

## TUZLA BELEDİYESİ BELEDİYE ÇALIŞAN TAKİP SİSTEMİ

**Web Servis Geliştirilmesi:** 8080 portu üzerine herhangi bir path eklenmeden rest api servisleri oluşturuldu.

**addTask:** @PathVariable ile URL'den gelen employeeId değerini metod parametresine alır. @RequestBody ile İstek gövdesindeki Task nesnesini alır ve eTask parametresine atar.

Tasks koleksiyonunda employeeId anahtarının olup olmadığını kontrol eder eğer null dönerse verilen hata mesajı NOT\_FOUND döner. employeeId mevcutsa, ilgili çalışanın görev listesine yeni görevi ekler.

Görev başarıyla eklendiğinde, HTTP 200 (OK) durumu ve "Task added successfully" mesajıyla yanıt döner.

```
@PostMapping("/addTask/{employeeId}")
public ResponseEntity<String> addTask(@PathVariable Integer employeeId, @RequestBody Task eTask)
{
    if (!Tasks.containsKey(employeeId))
        return ResponseEntity.status(HttpStatus.NOT_FOUND).body("Employee not found");

    Tasks.get(employeeId).add(eTask.task);

    return ResponseEntity.ok("Task added successfully");
}
```

**listAllTasks:** HTTP GET isteğiyle /taskList URL'sine gelen istekleri bu metoda yönlendirir.

Bu metodun dönüş tipi, Integer türünde anahtarlar ve List<String> türünde değerler içeren bir haritadır.

“return Tasks;” ile Tasks haritasını döner. Bu harita, çalışanların ID'lerini anahtar olarak, bu çalışanlara atanmış görevlerin (String) listelerini değer olarak içerir.

```
@GetMapping("/taskList")
public Map<Integer, List<String>> listAllTasks() {
    return Tasks;
}
```

**BİRİM TESTLER:** Programlama dilleri temel olarak giriş-input olarak çıktı-output üretir. Girişlerin ve çıkışların kaynağı farklı olsa da aynı girişlerin aynı sonuç vermesi beklenir. Bu amaç ile JUnit kütüphanesini kullanarak yazılan servislerin testleri gerçekleştirildi.

**listAllTasksTest:** MockMvc nesnesi kullanarak /taskList URL'sine bir GET isteği gönderir. Yanıt durumunun HTTP 200 (OK) olmasını bekler. Yanıtın içeriğinin JSON formatında olmasını bekler. JSON yanıtında 1 ve 2 anahtarına karşılık gelen listenin boyutunun 0 olmasını bekler. (öncesinde herhangi bir test taskı eklenmemiş olmasından kaynaklı)

```
@Test
public void listAllTasksTest() throws Exception {
    mockMvc.perform(get("/taskList"))
        .andExpect(status().isOk())
        .andExpect(content().contentType(MediaType.APPLICATION_JSON))
        .andExpect(jsonPath("$['1']", hasSize(0)))
        .andExpect(jsonPath("$['2']", hasSize(0)));
}
```

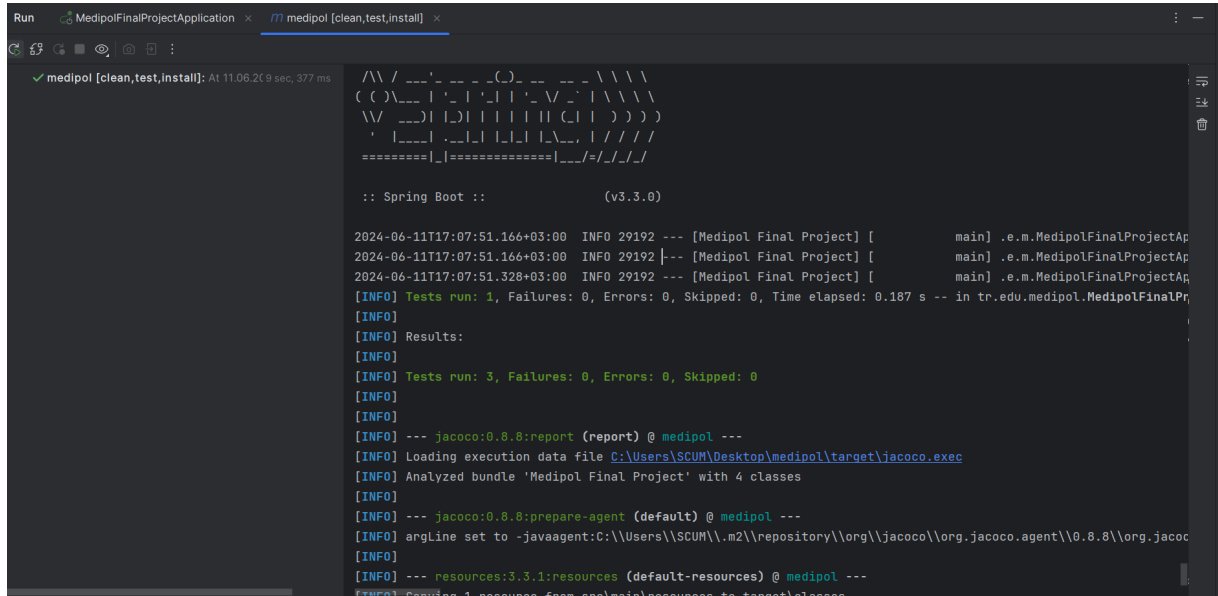
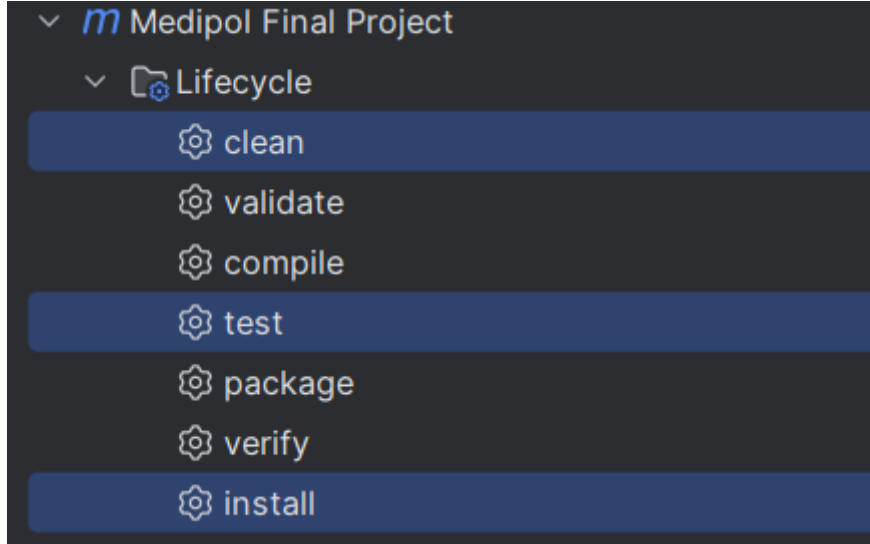
**addTaskTest:** "Test Task" adında yeni bir Task nesnesi oluşturur. MockMvc nesnesi kullanarak /addTask/1 URL'sine bir POST isteği gönderir. 1 burada employeeId'yi temsil eder. Yanıt durumunun HTTP 200 (OK) olmasını ve Yanıtın içeriğinin "Task added successfully" olmasını bekler. Son olarak bir önceki fonksiyondaki list isteğini gönderiyor fakat bu sefer beklenen değer "Test Task" olarak modifiye edildi.

```
@Test
public void addTaskTest() throws Exception {
    Task newTask = new Task("Test Task");

    mockMvc.perform(post("/addTask/1")
        .contentType(MediaType.APPLICATION_JSON)
        .content(objectMapper.writeValueAsString(newTask)))
        .andExpect(status().isOk())
        .andExpect(content().string("Task added successfully"));

    mockMvc.perform(get("/taskList"))
        .andExpect(status().isOk())
        .andExpect(content().contentType(MediaType.APPLICATION_JSON))
        .andExpect(jsonPath("$['1']", hasSize(1)))
        .andExpect(jsonPath("$['1'][0]", is("Test Task")));
}
```

**COVERAGE:** test kapsama oranını hesaplatmak için jacoco plugini “pom.xml” dosyasının en alt kısmındaki “build/plugins” etiketinin altında eklendi. Ardından aşağıdaki aksiyonlar çalıştırıldı.



Medipol Final Project

## Medipol Final Project

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxt	Missed Lines	Missed Methods	Missed Classes
tr.edu.medipol	<div><div></div></div>	%88	<div><div></div></div>	%50	4 13	3 23	3 12	0 4
Total	13 of 118	%88	1 of 2	%50	4 13	3 23	3 12	0 4

**Sürekli Entegrasyon:** yapılan değişikliklerini sürekli olarak entegre etmek için otomatikleştirilmiş bir süreçtir. Bu nedenle belirtilen kodu eklendi ve gerekli izinler sağlandı.

Bu action kod üzerindeki her güncellemede -pullrequestler dahil olmak üzere- test kapsama oranını hesaplayacak.

```
name: Java CI with Maven

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  build:

    runs-on: ubuntu-latest

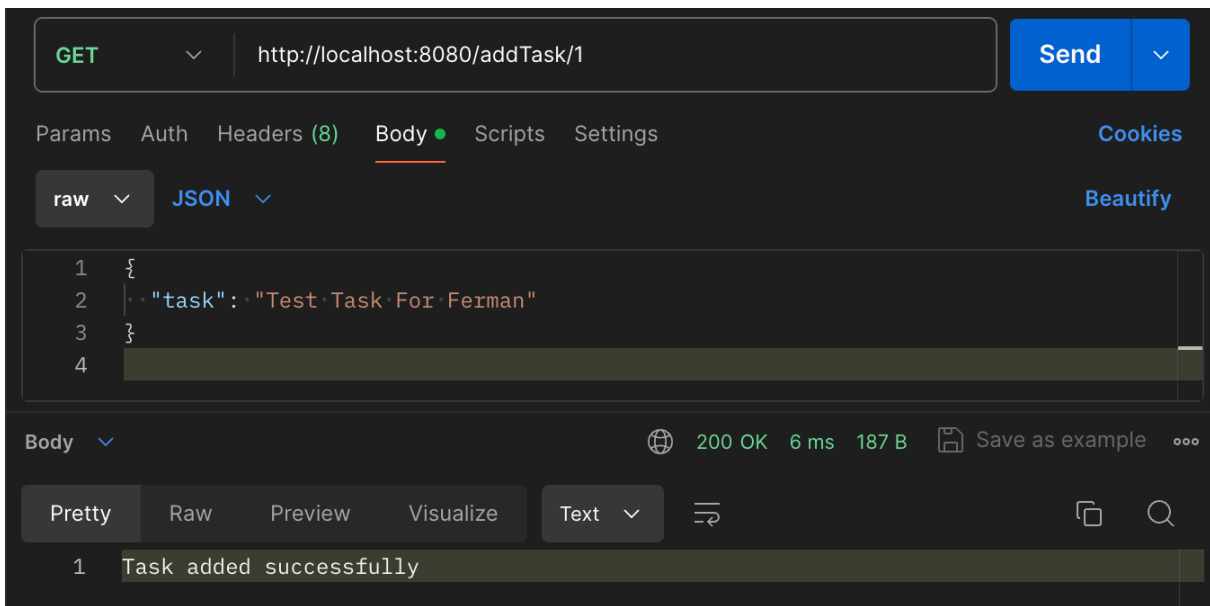
    permissions:
      pull-requests: 'write'

    steps:
      - uses: actions/checkout@v4
      - name: Set up JDK 17
        uses: actions/setup-java@v3
        with:
          java-version: '17'
          distribution: 'temurin'
          cache: maven
      - name: Build with Maven
        run: mvn -B package --file pom.xml

      - name: Coverage
        id: jacoco
        uses: madrapps/jacoco-report@v1.3
        with:
          paths: ${GITHUB_WORKSPACE}/target/site/jacoco/jacoco.xml
          token: ${GITHUB_TOKEN}
```

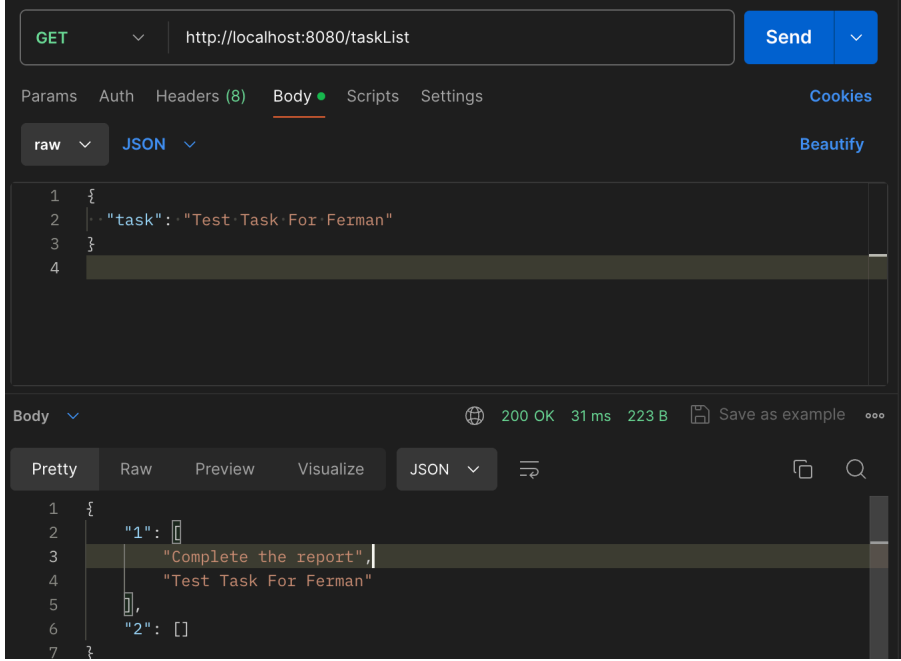
**Postman & JMeter Testleri:** Postman ile servisler, JMeter ile strese dayanıklılık testleri yapıldı.

**/addTask/{EmployeeId}:** Verilen URL adresine adrese çalışanın idsi ve “task” bodysi ile POST isteği gönderildi.

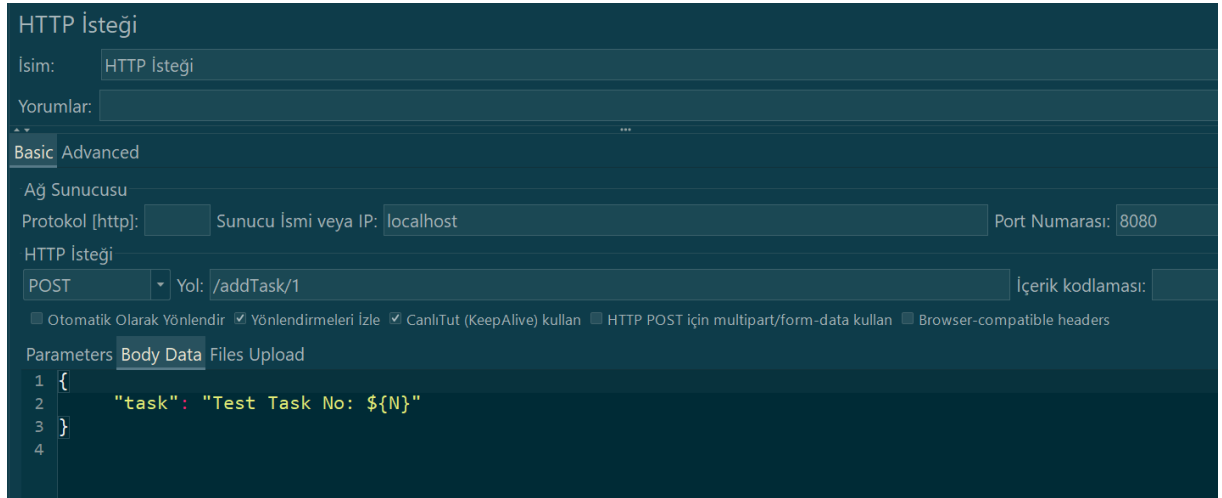


**/taskList:** Verilen URL adresine GET isteği gönderilerek öncesinde oluşturulan taskler listelendi.

**Http Request:** Göndereceğimiz isteğin türü, bodysi ve yolunu girerek testimizin requestini tanımlandı.



**Http Request:** Göndereceğimiz isteğin türü, body'si ve yolunu girerek testimizin requestini tanımlandı.



**Counter:** Counter ile her gönderilen istekte 1 artan bir değişken oluşturuldu.

Sayaç	
İsim:	Sayaç
Yorumlar:	
Starting value	5
Arttır	1
Maximum value	
Numara biçimi	
Referans İsmi	N
<input checked="" type="checkbox"/> Sayacı her kullanıcı için bağımsız çalıştır	
<input type="checkbox"/> Reset counter on each Thread Group Iteration	

**Result Tree:** Çalıştırılmanın sonrasında isteklerin başarılı olup olmadığını ve detaylarını görüntülemek için kullanıldı.

### Sonuçları Gösterme Ağacı

İsim:

Yorumlar:

Sonuçları dosyaya yaz / Dosyadan oku

Dosya ismi   Sadece Log/Görüntüleme: ☐ Hatalar ☐ Başarılar

Search:  ☐ Case sensitive ☐ Regular exp.

Metin Göster

- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği
- HTTP İsteği

Örnekleyici sonucu İstek Cevap verisi

Load time:3  
Connect Time:0  
Latency:3  
Size in bytes:185  
Sent bytes:219  
Headers size in bytes:162  
Body size in bytes:23  
Sample Count:1  
Error Count:0  
Data type ("text"|"bin"|""):text  
Response code:200  
Response message:

HTTPSampleResult fields:  
ContentType: text/plain;charset=UTF-8  
DataEncoding: UTF-8

```
1  {
2      "1": [
3          "Test Task For Berk",
4          "Test Task No: 82",
5          "Test Task No: 84",
6          "Test Task No: 63",
7          "Test Task No: 10",
8          "Test Task No: 81",
9          "Test Task No: 52",
10         "Test Task No: 59",
11         "Test Task No: 86",
12         "Test Task No: 66",
13         "Test Task No: 80",
14         "Test Task No: 48",
15         "Test Task No: 35",
16         "Test Task No: 16",
17         "Test Task No: 69",
18         "Test Task No: 37",
19         "Test Task No: 45",
20         "Test Task No: 21",
21         "Test Task No: 22",
22         "Test Task No: 31",
23         "Test Task No: 46",
```



**Git:** Verilen kodları sırasıyla kullanılan IDEnin terminale girilip repository oluşturuldu ve pushlandı.

```
echo "# yazilim" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/fermanarpck/yazilim.git
git push -u origin main
```