**29.03.2023**

İlgili günde geliştirilmekte olan vatandaşlık sistemi API servisinin güncelleme ve silme işlemleri için geliştirmeler yapılmıştır.

Veritabanında kayıtlı olan bir kişiye ait bilgilerin güncellenmesi için aşağıda belirtilen fonksiyon Controller katmanındaki CitizenController sınıfına tanımlanmıştır. Veri güncelleme işlemi yapılacağı için HTTP protokolünün PUT metodundan faydalanılmıştır. Sunucuya gelen PUT isteklerinin updateCitizen() fonksiyonuna yönlendirilebilmesi fonksiyonun başına @PutMaping(“/citizens/{id}) anotasyonu eklenmiştir. Kullanıcı güncelleme işlemi için alt adresin sonuna ilgili kaydın ID bilgisi eklemeli ve HTTP isteğinin body kısmına verinin güncel halini JSON notasyonunda yazmalıdır. Güncelleme işlemi için öncelikle veritabanı üzerinde kullanıcının göndermiş olduğu ID numarası ile arama yapılmaktadır. Eğer kayıt bulunursa kullanıcı body üzerinde göndermiş olduğu bilgiler ile setter metotlar çalıştırılır. Ardından kullanıcıya kaydın güncel hali ve OK anlamına gelen HTTP/200 kodu gönderilmektedir. Eğer veritabanı üzerinde belirtilen ID numarasına sahip kayıt bulunamazsa kullanıcıya NOT FOUND anlamına gelen HTTP/404 kodu gönderilir.

@PutMapping("/citizens/{id}")

public ResponseEntity<Citizen> updateCitizen(@PathVariable("id") long id, @RequestBody Citizen citizen) {

Optional<Citizen> citizenData = citizenRepository.findById(id);

if (citizenData.isPresent()) {

Citizen \_citizen = citizenData.get();

\_citizen.setName(citizen.getName());

\_citizen.setSurname(citizen.getSurname());

\_citizen.setCitizen\_number(citizen.getCitizen\_number());

return new ResponseEntity<>(citizenRepository.save(\_citizen), HttpStatus.OK);

} else {

return new ResponseEntity<>(HttpStatus.NOT\_FOUND);

}

}

Veritabanı üzerinde silme işlemi içi aşağıda belirtlen deleteCitizen fonksiyonu güncelleme işleminde olduğu gibi CitizenController sınıfına tanımlanmıştır. Silme işlemi için HTTP protokolünün DELETE fonksiyonundan yararlanılmıştır. Sunucuya gelen DELETE isteklerinin deleteCitizen fonksiyonuna yönlendirilmesi için fonksiyonun başına @DeleteMapping(“/citizens/{id}”) anotasyonu tanımlanmıştır. Kullanıcı silinmesini istediği kaydın ID değerini alt adresin sonuna eklemelidir. Bu ID bilgisi ile veritabanı üzerinde koşullu silme işlemi gerçekleştirilir. Silme işlemi başarıyla gerçekleştirilirse kullanıcıya NO\_CONTENT anlamına gelen HTTP/204 kodu yanıt olarak gönderilir.

@DeleteMapping("/citizens/{id}")

public ResponseEntity<HttpStatus> deleteCitizen(@PathVariable("id") long id) {

try {

citizenRepository.deleteById(id);

return new ResponseEntity<>(HttpStatus.NO\_CONTENT);

} catch (Exception e) {

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

}

}

Geliştirilen PUT ve DELETE fonksiyonlarının testi SoapUI programında gerçekleştirilmiştir. <http://localhost:8080/citizenApi/citizens>/1 adresine yeni bilgilerle yapılan PUT isteği ardından dönen yanıt EK-15 görselinde, <http://localhost:8080/citizenApi/citizens/1> adresine yapılan DELETE isteğinin ardından dönen HTTP/204 yanıtı EK-16 görselinde gözlemlenebilmektedir.

**KAYNAK: https://www.bezkoder.com/spring-boot-jpa-h2-example/**