**import** org.junit.Test;  
  
**public class** Main<TestRequest> {  
 **private** Object **TestRequest**;  
}  
 @Test  
 @WithUserDetails(value = **"doctor"**)  
 **public void** calling\_assignForConsultation\_with\_valid\_test\_request\_id\_should\_update\_the\_request\_status(){  
  
 getTestRequestByStatus(RequestStatus.***LAB\_TEST\_COMPLETED***);  
  
 *//Implement this method  
  
 //Create another object of the TestRequest method and explicitly assign this object for Consultation using assignForConsultation() method  
 // from consultationController class. Pass the request id of testRequest object.  
  
 //Use assertThat() methods to perform the following two comparisons  
 // 1. the request ids of both the objects created should be same  
 // 2. the status of the second object should be equal to 'DIAGNOSIS\_IN\_PROCESS'  
 // make use of assertNotNull() method to make sure that the consultation value of second object is not null  
 // use getConsultation() method to get the lab result* **public** <**TestRequest**> **TestRequest** Object status;  
 getTestRequestByStatus(RequestStatus status){  
 Object testRequestQueryService = **new** Object();  
 **return** testRequestQueryService.findBy(status).stream().findFirst().get();  
  
 Object testRequestQueryService =**null**;  
 **if** (testRequestQueryService != **null**) {  
 **return** testRequestQueryService.findByTester(RequestStatus.***INITIATED***);  
 }  
  
  
 }  
  
 }  
  
 **private void** getTestRequestByStatus(RequestStatus labTestCompleted) {  
 }  
  
  
}  
  
 **private class**{TestRequest getTestRequestByStatus(RequestStatus status){  
 **return** testRequestQueryService.findBy(status).stream().findFirst().get();  
 }  
 }  
  
 @Test  
 @WithUserDetails(value = **"doctor"**)  
 **public void** calling\_assignForConsultation\_with\_valid\_test\_request\_id\_should\_throw\_exception(){  
  
 Long InvalidRequestId= -34L;  
  
 *//Implement this method  
  
  
 // Create an object of ResponseStatusException . Use assertThrows() method and pass assignForConsultation() method  
 // of consultationController with InvalidRequestId as Id  
  
  
 //Use assertThat() method to perform the following comparison  
 // the exception message should be contain the string "Invalid ID"* TestRequest testRequest = Object userLoggedInService = **new** Object();  
 User tester = userLoggedInService.getLoggedInUser();  
  
 Object testRequestUpdateService = **null**;  
 Object id = **new** Object();  
 **assert** testRequestUpdateService != **null**;  
 **return** testRequestUpdateService.assignForLabTest(id,tester);  
 getTestRequestByStatus(RequestStatus.LAB\_TEST\_IN\_PROGRESS);  
  
  
  
  
  
  
 }  
  
  
  
  
  
 }  
  
 @Test  
 @WithUserDetails(value = **"doctor"**)  
 **public void** calling\_updateConsultation\_with\_valid\_test\_request\_id\_should\_update\_the\_request\_status\_and\_update\_consultation\_details(){  
  
 TestRequest testRequest = getTestRequestByStatus(RequestStatus.DIAGNOSIS\_IN\_PROCESS);  
  
 *//Implement this method  
 //Create an object of CreateConsultationRequest and call getCreateConsultationRequest() to create the object. Pass the above created object as the parameter  
  
 //Create another object of the TestRequest method and explicitly update the status of this object  
 // to be 'COMPLETED'. Make use of updateConsultation() method from labRequestController class (Pass the previously created two objects as parameters)  
  
 //Use assertThat() methods to perform the following three comparisons  
 // 1. the request ids of both the objects created should be same  
 // 2. the status of the second object should be equal to 'COMPLETED'  
 // 3. the suggestion of both the objects created should be same. Make use of getSuggestion() method to get the results.* Object userLoggedInService = **null**;  
 User tester = userLoggedInService.getLoggedInUser();  
  
 Object testRequestUpdateService = **null**;  
 Object createLabResult = **null**;  
  
 Object id = **null**;  
 **return** testRequestUpdateService.updateLabTest(id,createLabResult,tester);  
  
 }  
  
  
 @Test  
 @WithUserDetails(value = **"doctor"**)  
 **public void** calling\_updateConsultation\_with\_invalid\_test\_request\_id\_should\_throw\_exception(){  
  
 TestRequest testRequest = getTestRequestByStatus(RequestStatus.DIAGNOSIS\_IN\_PROCESS);  
  
 *//Implement this method  
  
 //Create an object of CreateConsultationRequest and call getCreateConsultationRequest() to create the object. Pass the above created object as the parameter  
  
 // Create an object of ResponseStatusException . Use assertThrows() method and pass updateConsultation() method  
 // of consultationController with a negative long value as Id and the above created object as second parameter  
 //Refer to the TestRequestControllerTest to check how to use assertThrows() method  
  
  
 //Use assertThat() method to perform the following comparison  
 // the exception message should be contain the string "Invalid ID"* Object userLoggedInService = **null**;  
 User tester = userLoggedInService.getLoggedInUser();  
  
 Object testRequestUpdateService = **null**;  
 Object createLabResult = **null**;  
 Object id = **null**;  
 **return** testRequestUpdateService.updateLabTest(id,createLabResult,tester);  
  
 **throw new** ResponseStatusException(HttpStatus.NOT\_IMPLEMENTED,**"Not implemented"**);  
  
 }  
  
  
 }  
  
 @Test  
 @WithUserDetails(value = **"doctor"**)  
 **public void** calling\_updateConsultation\_with\_invalid\_empty\_status\_should\_throw\_exception(){  
  
 TestRequest testRequest = getTestRequestByStatus(RequestStatus.DIAGNOSIS\_IN\_PROCESS);  
  
 *//Implement this method  
  
 //Create an object of CreateConsultationRequest and call getCreateConsultationRequest() to create the object. Pass the above created object as the parameter  
 // Set the suggestion of the above created object to null.  
  
 // Create an object of ResponseStatusException . Use assertThrows() method and pass updateConsultation() method  
 // of consultationController with request Id of the testRequest object and the above created object as second parameter  
 //Refer to the TestRequestControllerTest to check how to use assertThrows() method* Object userLoggedInService = **null**;  
 User tester = userLoggedInService.getLoggedInUser();  
  
 Object testRequestUpdateService = **null**;  
 Object createLabResult = **null**;  
 Object id = **null**;  
 **return** testRequestUpdateService.updateLabTest(id,createLabResult,tester);  
  
 **throw new** ResponseStatusException(HttpStatus.NOT\_IMPLEMENTED,**"ConstraintViolationException"**);  
  
  
  
 }  
  
 **public** CreateConsultationRequest getCreateConsultationRequest(TestRequest testRequest) {  
  
 *//Create an object of CreateLabResult and set all the values  
 // if the lab result test status is Positive, set the doctor suggestion as "HOME\_QUARANTINE" and comments accordingly  
 // else if the lab result status is Negative, set the doctor suggestion as "NO\_ISSUES" and comments as "Ok"  
 // Return the object* **return null**; *// Replace this line with your code* Object CreateLabResult = (Long Object id;  
 id, @Valid CreateLabResult Object createLabResult;  
 createLabResult, User Object tester;  
 tester);  
 Object newObject;  
 CreateLabResult createLabResult1 = (CreateLabResult) (CreateLabResult = newObject);  
 **return** createLabResult1;  
  
  
  
 **return null**; *// Replace this line with your code* }  
  
**public class** ResponseStatusException {  
 **public** ResponseStatusException(Object p0, String constraintViolationException) {  
  
  
 }  
  
}