**Spring Boot**

* *ResponseEntity* **represents the whole HTTP response: status code, headers, and body**. Because of it, we can use it to fully configure the HTTP response.
* @GetMapping(**"/age"**)
* ResponseEntity<String> age(
* @RequestParam(**"yearOfBirth"**) **int** yearOfBirth) {
* **if** (isInFuture(yearOfBirth)) {
* **return** **new** ResponseEntity<>(
* **"Year of birth cannot be in the future"**,
* HttpStatus.BAD\_REQUEST);
* }
* **return** **new** ResponseEntity<>(
* **"Your age is "** + calculateAge(yearOfBirth),
* HttpStatus.OK);
* **Additonally we can set Http Headers**
* @GetMapping(**"/customHeader"**)
* ResponseEntity<String> customHeader() {
* HttpHeaders headers = **new** HttpHeaders();
* headers.add(**"Custom-Header"**, **"foo"**);
* **return** **new** ResponseEntity<>(
* **"Custom header set"**, headers, HttpStatus.OK);
* }
* ResponseEntity<T> - Reference
* <https://stackoverflow.com/questions/26549379/when-use-responseentityt-and-restcontroller-for-spring-restful-applications>
* **Rest Service call from one service to another service**
* <https://howtodoinjava.com/spring-boot2/resttemplate/spring-restful-client-resttemplate-example/>
* <https://www.tutorialspoint.com/spring_boot/spring_boot_rest_template.htm>
* <https://www.baeldung.com/rest-template>
* Rest Template is used to create applications that consume RESTful Web Services.
* You can **use** the **exchange()** method to **consume the web services** for all HTTP methods.
* @GetMapping("/service1")
* public ResponseEntity<ResponseModel> showResult(){
* ResponseModel response = new ResponseModel();
* //response.setMessage("Welcomee");
* HttpHeaders headers = new HttpHeaders();
* headers.setAccept(Arrays.asList(MediaType.APPLICATION\_JSON));
* HttpEntity <ResponseModel> entity = new HttpEntity<ResponseModel>(headers);
* response = restTemplate.exchange("http://localhost:8081/api/employees/service2", HttpMethod.GET, entity, ResponseModel.class).getBody();
* return new ResponseEntity<ResponseModel>(response, HttpStatus.OK);
* }
* Where Use **HttpEntity** to **wrap the request object.**