**28.03.2023**

İlgili günde bir önceki gün geliştirilmeye başlanan vatandaşlık sisteminin Controller katmanı kodlanmaya başlanmıştır. Geliştirilen API servisi üzerinden kullanıcı veri göndermesini ve veri talep etmesini sağlayan yapılar başarıyla tanımlanmıştır.

Controller katmanında kullanıcıdan Citizen tablosunda işlem yapmak için gelen istekleri kabul etmek veya CitizenController isminde bir sınıf oluşmuştur. Bu sınıf Rest API olarak çalışacağından dolayı sınıfa @RestController anotasyonu verilmiştir. Kullanıcı Citizen tablosu ile ilgili işlem yapmayı talep etmek için URL adresinde alt adres olarak “/citizenApi” adresine gelmesi koşulu konulduğundan CitizenController sınıfına @RequestMapping(“/citizenApi”) anotasyonu verilmiştir. Controller katmanının Repository katmanına erişebilmesi için sınıfa @Autowired anotasyonuna sahip citizenRepository isminde bir alan adı (field) tanımlanmıştır.

API servisİ üzerinde tanımlanması planlanan fonksiyonlar, ne işe yarayacağı ve hangi alt adresle erişilmesi gerektiğine aşağıda yer verilmiştir.

**getAllCitizens():** Tüm vatandaş listesini geri döndürür. (Alt adres: /citizens)

**getCitizenById():** HTTP Body üzerinden gönderilen ID bilgisine göre bir vatandaşa ait bilgileri döndürür.

(Alt adres: /citizens/{id})

**createCitizen():** Gönderilen isim, soyisim , kimlik numarası ile yeni bir kayıt oluşturur. İstek POST yöntemiyle gönderilmelidir. (Alt adres: “/citizens”)

**updateCitizen():** Gönderilen ID bilgisi ve Body üzerinden gönderilen yeni bilgileri kullanılarak bir kayıt üzerinde güncelleme işlemi gerçekleştirir. İstek PUT yöntemiyle yapılmalıdır. (Alt adres: “/citizens/{id}”)

**deleteCitizen():** Gönderilen ID bilgisini kullanarak eşleşen kaydı tablodan siler. İstek DELETE yöntemiyle yapılmalıdır. (Alt adres: “/citizens/{id}”)

**deleteAllCitizens():** Tablodaki tüm vatandaş kayıtlarını siler. İstek DELETE yöntemiyle yapılmalıdır.

(Alt adres: “/citizens/{id}”)

Kullanıcının tüm verilere ulaşacağı API fonksiyonu olan getAllCitizens() fonksiyonu aşağıdaki gibi tanımlanmıştır. Fonksiyon yanıt olarak ResponseEntity içerisinde Citizen sınıfından nesneler barındıran bir liste döndürmektedir. ResponseEntity kullanıcının isteğine yanıt olarak gönderilen bir değer ve bir Http kodunu taşıyabilen yanıt paketidir. Eğer kullanıcı parametre olarak name parametresiyle isim bilgisi gönderdiyse Repository katmanında tanımlanan findByName() fonksiyonu aracılığıyla veritabanı üzerinde koşullu arama gerçekleştirilir ve eğer sonuç bulunursa bu sonuç kullanıcıya yanıt olarak gönderilir. Aşağıdaki kodun benzeri ID ile arama için de gerçekleştirilmiştir.

@GetMapping("/citizens")

public ResponseEntity<List<Citizen>> getAllCitizens(@RequestParam(required = false) String name) {

try {

List<Citizen> citizens = new ArrayList<Citizen>();

if (name == null) citizenRepository.findAll().forEach(citizens::add);

else citizenRepository.findByName(name).forEach(citizens::add);

if (citizens.isEmpty()) { return new ResponseEntity<>(HttpStatus.NO\_CONTENT); }

return new ResponseEntity<>(citizens, HttpStatus.OK);

} catch (Exception e) { return new ResponseEntity<>(null, HttpStatus.INTERNAL\_SERVER\_ERROR); }

}

Kullanıcının vatandaş tablosuna kayıt ekleyebilmesini sağlayan createCitizen() fonksiyonu aşağıdaki gibi tanımlanmıştır. Kullanıcı kaydetmek istediği vatandaşın bilgilerini sunucuya gönderdiği için POST yöntemi kullanılmıştır. Gönderilen bilgiler sunucu tarafında @RequestBody anotasyonuna sahip Citizen nesnesi ile fonksiyon içinde karşılanmıştır. Bu bilgiler otomatik olarak bu Citizen nesnesinin içerisine set edilmektedir. Ardından bu nesne citizenRepository nesnesinin sahip olduğu save fonksiyonu ile veritabanına kaydedilmiştir. Kullanıcıya yanıt olarak yeni kayıt ve HTTP/201 koduna sahip CREATED bilgisi döndürülmüştür. Eğer herhangi exception(istisna) gerçekleşirse kullanıcıya yanıt olarak HTTP/500 kodu döndürülmektedir.

@PostMapping("/citizens")

public ResponseEntity<Citizen> createCitizen(@RequestBody Citizen citizen) {

try {

Citizen \_citizen = citizenRepository

.save(new Citizen(citizen.getName(), citizen.getSurname(), citizen.getCitizen\_number()));

return new ResponseEntity<>(\_citizen, HttpStatus.CREATED);

} catch (Exception e) {

return new ResponseEntity<>(null, HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

Geliştirilen API fonksiyonlarının testi SoapUI programında gerçekleştirilmiştir. <http://localhost:8080/citizenApi/citizens> adresine yapılan isteği ardından dönen yanıt EK-12 görselinde, <http://localhost:8080/citizenApi/citizens?name=Eda> adresine yapılan “Eda” ismindeki kişilerin getirilmesi için yapılan isteğin yanıtı EK-13 görselinde gözlemlenebilmektedir.