



TURİZM ACENTESİ PROJESİ

Grup No: 9

Anıl Dursun İPEK - 031890131

Batuhan ARSLANDAŞ - 032190097

Yakup GÜRER - 031990081

Ramiz Can AKBIYIK - 032190062

Talha DAĞLAYAN - 032190070

DOÇ.DR. MURTAZA CİCİOĞLU

Proje Hakkında

- **Proje Konusu:** Bir turizm acentesinin tur türleri ve detayları, müşteriler, personel ve tur satışlarına dair detayları tutacak olan bir yazılım projesi
- **Proje Hedeflenen Çıktısı:** Kullanıcıların tatillerini ve gezi planlamalarını daha iyi ve kolay bir şekilde oluşturabilmeleri için bir web tabanlı yazılım projesi.

Kullanılan Teknolojiler

- **Kullanılan Yazılım Dilleri:** Proje içerisinde aktif olarak Javascript, C# ve Sql yazılım dilleri kullanılmaktadır.
- **Veri tabanı yönetimi:** İlişkisel veri tabanı türlerinden Mssql kullanılmaktadır. Programlama aşamasında ORM yapısı olarak Entity Framework kullanılmaktadır.
- **UI Tasarımı:** Javascript tabanlı Angular framework arayüz tasarımı için kullanılmaktadır.

Uygulama Arayüz Tasarımları

- Giriş Ekranı
- Dashboard
- Rezervasyonlar Ekranı
- Kullanıcılar Ekranı
- Kullanıcı Ekle Ekranı
- Tur Başlığı Ekle Ekranı
- Tur Ekle Ekranı
- Turlar Ekranı
- Tur Detayları Ekranı
- Harita Ekranı

Giriş Ekranı

Kullanıcı adı ve şifre girilerek veri tabanında bu bilgilere ait bir kullanıcı var mı diye kontrol edilir. Kullanıcı mevcutsa giriş yapar ve kullanıcının rolüne özel menü oluşmaktadır. Kullanıcı mevcut değilse hata mesajı döndürür.

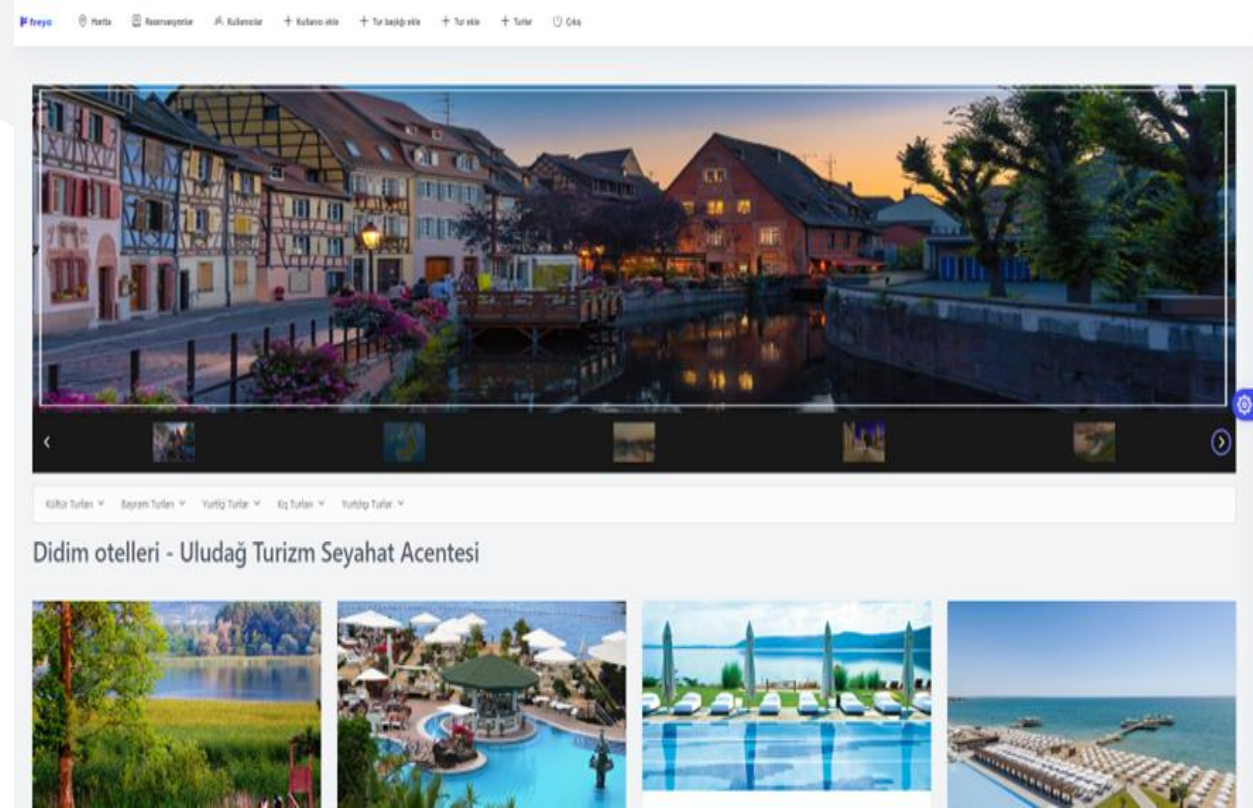


ULUDAG
Tourism

GİRİŞ YAP

Dashboard

Bu sayfa kullanıcıların turları görebilmesi için hazırlanmıştır. Seçilen alt başlığa göre turlar listelenmektedir. Listelenen turların üzerine tıklayarak tur detayları sayfasına geçiş yaparak tur hakkında daha detaylı bilgi alınmasını sağlamaktadır.



Rezervasyonlar Ekranı

Admin yetkisine sahip kullanıcı diğer kullanıcıların turlara yaptığı rezervasyonların bilgilerini bu sayfadan görebilmektedir.

freya

Harita

Rezervasyonlar

Kullanıcılar

Kullanıcı ekle

Tur başlığı ekle

Tur ekle

Turlar

Çıkış

+ Yeni

Search...

Tur adı	Ad Soyad	Telefon Numarası	Kalmacak gün sayısı	Yetişkin	Çocuk	Tarih	Fiyat
Ramada Resort By Wyndham Akbük	Ece DEMİR	(055) 368-7946	7	5	1	28/05/2023	867.00 TL
Venosa Beach Resort & Spa	Kerem YILMAZ	(055) 368-4346	2	2	0	01/08/2023	815.00 TL
Laur Hotels Experience&Elegance	Yakup GURER	(053) 247-4945	3	4	0	04/07/2023	1,168.00 TL
Ramada Resort By Wyndham Akbük	Hakan BAYRAK	(055) 596-4235	5	2	1	17/05/2023	867.00 TL
Ramada Resort By Wyndham Akbük	Harun ELMACI	(055) 695-8342	4	2	0	30/05/2023	867.00 TL
Ramada Resort By Wyndham Akbük	Ceyda OZKAN	(054) 295-9421	10	3	0	05/06/2023	867.00 TL
Ramada Resort By Wyndham Akbük	Ertan ASLAN	(055) 371-6964	3	3	0	21/05/2023	867.00 TL

1 of 1

<<

<

1

>

>>

Kullanıcılar Ekranı

Kaydedilen kullanıcıların bilgileri bu sayfada tutulmaktadır. Bu bilgiler butonlar yardımıyla düzenlenebilir veya silinebilir. "Yeni" butonuna tıklayarak sisteme yeni bir kullanıcı eklemek için kullanıcı ekleme sayfasına yönlendirir. Ek olarak kullanıcılar buton ile önceliklerine göre listelenebilmektedir.

Ad Soyad	Email	Telefon numarası	Tc	Kullanıcı Adı	Şifre	Rol	
Hakan BAYRAK	hakan.bayrak@gmail.com	05559642356	48612345064	hakanbyrk	bayrak12345	CUSTOMER	
Ertan ASLAN	ertan_aslan@gmail.com	05537169645	79493512853	ertanaslan55	aslanertn12345	CUSTOMER	
Ece DEMİR	ece_demir@hotmail.com	05536879464	49605312351	ecedmr	123456789ece	CUSTOMER	
Harun ELMACI	harun.elmaci@gmail.com	055669583429	41849531259	harunelmc68	123468harun	CUSTOMER	
Ceyda OZKAN	ceyda_ozkan123@hotmail.com	05429594219	41839506821	ceydaozkan34	ozkncyda123	CUSTOMER	
Ali GUNES	ali.gunes34@gmail.com	05424357766	46451930483	aligunes123	gunes34ali34	CUSTOMER	
Ethem HALICI	halici.ethem@gmail.com	05556843459	79468428459	ethemhalici17	hlocthem12345	PERSONEL	
Yakup GURER	yakup.gurer@gmail.com	05324749451	39285018403	yakup	gurer123	ADMIN	
Elif BAKIRCI	elifbakirci@outlook.com	0553951329	68492351294	elifbkrc13	elifbakir7954	VIP	
Erkan KAPICI	erkan.kapici11@outlook.com	05558471345	79284960313	erkankapici77	kpci123987	CUSTOMER	

Kullanıcı Ekle Ekranı

Kullanıcının bilgilerini girerek veri tabanına kaydedilmektedir. Bu bilgileri kullanarak kullanıcı sisteme giriş yapabilmektedir.

freya Harita Rezervasyonlar Kullanıcılar + Kullanıcı ekle + Tur başlığı ekle + Tur ekle + Turlar Çıkış

Kullanıcı ekle

Ad Soyad Telefon numarası

Email Tc kimlik

Kullanıcı Adı Şifre Rol

Kaydet

Tur Başlığı Ekle Ekranı

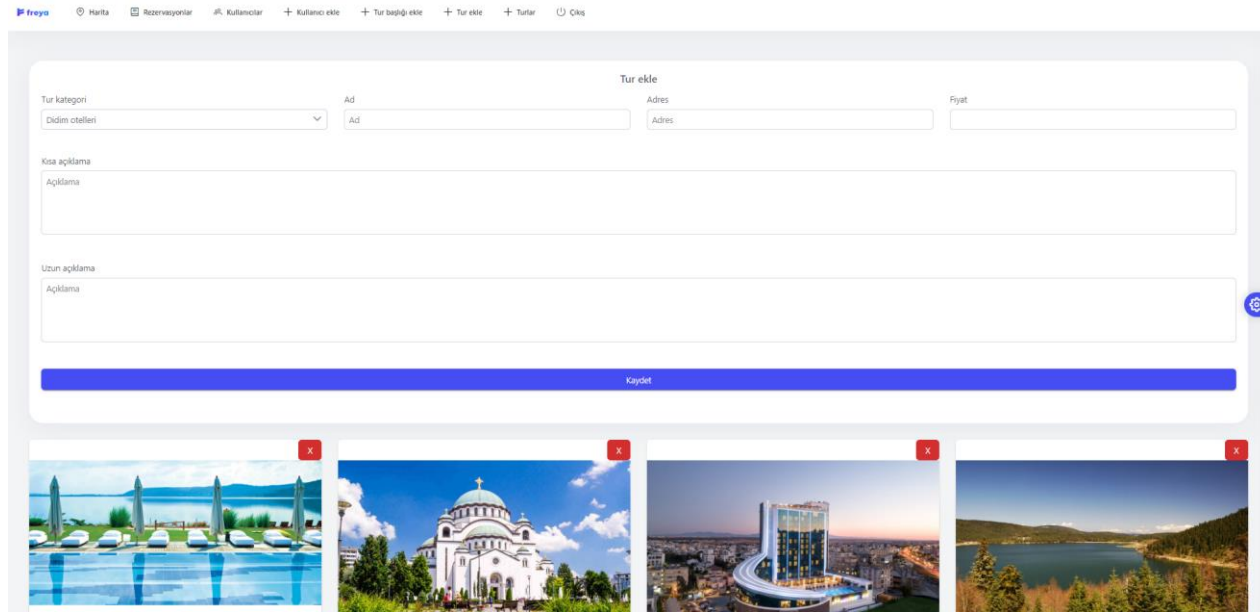
Bu sayfa üzerinde sisteme yeni tur başlıkları eklenebilmekte ve bu başlıklar dinamik bir şekilde dashboard'da gösterilebilmektedir. Var olan bir tura alt başlık da eklenebilmekte ve mevcut başlıklar listelenmektedir.

The screenshot displays the 'Tur ekle' (Add Tour) form and a table of existing tour titles. The form includes fields for 'Ad' (Name) and 'Tur kategori' (Tour category), with a 'Kaydet' (Save) button. The table below lists existing tour titles with their IDs and status indicators.

Ad	Kategori	
Nevşehir Turu	a8ee0ca7-d63a-46d0-938b-130957453328	✓ ⚠
Kültür Turları		✓ ⚠
Erciyes Turu	fd202a9e-41e6-4787-8222-e1fc4070cbb4	✓ ⚠
Bayram Turları		✓ ⚠
Kars Turu	fd202a9e-41e6-4787-8222-e1fc4070cbb4	✓ ⚠

Tur Ekle Ekranı

Seçilen tur başlığına ait yeni turlar eklenebilmektedir.
Seçilen tur başlığına ait turlar ek olarak sayfanın alt kısmında listelenmektedir.



The screenshot shows the 'Tur Ekle' (Add Tour) form in the Freya system. The form is divided into several sections:

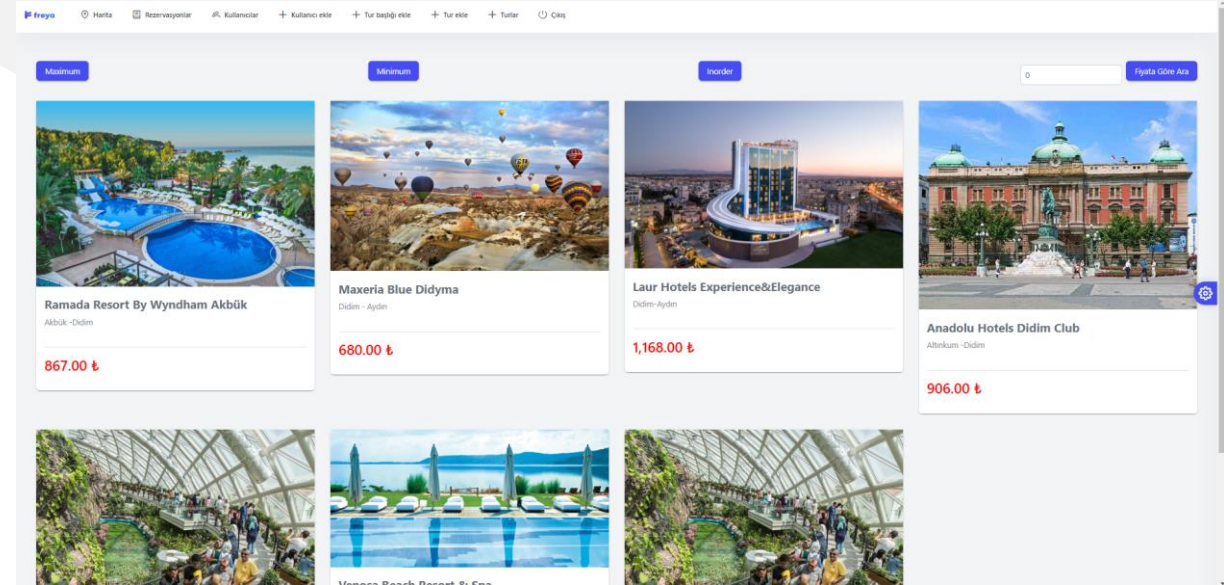
- Tur kategori:** A dropdown menu with 'Didim otelleri' selected.
- Ad:** A text input field with 'Ad' as a placeholder.
- Adres:** A text input field with 'Adres' as a placeholder.
- Fiyat:** A text input field.
- Kısa açıklama:** A text input field with 'Açıklama' as a placeholder.
- Uzun açıklama:** A text input field with 'Açıklama' as a placeholder.

Below the form is a blue 'Kaydet' (Save) button. At the bottom of the screen, there is a gallery of four images, each with a red 'X' icon in the top right corner:

- A resort pool with lounge chairs and umbrellas.
- A large white church with a dome.
- A modern building with a glass facade.
- A lake surrounded by trees and hills.

Turlar Ekranı

Bu sayfada sistemdeki mevcut tüm turlar görüntülenmektedir. Turların üzerinde bulunan butonlar ile minimum, maksimum fiyata ait turlar ekrana getirilebilmektedir. Ek olarak fiyata göre küçükten büyüğe göre sıralama yapılmakta ve fiyata göre arama işlemleri yapılabilmektedir.



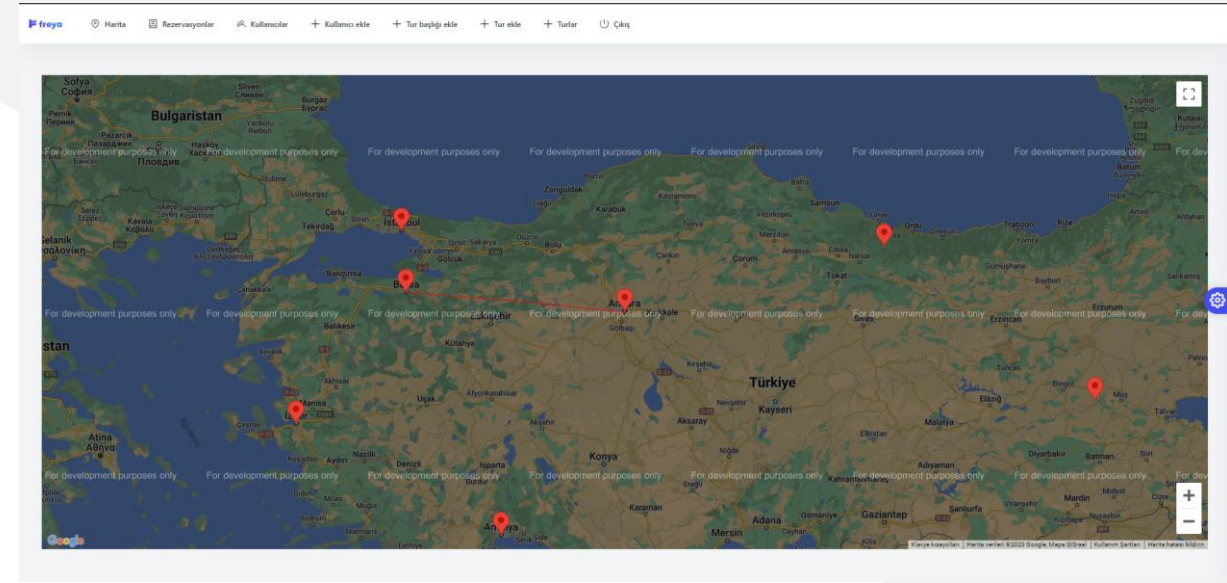
Tur Detayları Ekranı

Tur hakkında detaylı bilgiler ve fotoğraflar bulunmaktadır. Tur hakkında kişilerin yorum yapabilmesi için bir alan bulunmaktadır bu alanda kişiler tur ile ilgili yorum yaparak diğer kullanıcılara yardımcı olabilmektedir. Oturum açmış kullanıcı tura rezervasyon yapmak isterse gerekli alanları doldurarak bu sayfadan yapabilmektedir.

[illegible]

Harita Ekranı

Sistem üzerinde bulunan lokasyonlardaki belirtilen rotalar üzerinde işaretlemeler yapılabilir. Sistemin ilerleyen versiyonlarında harita geliştirilerek rota çizimler, en kısa yolun bulunması gibi sistemlerde iyileştirme yapılabilir.



- **Tree(Ağaç):** Ağaç yapısı, projede tur türlerini, tur fiyatları aramasını, tur fiyatları sıralamasını temsil etmek için kullanılacaktır. Bahsedilen türleri ve bu türler arasındaki ilişkileri düğüm yapısıyla temsil ederek hiyerarşik bir yapı oluşturulmaktadır. Bu sayede turlar daha iyi organize edilebilecek ve müşterilere daha kolay erişim sağlayacaktır.
- **Priority Queue(Öncelikli Kuyruk):** Müşteri yorumları, kullanıcı sıralaması gibi önceliklendirilmesi gereken işlemler için öncelikli kuyruk yapısı kullanılacaktır. Öncelik belirlenmiş yorumlar(örneğin VIP kullanıcılar) priority queue kullanılarak etkin bir şekilde yönetilecek ve ön sırada gösterilecektir. Böylece müşteri memnuniyeti ve önemli yorumlar önceliklendirilmiş olacaktır.
- **Graph:** Graf yapısı, turistik bölgeler ve bu bölgeler arasındaki ilişkileri temsil etmek için kullanılacaktır. Tur rotalarını yönetirken graf yapısını kullanarak müşterilere en iyi seyahat planı sunulması hedeflenmektedir.
- **Dairesel Bağlı Liste:** Dairesel bağlı dashboard ekranında görsellerin döngüsel bir şekilde belirli aralıklar ile geçiş yapması için kullanılmaktadır.
- **Hash:** Hash veri yapısı kullanıcı giriş ekranında veri tabanından alınan kullanıcı objelerinin girilen bilgiler ile eşleşmesinin kontrolü yapılırken kullanılmaktadır. $O(1)$ karmaşıklığa sahiptir.

Projenin Hedefleri

Bu yazılım projesi, turizm acentesinin iş süreçlerini optimize etmeyi ve müşterilere üstün bir deneyim sunmayı hedeflemektedir. Bu doğrultuda, tur türleri ve detaylarının daha etkin yönetimi, müşteri ilişkilerinin güçlendirilmesi, öncelikli işlemlerin etkin bir şekilde yönetilmesi ve turistik destinasyonlar arasındaki ilişkilerin analiz edilmesi gibi hedefler belirlenmiştir. Bu projenin başarılı bir şekilde hayata geçirilmesiyle turizm acentesi daha verimli çalışacak, müşteri memnuniyeti artacak ve iş performansını yükseltecektir.

Uygulama

```
def __init__(self, scene, modifier, type, use_x, use_y, use_z, use_clip, show_on_cage):
    self.scene = scene
    self.modifier = modifier
    self.type = type
    self.use_x = use_x
    self.use_y = use_y
    self.use_z = use_z
    self.use_clip = use_clip
    self.show_on_cage = show_on_cage

    self.modifier.add(type)
    self.modifier.modifiers[0].use_x = use_x
    self.modifier.modifiers[0].use_y = use_y
    self.modifier.modifiers[0].use_z = use_z
    self.modifier.modifiers[0].use_clip = use_clip
    self.modifier.modifiers[0].show_on_cage = show_on_cage

    self.modifier.apply(scene)

    self.modifier.name = 'MIRROR'

    self.modifier.parent = scene.modifiers

    self.modifier.hide()

    self.modifier.hide()
```

Dashboard Ekranında Dairesel Bağlı Liste Kullanımı

Dashboard sayfasında bulunan geçişli fotoğraf yapısında dairesel bağlı liste veri yapısı kullanılmıştır. Bu veri yapısının kullanım şekli şu şekildedir. Node classımız da url ve next adında 2 alanımız bulunmaktadır. Url alanı resmin bulunduğu dosyayı gösterir, next alanı ise bir sonraki resmin bulunduğu node u gösterir. CircularLinkedList classımız da sadece head alanı bulunur. Head alanı ilk Node u tutmaktadır. Dairesel bağlı listede son node ilk node ile bağlantılı olduğu için sistem çalıştığı süre boyunca ekrandaki fotoğraflar belirli bir süre ile dairesel bir şekilde değişmektedir.



```
export class CircularLinkedList{
  head:Node;

  public add(url:string){
    var node = new Node(url);

    if(this.head == null){
      this.head = node;
      node.next = this.head;
      return;
    }

    var temp = this.head;
    while(temp.next != this.head){
      temp = temp.next;
    }
    temp.next = node;
    node.next = this.head;
  }
}
```

```
setInterval(() => {
  this.currentNode = this.currentNode.next;
  this.currentImage = this.currentNode.url;
}, 2000)
}
```


Giriş İşlemi Yapan Kullanıcıların Kullanıcı Adına Göre Hash Veri Yapısı Kullanılarak Aranması

Sistemdeki kullanıcılar login işlemi yapıldığında oluşturduğumuz hash veri yapısı içerisinde tutulmaktadır. Hash veri yapısına eklediğimiz FindUser(string userName) metodu girilen userName'e göre hash içerisinde arama yapmakta ve sonuç bulunursa userName'e ait User objesini geri döndürmektedir. Eğer sonuç bulunamaz ise null değer geri döndürmektedir.



















ULUDAG
Tourism


```
1 reference
public User FindUser(string userName)
{
    int idx = this.HashCode(userName);
    Console.WriteLine(idx);
    int temp = idx;
    int count = 1;
    while (this.hashTable[temp] != null)
    {
        if (this.hashTable[temp] != null)
        {
            if (this.hashTable[temp].userName == userName)
            {
                break;
            }
        }
        temp = (int)((idx + Math.Pow(count, 2)) % this.N);
        count++;
    }
    if (this.hashTable[temp] != null)
    {
        return this.hashTable[temp].user;
    }
    return null;
}
```

```
var user = hash.FindUser(username);
if (user != null && user.Password == password)
{
    return Ok(_mapper.Map<UserDto>(user));
}
return Ok(null);
```

Kullanıcıların Role Göre Sıralanması

Sistem üzerindeki kullanıcılar öncelikli kuyruk yapısı kullanılarak tekrar listelenmektedir. Bu işlem kullanıcı tipine göre getir butonu ile çalışmaktadır. Sadece yetkili kullanıcılar bu işlemi gerçekleştirebilmektedir.

+ Kullanıcı tipine göre getir					Q Search...
İmarası	Tc	Kullanıcı Adı	Şifre	Rol	
451	39285018403	yakup	gurer123	ADMIN	 
459	79468428459	ethemhalici17	hlcethem12345	PERSONEL	 
465	74839583105	keremyilmaz46	keremylmz1881	PERSONEL	 
338	48592348135	mertbulut17	mrtblt9583	PERSONEL	 
329	68492351294	elifbkrc13	elifbakir7954	VIP	 
323	32946834519	haticebakan	hatice796bakan	VIP	 
132	49524964134	adnan.yazici	yaziciadnan3434	VIP	 
356	48612345064	hakanbyrk	bayrak12345	CUSTOMER	 

```
public void enqueue(int priority, Object data)
{
    Node node = new Node(priority, data);

    if (this.head == null) // If head is null, Add new node to head
    {
        this.head = node;
        return;
    }

    if (this.head.priority > node.priority) // If New node has a more priority queue, Prepend the new node
    {
        node.next = this.head;
        this.head = node;
        return;
    }

    Node temp = this.head;
    while (temp.next != null && temp.next.priority <= node.priority)
    {
        // Find the available place for new node
        temp = temp.next;
    }

    node.next = temp.next;
    temp.next = node;
}
```

```
foreach(var user in userDto)
{
    if(user.UserType == "ADMIN")
        priority = 1;
    else if(user.UserType == "PERSONEL")
        priority = 2;
    else if (user.UserType == "VIP")
        priority = 3;
    else
        priority = 4;

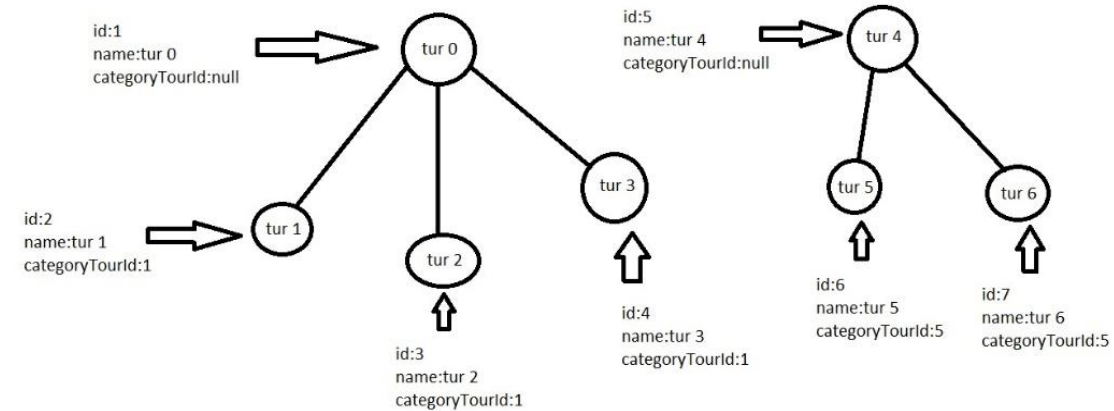
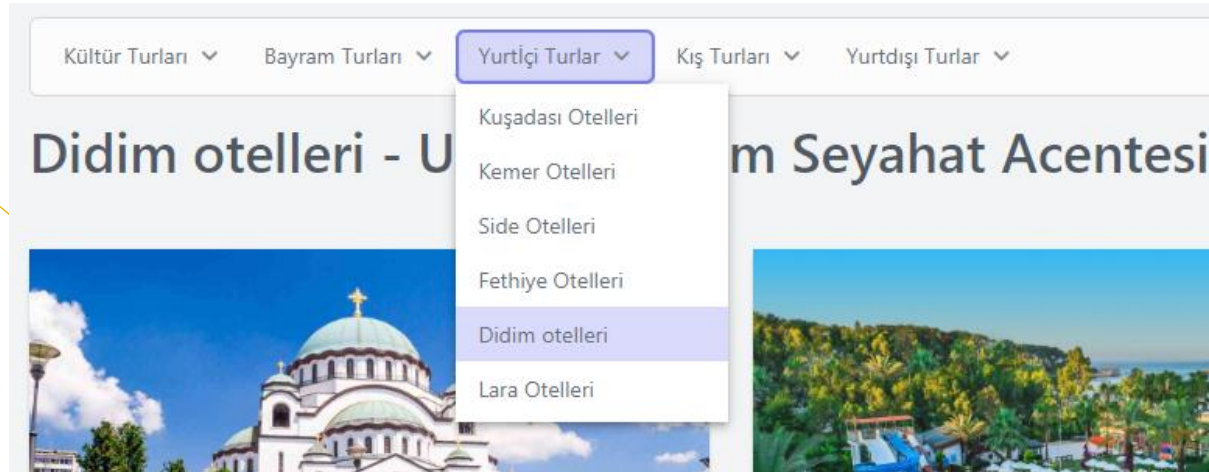
    priorityQueue.enqueue(priority, user);
}

List<UserDto> list = new List<UserDto>();
list = priorityQueue.transferToList(list);

return Ok(list);
```

Tur Başlıklarının Ağaç Yapısı Kullanılarak Kurgulanması

Turların categoryId'si null ise ağacın kökü olarak belirlenir turların categoryId'si başka bir turun idsini gösteriyorsa gösterilen turun bir dalı olduğunu temsil eder böylelikle menüler dinamik, sorunsuz ve basit bir yapıyla kullanılabilir.



Yorumların Kullanıcı Tipine Göre Öncelikli Olarak Sıralanması

Yorumların sıralanmasında öncelikli kuyruk yapısı kullanılmaktadır. Sistemdeki kullanıcı tipi VIP ve PERSONEL olan kullanıcılar öncelikli olarak üst sıralarda yer alırken CUSTOMER tipindeki kullanıcıların yorumları alt sıralarda yer almaktadır.



MB

Mert BULUT (PERSONEL)

Çok güzel bir deneyimdi



KY

Kerem YILMAZ
(PERSONEL)

Dünyanın en iyi oteli.



EB

Elif BAKIRCI

Uygun fiyatlı beklentiyi karşılayan bir apart otel.
Beklediğimizden daha güzel çıktı. 5 gün kaldık deniz de
yakın cidden çok hoş ve mütevazı bir otel fiyatını
karşılıyor.



AG

Ali GÜNEŞ

Harika bir otel.



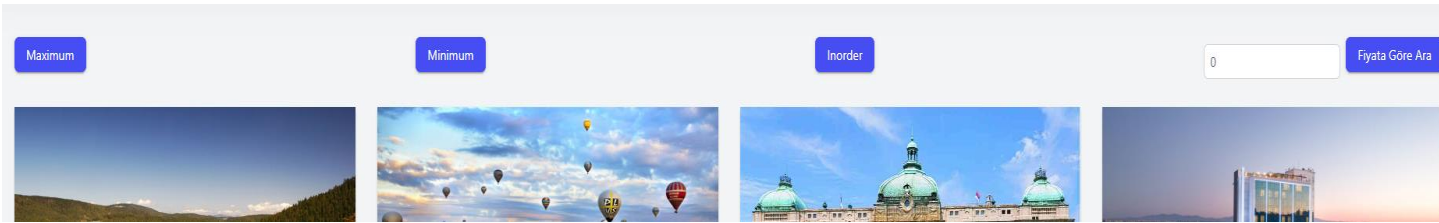
AP

Aysel POTUK

Herkese tavsiye ederim...

Turların İkili Ağaç İle Filtrelenmesi

Tüm turların listelenerek ikili ağaç metodları ile minimum, maksimum ve inorder sıralama getirilmektedir. Bunlara ek olarak fiyat bilgisi kullanıcıdan alınarak ikili arama işlemi gerçekleştirilmekte ve kullanıcıya aranan fiyatta bir tur getirilmektedir.



```
3 references
public Object search(decimal key, BinaryNode node)
{
    if (node == null)
        return null;

    if (node.key == key)
        return node.data;

    else
    {
        if (node.key > key)
            return search(key, node.left);
        else
            return search(key, node.right);
    }
}
```

```
5 references
public void inorder(List<TourItemDto> list, BinaryNode node)
{
    //inorder -> LNR
    if (node == null)
        return;

    if (node.left != null)
        inorder(list, node.left);

    //Console.WriteLine(node.key + " | ");
    list.Add((TourItemDto)node.data);

    if (node.right != null)
        inorder(list, node.right);
}
```

```
1 reference
public Object GetMin()
{
    if (this.root == null)
        return null;

    BinaryNode temp = this.root;
    while (temp.left != null)
        temp = temp.left;

    return temp.data;
}
```

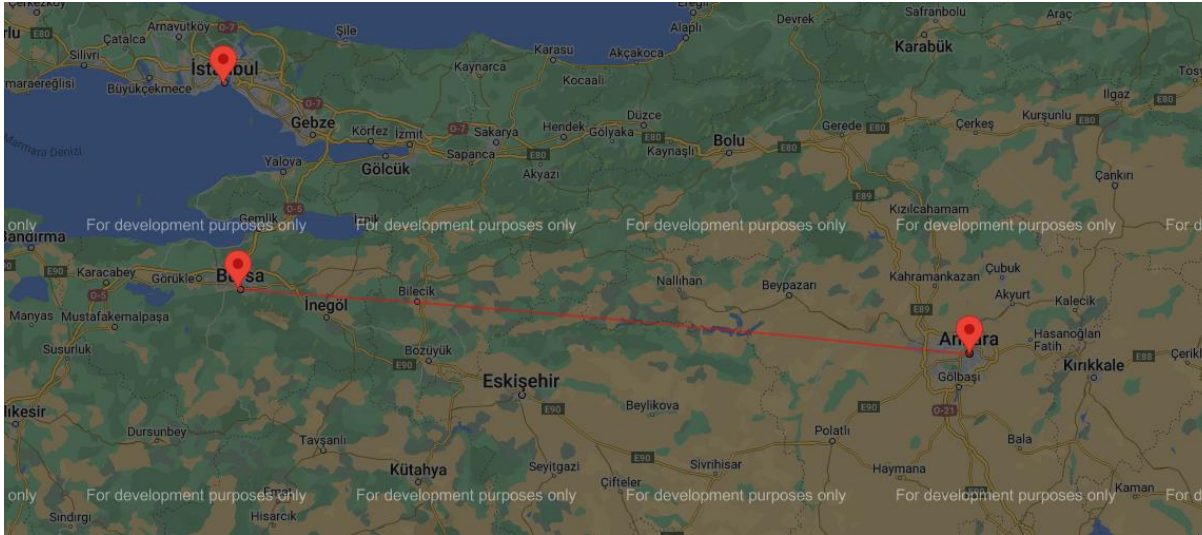
```
1 reference
public Object GetMax()
{
    if (this.root == null)
        return null;

    BinaryNode temp = this.root;
    while (temp.right != null)
        temp = temp.right;

    return temp.data;
}
```

Harita Üzerinde Graf Yapısının Kullanılması

Graf yapısı ile haritada belirtilen şehirler arasındaki minimum uzaklığın bulunması hedeflenmektedir. İlk aşamada şehirler statik olarak girilmektedir. Projenin ilerleyen aşamalarında dinamik yapılar ile çalışarak yeni graf yapıları oluşturulabilecektir.



```
1 reference
...public Graph(int dim)
...{
...    int i, j;
...    this.dim = dim;
...    edge = new int[dim, dim];
...    for (i = 0; i < dim; i++)
...    {
...        for (j = 0; j < dim; j++)
...        {
...            edge[i, j] = 0;
...        }
...    }
...}

7 references
...public void AddEdge(int start, int end, int weight)
...{
...    edge[start, end] = weight;
...}

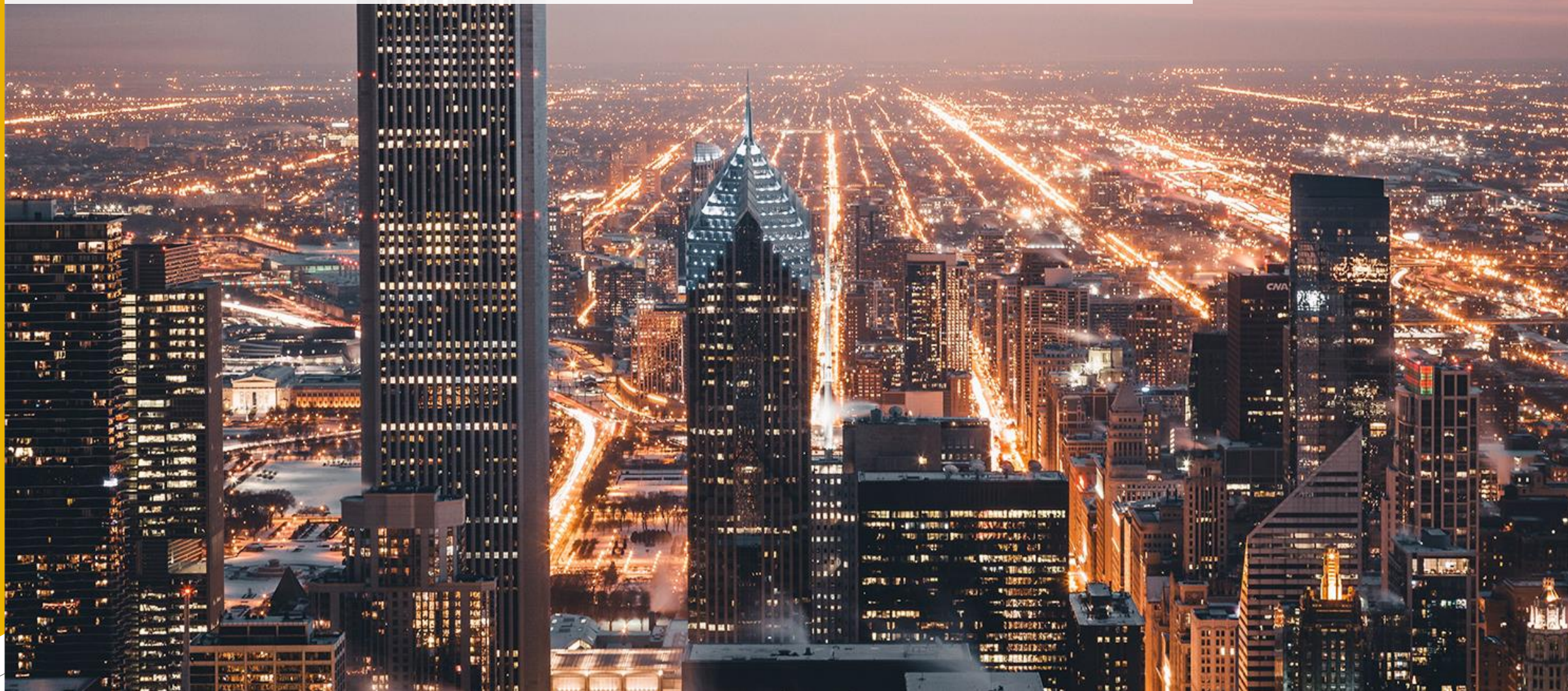
1 reference
...public City FindMin()
...{
...    int min = edge[0, 1];
...    City city = new City();

...    City[] cities = { new City(38.42, 27.14, "İzmir", 589), new City(4

...    for (int j = 1; j < 6; j++)
...    {
...        if (min > edge[0, j + 1])
...        {
...            min = edge[0, j + 1];
...            city = cities[j];
...        }
...    }

...    return city;
}
```


Algoritma Analizi



Ağaç Yapıları (Tree)

- **Ağacın Oluşturulması:** Ağaç yapısını oluşturmak için bir dizi veya listeden elemanlar eklememiz gerekebilir. Her elemanı eklemek için ağacın kökünden başlayarak bir yol izleriz. En kötü durumda, bir elemanı eklemek $O(n)$ zaman karmaşıklığına sahip olabilir, çünkü en kötü durumda ağaç tamamen dengesiz olabilir ve elemanları eklemek için tüm düğümleri ziyaret etmemiz gerekebilir. Bu projede ağaca eklenen elemanlar dashboard sayfasında tur başlıkları ve turların listelenmesi ile ilgilidir. Karmaşıklık düzeyi $O(n)$ 'dir.
- **Ağacın Gezinilmesi:** Ağaçtaki düğümleri gezinmek için farklı algoritmalar kullanılabilir. Üç yaygın gezinme yöntemi, Preorder, Inorder ve Postorder gezinme yöntemleridir. Bu gezinme yöntemlerinin her biri ağaçtaki düğümleri tamamen dolaşır ve düğümlerin değerlerini işler. Bu gezinme işlemleri $O(n)$ karmaşıklığına sahiptir çünkü en kötü durumda her düğümü ziyaret etmemiz gerekebilir. Bu projede ağacın gezinilmesi için InOrder yaklaşımı kullanılmıştır ve turların listelenmesi ile ilgilidir. Karmaşıklık düzeyi $O(n)$ 'dir.
- **Bir Elemanın Aranması:** Ağaç yapısında bir elemanı aramak için ağacın düğümlerini karşılaştırarak bir yol izleriz. Eğer ağaç dengeli bir şekilde yapılandırılmışsa, arama işlemi $O(\log n)$ karmaşıklığına sahip olabilir. Ancak en kötü durumda ağaç tamamen dengesiz olabilir ve aranan elemanı bulmak için tüm düğümleri ziyaret etmemiz gerekebilir. Bu durumda arama işlemi $O(n)$ karmaşıklığına sahip olur. Bu projede ağaçta bir elemanın aranması istenilen fiyatta turun bulunabilmesi ile ilgilidir. Karmaşıklık düzeyi $O(\log n)$ 'dir.

Öncelikli Kuyruk (Priority Queue)

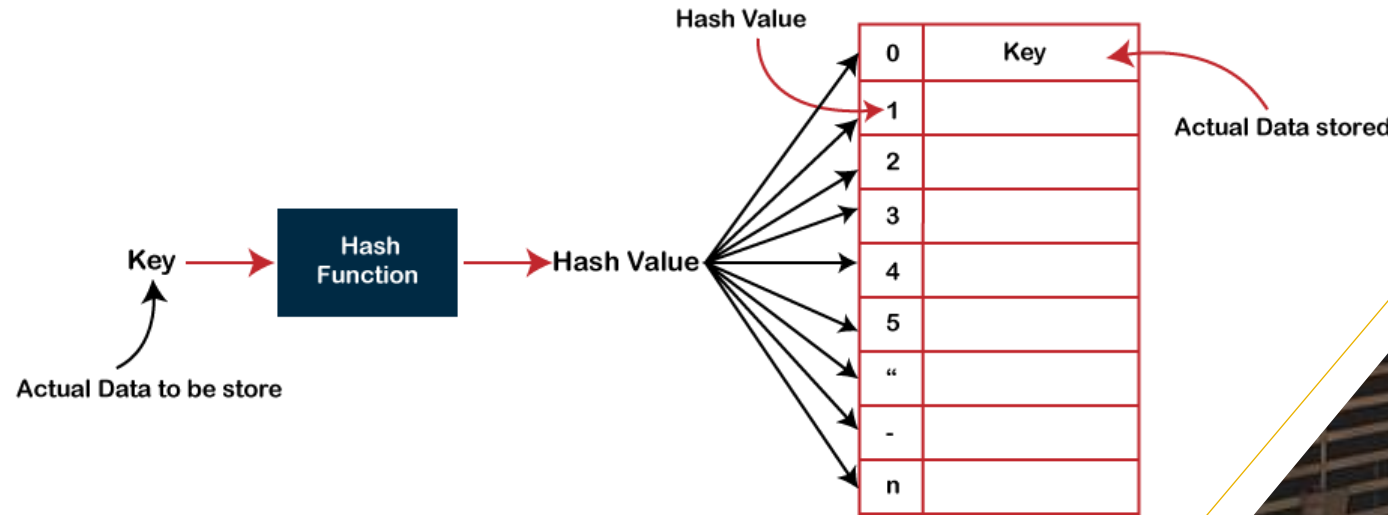
- **Eleman Ekleme:** Kuyruğa eleman ekleme $O(n)$ karmaşıklığa sahiptir. Elemanlar öncelik değerine göre sisteme sıra ile eklenmektedir. Bu projede kuyruğa eleman ekleme müşteri yorumlarının önceliklendirilmesi ve kullanıcıların sistemde sıralanması ile ilgilidir. Karmaşıklık düzeyi $O(n)$ 'dir.
- **Eleman Listeleme:** Kuyruk ekleme işleminden sonra kullanıcılar sıra ile listelendiğinde öncelikli olarak listelenmiş olmaktadır. Önceliği eşit olan verileri geliş sırasına göre kuyruk mantığı ile listelemektedir.
- **En Yüksek Önceliğe Sahip Elemanı Alma:** Öncelikli kuyruktan en yüksek önceliğe sahip elemanı almak, genellikle $O(1)$ karmaşıklığına sahip olan bir operasyondur. Öncelikli kuyruğun en yüksek önceliğe sahip elemanı, kuyruğun başında veya kök düğümünde bulunur. Bu nedenle, bu işlem zaman açısından hızlı bir şekilde gerçekleştirilebilir. Bu projede en yüksek önceliğe sahip elemanı alma VIP kullanıcıların müşteri yorumlarının belirlenmesi ve ön sıraya çıkartılması ile ilgilidir. Karmaşıklık düzeyi $O(1)$ 'dir.

Graf (Graph)

- Graf düğümler ve bu düğümler arasındaki bağlantıları temsil eden matematiksel bir yapıdır. Minimumu bulma problemleri, genellikle ağırlıklı bir graf üzerinde gerçekleştirilir. Sistemimizde ağırlıklı graflar kullanılarak belirli bir şehirden çevredeki şehirlere bakarak en yakın mesafede olan şehri bulmak üzerinde kurgulanmaktadır. Karmaşıklığı $O(n)$ 'dir.

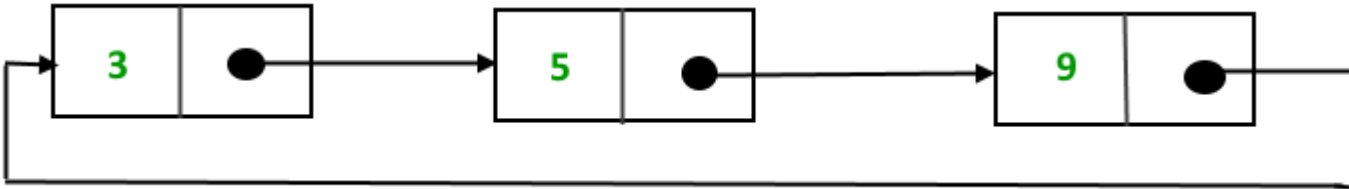
Hash

- Hash, verileri benzersiz bir temsilciye dönüştüren bir algoritma veya işlem şeklidir. Bu temsilciye hash değeri denir. Hash fonksiyonları, girdi verisini sabit bir boyutta bir temsilciye dönüştürür. Hash fonksiyonları, veri bütünlüğünü kontrol etmek, veri doğrulaması yapmak ve veri öğelerini hızlı bir şekilde erişmek gibi birçok alanda kullanılır.



Dairesel Bağlı Liste

- Dairesel bağlı liste, verilerin birbirine bağlı düğümlerle temsil edildiği bir veri yapısıdır. Her düğüm, veriyi depolayan bir eleman ve bir sonraki düğümün referansını içeren bir bağlantıya sahiptir. Dairesel bağlı listede, son düğümün bağlantısı başlangıç düğümüne (ilk düğüm) yönlendirilir, böylece bir döngü oluşur. Bu sayede liste dairese bir yapı kazanır.



Teşekkürler