



**MEDİPOL**  
**UNV-ISTANBUL**  
**ISTANBUL MEDIPOL UNIVERSITY**

Adı Soyadı / Name Surname: Serra Vuslat AKYÜZ

Numarası/ Number: H5220055

MYO/YO/Fakülte/ Enstitü(School):Meslek Yüksekokulu

Bölüm/Program(Dept.):Bilgisayar Programcılığı

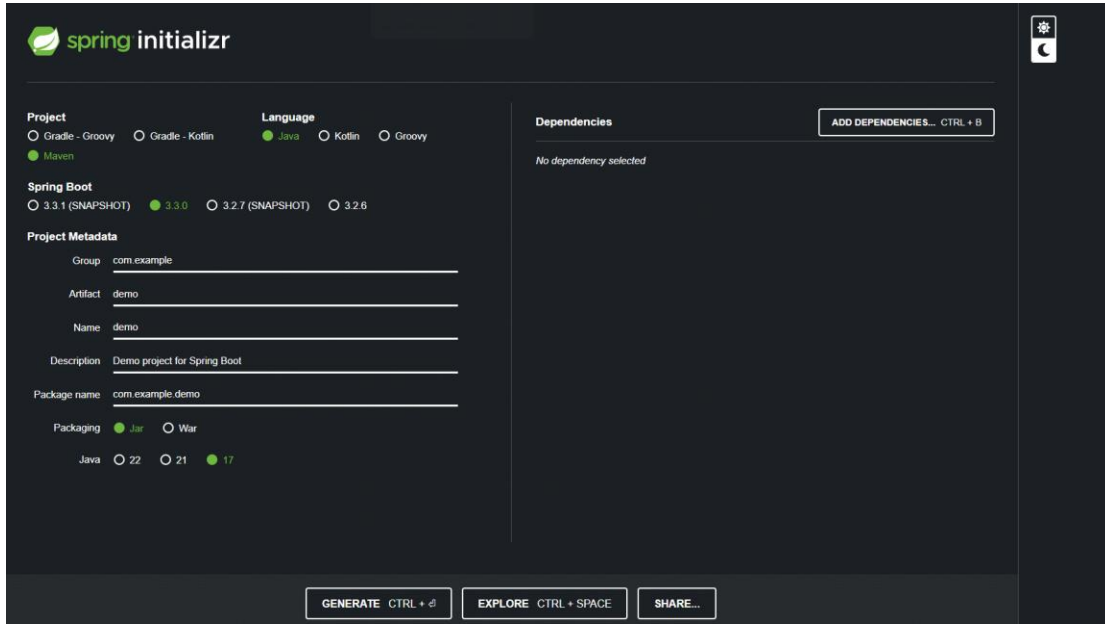
Dersin Adı/ Course Name: Yazılım Geliştirme Ortam Araçları

Öğretim Elemanı/ Instuctor: Özkan Sarı

Öğrenci İmzası/Student Signature:

# Proje: Fatih Belediye Çalışan Takibi ve Yönetim Sistemi Final Projesi

## 1- Proje Kurulumu:



The image shows the Spring Initializr web interface. The top left features the 'spring initializr' logo. Below it, the 'Project' section has radio buttons for 'Gradle - Groovy', 'Gradle - Kotlin', 'Maven' (selected), and 'Language' with 'Java' (selected), 'Kotlin', and 'Groovy'. The 'Spring Boot' section has radio buttons for '3.3.1 (SNAPSHOT)', '3.3.0' (selected), '3.2.7 (SNAPSHOT)', and '3.2.6'. The 'Project Metadata' section includes input fields for 'Group' (com.example), 'Artifact' (demo), 'Name' (demo), 'Description' (Demo project for Spring Boot), and 'Package name' (com.example.demo). The 'Packaging' section has radio buttons for 'Jar' (selected) and 'War'. The 'Java' section has radio buttons for '22', '21', and '17' (selected). The 'Dependencies' section on the right has a button 'ADD DEPENDENCIES... CTRL + B' and the text 'No dependency selected'. At the bottom, there are three buttons: 'GENERATE CTRL + G', 'EXPLORE CTRL + SPACE', and 'SHARE...'.

## 2. Proje Yapılandırması

src/main/resources/application.properties dosyasını açtım ardından yapılandırmayı ekledim:

```
lication.properties •
Users > HAS AKYÜZ LOJİSTİK > AppData > Local > Temp > Rar$Dla9000.7155 > application.properties
spring.application.name=demo
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.h2.console.enabled=true
|
```

### 3. Model Sınıfları Oluşturma:

#### a.Employee (Çalışan) Sınıfı:

```
File Edit Selection View Go Run Terminal Help
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C:\Users\HAS AKYÜZ LOJİSTİK\AppData\Local\Temp\Rar$Dla9000.46372\mune
1 package com.example.employee.management.model;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7
8 @Entity
9 public class Employee {
10     @Id
11     @GeneratedValue(strategy = GenerationType.IDENTITY)
12     private Long id;
13     private String firstName;
14     private String lastName;
15     private String startDate;
16     private String endDate;
17     private String employmentType; // Kadrolu veya İhtisari
18
19     // Getters and Setters
20     public Long getId() {
21         return id;
22     }
23
24     public void setId(Long id) {
25         this.id = id;
26     }
27
28     public String getFirstName() {
29         return firstName;
30     }
31
32     public void setFirstName(String firstName) {
33         this.firstName = firstName;
34     }
35
36     public String getLastName() {
37         return lastName;
38     }
39
40     public void setLastName(String lastName) {
41         this.lastName = lastName;
42     }
43
44     public String getStartDate() {
45         return startDate;
46     }
47 }
```

The screenshot shows an IDE window with the following code:

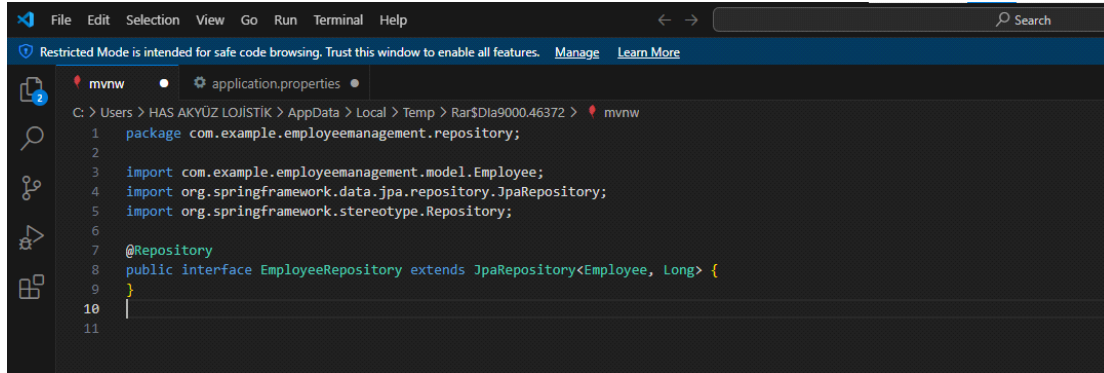
```
23 public class Employee {
24     public void setId(long id) {
25         this.id = id;
26     }
27
28     public String getFirstName() {
29         return firstName;
30     }
31
32     public void setFirstName(String firstName) {
33         this.firstName = firstName;
34     }
35
36     public String getLastName() {
37         return lastName;
38     }
39
40     public void setLastName(String lastName) {
41         this.lastName = lastName;
42     }
43
44     public String getStartDate() {
45         return startDate;
46     }
47
48     public void setStartDate(String startDate) {
49         this.startDate = startDate;
50     }
51
52     public String getEndDate() {
53         return endDate;
54     }
55
56     public void setEndDate(String endDate) {
57         this.endDate = endDate;
58     }
59
60     public String getEmploymentType() {
61         return employmentType;
62     }
63
64     public void setEmploymentType(String employmentType) {
65         this.employmentType = employmentType;
66     }
67
68 }
```

## b.Task (Görev) Sınıfı:

The screenshot shows an IDE window with the following code:

```
1 package com.example.employee.management.model;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7 @Entity
8 public class Task {
9     @Id
10     @GeneratedValue(strategy = GenerationType.IDENTITY)
11     private long id;
12     private String description;
13     private long employeeId;
14
15     public long getId() {
16         return id;
17     }
18
19     public void setId(long id) {
20         this.id = id;
21     }
22     public String getDescription() {
23         return description;
24     }
25     public void setDescription(String description) {
26         this.description = description;
27     }
28     public long getEmployeeId() {
29         return employeeId;
30     }
31     public void setEmployeeId(long employeeId) {
32         this.employeeId = employeeId;
33     }
34
35 }
```

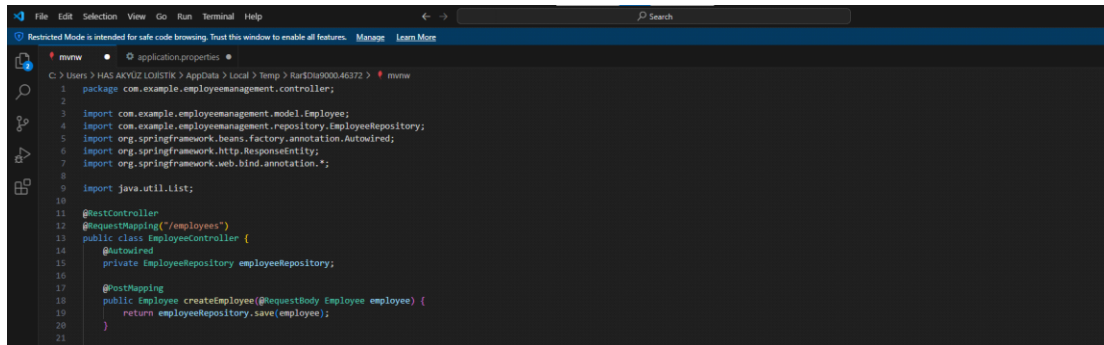
#### 4. Repository Sınıfları Oluşturma:



The screenshot shows an IDE window with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a package named 'com.example.employeeManagement.repository'. The code editor displays the following Java code:

```
1 package com.example.employeeManagement.repository;
2
3 import com.example.employeeManagement.model.Employee;
4 import org.springframework.data.jpa.repository.JpaRepository;
5 import org.springframework.stereotype.Repository;
6
7 @Repository
8 public interface EmployeeRepository extends JpaRepository<Employee, Long> {
9 }
10
11
```

#### 5. Controller Sınıfları Oluşturma



The screenshot shows an IDE window with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a package named 'com.example.employeeManagement.controller'. The code editor displays the following Java code:

```
1 package com.example.employeeManagement.controller;
2
3 import com.example.employeeManagement.model.Employee;
4 import com.example.employeeManagement.repository.EmployeeRepository;
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.http.ResponseEntity;
7 import org.springframework.web.bind.annotation.*;
8
9 import java.util.List;
10
11 @RestController
12 @RequestMapping("/employees")
13 public class EmployeeController {
14     @Autowired
15     private EmployeeRepository employeeRepository;
16
17     @PostMapping
18     public ResponseEntity createEmployee(@RequestBody Employee employee) {
19         return ResponseEntity.ok(employeeRepository.save(employee));
20     }
21 }
```

#### 6. Uygulamayı Başlatma:

```
package com.example.employeeeManagement;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class EmployeeManagementApplication {

    public static void main(String[] args) {
        SpringApplication.run(EmployeeManagementApplication.class, args);
    }

}
```

## 7. Projeyi Çalıştırma:

Proje çalışmaya müsait durumdadır.

## 8. API'yi Test Etme:Json formatı ile birlikte yazdım.

```
{
  "description": "Örnek Görev",
  "employeeId": 1
}
```

```
{
  "firstName": "Ahmet",
  "lastName": "Yılmaz",
  "startDate": "2024-01-01",
  "endDate": "2024-12-31",
  "employmentType": "Kadrolu"
}
```

Görevleri listelemek için (GET):GET isteği gönderilir.

Çalışan eklemek için (POST): JSON gönderilir:

Adım 7: Birim Testleri Yazma:

```
package com.example.employeeManagement.controller;

import com.example.employeeManagement.model.Employee;
import com.example.employeeManagement.repository.EmployeeRepository;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.http.MediaType;
import org.springframework.test.web.servlet.MockMvc;
import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;
import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.post;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.jsonPath;
import com.fasterxml.jackson.databind.ObjectMapper;

@SpringBootTest
@AutoConfigureMockMvc
public class EmployeeControllerTest {

    @Autowired
    private MockMvc mockMvc;

    @Autowired
    private EmployeeRepository employeeRepository;

    @Test
    public void createEmployeeTest() throws Exception {
        Employee employee = new Employee();
        employee.setFirstName("John");
        employee.setLastName("Doe");
        employee.setStartDate("2024-01-01");
        employee.setEndDate("2024-12-31");
        employee.setEmploymentType("Kadrolu");

        ObjectMapper objectMapper = new ObjectMapper();
        String json = objectMapper.writeValueAsString(employee);

        mockMvc.perform(post("/employees")
            .contentType(MediaType.APPLICATION_JSON)
            .content(json)
```

Postman ve JMeter ile Test Etme:

Görev Ekleme (POST) Servisini Test Etme:Body sekmesinden ve raw ve JSON formatını seçtim ve gönderilecek JSON verisini ekledim.

json

```
{  
  "description": "Yeni görev",  
  "employeeId": 1  
}
```

Body Data sekmesinde JSON verisini ekledim.

```
{  
  "description": "Yeni görev",  
  "employeeId": 1  
}
```