Testing Spring Boot Application using TestNG

#Dependency

- **Add TestNG Dependency in you pom.xml**

```
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.7.1</version>
    <scope>test</scope>
</dependency>
```

- **TestNG plugin in your pom.xml under plugin section**

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-surefire-plugin</artifactId>
    <version>3.1.2</version>
    <configuration>
        <suiteXmlFiles>
        <suiteXmlFile>testng.xml</suiteXmlFile>
        </suiteXmlFiles>
    </configuration>
</plugin>
```

- **Define your Service Class whose unit you want to test**
    - **CalculatorService.java**

```
@Service
public class CalculatorService {
public int add(int a, int b) {
return a + b;
}
public int multiply(int a, int b) {
return a * b;
}
}
```

- **Define a RestController to call the service layer methods**
    - **CalculatorController**

```
@RestController
@RequestMapping("/calc")
public class CalculatorController {

@Autowired
private CalculatorService calculatorService;

@GetMapping("/add")
public int add(@RequestParam int a, @RequestParam int b) {
return calculatorService.add(a, b);
}

@GetMapping("/multiply")
public int multiply(@RequestParam int a, @RequestParam int b) {
return calculatorService.multiply(a, b);
}
}
```

- **Add CalculatorServiceTest class inside src/text/java in service package of your app base package**
  - **CalculatorServiceTest.java**

```
public class CalculatorServiceTest {

private CalculatorService service;

@BeforeClass
public void setup() {
service = new CalculatorService();
}

@Test
public void testAdd() {
Assert.assertEquals(service.add(2, 3), 5);
}

@Test
public void testMultiply() {
Assert.assertEquals(service.multiply(4, 5), 20);
}
}
```

- **Mock MVC Test: TestNG Tests for REST API (using MockMvc)**
  - **CalculatorControllerTest.java**

```
@SpringBootTest
@AutoConfigureMockMvc
public class CalculatorControllerTest {


    @Autowired
    private MockMvc mockMvc;


    @Test
    public void testAddEndpoint() throws Exception {
        mockMvc.perform(get("/calc/add?a=10&b=20"))
                .andExpect(status().isOk())
                .andExpect(content().string("30"));
    }
}
```