**YAZILIM RAPORU**

1. **YAZILIM TANITIMI**

Yazılımcıların Adı-Soyadı: Melih Afşar, Kenan Baylan, Şerif Yılmaz, Aziz Eren Sağanda

*Geliştirilme Ortamları*

*Programlama Dilleri*: Program C dili kullanarak yazılacaktır.

*Programın Adı:* Online Sipariş Yazılımı

*Bilgisayar Sistemi*: Program yazılımcıların kendi bilgisayarlarında kodlanacaktır.

*İşletim Sistemi:* Program Windows 10 – 64bit işletim sisteminde DEV-C++ ve Visual Studio Code editörlerinde GCC compilerı kullanarak derlenmiştir. Aynı zamanda test amaçlı Windows Subsystem for Linux ile Debian işletim sisteminde Clang compilerı ile derlenerek test edilmiştir.

*Gereksinimler*: Program konsolda çalışacağı için hemen hemen bütün 64bit Windows kullanan bilgisayarlarda çalışacaktır.

*Kaynak Kod Satır Sayısı*: Programın bitmiş halinin 4500 satır civarında olması planlanmaktadır.

1. **YAZILIM TASARIMI**
2. Problemin Tanımlanması / Programın Amacı

Müşterilerin dilediği gibi sepet oluşturup sipariş verebildiği, aynı zamanda ürün ekleme-çıkarma, fiyat düzenleme vb. gibi özellikleri barındıran ve yapılan her işlemi düzenli olarak kayıt altında tutan ve bunları fiş olarak yazdırabilen bir C programı yapılması amaçlanmıştır.

1. Problemin Çözümü / Çözüm Tasarımı

Program menüsünde müşterilerin hesap oluşturarak veya giriş yaparak yemek menüsündeki yemeklerden istedikleri kadar seçmeleri sağlanır. Müşterilerin kendi hesaplarının güvenliği şifre ile koruma altına alınacaktır. Yemek seçen müşteriler dilerlerse içecek menüsünden istedikleri miktarda içeceği sepetlerine ekleyeceklerdir.

Admin menüsünde, admin kullanıcısı; şifre ile başarılı giriş yaparsa belli seçeneklerle karşılaşır. Admin bu aşamada yemekleri normal kullanıcı gibi görüntüleyebilir. Admin isterse menüye yeni yiyecek ve içecekler ekleyip çıkarabilir, fiyat güncellemesi yapabilir. Günlük bütçe dökümünü .txt dosyası olarak alabilir. Programın işleyişi bu şekilde tasarlanmıştır.

1. **NEDEN BU PROJE’Yİ SEÇTİK?**

Öncelikle ekip arkadaşlarımızla birlikte proje seçmeden önce neler yapabileceğimizi tartıştık ve aklımıza birçok fikir gelmesine rağmen bu projenin özgün olması adına diğer gruptaki arkadaşların projelerini Excel tablosuna eklemelerini bekledik.

Projeler belli olmaya başladıktan sonra Online Sipariş Yazılımında karar kıldık ve bu projeyi seçerken sadece proje ödevi olarak değil de ekipçe yazılım da ilerledikçe ilerleyen yıllarda geliştirilebilmesine önem gösterdik. Online Sipariş Yazılımına ilerleyen dönemlerde veri tabanı, çeşitli görsel tasarımlar ve fonksiyonlarla birlikte kullanılabilirliği arttırmayı hedefliyoruz.

1. **PROJE’DE YAPILMAK İSTENENLER**

Müşterilerine kaliteli bir hizmet sunmak isteyen restoranın; kebap, çorba, döner, ev yemekleri, pizza, balık, pide, hamburger, salata, tatlı gibi kategorilerde yiyeceklerini hazırlayıp bunları, müşterilerinin adres bilgileri ile değerlendirmesi; gerekli kredi kartı bilgilerinin doğrulanması gibi işlemleri yapması için siparişleri dinamik struct yapılarıyla oluşturulmuş olan bir programa ihtiyacı gerekmektedir.

**Proje İçeriği:**

* Müşteri Kayıt:

Restorana daha önce kayıt yapılmamış bir kişinin, program içerisinde isim, soy isim, e-posta, şifre, adres, kredi kartı gibi bilgileri alınır.

Bu alınan bilgiler daha sonra kullanılmak üzere .txt uzantılı dosyaya yazılır ve saklanır.

* Müşteri Girişi:

Kişi daha önce restorana kayıt yaptırmış ise kayıt esnasında girmiş olduğu e-posta ve şifre bilgilerini programın istediği şekilde girer.

Girilen bilgiler ile .txt uzantılı dosyada tutulan daha önceki girdiği bilgiler ile doğrulaması kontrol edilir. Doğrulama gerçekleştiği takdirde müşteri kendine ait bilgileri görüntüleyebilir.

Müşteri bu kısımdan kendine ait olan adresini ve şifresini değiştirebilir.

Müşterinin sipariş verebilmesi için müşteri giriş kısmından sisteme giriş yapması zorunludur.

Sisteme giriş yapmadığı takdirde yalnızca yiyeceklerin çeşitlerini ve fiyat bilgisini görebilmektedir.

* Sipariş:

Eğer müşteri numarası doğru biçimde eşleşirse müşteri sipariş ekranına yönlendirilir. Sipariş sonlandırıldıktan sonra, verilen sipariş özeti fiş halinde bir .txt dosyasında yazdırılır.

* Admin Girişi:

Admin kullanıcı, sistemde kayıtlı olan kendine özel bilgilerini girdiği takdirde doğrulanırsa müşteriden farklı olarak bazı özellikleri kullanır hale gelir.



* Admin Menüsünün görünüşü

Bu özellikler aşağıda belirtilmektedir:

Yiyecek kategorilerinin fiyatlarını güncelleyebilir.

Yiyecek, içecek gibi kategorilerin stok durumu hakkında bilgi sahibi olabilir.

Günlük bütçe dökümünü .txt uzantılı dosya olarak alabilir.

Ayrıca müşteri de olduğu gibi tüm yiyecek bilgilerini de görebilir.

Bu programı birden fazla kullanıcının kullanmasından dolayı yiyeceklerin bilgisine kolay ulaşılabilmesi için struct yapısına; programın kullanılabilirliğinin artması için de gerekli tasarım fonksiyonlarına başvurulacaktır.

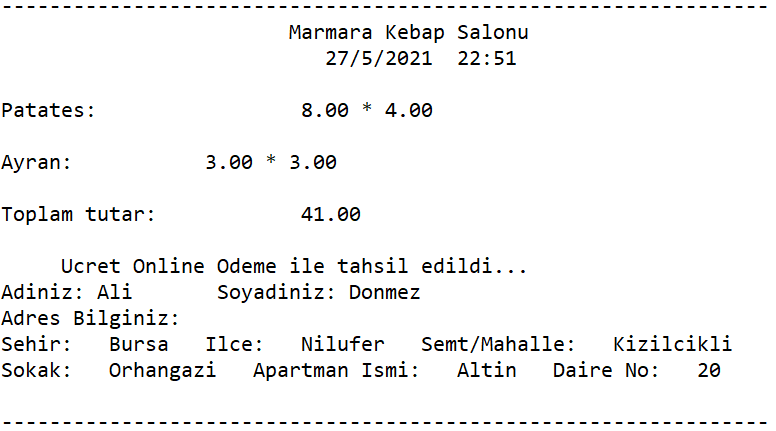
1. **PROJE İLE İLGİLİ BENZER ÇALIŞMALAR HAKKINDA BİLGİLER- FARKLILIKLAR:**

Online Sipariş Yazılımı projemize benzer çalışmalar üzerinde yaptığımız araştırmalarda karşımıza bazı eksiklikler ve farklılıklar çıktı. Bunları aşağıda detaylandırmaya çalışacağız:

1. Bunlardan ilki olan Food Order Management’e baktığımızda ise göze çarpan ilk fark sipariş sonucunda bir fatura yazdırılmamış olmasıdır. Bizim yaptığımız projede ise müşterilerin siparişleri ödemeden sonra fiş halinde bir .txt dosyasına kaydedilecektir. Ayrıca yemekler programa hardcoded olmadığı için diğer projelerin aksine yemeklerin adını ve fiyatını düzenlemek isteyen biri yemek türlerinin olduğu (örneğin çorba için çorbalar.txt, kebaplar için kebaplar.txt) dosyaları değiştirerek sorununu çözebilir. Bu bilgileri statik olarak structlarda tutmayıp dosyalar ile kontrol ederek programa esneklik kazandırmayı amaçladık.

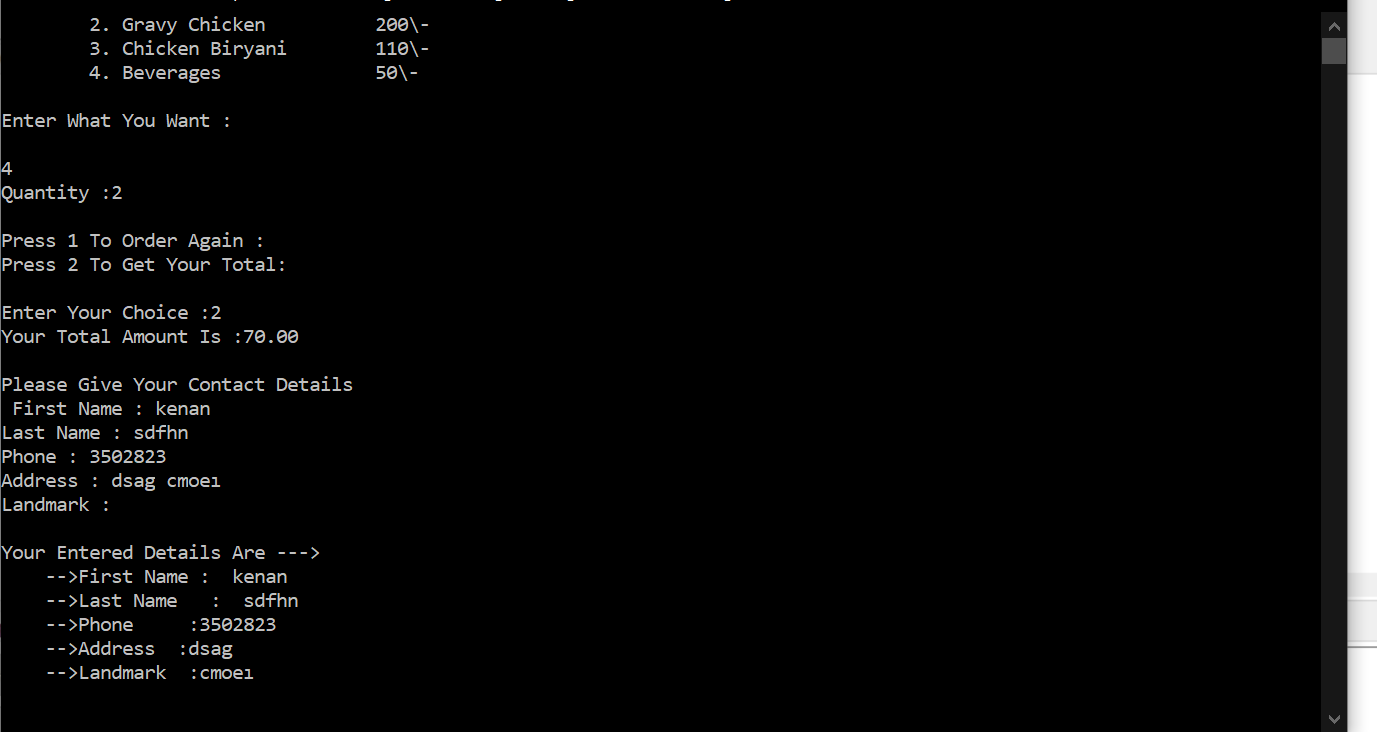


* Balıklar.txt dosyasının içeriği



* Sipariş fişinin tutulduğu siparisFis.txt dosyası

1. Aynı projenin kodları üzerinde biraz daha araştırma yaptıktan sonra gözümüze farklı bir fark daha çarpıyor menülerin aşağıdaki outputta göründüğü gibi düzensiz ve sıradan yazıların bir menü şeklinde değil de consol ekranında sadece sola yaslanmış gibi görünmesi kullanıcının gözüne hitap etmeyen hatalardandır .Bizim projemizde kaçınacağımız en önemli adımlardan biri bu olacak. Hedeflediğimiz menü tasarımın da consol ekranının ortasında kalan arka plan rengi ve yazı tipinin farklı olması, her farklı menüde kullanıcının karşısına farklı bir arka plan rengi ile projeyi sunarak sadece kodların işlevleri değil görsel olarak da efektif bir proje yapmaya çalışıyoruz.



* Food Order Management adlı programın menüsünden bir görünüm

1. Bir diğer projede karşımıza çıkan göze çarpan eksik detay ise sipariş projelerinde müşterilerin kayıtlı olmamasıydı bir müşteri yemek sipariş ettikten sonra aynı restorandan tekrar bir sipariş vermeye kalktığında müşteriye tekrardan telefon numarası ,e-mail ,ev adresi, ad, soyad gibi bilgileri tekrardan sorulması yerine biz müşteri kayıt sistemi ile müşteri bilgilerini dosyalama işlemleriyle .txt dosyasında tutabileceğimiz ve her kullanıcıya ait bir şifre ve e-posta ile kullanıcı sisteme rahatlıkla giriş-çıkış yapabileceği bir fonksiyon üzerinde çalışıyoruz .
2. Diğer projelerden farklılık olarak bir diğer konu ise ödeme bilgilerinin tutulması yemek ve içecek seçimlerinden sonra ödeme ekranına yönlendirildiğinde kullanıcı eğer daha önceden kayıtlı bir kullanıcı ise tekrardan kart bilgileri girmesine gerek kalmayacak eğer kullanıcı yeni bir müşteriyse kredi kart bilgilerinin veri tabanında tutulması için kullanıcıya soru sorması olacaktır. Sonradan ödeme bilgileri kredi kartı ve kapıda ödeme olarak ayrı şekillerle .txt dosyasına kaydedilecektir.
3. **PROJE’DE KULLANILAN KÜTÜPHANELER:**

Şu ana kadar kullanılan kütüphaneler aşağıda sıralanmıştır. Bundan sonra kullanılan kütüphaneler Rapor-2’ye dahil edilecektir.

#include <conio.h>

#include <locale.h>

#include <stdio.h>

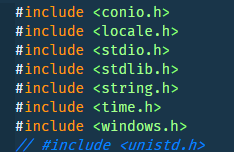
#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <windows.h>

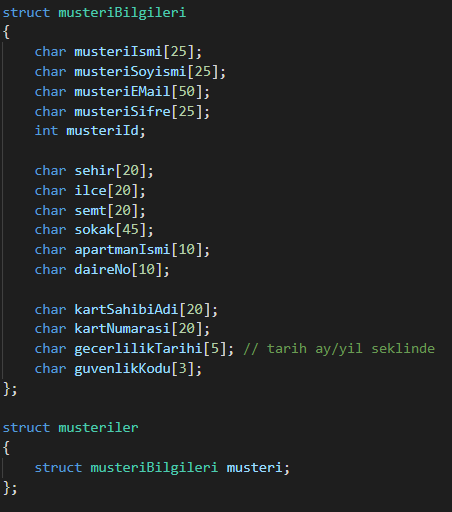
#include <unistd.h>



1. **C DİLİNİN HANGİ ÖZELLİKLERİNDEN FAYDALANDIK?**

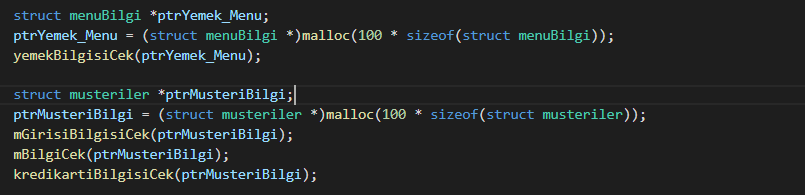
**Döngü Kullanımı**: Müşteri girişi yapılırken do-while döngüsüyle müşteri sayısını tuttuk. Aynı zamanda for döngüleriyle bazı tasarım fonksiyonlarını (cerceve, bosluk\_bas, tab\_bas, vs.) oluşturduk.

**STRUCT KULLANIMI**: Yemek ve müşteri bilgilerini structlarda saklayarak daha kolay erişilebilmesini sağladık. Ayrıca typedef kullanarak yeni tür tanımlamalarıyla tekrara düşmeden struct değişkenleri tanımladık.



* Struct tanımlarımızdan bir örnek

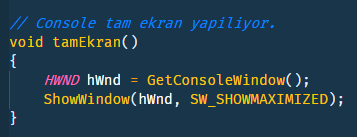
**Dinamik Bellek Atamaları**: malloc() ile dinamik bellek ataması yapıldı. Statik bellek ataması yerine dinamik ataması yapılarak gereksiz bellek kullanımının önüne geçilmiş oldu. free() komutuyla da belleğin kullandıktan sonra boşaltarak garbage collecting işini yapmış olduk.



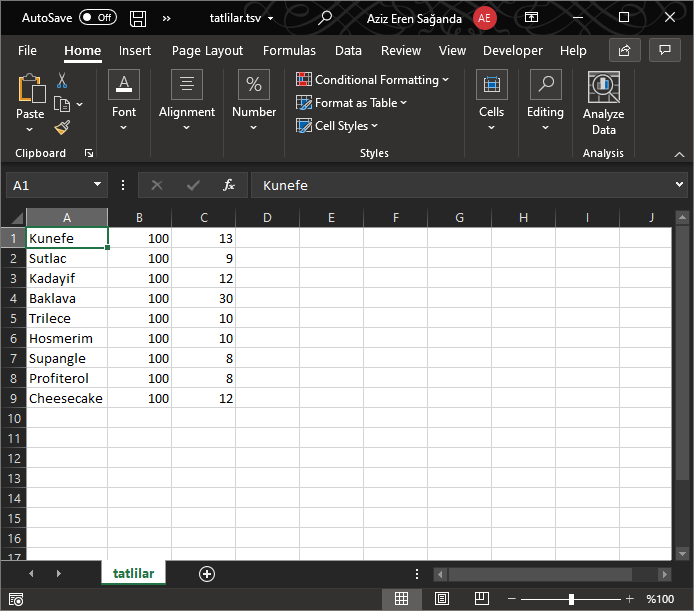
**Dosya İşlemleri**: FILE türünde pointerlar kullanılarak dosya yeri işaret edildi. fopen() komutuyla işaretçinin gösterdiği dosya açıldı ve EOF karakteri görünene kadar dosyanın parse edilmesini sağlayan feof() fonksiyonu kullanıldı. fscanf() ile dosyadan kullanıcı girişi okundu. fclose() komutuyla açtığımız dosyayı kapattık.

**Handle Kullanımı**: Ekran boyutunu algılayabilmek için Windows.API ile iletişime geçildi. Böylelikle farklı çözünürlükteki monitörlerde bile konsol metni ortalanmış oldu. ShowWindow() fonksiyonu ile alınan ekran bilgisine göre konsol penceresi tam ekran boyutuna getirildi.

* Konsolu maximize etmek için kullandığımız tamEkran() fonksiyonu



**Dosya Yönetimi:** Yemek dosyalarını kolayca yönetebilmek için .txt dosyası yerine MS Excel’de kolayca düzenlenebilecek .tsv (tab seperated values) uzantılı dosyalar kullandık. Siparişleri, fişleri ve müşteri adreslerini tablo olarak tutmaya ihtiyacımız olmadığından .txt dosyası olarak kaydettik.



* Tatlilar.tsv dosyasının MS Excel ile entegrasyonu

**Diğer Library fonksiyonları**: cls() kullanılarak konsol ekranı silindi, Sleep() kullanılarak menülerdeki yemeklerin listelenme animasyonu oluşturuldu. time.h kütüphanesinde bulunan tm structu ile ana menüde tarih-saat görüntülenmesi sağlandı. System(“color “) komutu ile konsol ekranının rengi değiştirildi. exit() fonksiyonu ile hata oluşması durumunda programın yürütülmesini durdurduk. Böylelikle işletim sistemine ekstra yük bindirilmeden hata kontrolü gerçekleştirilmiş oldu.

**H – GEÇMİŞ RAPORA GÖRE FARKLILIKLAR:**

Bundan iki – üç ay öncesinde hazırladığımız projenin geleceği hakkındaki raporda belirttiğimiz bazı özellikler ve fonksiyonlar hakkında yeterli kaynağa ulaşamadık. Bunlardan en önemlisi menüler arasında geçişleri sayılar ile değil klavyedeki yön tuşlarıyla yapmak gibi amacımız vardı fakat bu konu hakkında yeterli kaynağa ve benzer örneklere ulaşamadığımızı belirtmemiz ve bu projede ilerleyen zamanlarda eklenmesi gereken bir özellik olduğuna aramızda arkadaşlarla konuşup hem fikir sahibi olduk.

Bir diğer özellik ise kayıtlı kullanıcı verilerini silmekti. Bu konuda kendimizi oldukça geliştirdik hatta çeşitli projelerdeki örneklerle dosyalama işlemleriyle kullanıcı verilerini silmek konusunda yeterli kaynağa ulaşmamıza rağmen dosyalama işlemlerinde eksik tecrübemiz ve zaman sıkıntımız nedeniyle bu fonksiyon üzerinde yeterince ayıracak zamanımız kalmadığı için bu fonksiyonu geçici bir süreliğine rafa kaldırmak zorunda kaldık.

**I – PROJE SÜRECİ:**

Ekip arkadaşlarımızla projeye başlamadan önce yaptığımız toplantılarda projeyi çok daha verimli ve hızlı bir şekilde yapmak hedefimizdi fakat zaman ve hayat ilerledikçe ve Yemek Sipariş Otomasyonu üzerinde çalışmalarımızda çok farklı sorunlarla karşılaştık henüz birçoğumuzun yapmış olduğu ilk proje olduğu için sorunları anlama ve hataları bulma konusunda çok fazla uğraştık bazen günlerce farklı kaynaklardan yazacağımız fonksiyonlar hakkında araştırma yaptık. Bundan sonraki yapacağımız projeler adına bu proje çok iyi bir tecrübe oldu.

BURAYA GİTHUB LİNKİ EKLENECEKTİR….