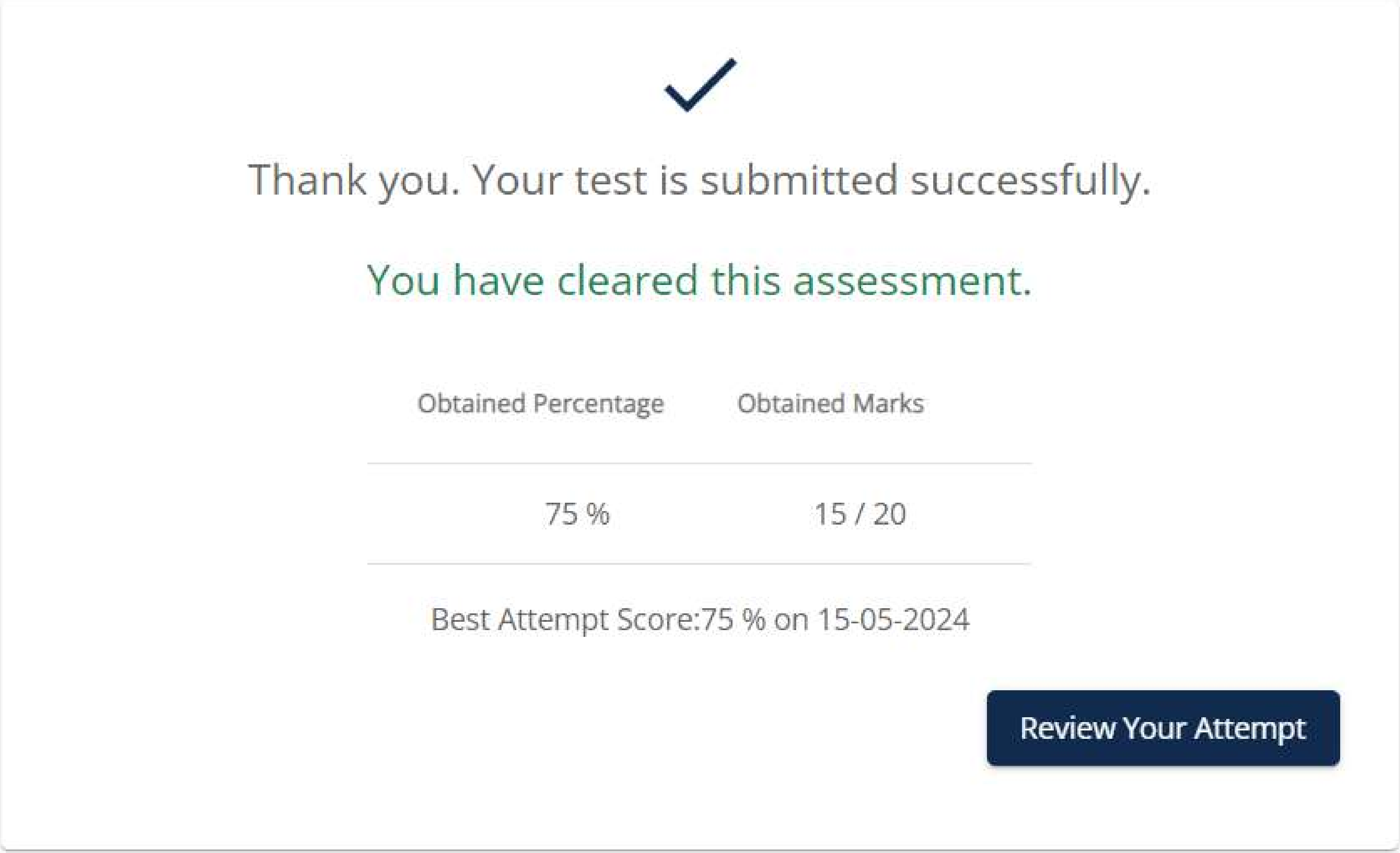


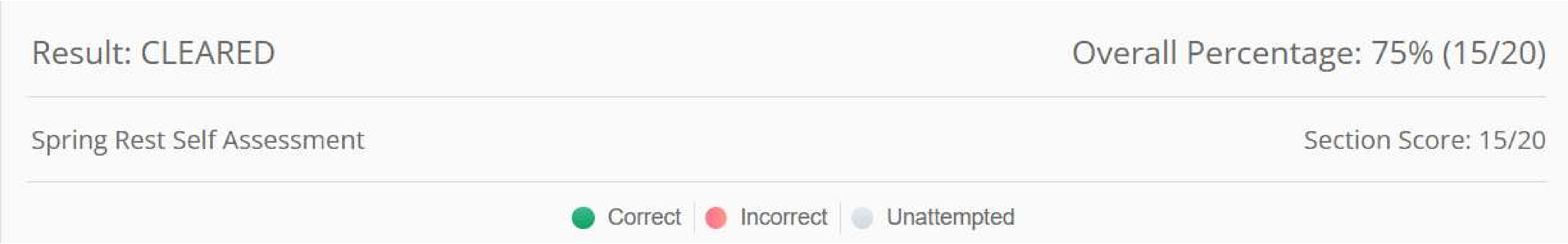
✓

Thank you. Your test is submitted successfully.

You have cleared this assessment.

Obtained Percentage	Obtained Marks
75 %	15 / 20

Best Attempt Score:75 % on 15-05-2024



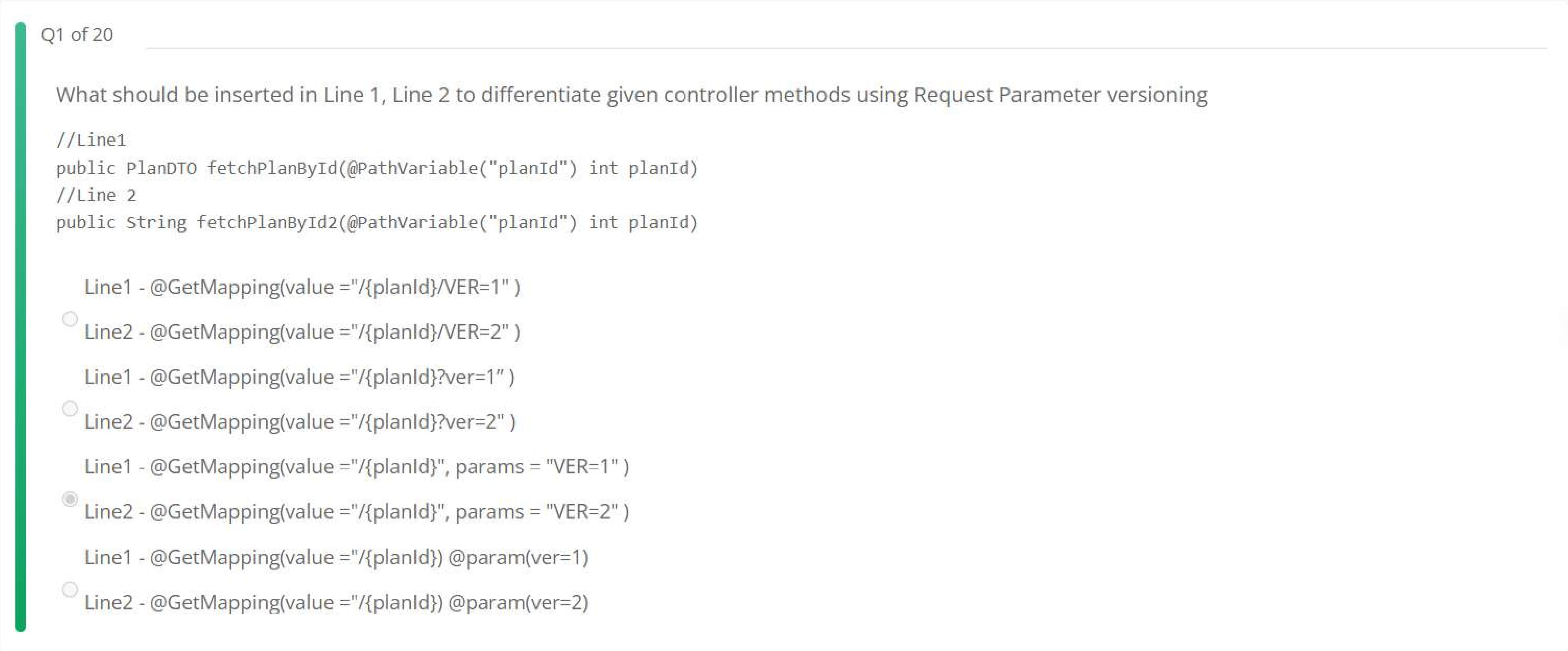
Result: CLEARED

Overall Percentage: 75% (15/20)

Spring Rest Self Assessment

Section Score: 15/20

● Correct | ● Incorrect | ● Unattempted



Q1 of 20

What should be inserted in Line 1, Line 2 to differentiate given controller methods using Request Parameter versioning

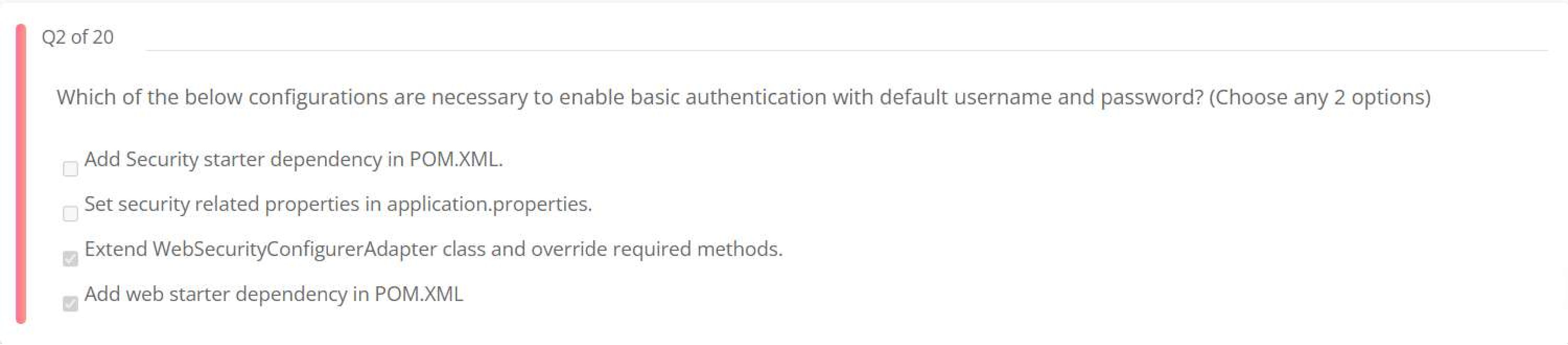
```
//Line1
public PlanDTO fetchPlanById(@PathVariable("planId") int planId)
//Line 2
public String fetchPlanById2(@PathVariable("planId") int planId)
```

○ Line1 - @GetMapping(value ="/{planId}/VER=1")
Line2 - @GetMapping(value ="/{planId}/VER=2")

○ Line1 - @GetMapping(value ="/{planId}?ver=1”)
Line2 - @GetMapping(value ="/{planId}?ver=2")

● Line1 - @GetMapping(value ="/{planId}", params = "VER=1")
Line2 - @GetMapping(value ="/{planId}", params = "VER=2")

○ Line1 - @GetMapping(value ="/{planId}) @param(ver=1)
Line2 - @GetMapping(value ="/{planId}) @param(ver=2)



Q2 of 20

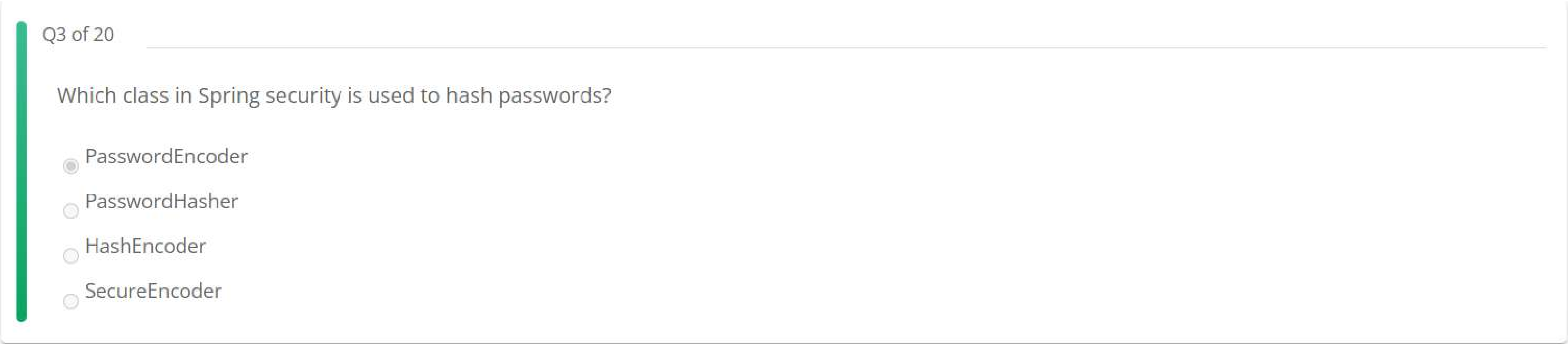
Which of the below configurations are necessary to enable basic authentication with default username and password? (Choose any 2 options)

☐ Add Security starter dependency in POM.XML.

☐ Set security related properties in application.properties.

☑ Extend WebSecurityConfigurerAdapter class and override required methods.

☑ Add web starter dependency in POM.XML



Q3 of 20

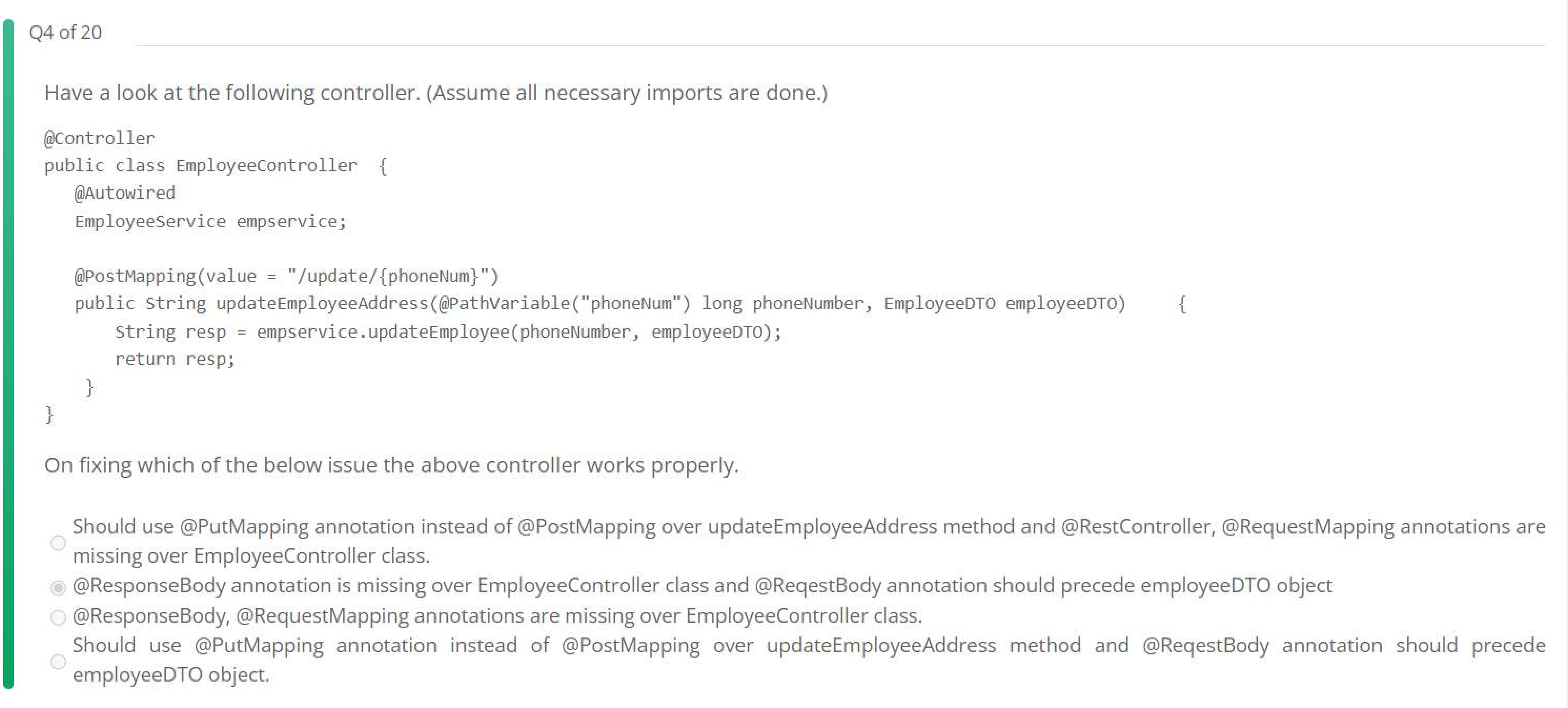
Which class in Spring security is used to hash passwords?

● PasswordEncoder

○ PasswordHasher

○ HashEncoder

○ SecureEncoder



Q4 of 20

Have a look at the following controller. (Assume all necessary imports are done.)

```
@Controller
public class EmployeeController  {
   @Autowired
   EmployeeService empservice;

   @PostMapping(value = "/update/{phoneNum}")
   public String updateEmployeeAddress(@PathVariable("phoneNum") long phoneNumber, EmployeeDTO employeeDTO)     {
       String resp = empservice.updateEmployee(phoneNumber, employeeDTO);
       return resp;
    }
}
```

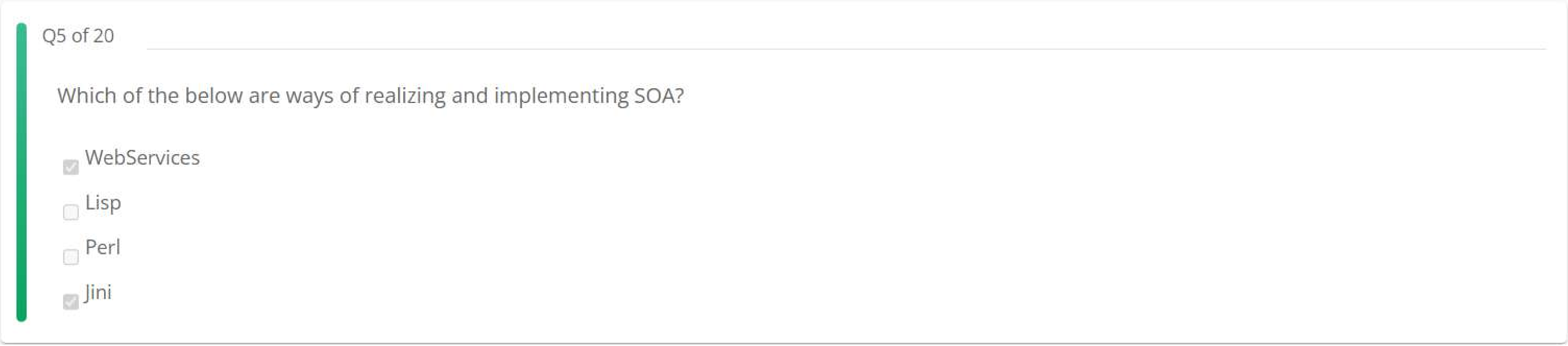
On fixing which of the below issue the above controller works properly.

○ Should use @PutMapping annotation instead of @PostMapping over updateEmployeeAddress method and @RestController, @RequestMapping annotations are missing over EmployeeController class.

● @ResponseBody annotation is missing over EmployeeController class and @ReqestBody annotation should precede employeeDTO object

○ @ResponseBody, @RequestMapping annotations are missing over EmployeeController class.

○ Should use @PutMapping annotation instead of @PostMapping over updateEmployeeAddress method and @ReqestBody annotation should precede employeeDTO object.



Q5 of 20

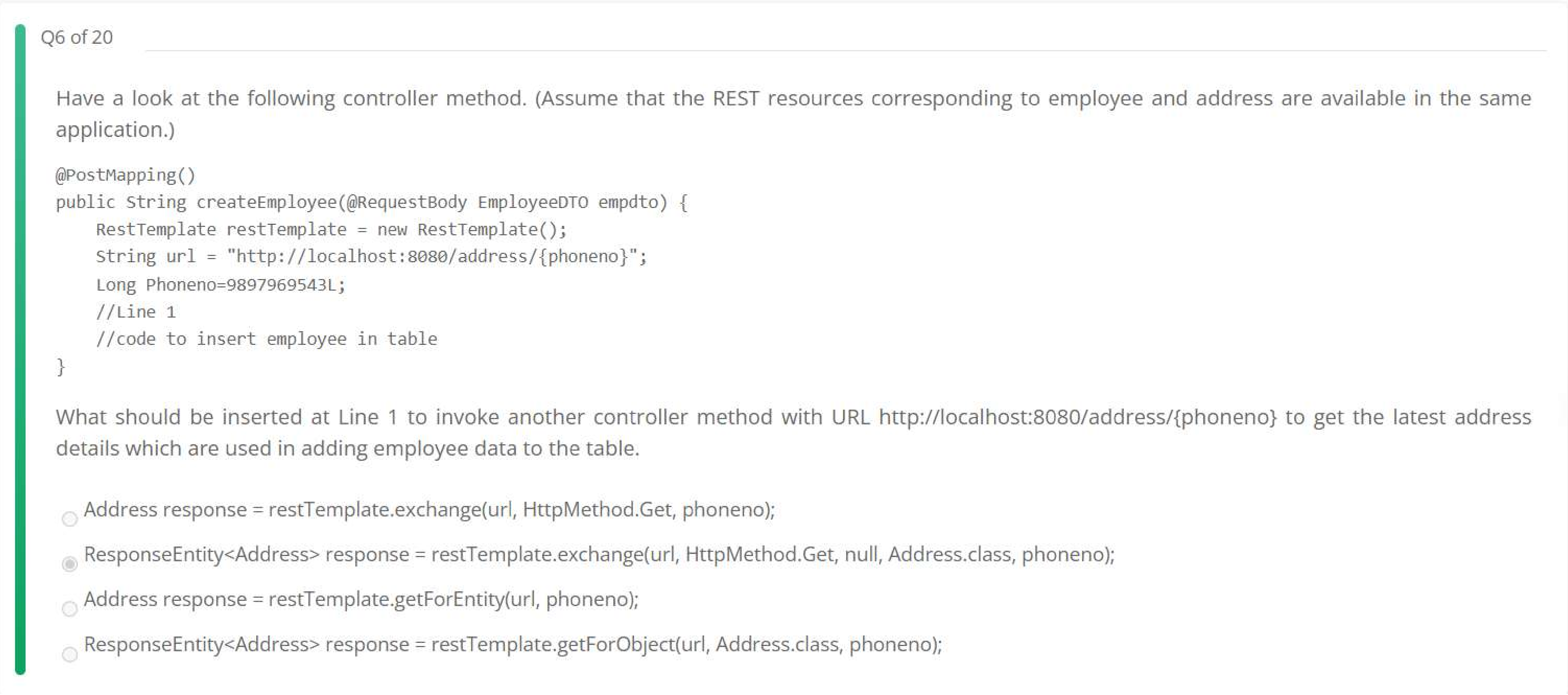
Which of the below are ways of realizing and implementing SOA?

☑ WebServices

□ Lisp

□ Perl

☑ Jini



Q6 of 20

Have a look at the following controller method. (Assume that the REST resources corresponding to employee and address are available in the same application.)

```
@PostMapping()
public String createEmployee(@RequestBody EmployeeDTO empdto) {
    RestTemplate restTemplate = new RestTemplate();
    String url = "http://localhost:8080/address/{phoneno}";
    Long Phoneno=9897969543L;
    //Line 1
    //code to insert employee in table
}
```

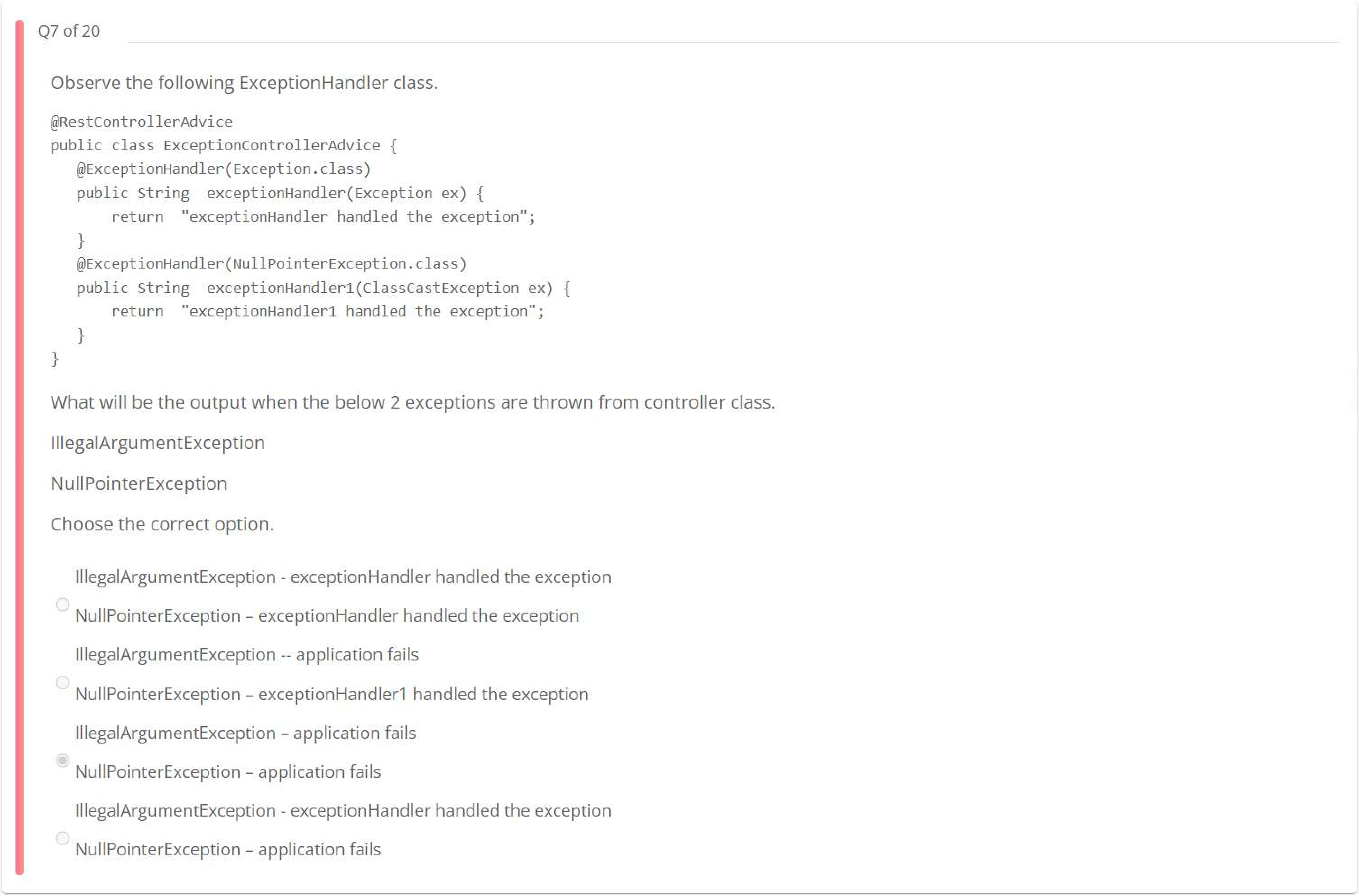
What should be inserted at Line 1 to invoke another controller method with URL http://localhost:8080/address/{phoneno} to get the latest address details which are used in adding employee data to the table.

○ Address response = restTemplate.exchange(url, HttpMethod.Get, phoneno);

● ResponseEntity<Address> response = restTemplate.exchange(url, HttpMethod.Get, null, Address.class, phoneno);

○ Address response = restTemplate.getForEntity(url, phoneno);

○ ResponseEntity<Address> response = restTemplate.getForObject(url, Address.class, phoneno);



Q7 of 20

Observe the following ExceptionHandler class.

```
@RestControllerAdvice
public class ExceptionControllerAdvice {
   @ExceptionHandler(Exception.class)
   public String  exceptionHandler(Exception ex) {
       return  "exceptionHandler handled the exception";
   }
   @ExceptionHandler(NullPointerException.class)
   public String  exceptionHandler1(ClassCastException ex) {
       return  "exceptionHandler1 handled the exception";
   }
}
```

What will be the output when the below 2 exceptions are thrown from controller class.

IllegalArgumentException

NullPointerException

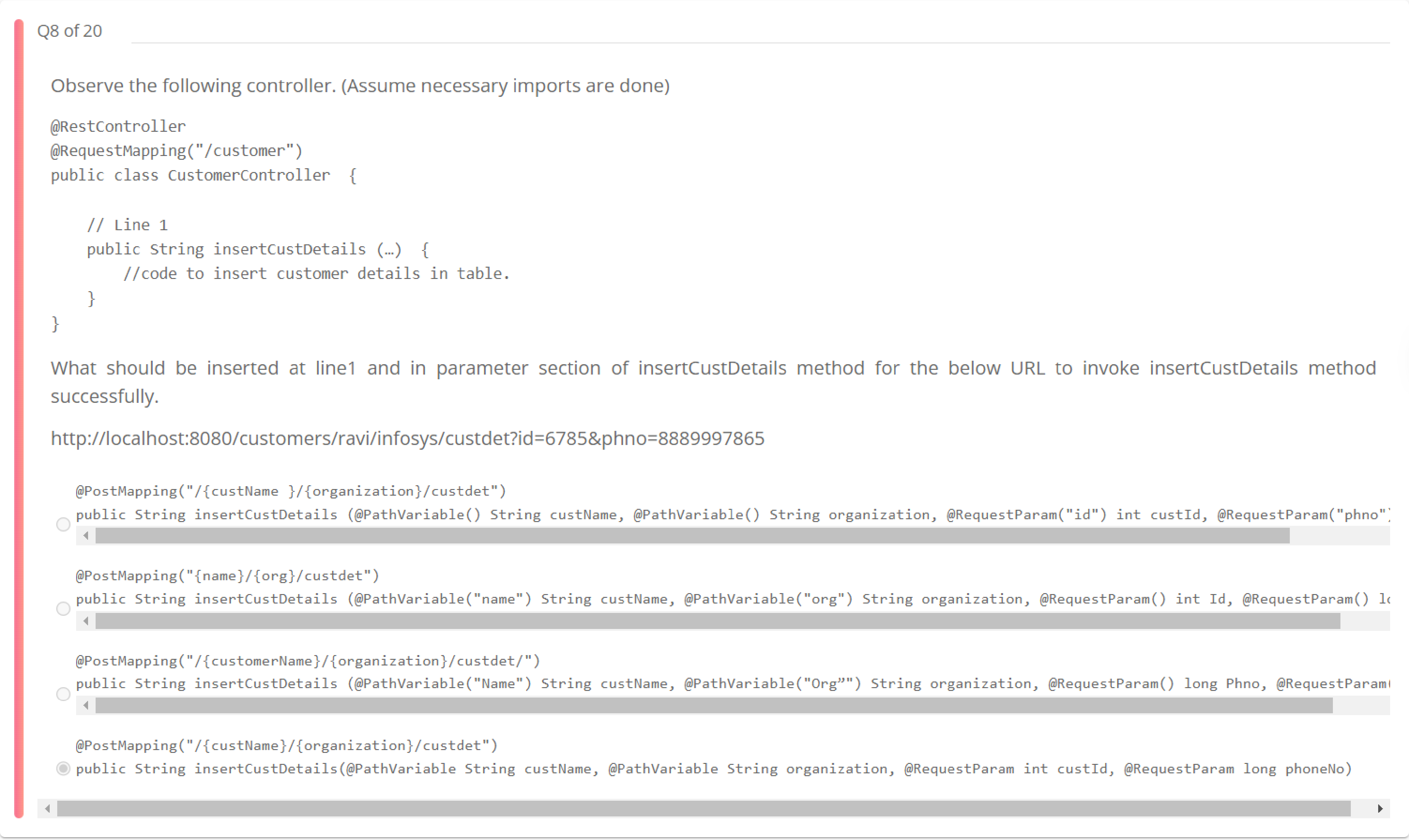
Choose the correct option.

○ IllegalArgumentException - exceptionHandler handled the exception
NullPointerException – exceptionHandler handled the exception

○ IllegalArgumentException -- application fails
NullPointerException – exceptionHandler1 handled the exception

● IllegalArgumentException – application fails
NullPointerException – application fails

○ IllegalArgumentException - exceptionHandler handled the exception
NullPointerException – application fails



Q8 of 20

Observe the following controller. (Assume necessary imports are done)

```
@RestController
@RequestMapping("/customer")
public class CustomerController  {

    // Line 1
    public String insertCustDetails (…)  {
        //code to insert customer details in table.
    }
}
```

What should be inserted at line1 and in parameter section of insertCustDetails method for the below URL to invoke insertCustDetails method successfully.

http://localhost:8080/customers/ravi/infosys/custdet?id=6785&phno=8889997865

○
```
@PostMapping("/{custName }/{organization}/custdet")
public String insertCustDetails (@PathVariable() String custName, @PathVariable() String organization, @RequestParam("id") int custId, @RequestParam("phno")
```

○
```
@PostMapping("{name}/{org}/custdet")
public String insertCustDetails (@PathVariable("name") String custName, @PathVariable("org") String organization, @RequestParam() int Id, @RequestParam() l
```

○
```
@PostMapping("/{customerName}/{organization}/custdet/")
public String insertCustDetails (@PathVariable("Name") String custName, @PathVariable("Org"") String organization, @RequestParam() long Phno, @RequestParam
```

●
```
@PostMapping("/{custName}/{organization}/custdet")
public String insertCustDetails(@PathVariable String custName, @PathVariable String organization, @RequestParam int custId, @RequestParam long phoneNo)
```