دانشکده مهندسی برق و کامپیوتر

مهندسی نرم‌افزار - پاییز ۱۴۰۱

تکلیف اول

آشنایی با فناوری‌های پروژه مبنا

در تکلیف‌های آینده در طول ترم، شما روی یک پروژه مبنا کار خواهید کرد و کارکردهایی را به آن اضافه می‌کنید. هدف این تکلیف آشنا کردن شما با فناوری‌های مورد استفاده در پروژه مبنا است. توجه داشته باشید این فناوری‌ها در این درس به صورت سطحی آموزش داده می‌شوند و یادگیری اصلی شما در درس‌های مهندسی اینترنت و آزمون نرم‌افزار شکل می‌گیرد.

شرح تکلیف

مرحله ۰ - راه‌اندازی محیط برنامه‌نویسی

محیط برنامه‌نویسی در این درس آیدیا[1] است. اگر امکان به دست آوردن لیسانس ویرایش آلتیمیت این محیط را دارید می‌توانید از آن استفاده کنید. در غیر این صورت (یا اگر ترجیح می‌دهید از ابزار سبک‌تری استفاده کنید)، تمام تکلیف‌های درس با ویرایش کامیونیتی این ابزار قابل انجام هستند. می‌توانید این نسخه را به طور مجانی از صفحه دانلود ابزار دریافت کنید.

مرحله ۱ - آشنایی با اسپرینگ‌بوت

متن آموزشی ساختن سرویس‌های رست با اسپرینگ را تا ابتدای بخش What makes something RESTful مطالعه کنید و مثال مورد مطالعه را روی کامپیوتر خود بسازید. توصیه می‌شود طبق آن‌چه در این متن ذکر شده، ابتدا با Spring Initializr چارچوب پروژه را تولید و دانلود کنید و بعد آن را در آیدیا باز کنید. دقت کنید در متن آموزشی به سه وابستگی[2] اشاره شده (که باید در Initializr آنها را انتخاب کنید)، اما نام آنها را به طور کامل ذکر نکرده که در زیر نام کامل آنها را آورده‌ایم:

Web → Spring Web
JPA → Spring Data JPA
H2 → H2 Database

بعد از اضافه کردن کلاس‌های مورد نیاز، طبق آنچه در متن آموزشی ذکر شده آن را با ابزار curl بیازمایید. یک ابزار ساده‌تر برای کار با سرویس‌های وب در ترمینال ابزار HTTPie است. استفاده از ابزار Postman هم برای این کار ممکن است.

---

[1] IntelliJ IDEA
[2] dependency

مرحله ۲ - اضافه کردن یک کارکرد ساده

پروژه‌ای که در مرحله قبل ساختید شامل یک موجودیت به نام Employee است. به این کلاس یک فیلد اضافه کنید که میزان ساعت کارکرد کارمند در ماه جاری را نگه دارد. همچنین متدی به این کلاس اضافه کنید که میزان اضافه‌کار کارمند را برگرداند. فرض کنید کارمندها موظف هستند ۲۰۰ ساعت در ماه کار کنند و میزان کار بیش از این عدد اضافه‌کار آنها محسوب می‌شود.

کاری کنید که سرویس وب شما در پاسخ یک درخواست GET روی /employees/2/overtime میزان اضافه‌کار کارمند با شناسه ۲ را در قالب زیر برگرداند:

```
{
    "id": 2,
    "name": "Dilbert",
    "overtime": 34
}
```

راهنمایی: برای ایجاد ساختار بالا، کافی است یک کلاس ساده جاوا تولید کنید که برای فیلدهای فوق getter عمومی داشته باشد. بقیه کارها را اسپرینگ‌بوت به طور خودکار انجام می‌دهد.

پس از انجام مرحله فوق، یک کارکرد جدید اضافه کنید که در پاسخ یک درخواست GET روی /employees/overtime فهرست تمام کارمندانی را برگرداند که مشمول اضافه‌کاری می‌شوند. لزومی ندارد این فهرست ترتیب خاصی داشته باشد.

```
[
    {
        "id": 1,
        "name": "Wally",
        "overtime": 2
    },
    {
        "id": 2,
        "name": "Dilbert",
        "overtime": 34
    }
]
```

راهنمایی: برای این کار کافی است یک لیست از کلاسی که در مرحله قبل ایجاد کردید برگردانید.

مرحله ۳ - استفاده از لومبوک

در این مرحله، از پروژه لومبوک[3] استفاده خواهیم کرد برای این که از تکرار واضحات در کد جلوگیری کنیم. استفاده از لومبوک در آیدیا توسط یک پلاگین به همین نام قابل انجام است. اگر نسخه آیدیای شما نسبتاً جدید باشد این پلاگین به طور خودکار نصب‌شده و فعال است. در غیر این صورت لازم است به صورت دستی آن را اضافه کنید.
پس از نصب، از دو حاشیه‌نویسی[4] @Getter و @Setter استفاده کنید تا کلاس Employee را خلاصه‌تر نمایید. نگاهی به امکانات لومبوک بیندازید و ببینید چگونه می‌توان تعریف کلاس Employee را خلاصه‌تر کرد. دقت کنید چون ما برای ذخیره اشیاء این کلاس در پایگاه داده از JPA استفاده می‌کنیم، استفاده از حاشیه‌نویسی‌های @Data و @EqualsAndHashCode بالقوه می‌تواند موجب مشکل شود و فعلا از آنها استفاده نکنید.

---

[3] Project Lombok
[4] annotation

مرحله ۴ - نوشتن آزمون واحد

برای آزمودن متدی که اضافه‌کار یک کارمند را برمی‌گرداند سه آزمایه[5] طراحی کنید. این آزمایه‌ها را با استفاده از چارچوب JUnit در کلاسی به نام EmployeeTest در همان بسته‌ای که Employee قرار دارد، اما در پوشه src/test/java پیاده‌سازی کنید. این آزمایه‌ها را در آیدیا اجرا کنید و از درستی متدی که نوشتید مطمئن شوید.

برای آشنایی با نوشتن ابتدایی آزمون‌ها در JUnit مثال زیر را ببینید که در آن یک کلاس Person توسط دو آزمایه در کلاس PersonTest آزموده می‌شود (هر چند نتیجه آزمون موفقیت‌آمیز نیست).

```java
package com.example.app;

public class Person {
   private String firstName;
   private String lastName;

   public Person(String firstName, String lastName) {
       this.firstName = firstName;
       this.lastName = lastName;
   }
   public Person(String firstName) {
       this(firstName, null);
   }
   public String getDisplayName() {
       return firstName + " " + lastName;
   }
}
```

```java
package com.example.app;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.assertEquals;

public class PersonTest {
   @Test
   void person_returns_full_name_correctly() {
       Person gholam = new Person("Gholam", "Patoobaf");
       String displayName = gholam.getDisplayName();
       assertEquals("Gholam Patoobaf", displayName);
   }

   @Test
   void person_without_first_name_displays_first_name_only() {
       Person gholam = new Person("Gholam");
       String displayName = gholam.getDisplayName();
       assertEquals("Gholam", displayName);
   }
}
```

[5] test case

تحویل تکلیف

انجام این تکلیف به صورت انفرادی است. برای تحویل تکلیف، پوشه اصلی پروژه را زیپ کنید و در محل مشخص‌شده در صفحه درس بارگذاری نمایید. این تکلیف تحویل حضوری نخواهد داشت و صرف بارگذاری کافی است.

ارزیابی

ارزیابی تکلیف طبق معیارهای زیر با امتیازهای مشخص‌شده انجام خواهد شد.

ساختار صحیح پروژه	۱۰
اجرای صحیح کارکردهای مرحله ۲	۲۰
استفاده مناسب از لومبوک	۵
کیفیت آزمون‌ها	۱۵