF9

TAG: Response Time < ID: FR8 TITLE: Edit Customer Information >
GIST: The fastness of editing customer information.
SCALE: The response time of editing customer information.
METER: Measurements obtained from 10 'edit customer information' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

ID: NF10

TAG: Response Time < ID: FR9 TITLE: Add Offer >
GIST: The fastness of adding an offer.
SCALE: The response time of adding an offer.
METER: Measurements obtained from 10 'add offer' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

ID: NF11

TAG: Response Time < ID: FR10 TITLE: Delete Offer >
GIST: The fastness of deleting an offer.
SCALE: The response time of deleting an offer.
METER: Measurements obtained from 10 'delete offer' scenarios during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.
RES: No more than 1 second.

5.2 Safety Requirements

Keeping the database and information secure is very important for our team. It is planned to make a backup every two weeks in order to ensure data security in order to eliminate bad advertisement situations that will occur after any errors and data loss. Because in such a case, there may be no compensation for the data loss that will occur in terms of the system and all agreements and customer information may be lost. The reason why it is so important is that the company will integrate all its information into the system and keep it only in the system.

5.3 Security Requirements

Since there is no cyber security expert in our developer team, in the development part we use Laravel 9, a PHP framework that takes care of security breaches. As security is not our client's primary goal, we do not dwell on this area much.

5.4 Organizational Requirements

Our development team must meet every week to talk about the changing requirements and job division in the software part. The progress of the project may change as a result of the feedback that our project manager Mehmet Mesut Yılmaz gave us as a result of the conversations he had with the customer. That's why it's important for us to get together every week.

5.5 Usability Requirements

For the usability test, we will put 5 different people who are capable of using the system to the usability test by asking them to try different scenarios. Thus, we plan to fix the system frontend and logical errors, if any.

6 Use Cases

6.1 Creating a New Account

Purpose of this use case is to explain the procedure of creating an account in our system.

Pre-Conditions:

-None.

Post-Conditions:

-The user will have an account.

Basic Flow:

1. User is on our create an account page.
2. The user choose 'Müşteri' or 'Çalışan'
3. The user enters all the information and clicks the 'Üye Ol' button.
4. If a username or email already exists an error message appears to the user and asking to choose another email or username. If no error appears, a confirmation appears and the user is being informed about she/he created her/his account successfully.

6.2 Login

Purpose of this use case is to explain the procedure of user login.

Pre-Conditions:

-Having an account.

Post-Conditions:

-The user successfully logged in to her/his account.

Basic Flow:

1. The user is on our login page.
2. The user enters the all needed information and then clicks the login button.
3. If all information is correct then users will be informed about successfully login and redirected to their own homepage. If no error message appears and the user checks all the information is correct and retry again.

6.3 Add a Customer

Purpose of this use case is to explain the procedure of adding a customer.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin, or field worker.

Post-Conditions:

- A new customer will be successfully added.

Basic Flow:

1. User is on our 'Müşteriler' page.
2. Clicks the button to the right of the 'Hızlı Kayıt' button.
3. The user fills the form correctly without blank inputs. In order to go to the end, the user clicks on the 'Sonraki' button after filling out the current form.
4. After filling out all the forms, clicks the 'Müşteri Ekle' button bottom of the page.
5. Customer will be added successfully if there is no other customer registered in the system matching the filled information. Otherwise, the system will return an error message.

6.4 Add an Employee

Purpose of this use case is to explain the procedure of adding an employee.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin.

Post-Conditions:

- A new employee will be successfully added.

Basic Flow:

1. User is on our 'Çalışanlar' page.
2. Clicks the 'Yeni Ekle' button at the top of the employees' list.
3. The user fills the form correctly without blank inputs. In order to go to the end, the user clicks on the 'Sonraki' button after filling out the current form.
4. After filling out all the forms, clicks the 'Çalışanı Ekle' button bottom of the page.
5. Employee will be added successfully if there is no other customer registered in the system matching the filled information. Otherwise, the system will return an error message

6.5 Delete an Employee

Purpose of this use case is to explain the procedure of deleting an employee.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin.

Post-Conditions:

- An existing employee will be successfully deleted.

Basic Flow:

1. User is on our 'Çalışanlar' page.
2. Clicks the three-dot icon on the right of the employee's row.
3. Clicks the 'Sil' button in the window that opens.
4. Employee will be deleted successfully.

6.6 Delete a Customer

Purpose of this use case is to explain the procedure of deleting a customer.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin.

Post-Conditions:

- An existing customer will be successfully deleted.

Basic Flow:

1. User is on our 'Müşteriler' page.
2. Clicks the three-dot icon on the right of the customer's row.
3. Clicks the 'Sil' button in the window that opens.
4. Customer will be deleted successfully.

6.7 Edit Customer Information

Purpose of this use case is to explain the procedure of editing customer information.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin.

Post-Conditions:

- The information of the selected customer will be changed.

Basic Flow:

1. User is on our 'Müşteriler' page
2. Clicks the three-dot icon on the right of the customer's row.
3. Clicks the 'Düzenle' button in the window that opens.
4. The user changes the information of the selected customer and clicks the 'Güncelle' button.
5. Customer information will be updated successfully.

6.8 Edit Employee Information

Purpose of this use case is to explain the procedure of editing employee information.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin.

Post-Conditions:

- The information of the selected employee will be changed.

Basic Flow:

1. User is on our 'Çalışanlar' page
2. Clicks the three-dot icon on the right of the employee's row.
3. Clicks the 'Düzenle' button in the window that opens.
4. The user changes the information of the selected customer and clicks the 'Güncelle' button.
5. Employee information will be updated successfully.

6.9 View an Offer

Purpose of this use case is to explain the procedure of viewing an offer.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin, or field worker.

Post-Conditions:

- The user can see the offer's information.

Basic Flow:

1. User is on our 'Teklifler' page.
2. Clicks the three-dot icon on the right of the offer's row.
3. Clicks the 'Düzenle' button in the window that opens.
4. Clicks the 'Önizle' button on the page that opens.
5. Offer information will be shown successfully.

6.10 Delete an Offer

Purpose of this use case is to explain the procedure of deleting an offer.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin.
- Must exist one or more offers.

Post-Conditions:

- An existing offer will be deleted successfully.

Basic Flow:

1. User is on our 'Teklifler' page.
2. Clicks the three-dot icon on the right of the offer's row.
3. Clicks the 'Sil' button in the window that opens.
4. Offer will be deleted successfully.

6.11 Edit an Offer

Purpose of this use case is to explain the procedure of editing an offer.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin, or field worker.

Post-Conditions:

- An existing offer will be updated successfully.

Basic Flow:

1. User is on our 'Teklifler' page.
2. Clicks the three-dot icon on the right of the offer's row.
3. Clicks the 'Düzenle' button in the window that opens.
4. The user changes the offer information and clicks the 'Kaydet' button.
5. Offer will be updated successfully.

6.12 Send an Offer

Purpose of this use case is to explain the procedure of sending an offer.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin, or field worker.

Post-Conditions:

- The user sends an offer to the customer.

Basic Flow:

1. User is on our 'Teklifler' page.
2. Clicks the 'Yeni Ekle' button at the top of the offers list.
3. The user fills out the form and clicks the 'Teklifi Gönder' button.
4. Offer will be sent to the customer successfully.

6.13 Edit Profile Information

Purpose of this use case is to explain the procedure of editing profile informations of users of the customer management system.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin, customer, or field worker.

Post-Conditions:

- The user changed the profile informations.

Basic Flow:

1. User is on our 'Anasayfa' page.
2. The user clicks on her/his name or on the part with her profile photo at the top right of the webpage.
3. The user clicks on the 'Profil' button in the opened section.
4. The user will be able to see their own information on the opened page. The 'Güncelle' button will not be visible unless it changes any information. When he changes any information, he changes the information by clicking the 'Güncelle' button that appears.

6.14 Fill Form

Purpose of this use case is to explain the procedure of filling the form that is related to customer's offer from field worker.

Pre-Conditions:

- Having an account.
- Must be logged in to the system as an admin, or field worker.

Post-Conditions:

- The user filled the form successfully.

Basic Flow:

1. User is on our 'Randevu Yönetimi' page.
2. The user click the related form button which is in the list of appointment.
3. After the form page is opened, the user must fill in the information in the form correctly.
4. After the information is filled, press the 'Finish' button and the process is completed successfully.

7 Interaction Models

7.1 Use Case Models

- General Use Case Model

GENERAL USE CASE MODEL

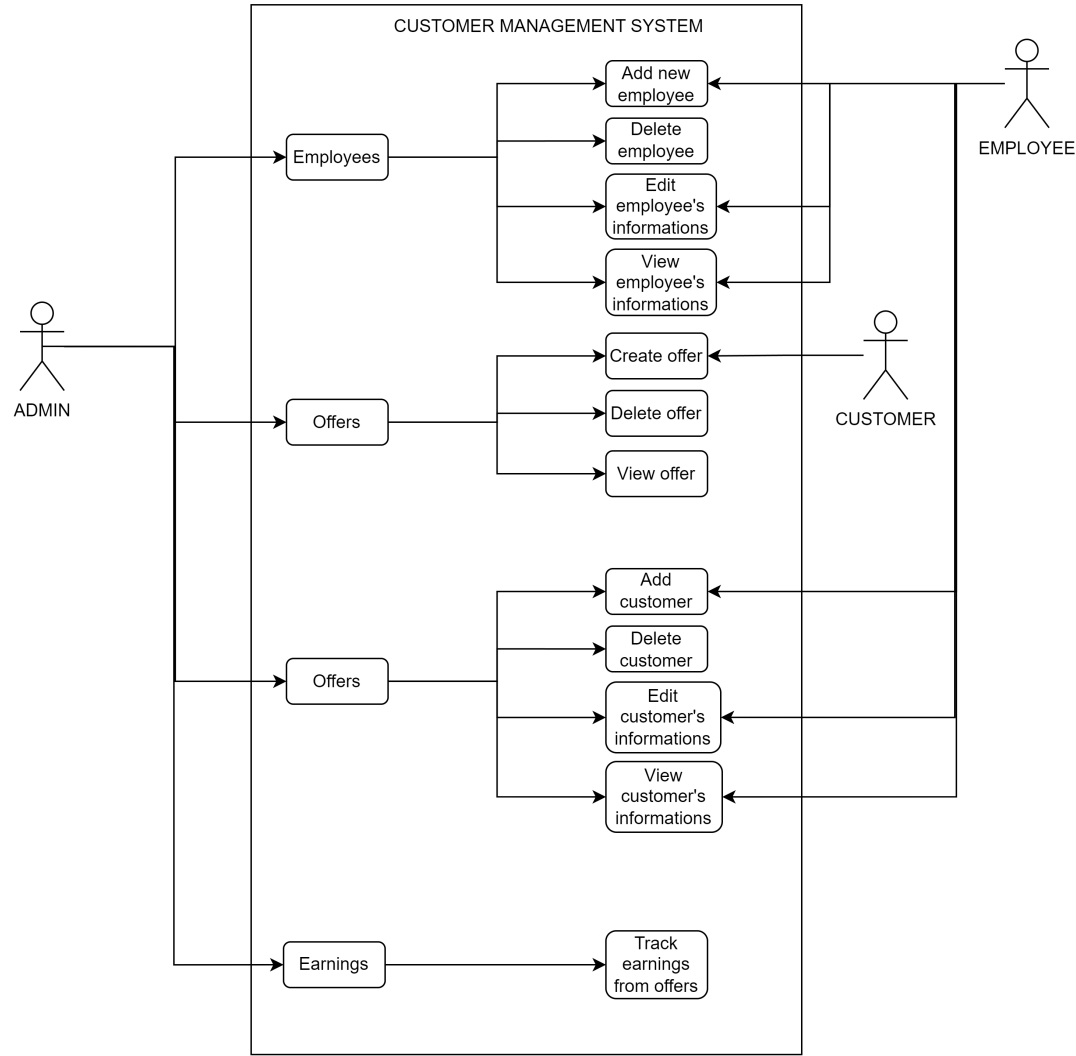


Figure 29: Admin - Customer - Employee Use Case Model

7.2 Sequence Diagrams

- User Login

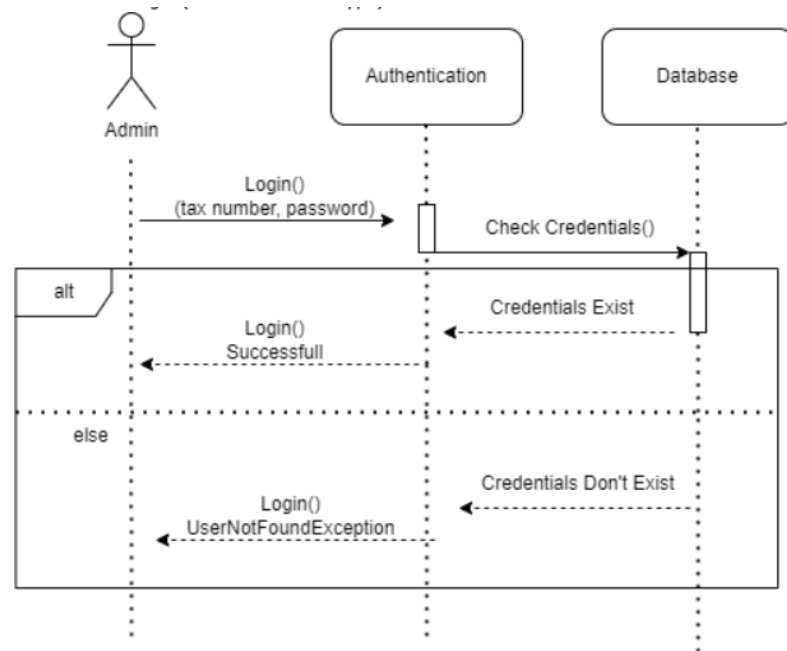


Figure 30: User Login Sequence Diagram

- Create New Customer

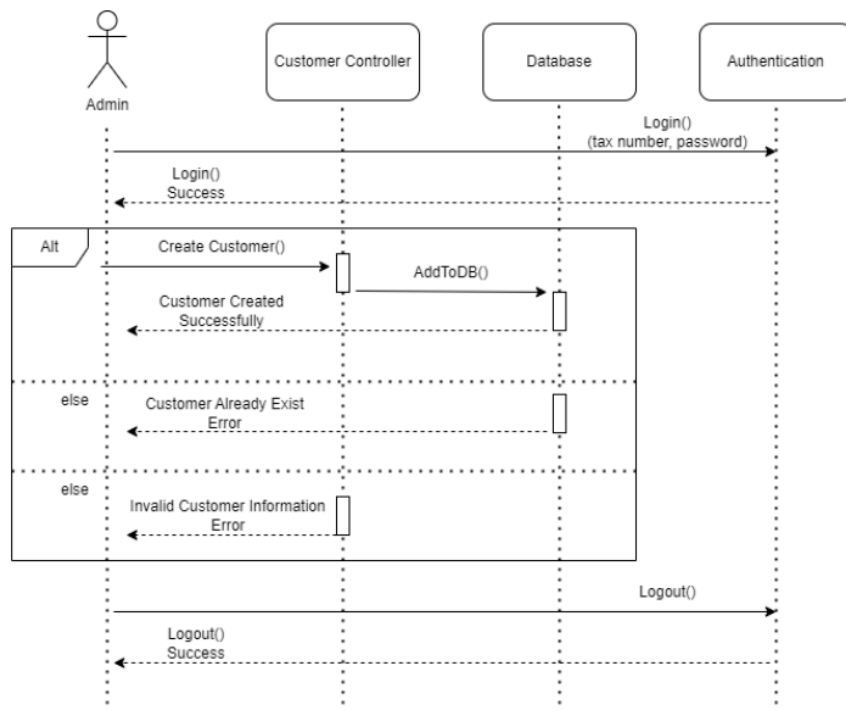


Figure 31: Create New Customer Sequence Diagram

7.3 Class Diagrams

- All Classes

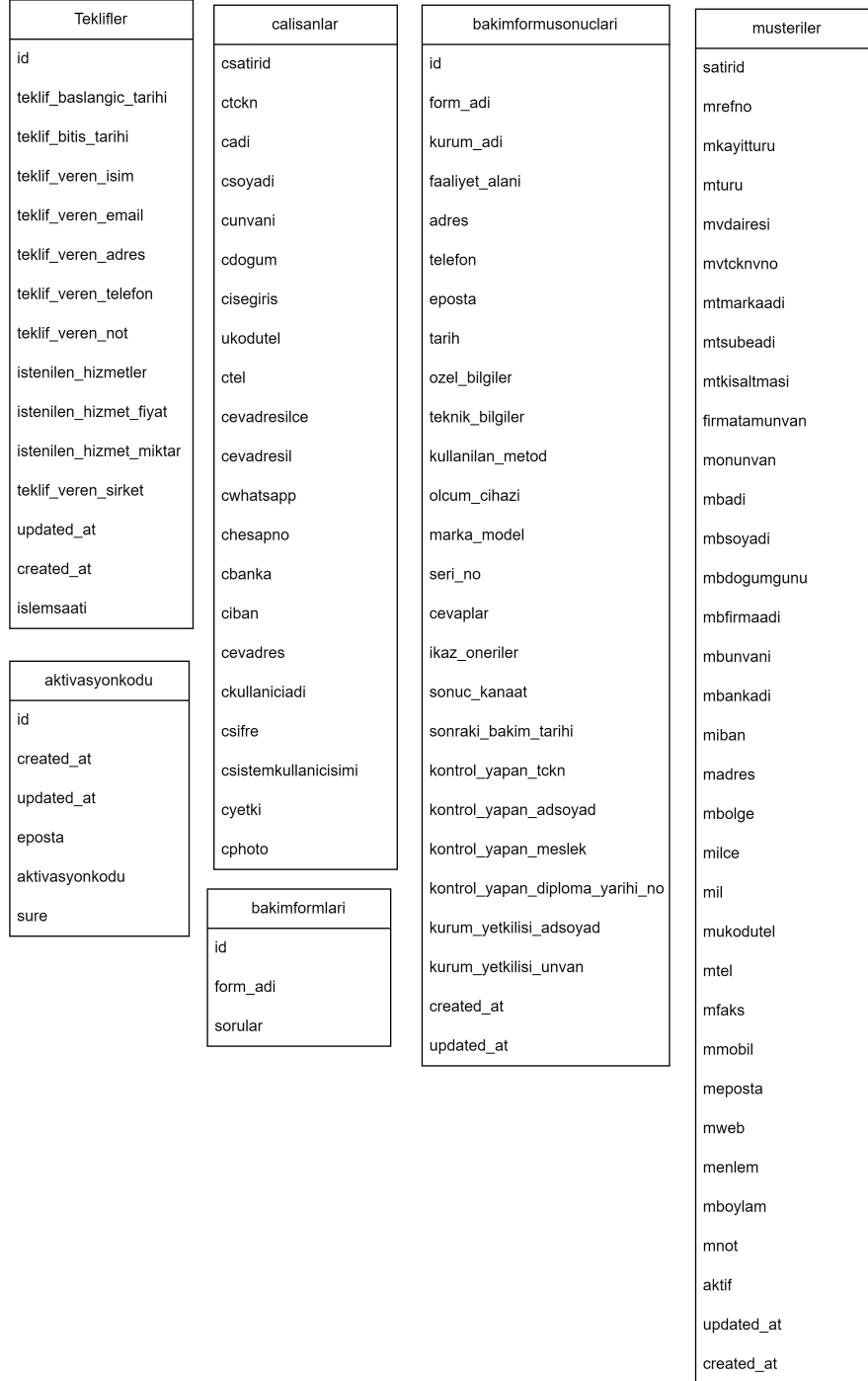


Figure 32: All Classes That System Has

- Entity Relationship Diagram

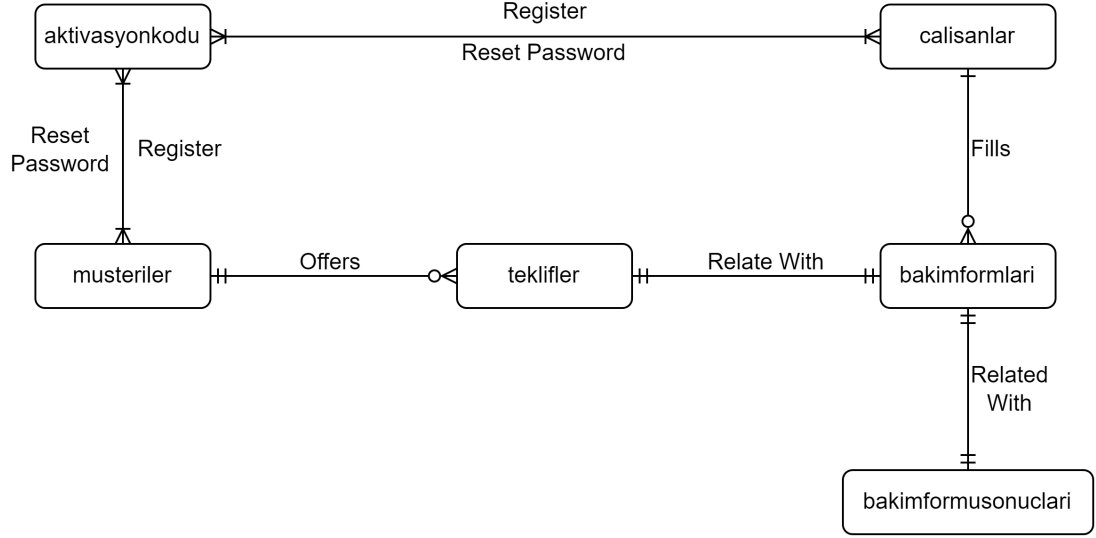


Figure 33: Entity Relationship Diagram

8 Further Development

We're going to increase the accessibility. We will review our code and we will use more "semantic" HTML to support screen readers. It will also increase the SEO score of our application. Semantic HTML refers to the practice of using HTML elements to convey the meaning and structure of the content on a webpage, rather than just focusing on its visual presentation. In other words, it involves using HTML tags in a way that accurately represents the purpose and significance of the information being presented. This helps search engines understand the webpage better and improves accessibility for users with disabilities who may rely on assistive technologies.

We are planning to add a cache mechanism to our project. A cache mechanism is a technique used to store and retrieve data in a faster and more efficient manner. It involves temporarily storing frequently accessed or computationally expensive data in a cache, which is a high-speed storage location. When a requested data is found in the cache, it can be retrieved quickly without having to perform the original, time-consuming operation. This helps to reduce latency, improve performance, and alleviate the load on the underlying resources, such as databases or servers. Redis is the most common and perfect solution to do that.

Also we have an another plan to implement a modern artificial intelligence (AI) system will completely transform how we communicate with our clients in the future. By providing tailored price recommendations based on particular times, we hope to improve their shopping experience. With the help of this new AI technology, we hope to offer customers personalized pricing recommendations that take into account a range of elements including current market conditions, client preferences, and historical data. This data will be analyzed by the AI system to produce precise and timely price suggestions that fit the needs and budget of the customers. We will be able to recognize patterns and adjust our pricing strategies in response thanks to the AI's ability to analyze historical data and market trends. The AI will adjust and suggest competitive prices that entice customers during seasonal sales, special events, or other times of peak demand.