VERİ YAPILARI VE ALGORİTMALAR
PROJESİ

MARMARA ÜNİVERSİTESİ TEKNOLOJİ FAKÜLTESİ

MÜŞTERİ KUYRUĞU YÖNETİMİ UYGULAMASI

HAZIRLAYANLAR

Şeyma Nur ALTAN – 170423508

İpek AKPINAR – 170423822

1. Giriş:

- Projenin Tanımı: Uygulama bir bankada işlem yaptıracak her müşteri için kayıt sırasına göre bir sıra numarası verir. Veznede son işlem bittiğinde banka görevlisi butona basar ve henüz çağırılmamış müşterilerden en öndekinin sıra numarası ekranda görünür. Yeni bir müşteri ise kuyruğa girmek istediğinde müşterinin bilgileri alınır ve müşteriye bir sıra numarası verilir.

- Projenin Amacı: Bankada karmaşıklığı önlemek için müşterilerin her birine bir sıra numarası vererek düzenli bir sıra oluşturmak ve ilk kayıt açandan son kayıt açana doğru bu müşterilere hizmet sunmak. Yeni müşterinin geliş sebebini ve bilgirlerini de alarak gişedeki işlem süresini azaltmak.

- Proje: Proje kod dosyası içeriğinde altı class oluşturuldu. Bu class'lardan biri ana uygulama için oluşturulmuş App.java dosyası bir diğeri ise uygulama kontrolü için oluşturulmuş AppController.java dosyasıdır.

Bankadaki kuyruk sistemi için Linked List kullanarak bir Circular Queue oluşturacak şekilde Queue.java class oluşturuldu. Bu dosyada kuyruğa ekleme yapacak bir enqueue(), kuyruktan silme işlevini gerçekleştiren deque(), kuyruğun ilk elemanını döndüren peek() ve kuyruk boş mu değil mi kontrol eden isEmpty() fonksiyonları yazıldı. Ayrıca dosyada kuyruğun ilk node'unu tutan bir front, son node'unu tutan bir rear ve kaç elemana sahip olduğunu gösteren size değişkenleri tanımlandı.

Yeni kayıt ekleme işleminin yapılacağı ekran için ise NewEntryController.java oluşturuldu. Bu class'ın içinde girilen bilgilerin kaydedilmesi işlemi için handleSaveButton() fonksiyonu oluşturuldu. Bu fonksiyon ile hem kullanıcıdan ad, soyad ve geliş sebebi bilgileri alındı; hem de alınan bu bilgiler veri tabanına kaydedildi. Veri tabanı uygulaması olarak MySQL kullanıldı. Ve veri tabanını uygulamamıza bağlamak için DatabaseConnection.java class'ı oluşturuldu.

2. Kullanım

Müşteri Girişi: Kuyruğa yeni müşteri eklemek için "Yeni Kayıt" butonunu kullanın.

Kuyruk İşlemleri: Müşteri numaralarını görüntülemek ve sıradaki müşteriyi çağırmak için "Next" butonlarını kullanın.

Müşteri Bilgileri Kaydetme: Yeni müşteri bilgilerini kaydetmek için giriş formunu kullanın ve "Tamam" butonuna basın.

3. Yazılım Tasarımı

a. App Sınıfı:

App sınıfı, JavaFX uygulamasının başlangıç noktasını temsil eder. Bu sınıf, uygulamanın başlatılmasını sağlar ve kullanıcı arayüzünü yükler.

i. Özellikler:

- **primaryStage**: JavaFX sahnesinin ana penceresini temsil eden bir Stage referansı.

ii. Metotlar:

- **start(Stage stage):** Bu metot, uygulamanın başlatılmasını sağlar. start metodu, bir Stage örneği alır ve ana sahneyi yükler. app-view.fxml dosyasından bir FXML yükleyicisi oluşturur ve JavaFX sahnesini yükler. Son olarak, sahneye başlık ekler ve sahneyi gösterir.
- **getPrimaryStage():** Bu metot, diğer sınıfların primaryStage örneğine erişmesini sağlar. primaryStage örneğini döndürür.
- **main(String[] args):** Bu metot, uygulamanın ana başlangıç noktasıdır. launch metodu, uygulamayı başlatır ve start metotunu çağırır.

b. AppController Sınıfı:

AppController sınıfı, JavaFX uygulamasının kullanıcı arayüzündeki olayları yönetir ve kontrol eder. Bu sınıf, kullanıcının etkileşimde bulunduğu arayüz öğelerine (butonlar, etiketler vb.) tepki verir ve uygun işlevleri çağırır.

i. Özellikler:

- **seqA, seqB, seqC, seqD:** Sıradaki müşterilerin numaralarını gösteren Label öğeleri.
- **infoText:** Bilgi metnini gösteren Label öğesi.
- **num_of_waiting:** Sırada bekleyen kişi sayısını tutan bir tam sayı.

ii. Metotlar:

- **onCLick_btn1(), onCLick_btn2(), onCLick_btn3(), onCLick_btn4():** Bu metotlar, sıradaki müşteriyi çağırmak için kullanılan butonların tıklanma olaylarına tepki verir. Her bir metot, ilgili butona tıklandığında çalışır. Eğer sırada bekleyen müşteri yoksa, bilgi metnini günceller; aksi

halde, bir sonraki müşteriyi sıradan çıkarır ve ilgili etikete numarasını yazar.
- **newCustomer_btn():** Bu metot, yeni bir müşterinin kuyruğa eklenmesini sağlayan butona tıklanma olayına tepki verir. Kuyruğa yeni bir müşteri ekler ve sırada bekleyen kişi sayısını artırır.
- **initialize(URL url, ResourceBundle resourceBundle):** Bu metot, controller sınıfı başlatıldığında otomatik olarak çağrılır. Başlangıçta, sistem bir kuyruk oluşturur, kuyruktaki müşteri sayısını hesaplar ve ilgili etikete bu sayıyı yazar.

c. DatabaseConnection Sınıfı:

DatabaseConnection sınıfı, veritabanı bağlantısını yöneten bir yardımcı sınıftır. Bu sınıf, MySQL veritabanına bağlanmayı sağlar ve bağlantıyı döndürür.

i. Sabitler:
- **DATABASE_NAME**: Veritabanı adı.
- **DATABASE_USER**: Veritabanı kullanıcı adı.
- **DATABASE_PASSWORD:** Veritabanı kullanıcı parolası.
- **URL:** Veritabanına bağlantı URL'si.

ii. Özellikler:
- **databaseLink:** Veritabanı bağlantısını tutan bir Connection örneği.

iii. Metotlar:
- **getConnection():** Bu metot, MySQL veritabanına bir bağlantı oluşturur. Öncelikle MySQL sürücüsünü yükler ve sonra belirtilen URL, kullanıcı adı ve parola ile bir bağlantı kurar. Bağlantıyı döndürür veya bir hata durumunda istisna fırlatır.

d. NewEntryController Sınıfı:

NewEntryController sınıfı, yeni müşteri girişi formunu yöneten bir controller sınıfıdır. Bu sınıf, kullanıcı girdilerini alır, doğrular, veritabanına kaydeder ve bir uyarı mesajı gösterir.

i. Özellikler:
- **adtext, soyadtext:** Kullanıcının adını ve soyadını girmesi için metin alanları.
- **gelisSebebi:** Kullanıcının geliş sebebini seçmesi için bir seçim kutusu.
- **OKbutton:** Girişi onaylamak için bir düğme.
- **WarningLabel:** Kullanıcıya uyarı mesajlarını göstermek için bir etiket.
- **databaseConnection:** Veritabanı bağlantısını yöneten DatabaseConnection örneği.

ii. **Metotlar:**
- **initialize():** Bu metot, controller sınıfı başlatıldığında otomatik olarak çağrılır. databaseConnection örneğini oluşturur ve gelisSebebi seçim kutusuna seçenekler ekler.
- **handleSaveButton():** Bu metot, "Kaydet" düğmesine tıklanma olayına tepki verir. Kullanıcı girdilerini alır, doğrular ve veritabanına kaydeder. Eğer gerekli alanlar boşsa, bir uyarı mesajı gösterir. Kayıt işlemi tamamlandıktan sonra pencereyi kapatır.

e. Queue Sınıfı:

Queue sınıfı, dairesel bir kuyruk veri yapısını temsil eder. Bu sınıf, kuyruğa eleman eklemek, kuyruğun başından eleman çıkarmak, kuyruğun başındaki elemanı görmek ve kuyruğun boyutunu almak gibi işlevleri gerçekleştirir.

i. **Sabitler:**
- **CAPACITY:** Kuyruğun kapasitesini temsil eden bir sabit.

ii. **İç Sınıf - Node:**
- **data:** Düğümün verisini temsil eden bir tam sayı.
- **node:** Düğümün bir sonraki düğümüne işaret eden bir referans.

iii. **Özellikler:**
- **front, rear:** Kuyruğun başını ve sonunu işaret eden düğüm referansları.
- **size:** Kuyruktaki eleman sayısını tutan bir tam sayı.
- **capacity:** Kuyruğun kapasitesini temsil eden bir tam sayı.

iv. **Metotlar:**
- **enqueue(Integer seq):** Kuyruğun sonuna yeni bir eleman ekler.
- **dequeue():** Kuyruğun başındaki elemanı kaldırır ve değerini döndürür.
- **peek():** Kuyruğun başındaki elemanın değerini döndürür ancak kuyruktan kaldırmaz.
- **size():** Kuyruktaki eleman sayısını hesaplar.
- **isEmpty():** Kuyruğun boş olup olmadığını kontrol eder.

f. QueueController Sınıfı:

QueueController sınıfı, kuyruk işlemlerini yöneten bir controller sınıfıdır. Bu sınıf, bir kuyruk oluşturur, hazır bir kuyruk oluşturur ve yeni müşteri girişi için bir işlev sağlar.

i. **Özellikler:**
- **queue**: Kuyruğu temsil eden bir Queue örneği.

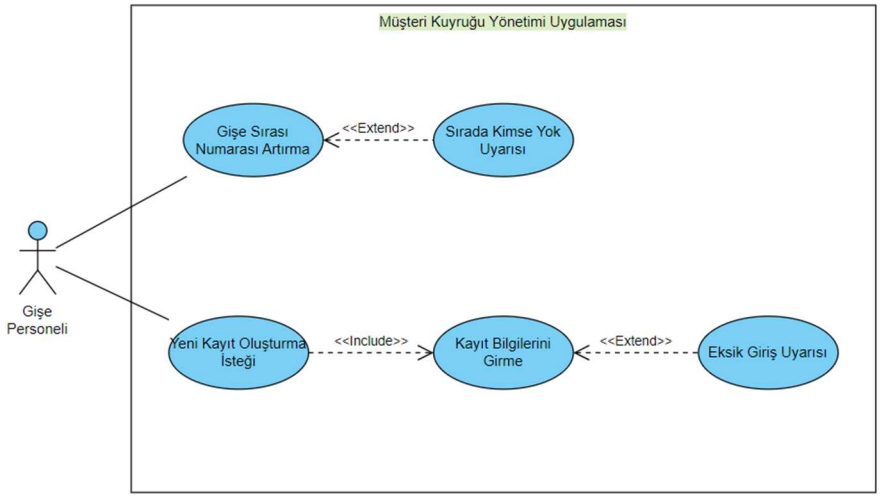
ii. **Metotlar:**
- **createQueue():** Oluşturulan hazır bir kuyruk oluşturur ve başlangıç değerlerini ekler.

- **newCustomer():** Yeni bir müşteri girişi için bir işlev sağlar. Kuyruğa yeni bir müşteri ekler ve ardından yeni müşteri girişi formunu açar.

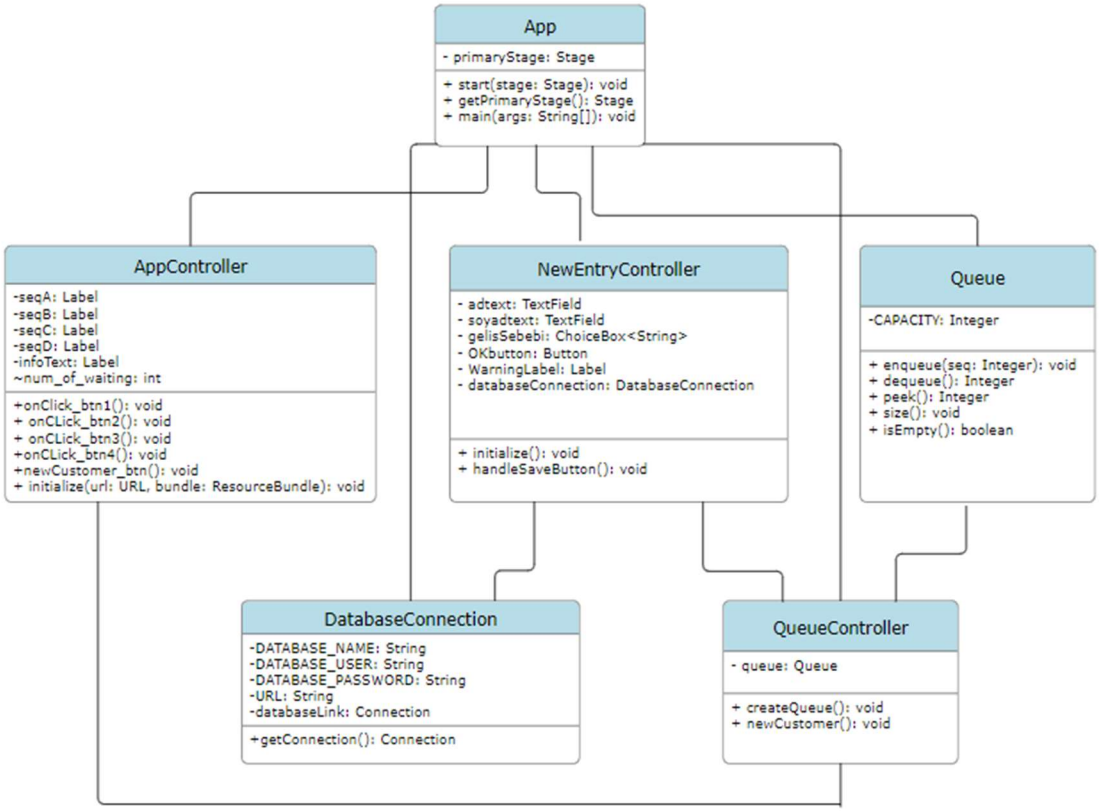
g. MySQL:

```
CREATE TABLE `users` (
 `user_no` int NOT NULL AUTO_INCREMENT,
 `user_name` varchar(255) DEFAULT NULL,
 `user_surname` varchar(255) DEFAULT NULL,
 `reason` varchar(255) DEFAULT NULL,
 PRIMARY KEY (`user_no`)
) ENGINE=InnoDB AUTO_INCREMENT=14 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

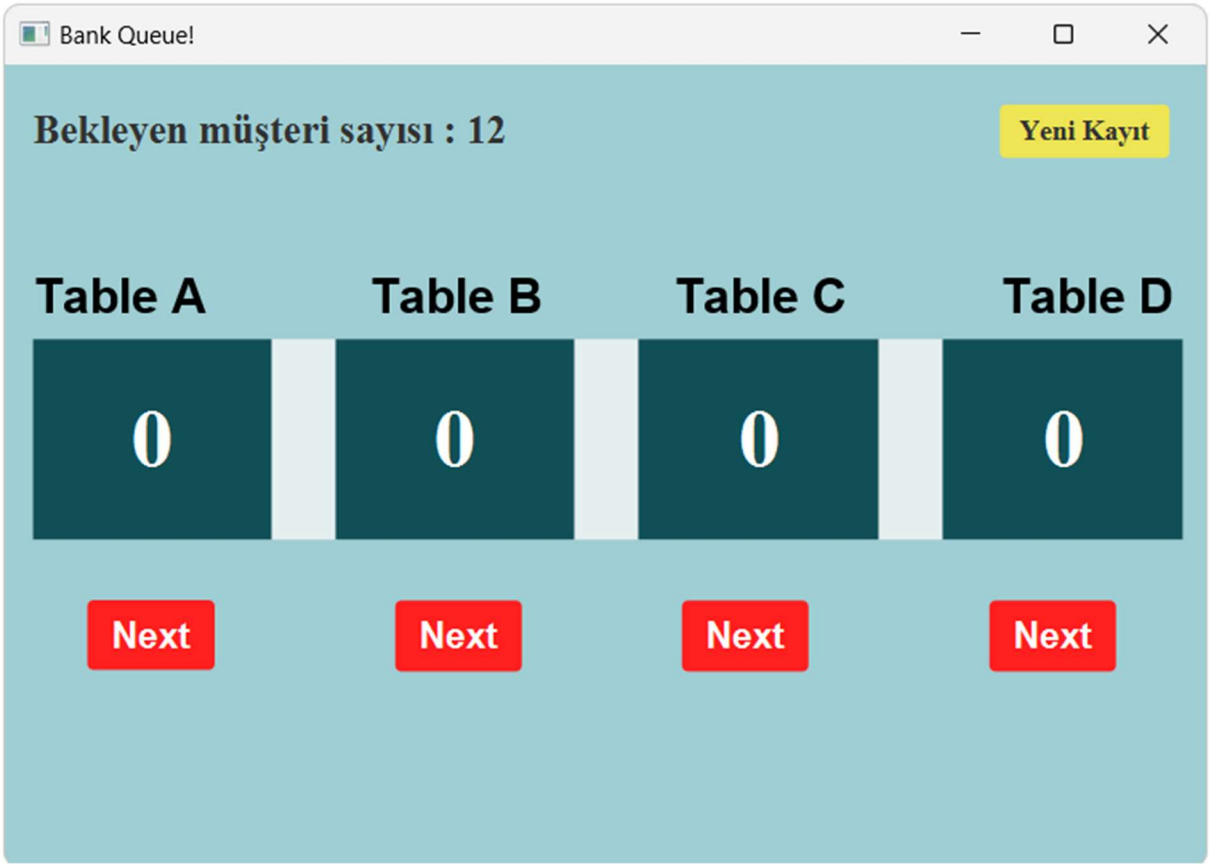
4. Use Case



5. UML Diyagramı



6. Arayüz


Bank Queue!
Bekleyen müşteri sayısı : 12
Yeni Kayıt
Table A
Table B
Table C
Table D
0
0
0
0
Next
Next
Next
Next


New Customer Entry
Yeni Müşteri Girişi
Adınız:
Soyadınız:
Geliş Sebebiniz:
Tamam