REST SERVİS VERSİYONLAMA

Api versiyonlama bazı durumlarda ihtiyaç duyduğumuz bir yöntemdir. En çok ihtiyaç duyulan alan, geliştirilen yazılımın apilerin diğer müşterilere göre şekilleniyor olması gerektiğidir. Bu tür durumlarda elimizde bulunan api üzerinde değişikliklerimizi yaparak bunları diğer müşterilerimize sunarız.

Şimdi api versiyonlama için oluşturulan bazı best practice durumları inceleyelim.

1- URL PATH DÜZENLEME

Elimizde var olan API yi versiyonladığımızda bunu bir endpointte sunmamız gerekir. Fakat yapıyı bozmamak adına pathlerimizin başına v1,v2 gibi belirteçler koyarak rahatça ayırt edebiliriz

ÖR:

```
@GetMapping(value = "/v1/messages")
public ResponseEntity<List<MessageResponse>> getAllMessages() {
    return new ResponseEntity<>(messageService.getAllMessageForVersion1(), HttpStatus.OK);
}

@GetMapping(value = "/v2/messages")
public ResponseEntity<List<MessageResponse>> getAllMessages() {
    return new ResponseEntity<>(messageService.getAllMessageForVersion2(), HttpStatus.OK);
}
```

2-URL PARAMETRESİ ARACILIĞIYLA VERSİYONLAMA

Bazı zamanlarda 1.yöntemde anlatıldığı gibi endpointlerimizi çoklamak istemeyiz. Bu tür durumlarda endpointlerimize bir parametre geçerek burada hangi versiyonumuz için işlem görmek istediğimizi atarız.

Atamış olduğumuz parametreye göre endpointlerimiz içerisinde işlemlerimizi gerçekleştirir ve return olarak sonucu döneriz.

ÖR:

```
@GetMapping(value = "/messages" params = "version=1")
public ResponseEntity<List<MessageResponse>> getAllMessages() {
    return new ResponseEntity<>(messageService.getAllMessageForVersion1(), HttpStatus.OK);
}

@GetMapping(value = "/messages" params = "version=2")
public ResponseEntity<List<MessageResponse>> getAllMessages() {
    return new ResponseEntity<>(messageService.getAllMessageForVersion2(), HttpStatus.OK);
}
```

3- CUSTOM HEADER ARACILIĞIYLA VERSİYONLAMA

Endpointimizde Mapping anotasyonu geçtiğimiz yerde header string olarak kendi belirlediğimiz bir değer verilebilir.

ÖR:

```
@GetMapping(value = "/messages" headers = "X-API-VERSION=1")
public ResponseEntity<List<MessageResponse>> getAllMessages() {
    return new ResponseEntity<>(messageService.getAllMessageForVersion1(), HttpStatus.OK);
}

@GetMapping(value = "/messages" headers = "X-API-VERSION=2")
public ResponseEntity<List<MessageResponse>> getAllMessages() {
    return new ResponseEntity<>(messageService.getAllMessageForVersion2(), HttpStatus.OK);
}
```

Verdiğimiz olduğumuz bu string değerin başına X harfi konulması tavsiye edilmektedir.

Daha sonra postman aracılığıyla göndereceğimiz istekte headers altında bu string değeri geçeriz ve istediğimiz endpointe istek atmış oluruz.

ÖR: