

ÇANKIRI KARATEKİN ÜNİVERSİTESİ
Mühendislik Mimarlık Fakültesi
Bilgisayar Mühendisliği Bölümü



Staj Defteri

Burhan Çavdaroğlu
220905003

Ağustos, 2024

FOTOĞRAF

Öğrencinin	
Adı Soyadı	Burhan Çavdaroglu
Fakülte Numarası	220905003
Firmanın	
Adı	Look and Cash
Staja Başlama ve Bitiş Tarihi	29.07.2024- 23.08.2024
Staj Süresi (İş günü olarak)	25
Staj Değerlendirmesi (Bu bölüm üniversite tarafından doldurulacaktır)	
Kabul edilen gün sayısı	
Onaylayan	İmza / Tarih

Bu staj defterinde stajıma destek veren firmanın ticari değeri olan özel bilgileri içermediği doğrularım.

__/__/__

Öğrenci Adı ve Soyadı
İmza

Bu staj defteri aşağıda bilgileri yer alan öğrenciniz tarafından firmamızda yapmış olduğu staj süresince doldurulmuş olup, yapılan çalışmaları ve öğrencinin stajına devam ettiğini onaylarım.

__/__/__

Sorumlu Mühendis
Kaşe/İmza

STAJIN KONUSU

Yapmış olduğum staj bir yazılım stajıdır ve bu staj süresince öğrenmiş olduğum bazı diller , frameworkler , kütüphaneler ve işletim sistemi şunlardır Python, React, FastAPI, Kali Linux(Debian Bağımlı) bu gibi araçları kullanarak Back-End kısmında FastAPI kütüphanesi ve Front-End kısmında ise Html , Css, Js, React kullanarak Full-Stack Web Projesi ortaya çıkarttım aynı zamanda FastAPI kullanarak sadece Back-End kısmında çalışan birden fazla farklı fonksiyon işlevlerini yerine getiren aynı zamanda çoklu fonksiyonel ve asenkron olmakla birlikte WEB API(Restful API) gibi projeleri yaptım bunun dışında başlangıç kısmında algoritmayı daha iyi oturtup , sindirebilmek için çözdüğüm algoritmik problemler için C , Python , SQL gibi diller kullandım.

STAJIN AMACI

Yazılım stajı sürecimde çalışanlarla iletişim kurmayı bir şirket ortamında insanlarla nasıl iletişime geçileceği konusunda tecrübe edindim aynı zamanda okul hayatım boyunca öğrenmiş olduğum teorik bilgileri kodlama pratiğine dökme fırsatı buldum bu süre zarfında binlerce satır kod yazarak algoritmik becerimi ve bilgilerimi pekiştirdim ve iş hayatına bir adım hatta birden fazla adım atmış bulunmaktayım , aynı zamanda bana verilen taskları ve bu taskları yerine getirmek için öğrenmem gereken bütün kütüphaneleri , frameworkleri , yazılım dillerini veyahut işletim sistemlerini öğrenip bana verilen görevleri sorunsuz ve mentörümün de istediği şekilde yerine getirdim.

STAJIN YAPILDIĞI KURULUŞ HAKKINDA BİLGİLER

Stajı yapmış olduğum kurumum bir start-up firması ve bu firmada genelde web ve mobil uygulamalar geliştiriliyor ben de bu kısımda web üzerine yoğunlaşma fırsatım oldu bu süreçte de kendimi çok geliştirme fırsatı buldum aynı zamanda mentörüm İlteriş Yücel bey benimle bizzat bu süreç boyunca kendisi ilgilenmiş olup kendileri Hacettepe Üniversitesi Bilgisayar Mühendisliği mezunu olmakla birlikte yazılım sektöründe uzun süre tecrübesi ve bilgisi mevcut.

...../..... /..... TARİHİNDEN/..... /..... TARİHİNE KADAR BİR HAFTALIK ÇALIŞMA

GÜNLER	YAPILAN İŞ	İZAHATIN BULUNDUĞU YAPRAK NO'SU
PAZARTESİ	KALİ LİNX KURULUMU	7
SALI	KALİ LİNX KURULUMU	8
ÇARŞAMBA	KALİ LİNX KURULUMU	9
PERŞEMBE	C DİLİNDE ALGORİTMA	10
CUMA	C DİLİNDE ALGORİTMA	11
CUMARTESİ		
ÖĞRENCİNİN İMZASI :		
SORUMLU MÜHENDİSİN İMZASI :		

...../..... /..... TARİHİNDEN/..... /..... TARİHİNE KADAR BİR HAFTALIK ÇALIŞMA

GÜNLER	YAPILAN İŞ	İZAHATIN BULUNDUĞU YAPRAK NO'SU
PAZARTESİ	C DİLİNDE ALGORİTMA	12
SALI	C DİLİNDE ALGORİTMA	13
ÇARŞAMBA	C DİLİNDE ALGORİTMA	14
PERŞEMBE	C DİLİNDE ALGORİTMA	15
CUMA	C DİLİNDE ALGORİTMA	16
CUMARTESİ		
ÖĞRENCİNİN İMZASI :		
SORUMLU MÜHENDİSİN İMZASI :		

...../..... /..... TARİHİNDEN/..... /..... TARİHİNE KADAR BİR HAFTALIK ÇALIŞMA

GÜNLER	YAPILAN İŞ	İZAHATIN BULUNDUĞU YAPRAK NO'SU
PAZARTESİ	C DİLİNDE ALGORİTMA	17
SALI	C DİLİNDE ALGORİTMA	18
ÇARŞAMBA	C DİLİNDE ALGORİTMA	19
PERŞEMBE	RESTFUL API(WEB API)	20
CUMA	RESTFUL API(WEB API)	21
CUMARTESİ		
ÖĞRENCİNİN İMZASI :		
SORUMLU MÜHENDİSİN İMZASI :		

...../..... /..... TARİHİNDEN/..... /..... TARİHİNE KADAR BİR HAFTALIK ÇALIŞMA

GÜNLER	YAPILAN İŞ	İZAHATIN BULUNDUĞU YAPRAK NO'SU
PAZARTESİ	RESTFUL API(WEB API)	22
SALI	RESTFUL API(WEB API)	23
ÇARŞAMBA	RESTFUL API(WEB API)	24
PERŞEMBE	RESTFUL API(WEB API)	25
CUMA	RESTFUL API(WEB API)	26
CUMARTESİ	KAYNAKÇA YAZMAK	27
ÖĞRENCİNİN İMZASI :		
SORUMLU MÜHENDİSİN İMZASI :		

KISIM	SON	YAPRAK NO:27
YAPILAN İŞ	KAYNAKÇA YAZMAK	TARİH:24/08/2024

KAYNAKÇA

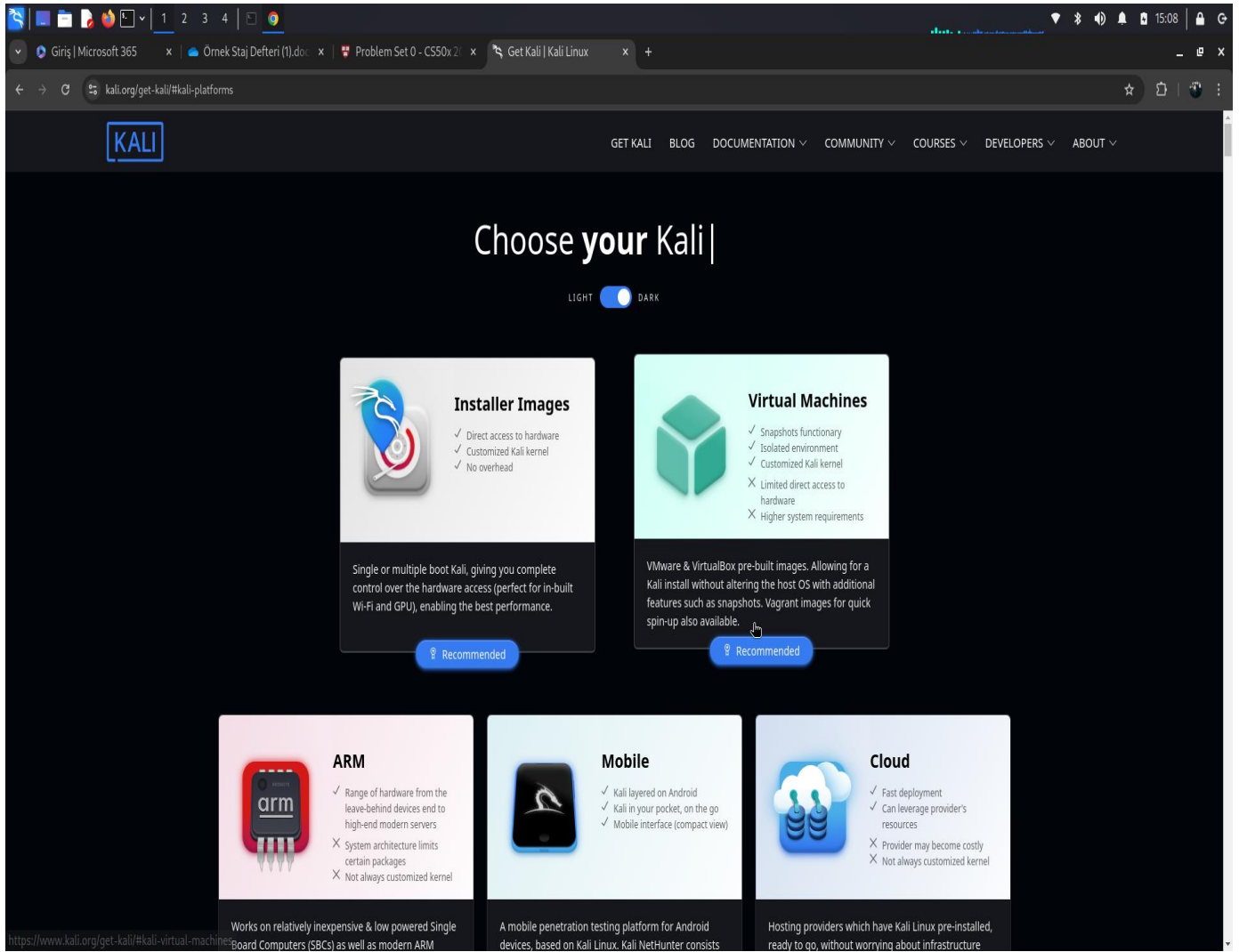
- <https://github.com/Burhan0664>
- <https://github.com/Burhan0664/1-cstaj-4>
- <https://github.com/Burhan0664/1-cstaj-3>
- <https://github.com/Burhan0664/1-cstaj-2>
- <https://github.com/Burhan0664/1-cstaj-1>
- <https://certificates.cs50.io/40c0f40e-2a84-4168-9bec-80adc0264e59.pdf?size=letter>

Algoritmaların kodunu yasa gereği githuba koyamıyorum fakat staj süresince yapmış olduğum projeler burda open-source olarak linklerden ulaşabilirsiniz.

KONTROL SONUCU

KISIM	KURULUM	YAPRAK NO:7
YAPILAN İŞ	KALİ LİNX KURULUMU	TARİH: 29/07/2024

Öncelikle ben staja başlamadan önce mentörümle konuştuğumda kendisi windows kullanmıyordu onun yerine Ubuntu kullanıyordu ben de artık Windows'u bırakıp Kali Linux'e(Debian) geçmeye karar verdim çünkü hem yazılımsal hemde open source bir işletim sistemi olduğundan dolayı çok fazla avantajı vardı hem güvenli olması hem de command line interface kullanımı açısından size bir çok imkan sağlaması kurulumuyla birlikte yazılımsal açıdan veyahut siber güvenlik açısından işimize yarayacak çok fazla paketi kendisiyle birlikte gelmesi de avantajlarından birisi ayrıca command line interface'in sunduğu işlerinizi daha hızlı yapma olasılığı sunması da gayet tatmin edici sebeplerin arasında .

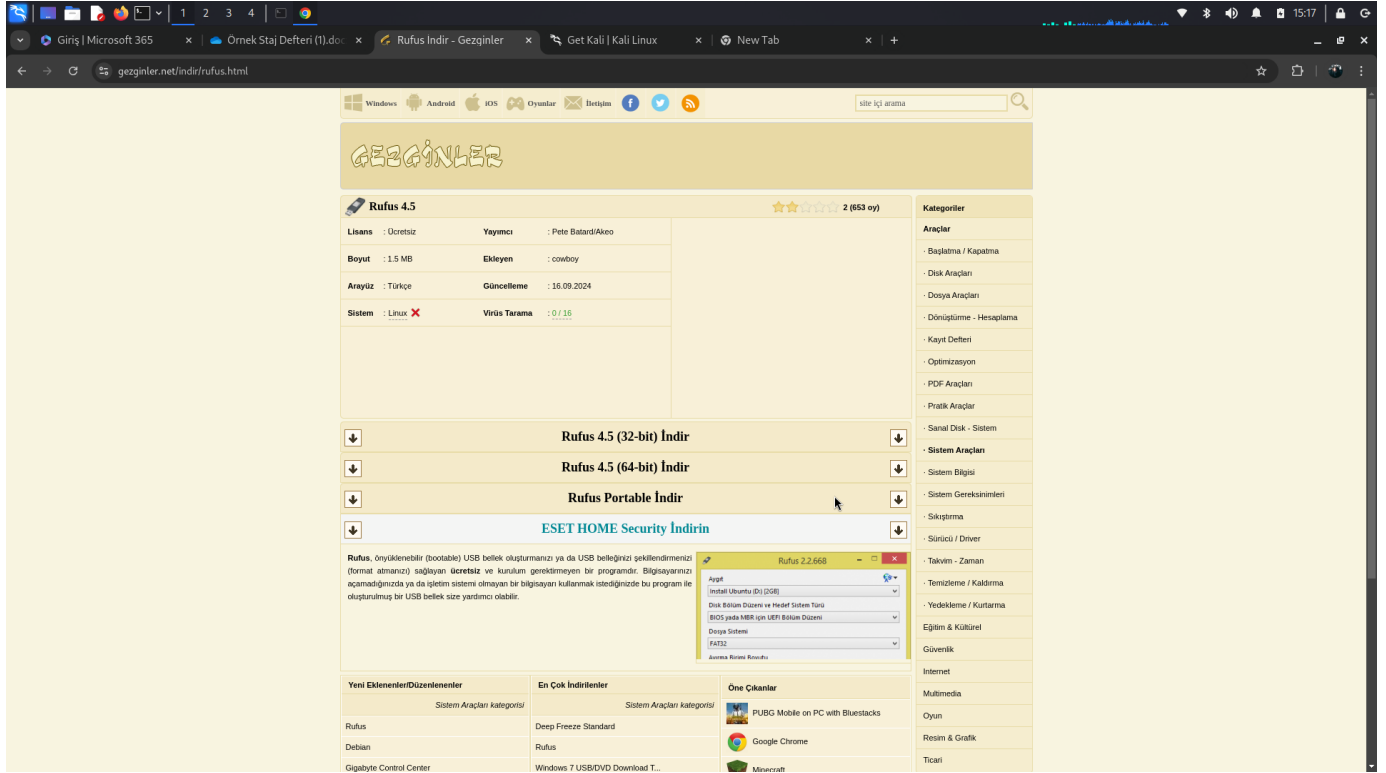


Yukarda paylaştığım resimde kali linuxun kendi sitesinden bu kurulumu gerçekleştirebilirsiniz isterseniz sanal ya da direk kendi donanımınızın üzerine kurma şansınız mevcut bu durumda ben direk olarak artık hep linux kullanmaya karar verdiğimden bilgisayarımı komple yalnızca kali linux kuruyorum windowsu kaldırıp bunun için sol üstteki installer image kısmından indirdiğim linux tabanlı iso dosyasını indiriyorum.

KONTROL SONUCU

KISIM	KURULUM	YAPRAK NO:8
YAPILAN İŞ	KALİ LINUX KURULUMU	TARİH: 30/07/2024

İndirdiğimiz iso dosyasını kullanacağımız bir uygulama ile en az 8 gb olan bir flash belleğe işletim sistemini kurulabilir hale getireceğiz bu işlem sırasında ben rufus uygulamasını kullandım , aşağıda paylaştığım resimden bilgisayarınızın donanımına göre indirebilirsiniz ben 64 bit indiriyorum.

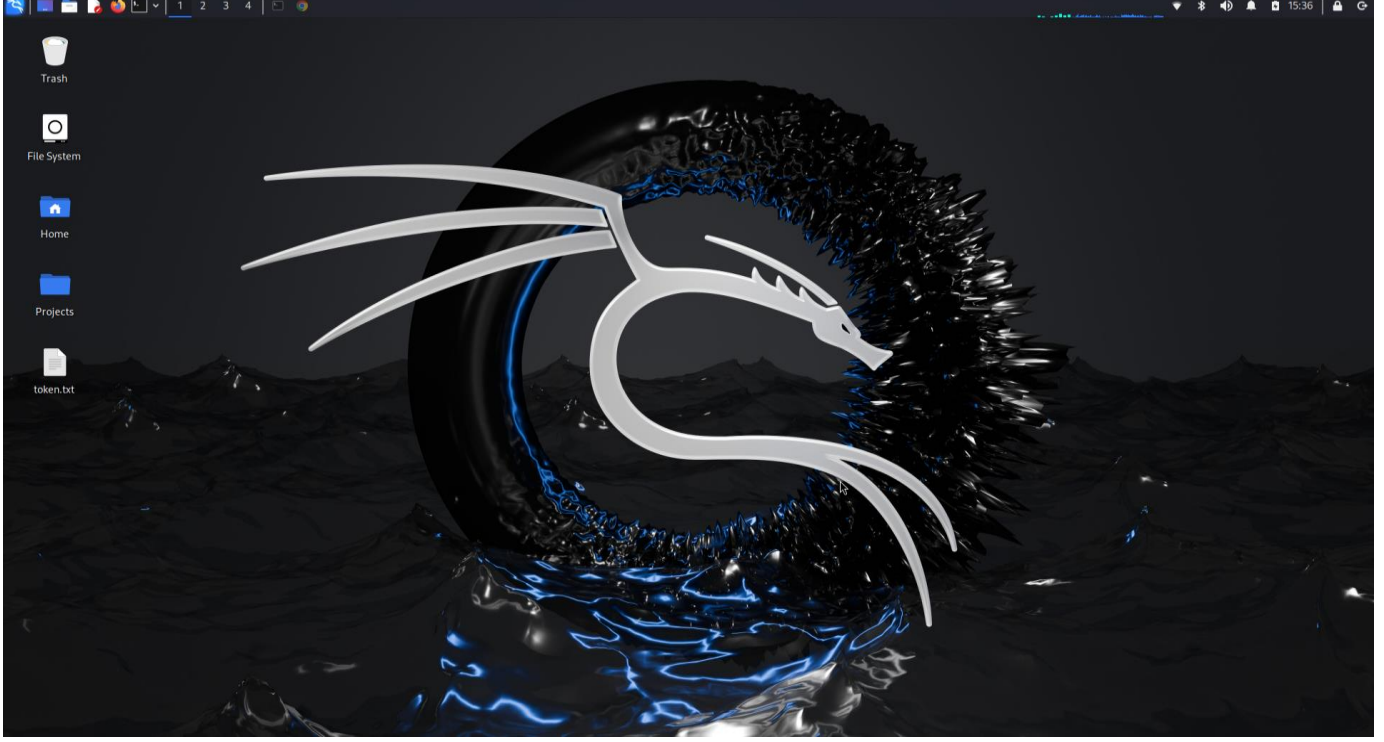


Rufusu kurduktan sonra tek yapmanız flashı bilgisayarınıza takıp uygulamayı açtıktan sonra indirdiğiniz kali linux iso dosyasının konumunu seçip işlemi başlattım. Daha sonrasında gerekli bios işlemlerini bilgisayarı kapattıktan sonra benim laptopumda ayrıyeten güvenlik segmenti var sağ tarafında bios ekranına girmek için toplu iğne ile ona batırıp biosda gerekli işlemleri yani usb yi devreye sokuyorum daha sonrasında tekrar kapatıp açıp aynı şekilde toplu iğne ile basıp kurulum için gerekli usb yi seçip linux kurulumunu başlatıyorum.

Geri kalan kurulum kısmını artık burda anlatamayacağım çünkü hepsini fotoğrafını paylaşmam çok fazla yer kaplıyor ben ondan sonrası ise belli başlı aşamalar var size uygun klavyeyi , dili , tarihi ve bazı linux ayarlarını yaptıktan sonra yazılım yüklemesi yapıyorsunuz ve yazılım yüklemenden önce hard diskinizi manage ediyosunuz daha sonrasında gerekli yerlere linux yazılımını kurup kurulum işlemini bitiriyosunuz.

KONTROL SONUCU

KISIM	KURULUM	YAPRAK NO:9
YAPILAN İŞ	KALİ LİNX KURULUMU	TARİH: 31/07/2024



Benim kurulumdan sonraki ekranım bu şekilde bu iş sadece linuxu kurmakla bitmiyor tabi ki cli komutlarını iyi bilmeniz özümsemeniz lazım örnek veriyorum en basitlerinden `cd` , `ls` , `rm` , `mv` , `cp` , `mkdir` , `touch` , `git` , `wget` , `sudo` , `apt-get install` , `apt-get update` , `apt install` şeklinde gidiyor .

Linux kurulumunun hemen ardından yapılması gereken paket kurulum işlemlerini sizinle paylaşmak istedim bu noktada sadece bir tane komutumuz var bunu cli kısmına yazarsanız bu sorun da ortadan kalkacaktır ancak bu işlem için internet gereklidir :

```
[sudo apt-get update && sudo apt-get upgrade -y]
```

Bu komutu yazmanız yeterlidir o gerekli kurulumları kendi yapacaktır ve artık kali linux işletim sistemi kullanmaya hazır hale gelmiş bulunmakta artık linux kurulumunu bu günden itibaren bitirmiş bulunmaktayız geriye kalan işlemlerimizin hepsi linux işletim sisteminde gerçekleştirilecektir.

KONTROL SONUCU

KISIM

YAZILIM

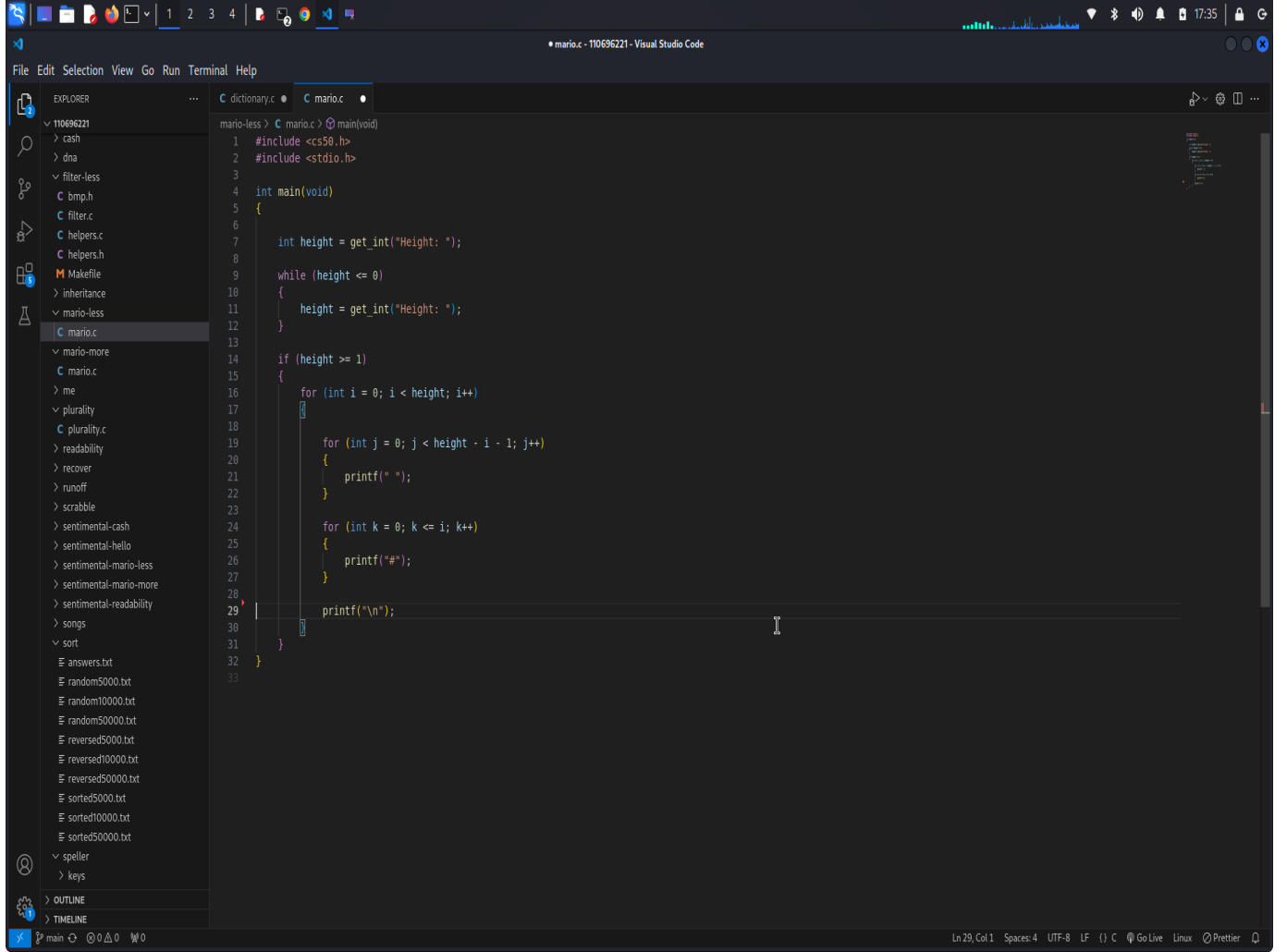
YAPRAK NO:10

YAPILAN İŞ

C DİLİNDE ALGORİTMA

TARİH: 01/08/2024

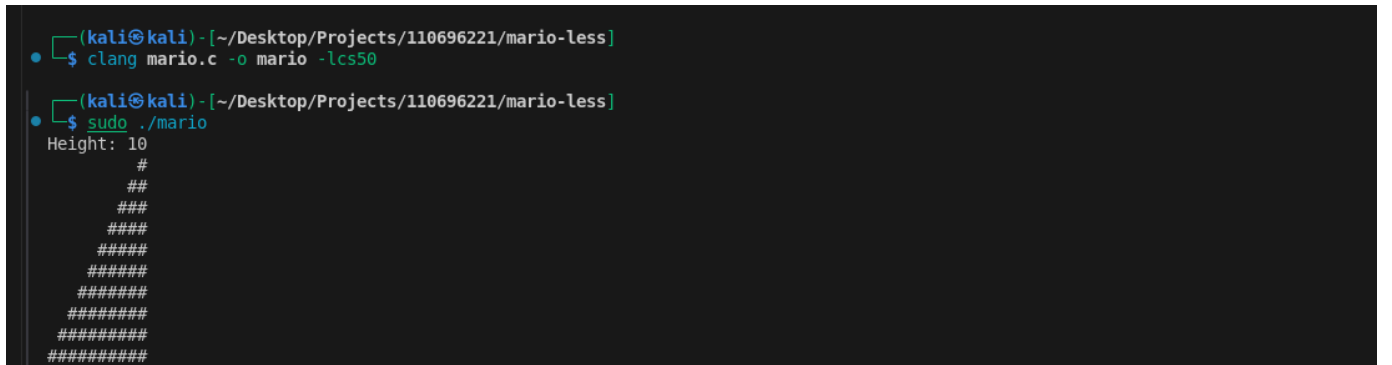
Artık linux kurulumunu bitirmiş bulunmaktayım bu günden itibaren çözmüş olduğum algoritmaları paylaşacağım bu algoritmalar C ve Python dilinde çözülmüştür elimden geldiğince sıdırmaya çalışacağım bu algoritmalar tamamen kendi çözümüm olup dahil olduğum CS50 kursunda çözdüğüm algoritmalar dır. Öncelikle C dilinde kolay dan başlayıp zorlara doğru gidiyor bu süre zarfında Python da var kursun devamında da bazı diller mevcut ama burda sizinle sadece bu ikisini paylaşacağım sıđması açısından.



```
mario-less > C mario.c > main(void)
1 #include <cs50.h>
2 #include <stdio.h>
3
4 int main(void)
5 {
6
7     int height = get_int("Height: ");
8
9     while (height <= 0)
10     {
11         height = get_int("Height: ");
12     }
13
14     if (height >= 1)
15     {
16         for (int i = 0; i < height; i++)
17         {
18             for (int j = 0; j < height - i - 1; j++)
19             {
20                 printf(" ");
21             }
22
23             for (int k = 0; k <= i; k++)
24             {
25                 printf("#");
26             }
27
28             printf("\n");
29         }
30     }
31 }
32
33
```

Yukarda paylaştığım kod parçası tıpkı mariodaki gibi blokları yukardan aşağıya artacak şekilde kullanıcıdan alınan input ile height sayısına göre bir blok dizisi oluşturur .

OUTPUT:



```
(kali@kali) - [~/Desktop/Projects/110696221/mario-less]
$ clang mario.c -o mario -lcs50

(kali@kali) - [~/Desktop/Projects/110696221/mario-less]
$ sudo ./mario
Height: 10
#
##
###
####
#####
#####
#####
#####
#####
#####
```

KONTROL SONUCU		
KISIM	YAZILIM	YAPRAK NO:11
YAPILAN İŞ	C DİLİNDE ALGORİTMA	TARİH:02/08/2024

Bir önceki gün yaptığım projenin biraz daha zor versiyonunu bugün yapacağız aslında çok bir farkı yok ama daha kompleks bir yapıya sahip sadece diğer tarafa simetrik olarak yansımısını da yapmış olacağız.

The screenshot shows a Visual Studio Code editor window with the following details:

- File Explorer (Left Sidebar):** Displays a file tree with folders like 'dictionary.c', 'mario.c', and 'mario-more'. The 'mario.c' file is selected.
- Editor Window:** Contains the source code for 'mario.c'. The code is as follows:


```

1 #include <cs50.h>
2 #include <stdio.h>
3
4 int main(void)
5 {
6     int height = get_int("Height: ");
7     while (height <= 0)
8     {
9         height = get_int("Height: ");
10    }
11
12    if (height >= 1 && height <= 8)
13    {
14        for (int i = 0; i < height; i++)
15        {
16            for (int j = 0; j < height - i - 1; j++)
17            {
18                printf(" ");
19            }
20
21            for (int k = 0; k <= i; k++)
22            {
23                printf("#");
24            }
25
26            for (int a = 0; a < 2; a++)
27            {
28                printf(" ");
29            }
30
31            for (int b = 0; b <= i; b++)
32            {
33                printf("#");
34            }
35
36            printf("\n");
37        }
38    }
39 }
```
- Terminal (Bottom):** Shows the output of the program, which is a pyramid of hashes:


```

#####
#####
#####
#####
#####
#####
#####
#####
```
- Status Bar (Bottom):** Indicates the current file is 'mario.c' and the editor is in 'C' mode.

Burda yapmış olduğumuz işlem sadece öncekinden sonra blokların ardından iki boşluk bıraktırıp karşısına devam ettirmek kodu ikiyle çarpmış gibi olduk bir nevi alt kısımda da sizinle outputunu paylaşacağım.

OUTPUT:

```
(kali㉿kali)-[~/Desktop/Projects/110696221/mario-more]
└─$ clang mario.c -o mario -lcs50

(kali㉿kali)-[~/Desktop/Projects/110696221/mario-more]
└─$ ./mario
Height: 7
#
##
###
####
#####
#####
#####
#####
```

KONTROL SONUCU

KISIM

YAZILIM

YAPRAK NO:12

YAPILAN İŞ

C DİLİNDE ALGORİTMA

TARİH:05/08/2024

```
caesar.c - 110696221 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C dictionary.c C main.c C caesar.c
caesar > C caesar.c >...
1 #include <cs50.h>
2 #include <ctype.h>
3 #include <limits.h>
4 #include <math.h>
5 #include <stdio.h>
6 #include <stdlib.h>
7 #include <string.h>
8
9 int main(int argc, char *argv[])
10 {
11     if (argc != 2)
12     {
13         printf("Usage: ./cipher key\n");
14         return 1;
15     }
16     for (int i = 0; i < strlen(argv[1]); i++)
17     {
18         if (!isdigit(argv[1][i]))
19         {
20             printf("Usage: ./cipher key\n");
21             return 1;
22         }
23     }
24
25     if (true)
26     {
27         int key = atoi(argv[1]);
28         string plaintext = get_string("plaintext: ");
29
30         if (strlen(plaintext))
31         {
32             key = key % 26;
33
34             for (int i = 0; i < strlen(plaintext); i++)
35             {
36                 if (isupper(plaintext[i]))
37                 {
38                     if (plaintext[i] + key > 90)
39                     {
40                         plaintext[i] = (plaintext[i] + key) % 90 + 64;
41                     }
42                     else
43                     {
44                         plaintext[i] += key;
45                     }
46                 }
47             }
48
49             if (islower(plaintext[i]))
50             {
51                 if (plaintext[i] + key > 122)
52                 {
53                     plaintext[i] = ((plaintext[i] + key) % 122) + 96;
54                 }
55                 else
56                 {
57                     plaintext[i] += key;
58                 }
59             }
60         }
61         string ciphertext = plaintext;
62         printf("ciphertext: %s\n", ciphertext);
63         return 0;
64     }
65 }
```

Yukarda gördüğünüz kod caeaser adlı bir şifreleme yöntemidir bu yöntem ile yazdığınız mesajları göndermeden önce şifreleyebilirsiniz yapmanız gereken sadece harflerin ascı kodlarını öğrenip onların üstüne bir key vererek bu keyi ekledikten sonraki orjinal harf yerine şifrenlenmiş harf getiriliyor bunu yaparken de ascı numaraları çok fazla işe yaramakta . Aşağı kısımda outputu sizinle paylaşacağım bu arada bu uygulama ingiliz alfabesine göre yazılmıştır.

OUTPUT:

```
(kali@kali)-[~/Desktop/Projects/110696221/caesar]
$ clang caesar.c -o caesar -lcs50

(kali@kali)-[~/Desktop/Projects/110696221/caesar]
$ ./caesar 10
plaintext: Merhaba Cankiri Karatekin Ailesi Hepinizi Burdan Selamlarimi İletiyorum Hoscalalinn
ciphertext: Wobrkkl Mxksus Ukbkdousx Ksvocs Rozsxsjs Lebnkx Covkwvkbsws İvodsıybew Rycmkukvsxx
```

KONTROL SONUCU

KISIM

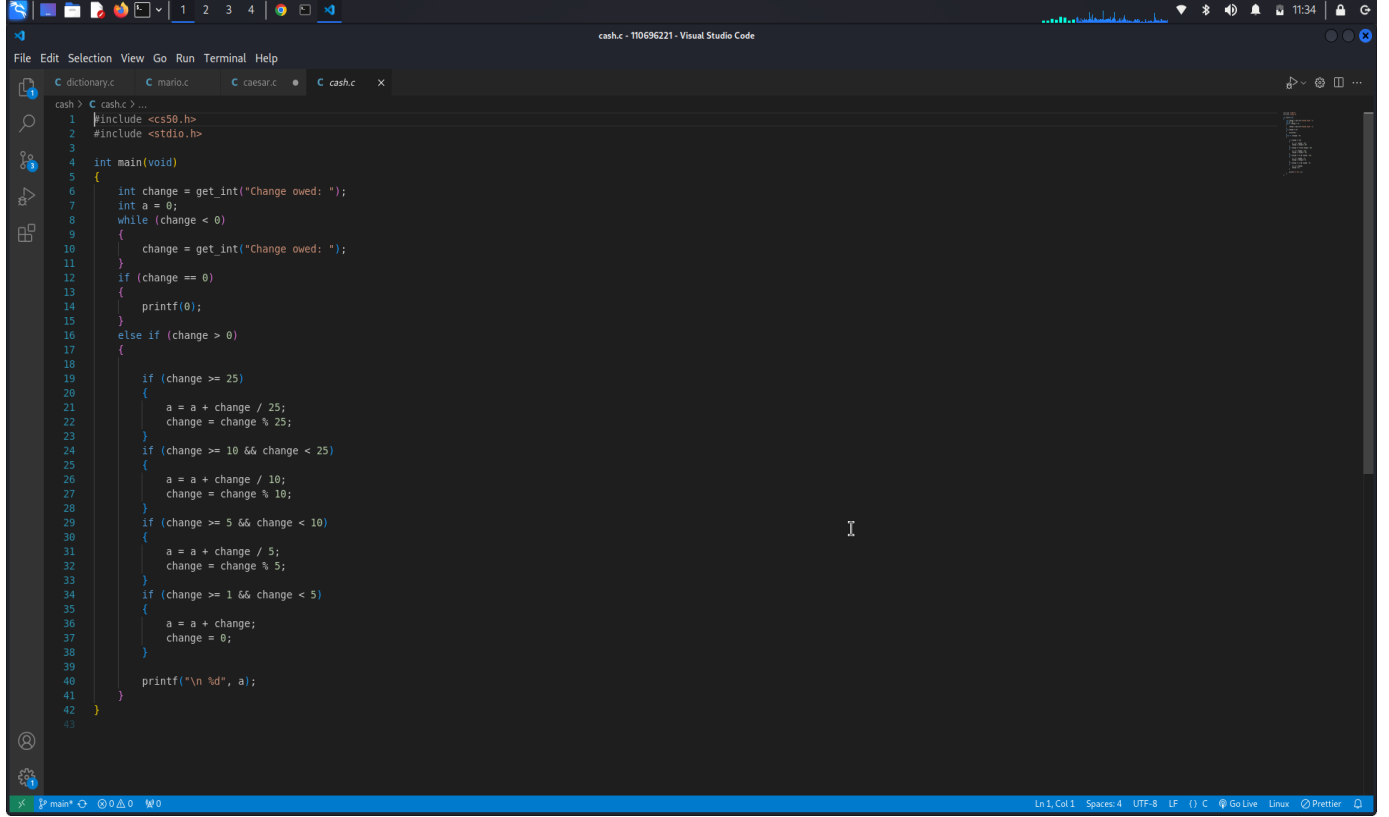
YAZILIM

YAPRAK NO: 13

YAPILAN İŞ

C DİLİNDE ALGORİTMA

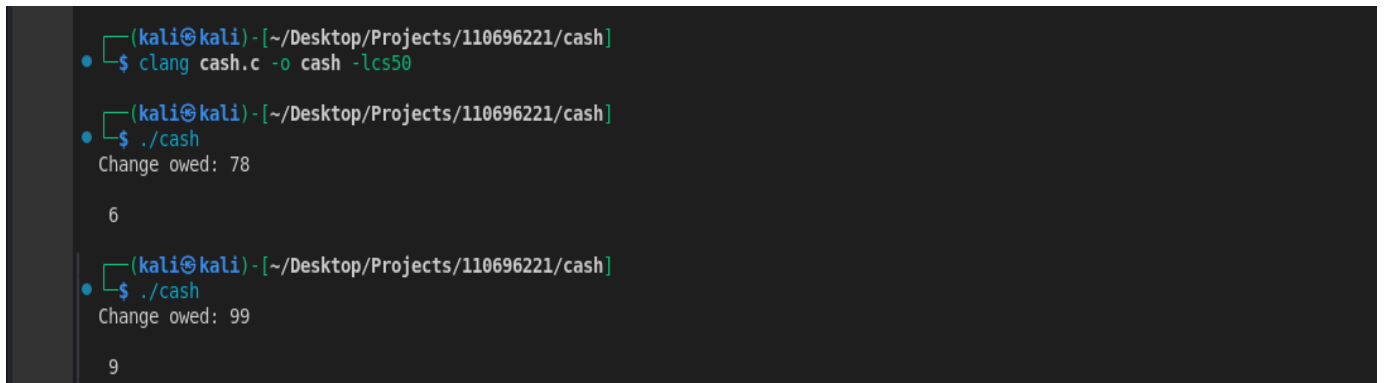
TARİH: 06/08/2024



```
1 #include <cs50.h>
2 #include <stdio.h>
3
4 int main(void)
5 {
6     int change = get_int("Change owed: ");
7     int a = 0;
8     while (change < 0)
9     {
10         change = get_int("Change owed: ");
11     }
12     if (change == 0)
13     {
14         printf(0);
15     }
16     else if (change > 0)
17     {
18         if (change >= 25)
19         {
20             a = a + change / 25;
21             change = change % 25;
22         }
23         if (change >= 10 && change < 25)
24         {
25             a = a + change / 10;
26             change = change % 10;
27         }
28         if (change >= 5 && change < 10)
29         {
30             a = a + change / 5;
31             change = change % 5;
32         }
33         if (change >= 1 && change < 5)
34         {
35             a = a + change;
36             change = 0;
37         }
38         printf("\n %d", a);
39     }
40 }
41
42 }
```

Yukardaki kod parçacığına göre yazılan algoritma para üstünün kaç adet para ile bu para üstünü tamamlayabileceğinizi size output olarak veriyor bunu yaparken de 25 ,10 ,5 ve 1 bizim elimizde olan paralar olarak kabul edip bunlar ile bu para üstünü tamamlayabiliyoruz bunu yaparken de sürekli yukarda verdiğim rakamlara bölerek bu değeri kaydedip daha sonra paranın modunu alarak devam etmesi işleminden ibaret.

OUTPUT:



```
(kali@kali)-[~/Desktop/Projects/110696221/cash]
$ clang cash.c -o cash -lcs50

(kali@kali)-[~/Desktop/Projects/110696221/cash]
$ ./cash
Change owed: 78

6

(kali@kali)-[~/Desktop/Projects/110696221/cash]
$ ./cash
Change owed: 99

9
```

KONTROL SONUCU

KISIM

YAZILIM

YAPRAK NO:14

YAPILAN İŞ

C DİLİNDE ALGORİTMA

TARİH: 07/08/2024

```
1 #include <cs50.h>
2 #include <ctype.h>
3 #include <limits.h>
4 #include <math.h>
5 #include <stdio.h>
6 #include <string.h>
7
8 int main(void)
9 {
10     string text = get_string("Text : ");
11     float letter = 0;
12     float sentence = 0;
13     float space = 0;
14
15     for (int i = 0; i < strlen(text); i++)
16     {
17         if (text[i] != ' ' && text[i] != '.' && text[i] != '?' && text[i] != '\n' &&
18             text[i] != ',' && text[i] != '!' && text[i] != '-' && text[i] != '-')
19         {
20             letter++;
21         }
22         if (text[i] == ' ')
23         {
24             space++;
25         }
26         if (text[i] == '.' || text[i] == '!' || text[i] == '?')
27         {
28             sentence++;
29         }
30     }
31
32     float L = (letter / (space + 1)) * 100;
33     float S = (sentence / (space + 1)) * 100;
34
35     int index = round(0.0588 * L - 0.296 * S - 15.8);
36
37     if (index < 1)
38     {
39         printf("Before Grade 1\n");
40     }
41     else if (index >= 1 && index < 16)
42     {
43         printf("Grade %d\n", index);
44     }
45     else
46     {
47         printf("Grade 16+\n");
48     }
49 }
```

Yukarda paylaştığım resimde bir metnin okunabilirlik seviyesini hesaplayabilen bir algoritma yazdım. Bu algoritma önce metindeki harfleri, boşlukları ve kaç tane cümle olduğunu hesapladıktan sonra bazı matematiksel işlemler yapıyor. Örnek veriyorum: L metindeki 100 kelimedeki harfin ortalamasını hesaplıyor, S ise 100 kelimedeki bir cümlemin ortalamasını hesaplıyor. Bunlara göre seviyeler belirlenip output yazılıyor.

OUTPUT:

```
(kali@kali) - [~/Desktop/Projects/110696221/readability]
• $ clang -o readability readability.c -lcs50 -lm

(kali@kali) - [~/Desktop/Projects/110696221/readability]
• $ ./readability
Text : Merhaba herkese cankiri ailesi burda mi ? selam.
Grade 5

(kali@kali) - [~/Desktop/Projects/110696221/readability]
• $ ./readability
Text : bugün sizlere sunum yapacagiz word,excel ,pdf kullanılacaktır.
Grade 16+
```

KONTROL SONUCU

KISIM

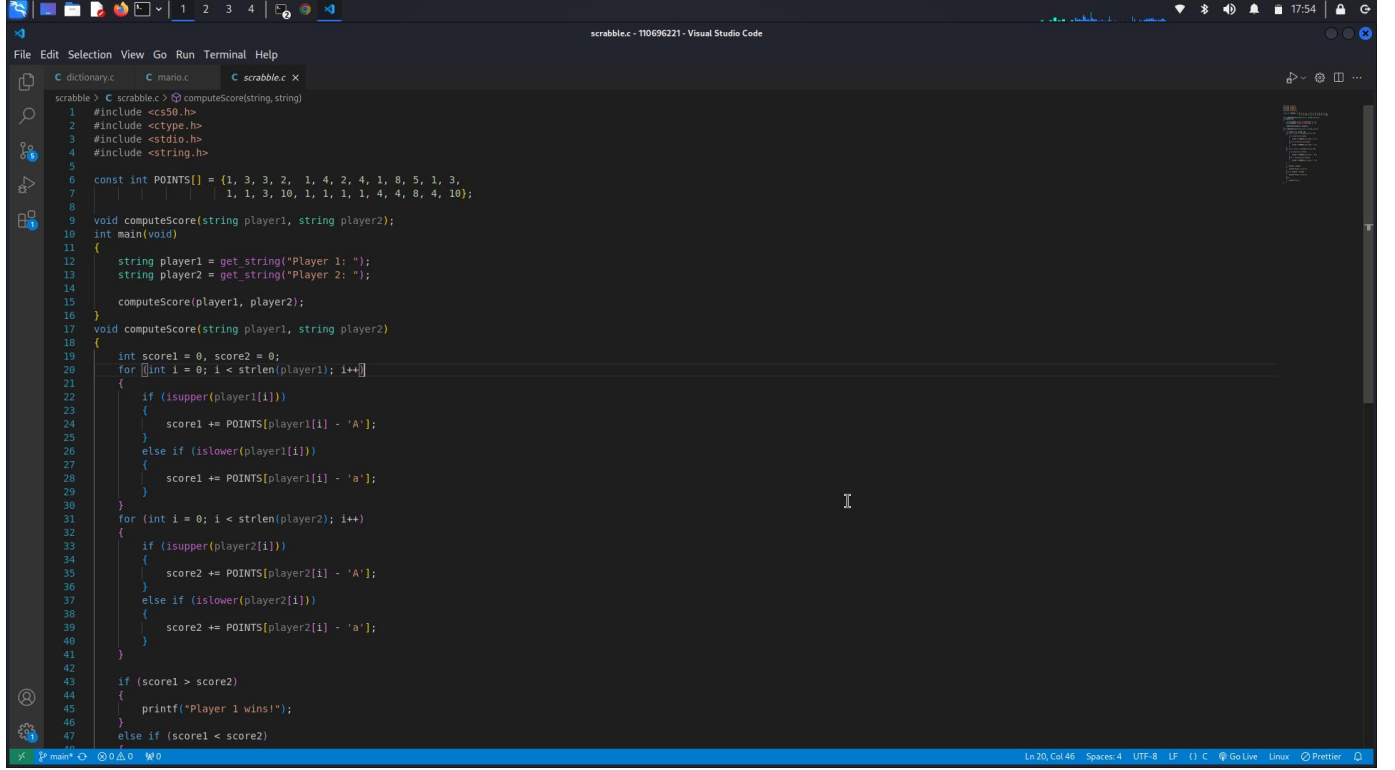
YAZILIM

YAPRAK NO:15

YAPILAN İŞ

C DİLİNDE ALGORİTMA

TARİH: 08/08/2024



```
1 #include <cs50.h>
2 #include <ctype.h>
3 #include <stdio.h>
4 #include <string.h>
5
6 const int POINTS[] = {1, 3, 3, 2, 1, 4, 2, 4, 1, 8, 5, 1, 3,
7                       1, 1, 3, 10, 1, 1, 1, 1, 4, 4, 8, 4, 10};
8
9 void computeScore(string player1, string player2);
10 int main(void)
11 {
12     string player1 = get_string("Player 1: ");
13     string player2 = get_string("Player 2: ");
14
15     computeScore(player1, player2);
16 }
17 void computeScore(string player1, string player2)
18 {
19     int score1 = 0, score2 = 0;
20     for (int i = 0; i < strlen(player1); i++)
21     {
22         if (isupper(player1[i]))
23         {
24             score1 += POINTS[player1[i] - 'A'];
25         }
26         else if (islower(player1[i]))
27         {
28             score1 += POINTS[player1[i] - 'a'];
29         }
30     }
31     for (int i = 0; i < strlen(player2); i++)
32     {
33         if (isupper(player2[i]))
34         {
35             score2 += POINTS[player2[i] - 'A'];
36         }
37         else if (islower(player2[i]))
38         {
39             score2 += POINTS[player2[i] - 'a'];
40         }
41     }
42     if (score1 > score2)
43     {
44         printf("Player 1 wins!");
45     }
46     else if (score1 < score2)
```

```
47     else if (score1 < score2)
48     {
49         printf("Player 2 wins!");
50     }
51     else
52     {
53         printf("Tie!");
54     }
55 }
56
```

Yukardaki algoritmaya göre Amerikan kod standartlarına göre yazılmış ascı tablosundan aldığım verilere göre 26 tane ayrı ayrı büyük ve küçük harfleri Points dizisinde puanlandırıp daha sonra algoritmasını yazarak tek tek ascı tablosundaki yerlerine göre harflerin puanını bulup yazmış olduğum bir algırtmadır output bu sayfaya sığmadığı için aşağıda paylaşacağım .

KONTROL SONUCU

KISIM

YAZILIM

YAPRAK NO:16

YAPILAN İŞ

C DİLİNDE ALGORİTMA

TARİH: 09/08/2024

OUTPUT:

```
(kali@kali)-[~/Desktop/Projects/110696221/scrabble]
$ clang -o scrabble scrabble.c -lcs50 -lm

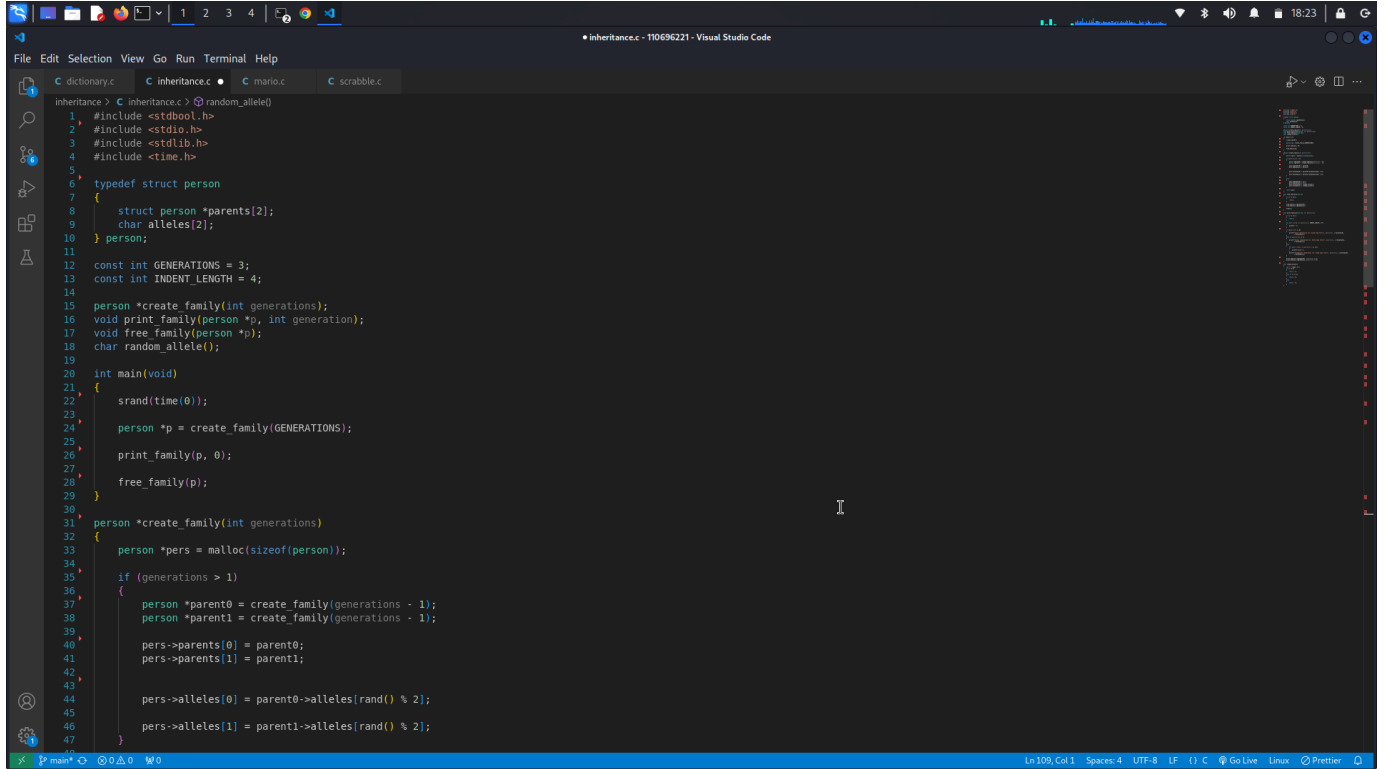
(kali@kali)-[~/Desktop/Projects/110696221/scrabble]
$ ./scrabble
Player 1: Ses
Player 2: Kalem
Player 2 wins!

(kali@kali)-[~/Desktop/Projects/110696221/scrabble]
$ ./scrabble
Player 1: Ahmet
Player 2: Mehmet
Player 2 wins!

(kali@kali)-[~/Desktop/Projects/110696221/scrabble]
$ ./scrabble
Player 1: BurhanCavdar
Player 2: Selam
Player 1 wins!

(kali@kali)-[~/Desktop/Projects/110696221/scrabble]
$ ./scrabble
Player 1: fatih
Player 2: emre
Player 1 wins!
```

Son olarak algoritmaları bitirmeden önce bundan önceki örneklere göre bir tık daha üst seviye olan çözmüş olduğum bir algoritmayı paylaşacağım daha sonrasında yapmış olduğum projelere geçiş yapacağım.



```
inheritance.c - 110696221 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C dictionary C inheritance.c C main.c C scrabble.c
inheritance.c
1 #include <stdbool.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <time.h>
5
6 typedef struct person
7 {
8     struct person *parents[2];
9     char alleles[2];
10 } person;
11
12 const int GENERATIONS = 3;
13 const int INDENT_LENGTH = 4;
14
15 person *create_family(int generations);
16 void print_family(person *p, int generation);
17 void free_family(person *p);
18 char random_allele();
19
20 int main(void)
21 {
22     srand(time(0));
23
24     person *p = create_family(GENERATIONS);
25
26     print_family(p, 0);
27
28     free_family(p);
29 }
30
31 person *create_family(int generations)
32 {
33     person *pers = malloc(sizeof(person));
34
35     if (generations > 1)
36     {
37         person *parent0 = create_family(generations - 1);
38         person *parent1 = create_family(generations - 1);
39
40         pers->parents[0] = parent0;
41         pers->parents[1] = parent1;
42
43         pers->alleles[0] = parent0->alleles[rand() % 2];
44         pers->alleles[1] = parent1->alleles[rand() % 2];
45     }
46
47     return pers;
48 }
```


KONTROL SONUCU

KISIM

YAZILIM

YAPRAK NO:17

YAPILAN İŞ

C DİLİNDE ALGORİTMA

TARİH: 12/08/2024

The screenshot shows a Visual Studio Code editor with a C program titled "inheritance.c". The code is a simulation of a family tree over 10 generations. It includes a header file "random.h" and a "main" function that calls "create_family" and "print_family" for each generation. The "create_family" function recursively creates a family tree structure, and the "print_family" function prints the tree structure. The output console on the right shows the execution results, including the generation number and the blood type of each individual.

```

1 // random.h
2 #ifndef RANDOM_H
3 #define RANDOM_H
4
5 #include <stdio.h>
6 #include <stdlib.h>
7 #include <string.h>
8
9 typedef struct person {
10     char *name;
11     int generation;
12     int blood_type;
13     struct person *parents[2];
14     char *alleles[2];
15 } person_t;
16
17 person_t *create_family(int generations);
18 void free_family(person_t *p);
19 void print_family(person_t *p, int generation);
20
21 #endif
22
23 // inheritance.c
24 #include "random.h"
25
26 // Function to create a family tree
27 person_t *create_family(int generations) {
28     if (generations == 0) {
29         // Create a new person
30         person_t *pers = (person_t *) malloc(sizeof(person_t));
31         pers->name = random_name();
32         pers->generation = generations;
33         pers->blood_type = random_blood_type();
34         pers->parents[0] = NULL;
35         pers->parents[1] = NULL;
36         pers->alleles[0] = random_allele();
37         pers->alleles[1] = random_allele();
38         return pers;
39     } else {
40         // Create parents
41         person_t *parent1 = create_family(generations - 1);
42         person_t *parent2 = create_family(generations - 1);
43
44         // Create child
45         person_t *child = (person_t *) malloc(sizeof(person_t));
46         child->name = random_name();
47         child->generation = generations;
48         child->blood_type = random_blood_type();
49         child->parents[0] = parent1;
50         child->parents[1] = parent2;
51         child->alleles[0] = random_allele();
52         child->alleles[1] = random_allele();
53         free_family(parent1);
54         free_family(parent2);
55         return child;
56     }
57 }
58
59 // Function to free a family tree
60 void free_family(person_t *p) {
61     if (p == NULL) {
62         return;
63     }
64     free_family(p->parents[0]);
65     free_family(p->parents[1]);
66     free(p);
67 }
68
69 // Function to print a family tree
70 void print_family(person_t *p, int generation) {
71     if (p == NULL) {
72         return;
73     }
74     printf("Generation %d: %s (Blood Type: %c)\n", generation, p->name, p->blood_type);
75     for (int i = 0; i < generation; i++) {
76         printf("  ");
77     }
78     printf("\n");
79     if (generation == 0) {
80         printf("Child (Generation %d): blood type %c\n", generation, p->alleles[0], p->alleles[1]);
81     } else if (generation == 1) {
82         printf("Parent (Generation %d): blood type %c\n", generation, p->alleles[0], p->alleles[1]);
83     }
84 }
85
86 // Main function
87 int main() {
88     person_t *family = create_family(10);
89     print_family(family, 10);
90     free_family(family);
91     return 0;
92 }

```

The screenshot displays the Visual Studio Code editor with a C program titled "inheritance.c". The code is as follows:

```

inheritance.c > inheritance.c > random_allele()
73 void print_family(person *p, int generation)
74 {
75     printf("Parent (Generation %i): blood type %c%c\n", generation, p->alleles[0],
76           p->alleles[1]);
77 }
78 else
79 {
80     for (int i = 0; i < generation - 2; i++)
81     {
82         printf("Great-");
83     }
84     printf("Grandparent (Generation %i): blood type %c%c\n", generation, p->alleles[0],
85           p->alleles[1]);
86 }
87
88 print_family(p->parents[0], generation + 1);
89 print_family(p->parents[1], generation + 1);
90 }
91
92 char random_allele()
93 {
94     int r = rand() % 3;
95     if (r == 0)
96     {
97         return 'A';
98     }
99     else if (r == 1)
100     {
101         return 'B';
102     }
103     else
104     {
105         return 'O';
106     }
107 }
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125

```

The interface includes a file explorer on the left showing the project structure, a code editor in the center, and an output console on the right. The status bar at the bottom indicates the current file is "main.c" and the editor is in "main" mode.

KONTROL SONUCU

KISIM	YAZILIM	YAPRAK NO:18
YAPILAN İŞ	C DİLİNDE ALGORİTMA	TARİH: 13/08/2024

Yukardaki algoritmaya göre 3 çeşit jenerasyon belirlenip bu jenerasyonlara göre yukardan aşağıya grandparent , parent, child olarak gen aktarımı algoritmasını görmekteyiz bu algorithmada create kısmında özyinelemeli fonksiyon kullanılmıştır.

OUTPUT:

```
(kali@kali) - [~/Desktop/Projects/110696221/inheritance]
• $ clang -o inheritance inheritance.c -lcs50 -lm

(kali@kali) - [~/Desktop/Projects/110696221/inheritance]
• $ ./inheritance
Child (Generation 0): blood type OA
Parent (Generation 1): blood type OB
Grandparent (Generation 2): blood type OA
Grandparent (Generation 2): blood type OB
Parent (Generation 1): blood type AB
Grandparent (Generation 2): blood type AA
Grandparent (Generation 2): blood type BA

(kali@kali) - [~/Desktop/Projects/110696221/inheritance]
• $ ./inheritance
Child (Generation 0): blood type BO
Parent (Generation 1): blood type BA
Grandparent (Generation 2): blood type BA
Grandparent (Generation 2): blood type AA
Parent (Generation 1): blood type OO
Grandparent (Generation 2): blood type OO
Grandparent (Generation 2): blood type AO

(kali@kali) - [~/Desktop/Projects/110696221/inheritance]
• $ ./inheritance
Child (Generation 0): blood type BB
Parent (Generation 1): blood type BA
Grandparent (Generation 2): blood type BB
Grandparent (Generation 2): blood type OA
Parent (Generation 1): blood type BA
Grandparent (Generation 2): blood type BB
Grandparent (Generation 2): blood type BA
```

Buraya kadar paylaştığım kodlar algoritmayı anlayıp çözebilmek bunu beceri haline getirebilmekten oluşuyordu artık biraz daha gerçek hayata yönelik yaptığım projelerden bahsetmek istiyorum bu noktada ben Python ile FastAPI kütüphanesini kullandım daha sonra React.js kullanarak basit bir arayüz yaptım ve bu ikisi arasında local ağda bağlantı kurarak arayüzden crud işlemlerini back end kısmına gerek database olsun gerek json dosyası olsun iki taraf arasında halk dilinde Web API olarak bilinen uygulamayı yazıp hem back end hem de front end kısmında çalışmış oldum ve bunu yaparken birden fazla kütüphane ve paket kullandım.

KONTROL SONUCU

KISIM

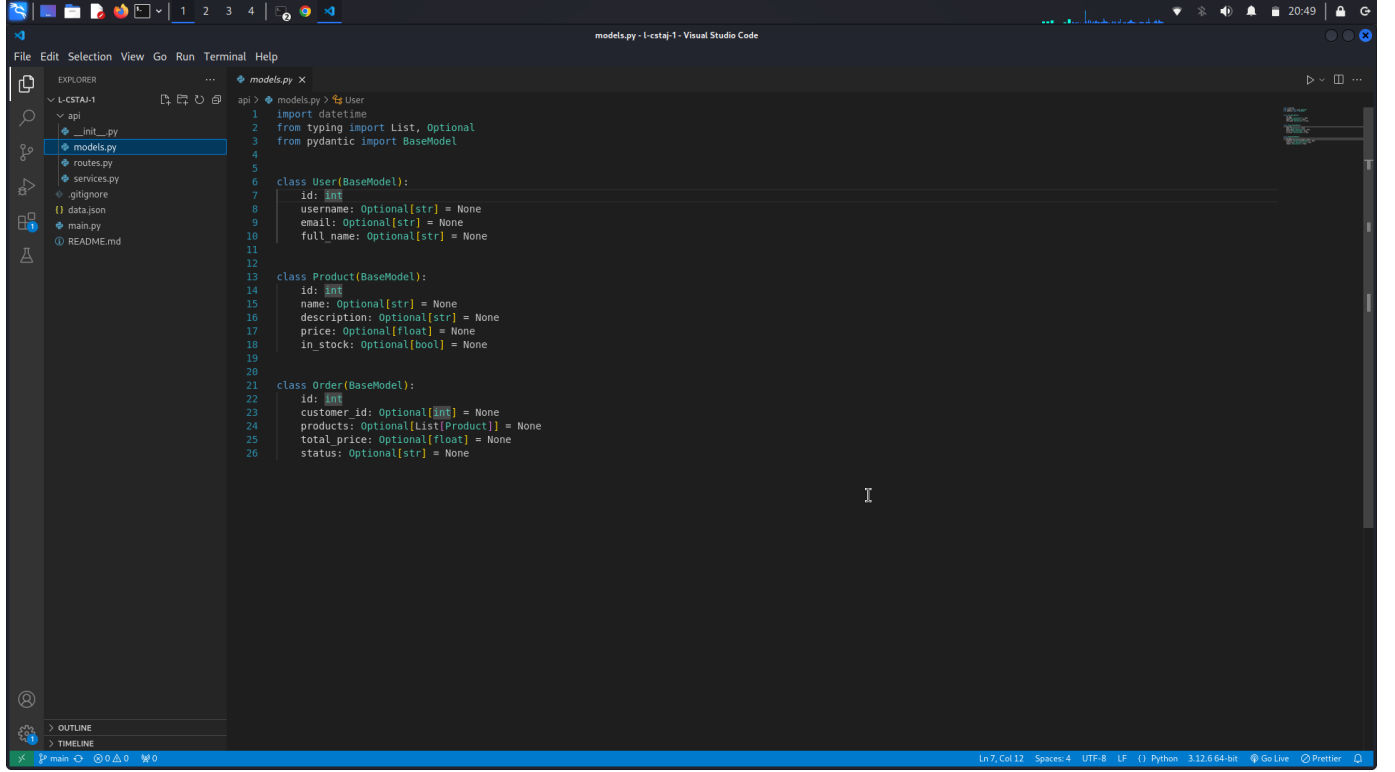
YAZILIM

YAPRAK NO:19

YAPILAN İŞ

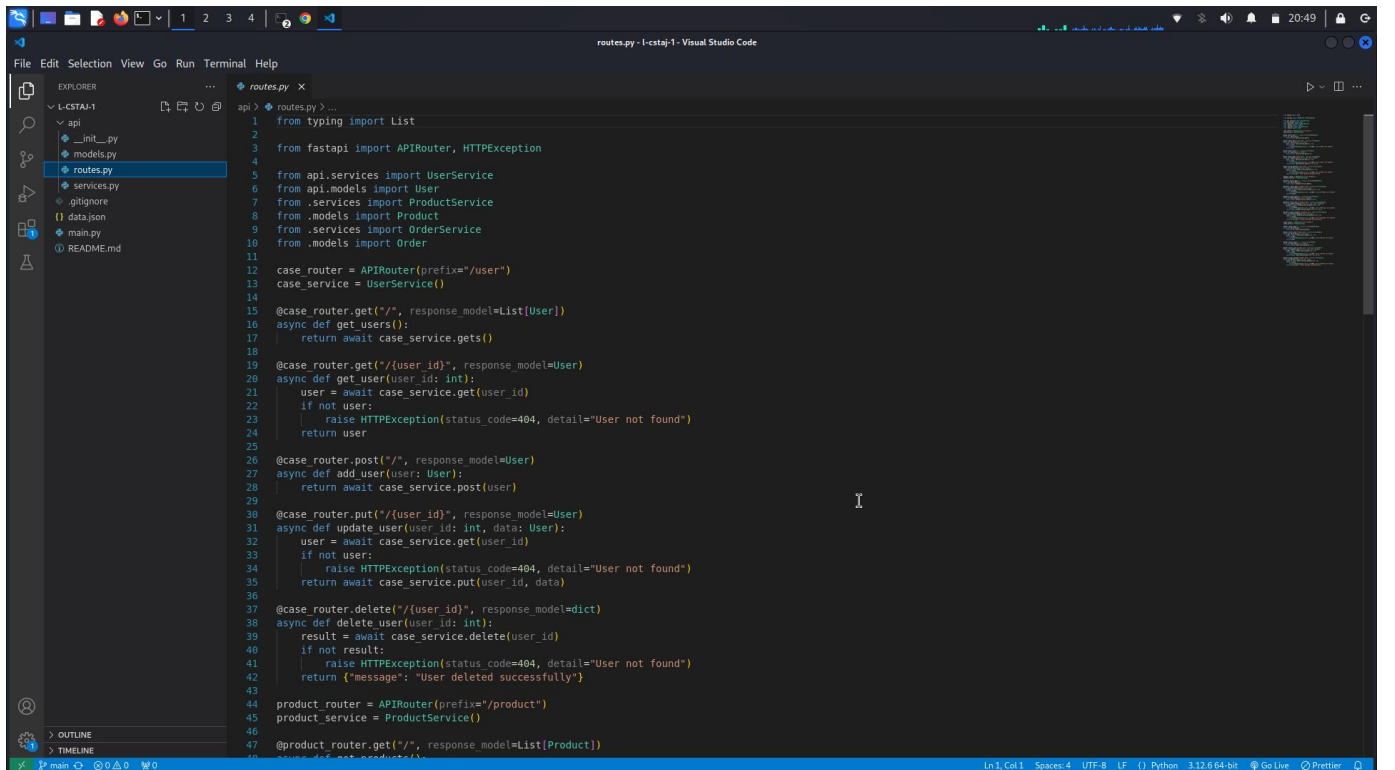
C DİLİNDE ALGORİTMA

TARİH: 14/08/2024



```
1 import datetime
2 from typing import List, Optional
3 from pydantic import BaseModel
4
5
6 class User(BaseModel):
7     id: int
8     username: Optional[str] = None
9     email: Optional[str] = None
10     full_name: Optional[str] = None
11
12
13 class Product(BaseModel):
14     id: int
15     name: Optional[str] = None
16     description: Optional[str] = None
17     price: Optional[float] = None
18     in_stock: Optional[bool] = None
19
20
21 class Order(BaseModel):
22     id: int
23     customer_id: Optional[int] = None
24     products: Optional[List[Product]] = None
25     total_price: Optional[float] = None
26     status: Optional[str] = None
```

Yukarda gördüğünüz FastAPI yapısında kullanılan tier design yapısını kullanmış bulunmaktayız bu yapı gördüğünüz üzere 3'e ayrılıyor models.py , routes.py , services.py öncelikle burda gördüğünüz model yapısı bizim database ile python sınıfları arasında bağlantı kurmamızı yarıyor gördüğünüz üzere burda pydantic yapılı 3 tane model kullanmışız.



```
1 from typing import List
2 from fastapi import APIRouter, HTTPException
3 from api.services import UserService
4 from api.models import User
5 from .services import ProductService
6 from .models import Product
7 from .services import OrderService
8 from .models import Order
9
10 case_router = APIRouter(prefix="/user")
11 case_service = UserService()
12
13 @case_router.get("/", response_model=List[User])
14 async def get_users():
15     return await case_service.gets()
16
17 @case_router.get("/{user_id}", response_model=User)
18 async def get_user(user_id: int):
19     user = await case_service.get(user_id)
20     if not user:
21         raise HTTPException(status_code=404, detail="User not found")
22     return user
23
24 @case_router.post("/", response_model=User)
25 async def add_user(user: User):
26     return await case_service.post(user)
27
28 @case_router.put("/{user_id}", response_model=User)
29 async def update_user(user_id: int, data: User):
30     user = await case_service.get(user_id)
31     if not user:
32         raise HTTPException(status_code=404, detail="User not found")
33     return await case_service.put(user_id, data)
34
35 @case_router.delete("/{user_id}", response_model=dict)
36 async def delete_user(user_id: int):
37     result = await case_service.delete(user_id)
38     if not result:
39         raise HTTPException(status_code=404, detail="User not found")
40     return {"message": "User deleted successfully"}
41
42 product_router = APIRouter(prefix="/product")
43 product_service = ProductService()
44
45 @product_router.get("/", response_model=List[Product])
46 async def get_products():
47     return await product_service.gets()
```

KONTROL SONUCU

KISIM

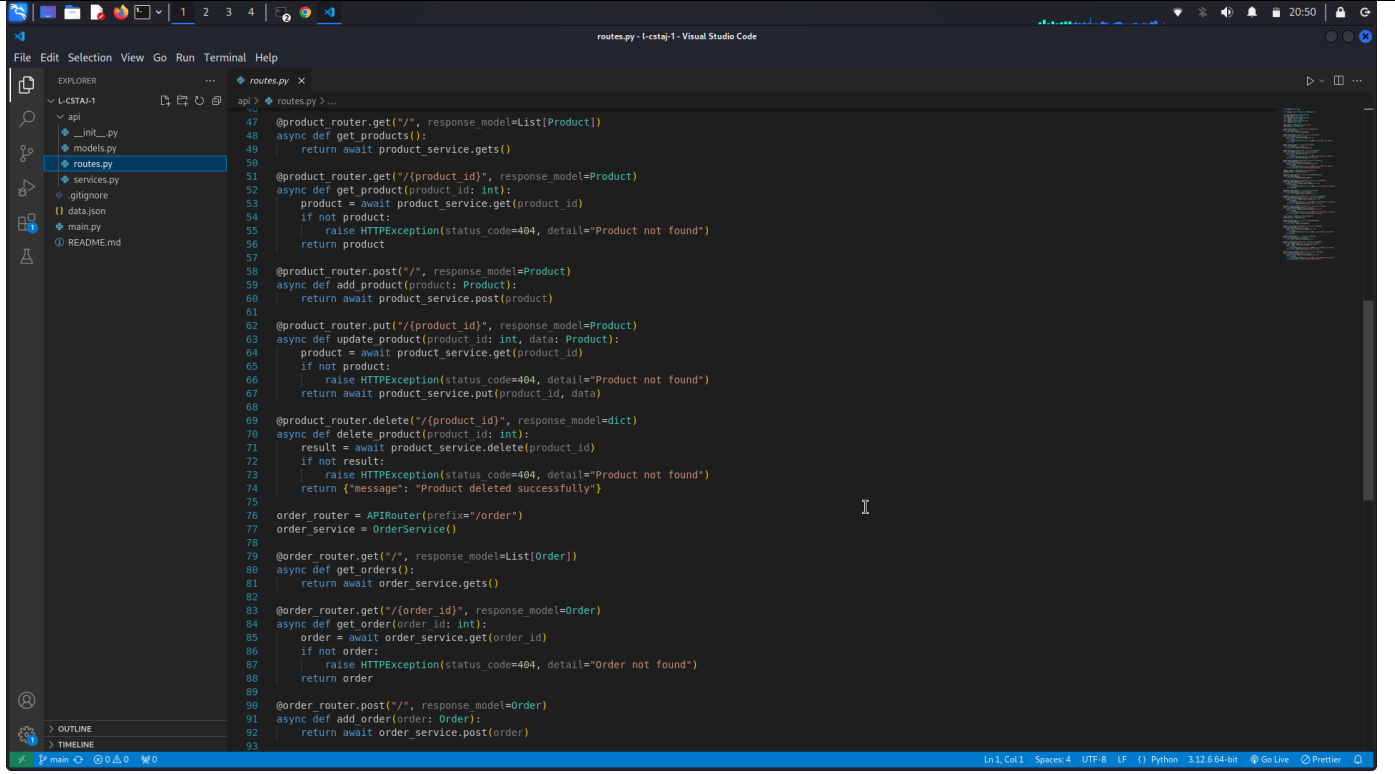
YAZILIM

YAPRAK NO:20

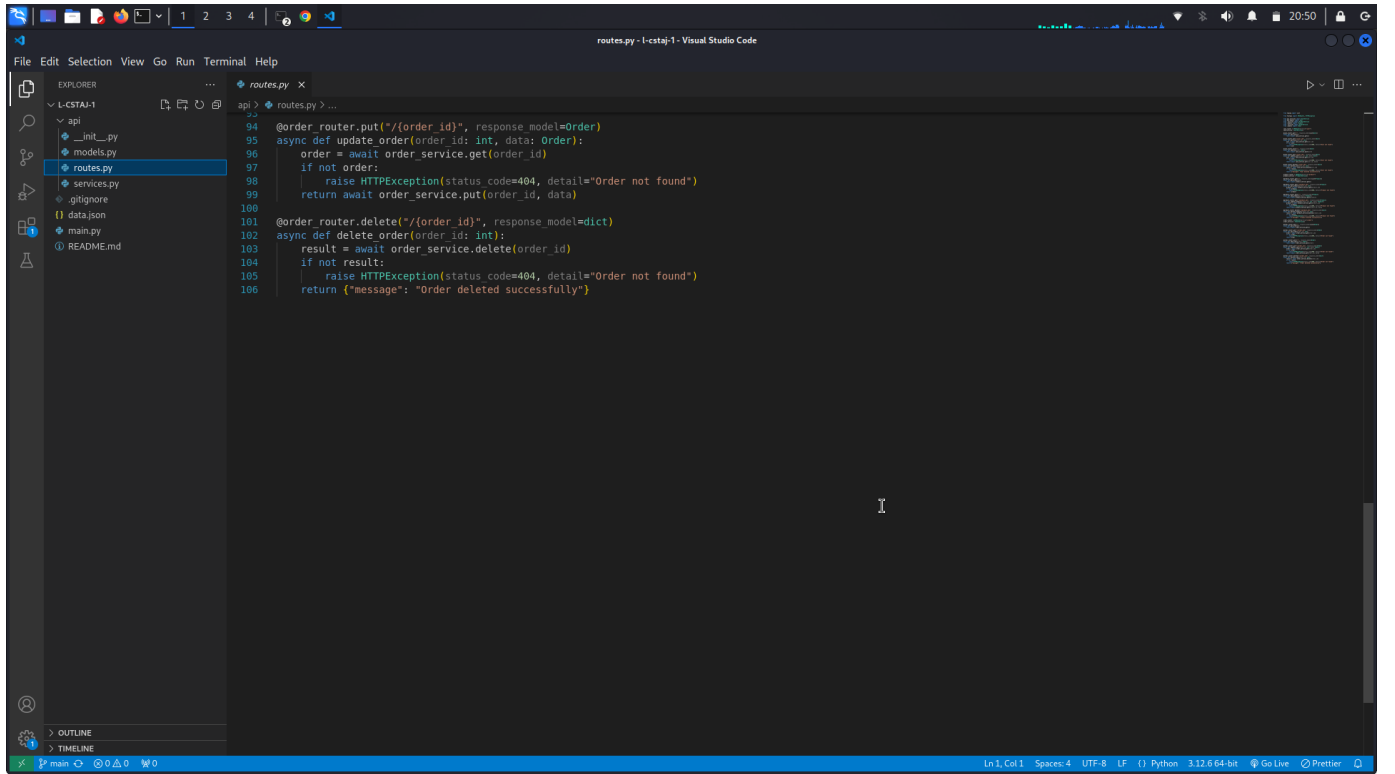
YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 15/08/2024



```
47 @product_router.get("/", response_model=List[Product])
48 async def get_products():
49     return await product_service.gets()
50
51 @product_router.get("/{product_id}", response_model=Product)
52 async def get_product(product_id: int):
53     product = await product_service.get(product_id)
54     if not product:
55         raise HTTPException(status_code=404, detail="Product not found")
56     return product
57
58 @product_router.post("/", response_model=Product)
59 async def add_product(product: Product):
60     return await product_service.post(product)
61
62 @product_router.put("/{product_id}", response_model=Product)
63 async def update_product(product_id: int, data: Product):
64     product = await product_service.get(product_id)
65     if not product:
66         raise HTTPException(status_code=404, detail="Product not found")
67     return await product_service.put(product_id, data)
68
69 @product_router.delete("/{product_id}", response_model=dict)
70 async def delete_product(product_id: int):
71     result = await product_service.delete(product_id)
72     if not result:
73         raise HTTPException(status_code=404, detail="Product not found")
74     return {"message": "Product deleted successfully"}
75
76 order_router = APIRouter(prefix="/order")
77 order_service = OrderService()
78
79 @order_router.get("/", response_model=List[Order])
80 async def get_orders():
81     return await order_service.gets()
82
83 @order_router.get("/{order_id}", response_model=Order)
84 async def get_order(order_id: int):
85     order = await order_service.get(order_id)
86     if not order:
87         raise HTTPException(status_code=404, detail="Order not found")
88     return order
89
90 @order_router.post("/", response_model=Order)
91 async def add_order(order: Order):
92     return await order_service.post(order)
93
```



```
94 @order_router.put("/{order_id}", response_model=Order)
95 async def update_order(order_id: int, data: Order):
96     order = await order_service.get(order_id)
97     if not order:
98         raise HTTPException(status_code=404, detail="Order not found")
99     return await order_service.put(order_id, data)
100
101 @order_router.delete("/{order_id}", response_model=dict)
102 async def delete_order(order_id: int):
103     result = await order_service.delete(order_id)
104     if not result:
105         raise HTTPException(status_code=404, detail="Order not found")
106     return {"message": "Order deleted successfully"}

```

Burda gördüğünüz router yapısında yapılan işlemlerin google’da tarama yapmış olduğunuz linklerin yolu tanımlandırılma işlemi ve gelen requestlerin doğruluğu try catch bloklarıyla belirlenip bir response döndürülür. Bir sonraki aşamada service yapısını anlatacağım.

KONTROL SONUCU

KISIM

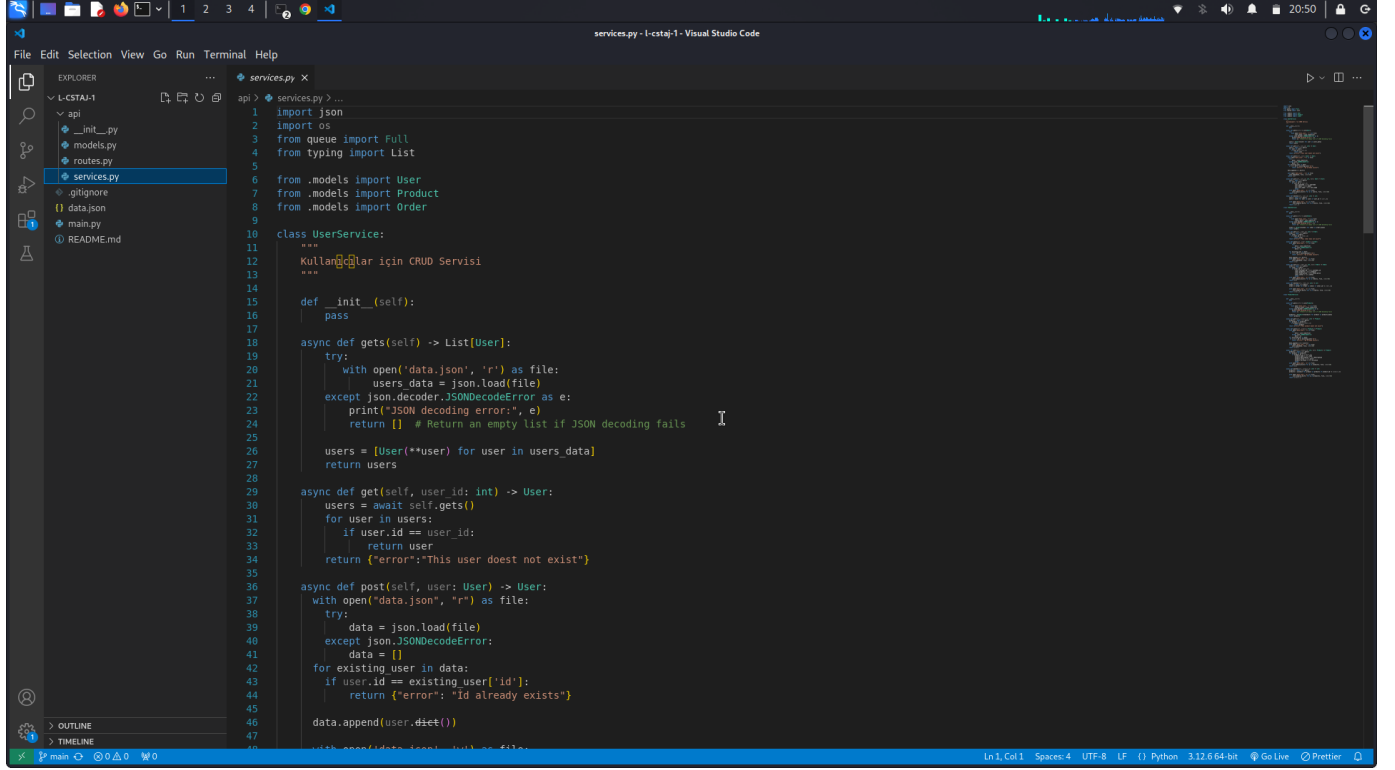
YAZILIM

YAPRAK NO: 21

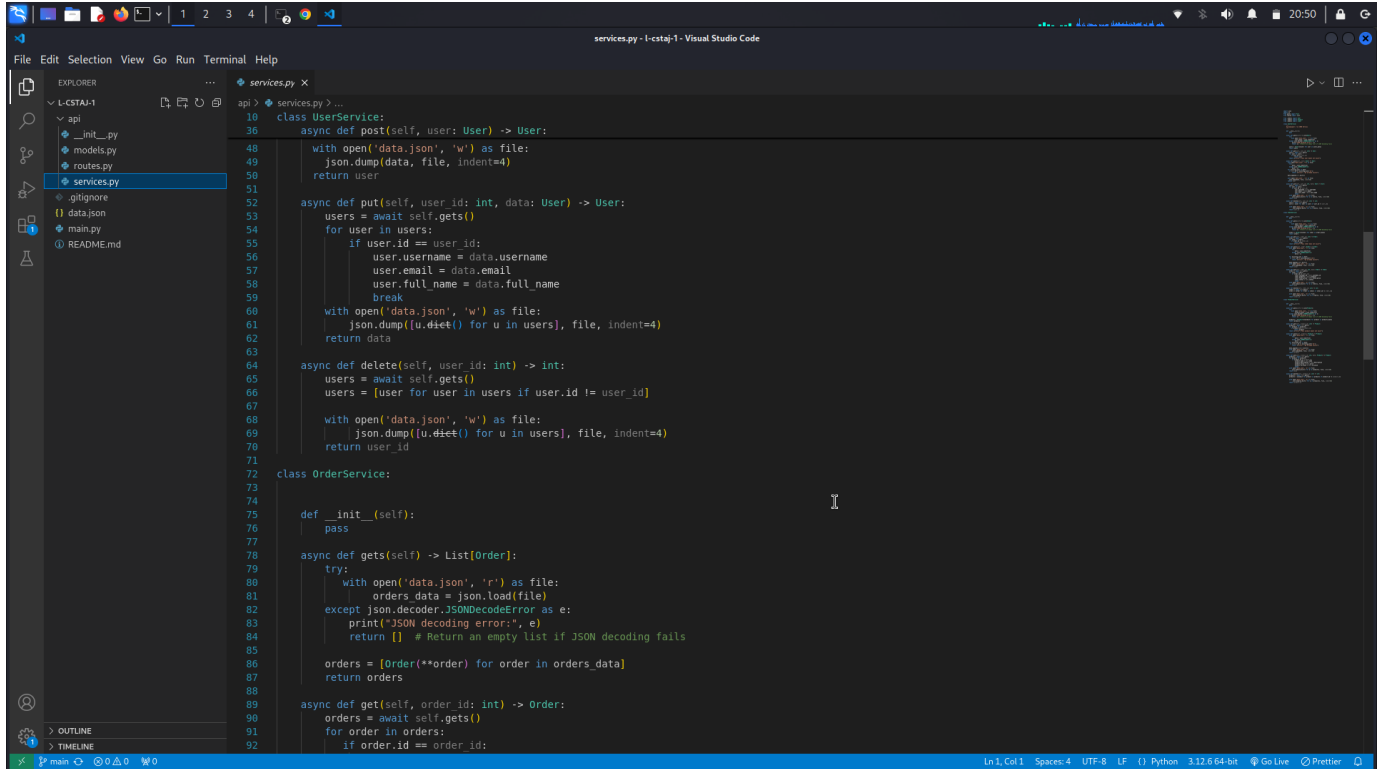
YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 16/08/2024



```
1 import json
2 import os
3 from queue import Full
4 from typing import List
5
6 from .models import User
7 from .models import Product
8 from .models import Order
9
10 class UserService:
11     """
12     Kullanıcılar için CRUD Servisi
13     """
14
15     def __init__(self):
16         pass
17
18     async def gets(self) -> List[User]:
19         try:
20             with open('data.json', 'r') as file:
21                 users_data = json.load(file)
22             except json.decoder.JSONDecodeError as e:
23                 print("JSON decoding error:", e)
24                 return [] # Return an empty list if JSON decoding fails
25
26             users = [User(**user) for user in users_data]
27             return users
28
29     async def get(self, user_id: int) -> User:
30         users = await self.gets()
31         for user in users:
32             if user.id == user_id:
33                 return user
34         return {"error": "This user does not exist"}
35
36     async def post(self, user: User) -> User:
37         with open('data.json', 'r') as file:
38             try:
39                 data = json.load(file)
40             except json.decoder.JSONDecodeError:
41                 data = []
42         for existing_user in data:
43             if existing_user['id'] == user.id:
44                 return {"error": "Id already exists"}
45         data.append(user.dict())
46         with open('data.json', 'w') as file:
47             json.dump(data, file, indent=4)
```



```
10 class UserService:
11     async def post(self, user: User) -> User:
12         with open('data.json', 'w') as file:
13             json.dump(data, file, indent=4)
14         return user
15
16     async def put(self, user_id: int, data: User) -> User:
17         users = await self.gets()
18         for user in users:
19             if user.id == user_id:
20                 user.username = data.username
21                 user.email = data.email
22                 user.full_name = data.full_name
23                 break
24         with open('data.json', 'w') as file:
25             json.dump([u.dict() for u in users], file, indent=4)
26         return data
27
28     async def delete(self, user_id: int) -> int:
29         users = await self.gets()
30         users = [user for user in users if user.id != user_id]
31
32         with open('data.json', 'w') as file:
33             json.dump([u.dict() for u in users], file, indent=4)
34         return user_id
35
36 class OrderService:
37
38     def __init__(self):
39         pass
40
41     async def gets(self) -> List[Order]:
42         try:
43             with open('data.json', 'r') as file:
44                 orders_data = json.load(file)
45             except json.decoder.JSONDecodeError as e:
46                 print("JSON decoding error:", e)
47                 return [] # Return an empty list if JSON decoding fails
48
49             orders = [Order(**order) for order in orders_data]
50             return orders
51
52     async def get(self, order_id: int) -> Order:
53         orders = await self.gets()
54         for order in orders:
55             if order.id == order_id:
```

KONTROL SONUCU

KISIM

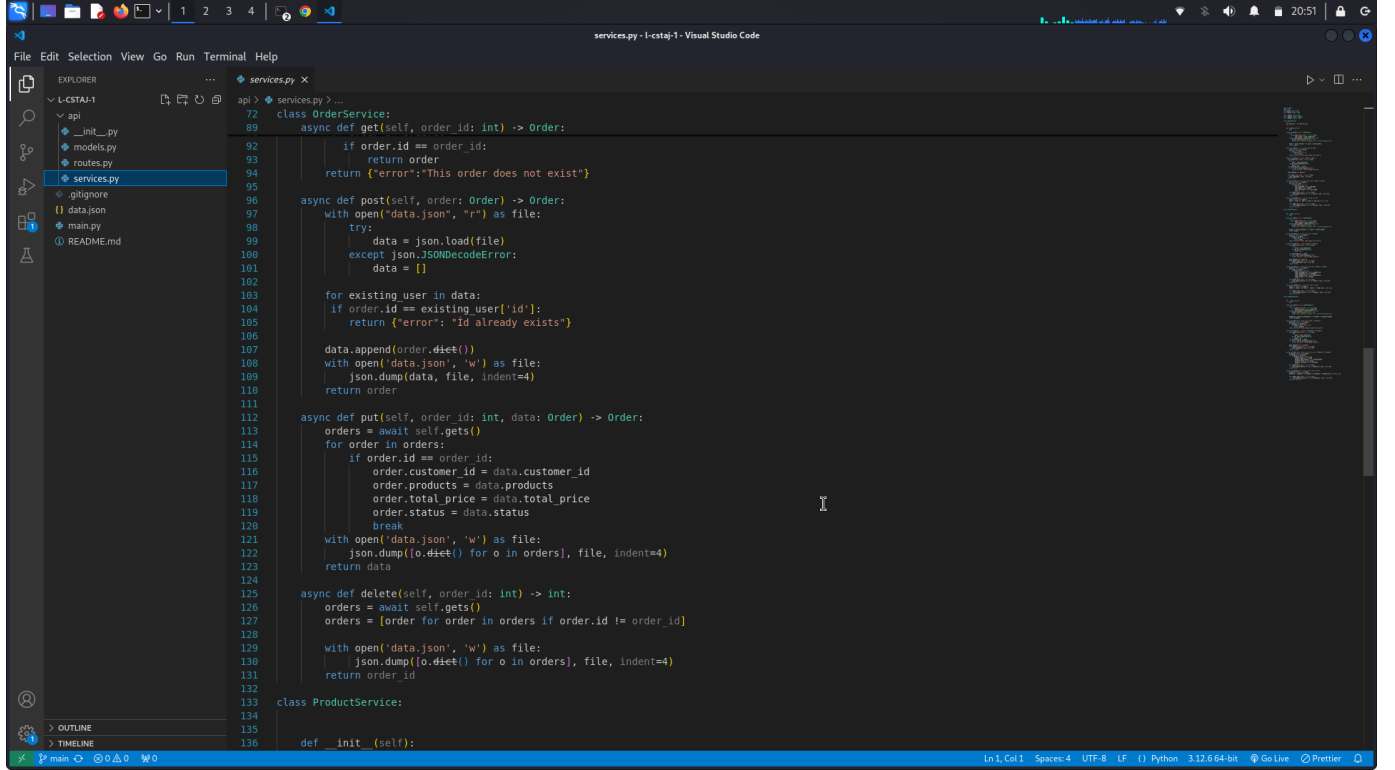
YAZILIM

YAPRAK NO:22

YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 19/08/2024

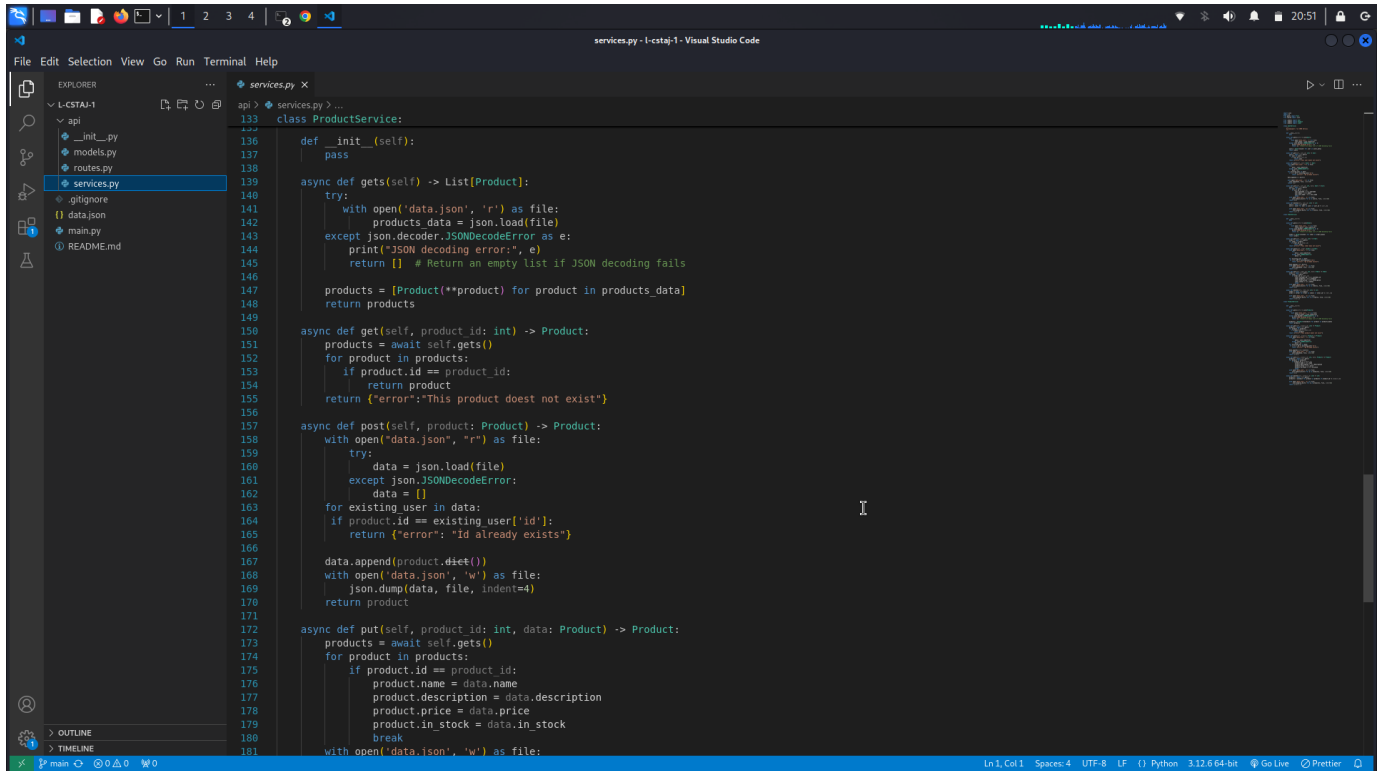


```
services.py - l-cstaj-1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER
  L-CSTAJ-1
  api
  __init__.py
  models.py
  routes.py
  services.py
  .gitignore
  data.json
  main.py
  README.md

api > services.py
121 class OrderService:
122     async def get(self, order_id: int) -> Order:
123         if order_id == order_id:
124             return order
125         return {"error": "This order does not exist"}
126
127     async def post(self, order: Order) -> Order:
128         with open("data.json", "r") as file:
129             try:
130                 data = json.load(file)
131             except json.JSONDecodeError:
132                 data = []
133
134         for existing_user in data:
135             if order.id == existing_user['id']:
136                 return {"error": "Id already exists"}
137
138         data.append(order.dict())
139         with open("data.json", "w") as file:
140             json.dump(data, file, indent=4)
141         return order
142
143     async def put(self, order_id: int, data: Order) -> Order:
144         orders = await self.gets()
145         for order in orders:
146             if order.id == order_id:
147                 order.customer_id = data.customer_id
148                 order.products = data.products
149                 order.total_price = data.total_price
150                 order.status = data.status
151                 break
152         with open("data.json", "w") as file:
153             json.dump([o.dict() for o in orders], file, indent=4)
154         return data
155
156     async def delete(self, order_id: int) -> int:
157         orders = await self.gets()
158         orders = [order for order in orders if order.id != order_id]
159
160         with open("data.json", "w") as file:
161             json.dump([o.dict() for o in orders], file, indent=4)
162         return order_id
163
164 class ProductService:
165     def __init__(self):
166         pass
```



```
services.py - l-cstaj-1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER
  L-CSTAJ-1
  api
  __init__.py
  models.py
  routes.py
  services.py
  .gitignore
  data.json
  main.py
  README.md

api > services.py
165 class ProductService:
166     def __init__(self):
167         pass
168
169     async def gets(self) -> List[Product]:
170         try:
171             with open("data.json", "r") as file:
172                 products_data = json.load(file)
173         except json.decoder.JSONDecodeError as e:
174             print("JSON decoding error:", e)
175             return [] # Return an empty list if JSON decoding fails
176         products = [Product(**product) for product in products_data]
177         return products
178
179     async def get(self, product_id: int) -> Product:
180         products = await self.gets()
181         for product in products:
182             if product.id == product_id:
183                 return product
184         return {"error": "This product does not exist"}
185
186     async def post(self, product: Product) -> Product:
187         with open("data.json", "r") as file:
188             try:
189                 data = json.load(file)
190             except json.JSONDecodeError:
191                 data = []
192
193         for existing_user in data:
194             if product.id == existing_user['id']:
195                 return {"error": "Id already exists"}
196
197         data.append(product.dict())
198         with open("data.json", "w") as file:
199             json.dump(data, file, indent=4)
200         return product
201
202     async def put(self, product_id: int, data: Product) -> Product:
203         products = await self.gets()
204         for product in products:
205             if product.id == product_id:
206                 product.name = data.name
207                 product.description = data.description
208                 product.price = data.price
209                 product.in_stock = data.in_stock
210                 break
211         with open("data.json", "w") as file:
```

KONTROL SONUCU

KISIM

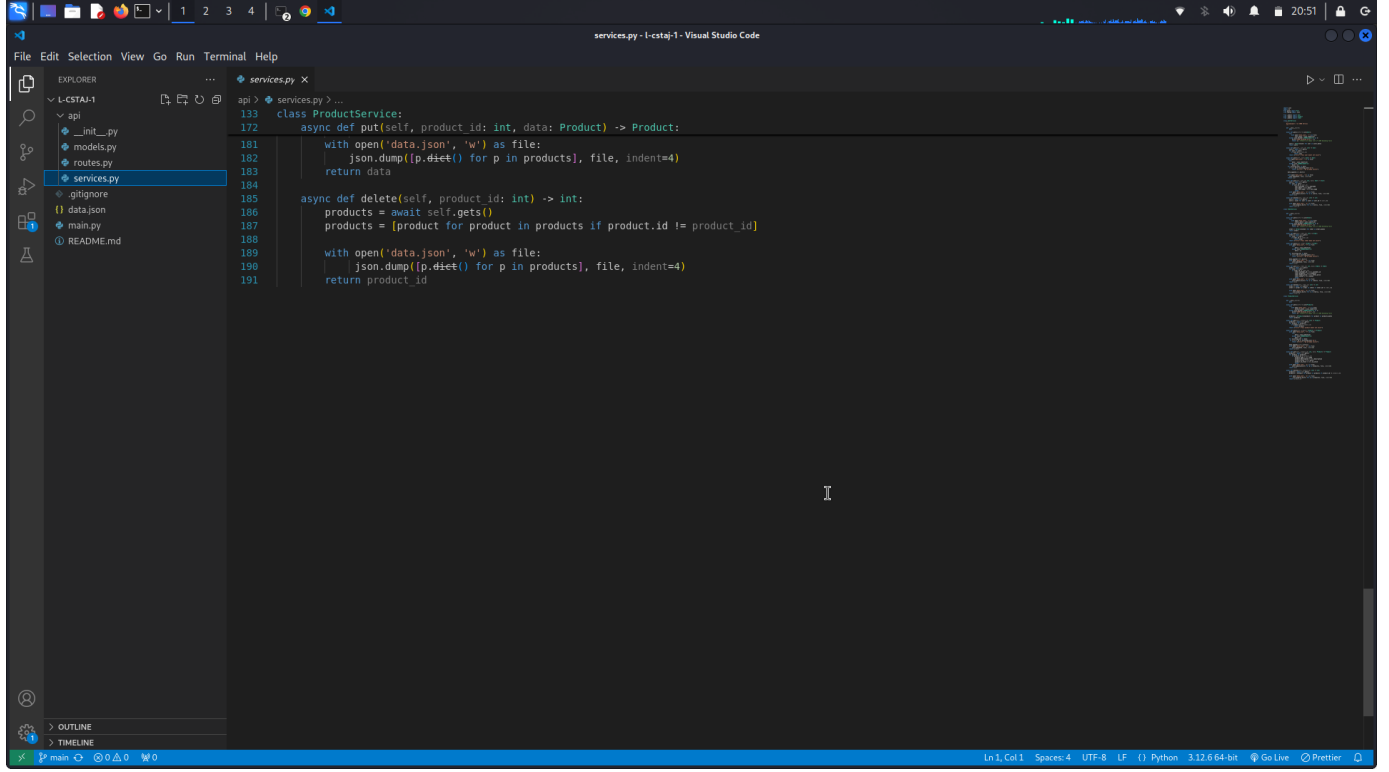
YAZILIM

YAPRAK NO:23

YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 20/08/2024



```
133 class ProductService:
134     async def put(self, product_id: int, data: Product) -> Product:
135         with open('data.json', 'w') as file:
136             json.dump([p.dict() for p in products], file, indent=4)
137         return data
138
139     async def delete(self, product_id: int) -> int:
140         products = await self.gets()
141         products = [product for product in products if product.id != product_id]
142         with open('data.json', 'w') as file:
143             json.dump([p.dict() for p in products], file, indent=4)
144         return product_id
```

Yukarda görmüş olduğunuz service yapısında alınan verilere göre ilgili veritabanıyla ilişki kurularak bu bilgiler istenilen yapımış olduğunuz web projesine uygun hale getirilerek kullanıcıya bir response döndürülür. Yani kısacası biz web dilinde bu işlemlere dil farketmeksizin CRUD işlemi deriz eğer açacak olursam CREATE , READ , UPDATE , DELETE işlemleri olarak 4'e ayrılır tabi bu bir web projesi için başlangıç diyebiliriz işin derinliklerine indiğiniz zaman bunlar sizin için sıradanlaşmaya başlayacaktır.

KONTROL SONUCU

KISIM

YAZILIM

YAPRAK NO: 24

YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 21/08/2024

Yukarda görmüş olduğunuz son olarak paylaştığım kod dizini python projemizim main kısmı burda yani yazmış olduğum kodların hepsini burda FastAPI kütüphanesini ve diğer tier design yapısındaki yapıları import ederek tüm bunları burda combine ederek kodumuzun çalışması için terminalimizi açıp şu kodu yazıyoruz .

--python main.py

bunu yazdıktan sonra local ağımızda çalışacak olan web api projemiz kullanıma hazır hale gelecektir .

--<http://0.0.0.0:8000/docs>

kullanacağınız tarayıcı açıp url kısmına bu linki yazmanız yeterlidir ben google kullanıyorum bu aşamadan sonra sonuçları sizinle paylaşacağım.

KONTROL SONUCU

KISIM

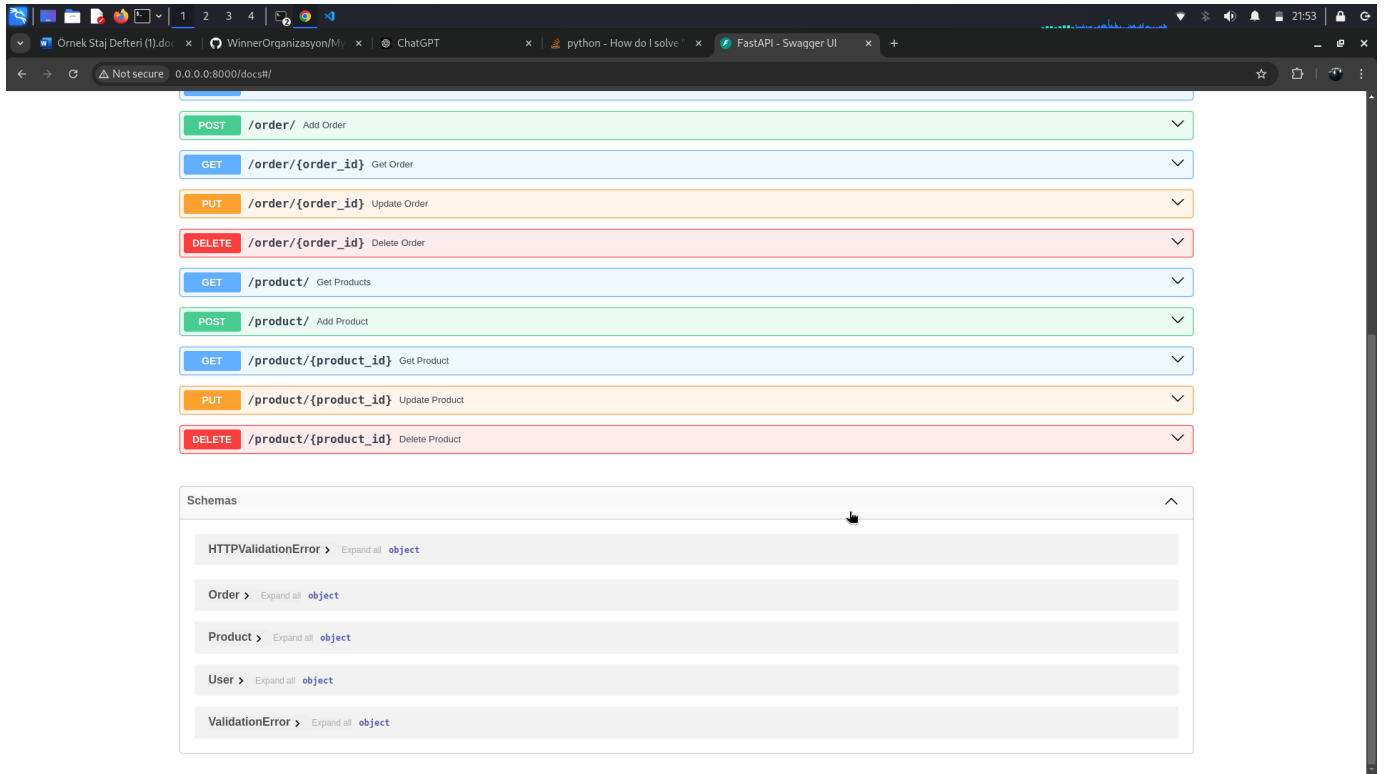
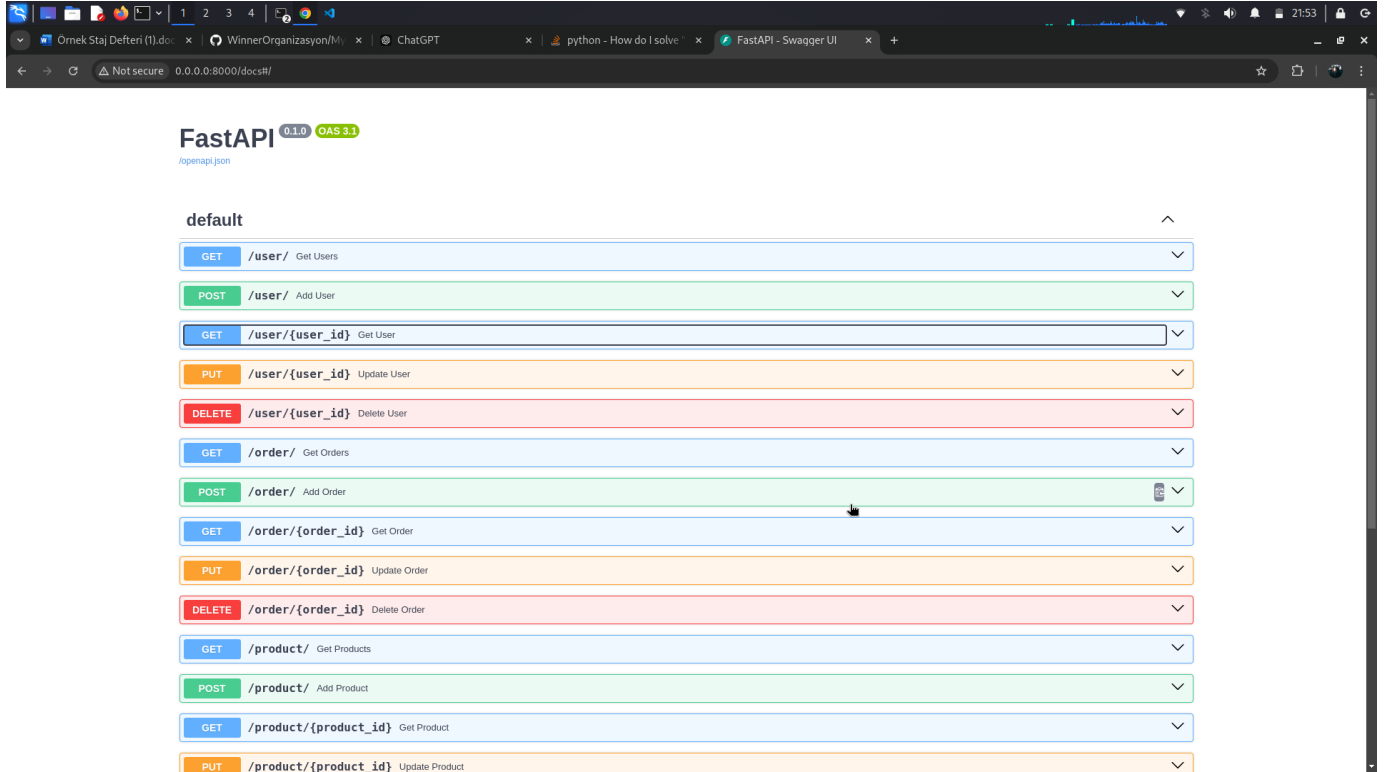
YAZILIM

YAPRAK NO:25

YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 22/08/2024



Bütün CRUD işlemlerini görmüş olduğunuz pencereden gerçekleştirebilirsiniz ben ilk öğrenme amaçlı yaptığım için bir database kullanmadım ancak projede görmüş olduğunuz üzere benim verilerim .json uzantılı konuma işleniyor.

KONTROL SONUCU

KISIM

YAZILIM

YAPRAK NO:26

YAPILAN İŞ

RESTFUL API(WEB API)

TARİH: 23/08/2024

Oct 19 20:41

GET /user/ Get Users

Parameters

No parameters

Servers

These operation-level options override the global server options.

/

Execute Clear

Responses

200

Response body

```
{
  "id": 1,
  "username": "burhan",
  "email": "asdasdeewqe",
  "full_name": "burhan cavdaro"
}
```

Response headers

content-length: 82

Oct 19 20:40

POST /user/ Add User

Parameters

No parameters

Request body required

application/json

```
{
  "id": 1,
  "username": "burhan",
  "email": "asdasdeewqe",
  "full_name": "burhan cavdaro"
}
```

Servers

These operation-level options override the global server options.

/

Execute Clear

Responses

200

Response body

```
{
  "id": 1,
  "username": "burhan",
  "email": "asdasdeewqe",
  "full_name": "burhan cavdaro"
}
```

Response headers

content-length: 82

Gördüğümüz üzere yukarda önce POST sonrasında ise GET methodunu çalıştırarak verileri almış bulunmaktayım.