ATM Projesi

Bütün projelerde:

- Ön yüz olarak (Thymeleaf,React,Angular) herhangi bir arayüzle yapabilirsiniz.
- Aşağıdaki proje fikirlerinde hepsinde olmasını istediğimiz kurallar:
- Projelerde Roller önem arz etmektedir. Her projede mutlaka en az 2 rol olmalı örneğin: Admin,User v.s
- Veriler Mutlaka Database kaydedilmelidir. (Mysql-Postgresql-H2Database)
- Projelerde mutlaka servisler yazılması gerekiyor.
- Spring Boot 22 videodan itibaren Katmanlı mimari yapısı olacak şekilde projelerimizi yapabiliriz ve kullanılan librariesler kullanmak lombok,swagger-ui v.s

FrontEnd için kendimiz tasarım yapmamız beklenmektedir

- Html5
- Css3
- Js(React veya Angular) bir tanesini seçerek devam edebiliriz benim size tavsiyem React ilerlemeniz daha revaçta.
- Jquery
- Bootstrap

Backend

- OOP kullanmak çok önemli(interface,abstract,inheritance), Stream API, Optional, enum kullanılmalıdır.

Projelerde yapmamız gerekenler:

- Design pattern kullanmalıyız.
- Clean code mantığında ilerlemeliyiz.
- S.O.L.I.D prensibine uygun kodlar yazmalıyız.
- Projede UML diyagram yapmalıyız.
- Database EER diyagram yapmalıyız.
- Loglama tutmak.
- Şifreler maskelenmelidir.
- Sistemdeki kullanıcılar için Hem database kaydetmek ve io(input/output) dosya kaydetmek.
- Paging yapılabilmeli(Sayfalama).

Atm projesi:

Not: Aşağıdaki gereksinimlere göre ui(thymeleaf,react veya angular) sayfa yapalım.

- Roller: Admin(Banka) user(Müşteri)
- Database relation: admin(1) - user (N) ==> Spring Data(@OneToMany @ManyToOne ilişki üzerinde olmalıdır)
- Register/Login: Eğer kullanıcı üye değilse üye olması gerekiyor şifreler database maskelenmiş şekilde kaydedilmelidir. (Spring Security)
- Tanımlama:Müşteri şifresini girerek sisteme giriş yapar (Login için 3 giriş hakkı vardır yoksa bloke olur)
- Müşteri eğer para eklerse parasına ekleme , para çekerse hesaptan para eksilecek
- Validation: Eğer Müşterinin parası yoksa para çekemez
- Loglama: Yapılan her bir işlem için mutlaka loglama tutmak gerekiyor

Müşteri ekranında:

- Müşteri para ekleme
- Müşteri para çekme
- Müşteri para havale

Admin : admin gün içinde sisteme giren müşterileri takip edebilmeli. Her bir müşteri için bilgilerine erişim sağlama örneğin hangi müşterinin ne kadar parası var (ipucu: sql inner join)

Dikkat:

- OOP kullanmak çok önemli(interface,abstract,inheritance) , Stream API,Optional,enum kullanılmalıdır.
- Design pattern kullanmalıyız.
- Clean code mantığında ilerlemeliyiz
- S.O.L.I.D prensibine uygun kodlar yazmalıyız.