EXPERIMENT NO. 10: TESTING FRONTEND INTEGRATION WITH SPRING BOOT BACKEND

Objective

To create a Java application using Spring Boot that demonstrates testing of a frontend web application integrated with a Spring Boot backend.

Theory

Testing integration between frontend and backend ensures that both layers work together seamlessly. Frontend testing verifies that user interactions are processed correctly and result in the expected server responses. Tools like Selenium simulate browser-based interactions, making it possible to test the UI and REST integration end-to-end.

Create the Spring Boot Backend

Use Spring Initializr to generate a Spring Boot project with Spring Web and Spring Boot Test dependencies.

Develop the Frontend Application

Create a simple `index.html` file in `src/main/resources/static` that interacts with the backend via JavaScript or a frontend framework like React, Angular, or Vue.

Sample HTML Code

```
<!-- index.html -->

<!DOCTYPE html>
<html>
 <head>
  <title>Employee App</title>
 </head>
 <body>
  <h1>Employee Management</h1>
  <button onclick="loadEmployees()">Load Employees</button>
  <ul id="employeeList"></ul>

  <script>
   function loadEmployees() {
    fetch("/api/employees")
     .then((response) => response.json())
     .then((data) => {
```

```
        const list = document.getElementById("employeeList");
        list.innerHTML = "";
        data.forEach((emp) => {
          const li = document.createElement("li");
          li.textContent = emp.name + " - " + emp.department;
          list.appendChild(li);
        });
      });
    }
  </script>
 </body>
</html>
```

Write Selenium Integration Tests

```
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.web.server.LocalServerPort;

@SpringBootTest(webEnvironment = SpringBootTest.WebEnvironment.RANDOM_PORT)
public class FrontendIntegrationTest {

    @LocalServerPort
    private int port;

    private WebDriver driver;

    @BeforeEach
    public void setUp() {
        System.setProperty("webdriver.chrome.driver", "path_to_chromedriver");
        driver = new ChromeDriver();
    }

    @AfterEach
    public void tearDown() {
        if (driver != null) {
            driver.quit();
        }
    }

    @Test
```

```
    public void testHomePage() {
        driver.get("http://localhost:" + port);
        // Perform actions and assertions on the page using Selenium
    }
}
```

Important Notes

• Replace `path_to_chromedriver` with the full path to your ChromeDriver executable.
• Ensure the Spring Boot app serves the HTML frontend correctly (from `src/main/resources/static`).
• Selenium opens a real browser to interact with your app. Ensure browser automation is allowed.

Conclusion

This experiment illustrates end-to-end integration testing of a Spring Boot backend and a static frontend UI using Selenium. It verifies that frontend elements correctly communicate with backend APIs, ensuring production readiness.