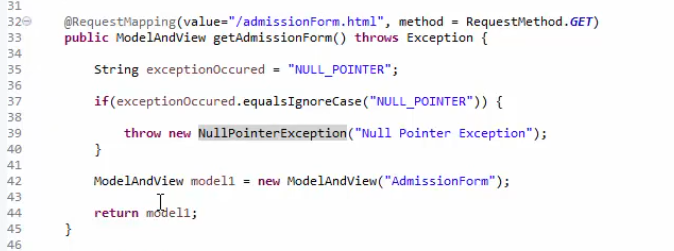
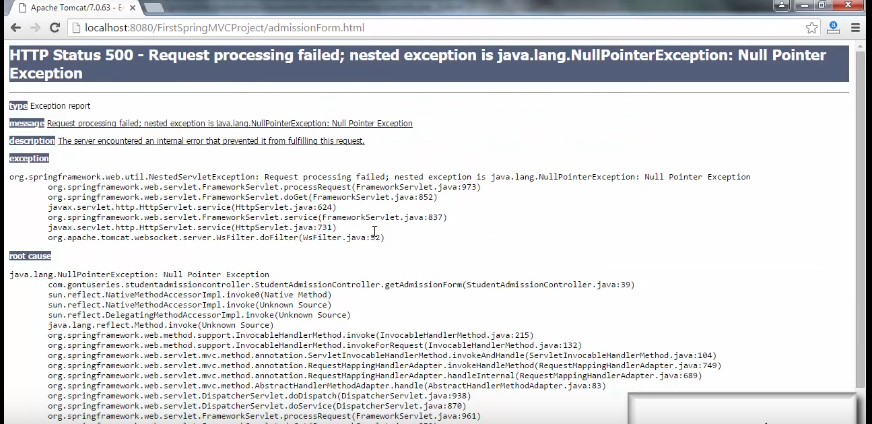
**Exception Handling in Spring** :

In Spring mvc web application If we get any exception(like NullPointerException) in Controller’s mapping method and if that exception is not handled then on calling that method by hitting url we get an error page as response .



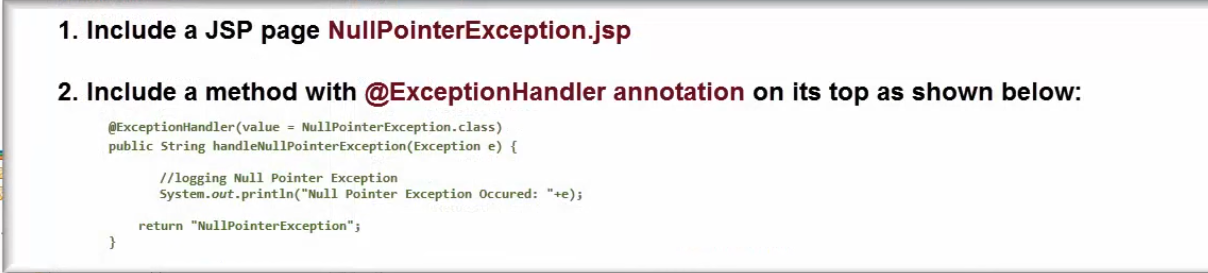
We get this response on hitting url :



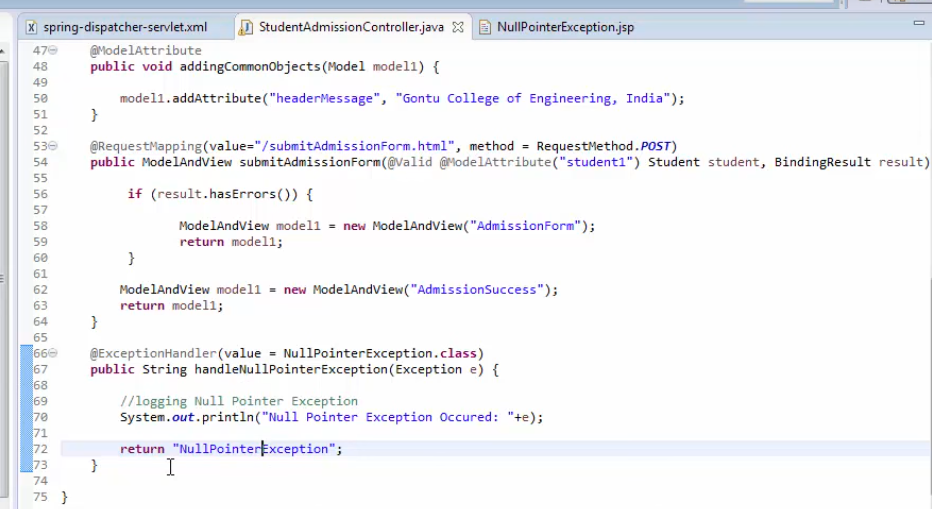
This error page is not user friendly so developer uses some customized exception handling mechanism so that instead of getting this unfriendly page user gets some other alternative meaningful customized webpage.

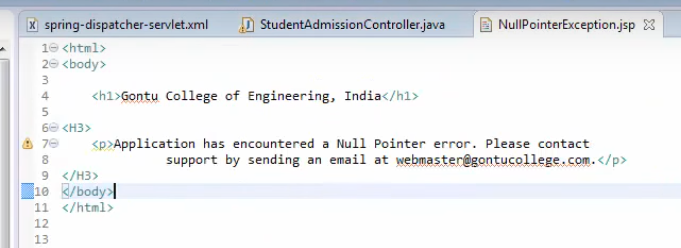
**To do so what are the task needs to be performed :**

**We need to do two simple steps to achieve this new web page:-**



We are adding one method in that controller **handleNullPointerException with @ExceptionHandler (value= NullPointerException.class )** this method will return **NullpOinterException.jsp**

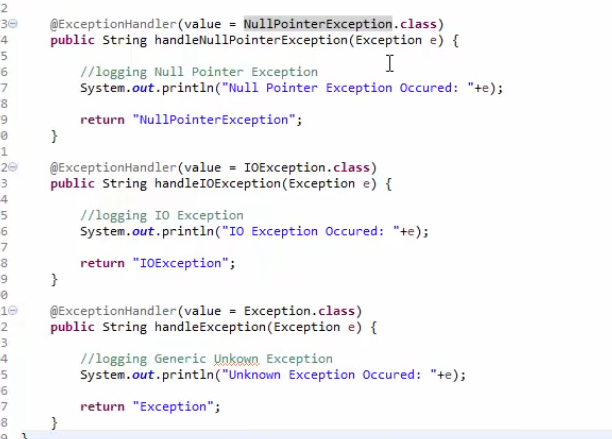




* On hitting url of method /admissionForm.html , it throws **NullPointerException** (as discussed in above code snippet) now Spring MVC comes into picture and finds the method with **@ExceptionHandler** with value **NullPointerException** and calls that method. This method returns **NullPointerException.jsp** with following details :



**Note:** As we have handled the Controller Class method for NullPointerException. If in any method in Controller class gets IOException we have to repeat the same things for the IOException .We add one more method with annotation **@ExceptionHandler( value = IOException.class)** and this method also returns a webpage IOException.jsp with some meaningful messages. But in a controller class there could be many exception so to handle that each type of exception in a single method we create a generic exception handling by creating a method with **@ExceptionHandler( value = Exception.class)** and this will return the **Exception.jsp** with proper message.

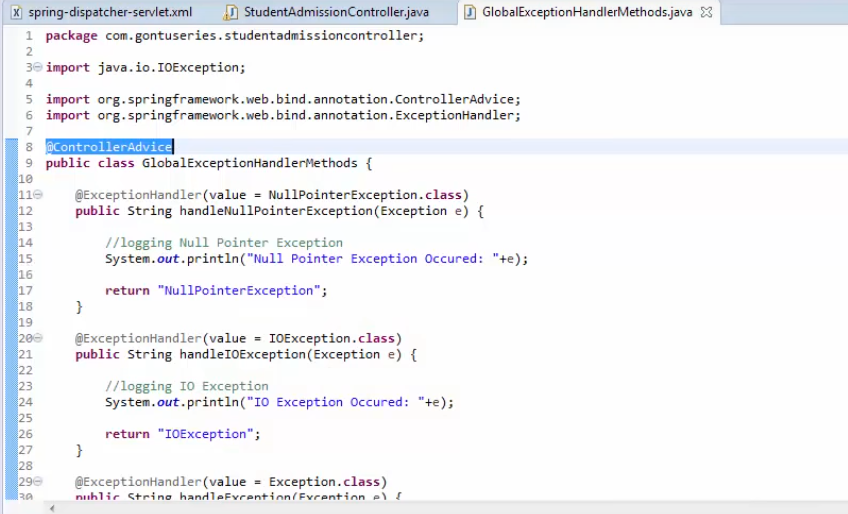


Exception handling methods are defined from most specific to generic.

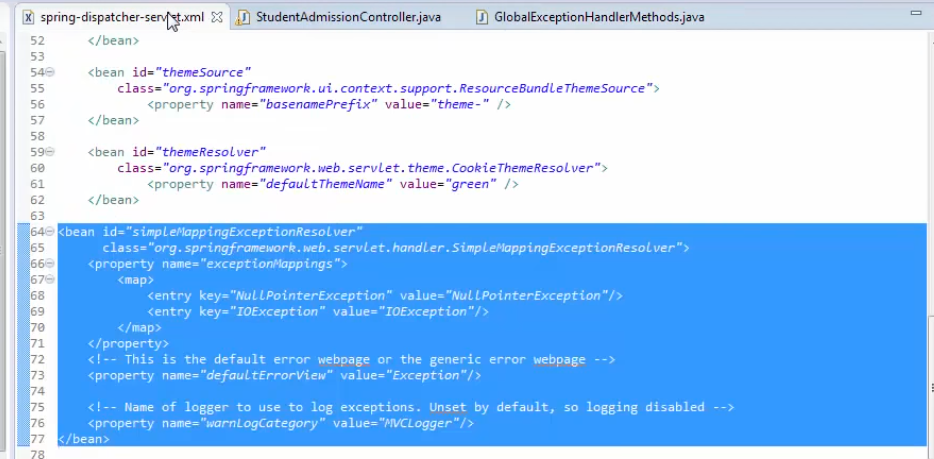
**Drawback of above approach:**

The only drawback of this approach is this @ExceptionHandler method is applicable for each Controller class level.so if our application has 10 controller class and the chances so we will have to add these all method in each controller. So we will have too much code redundancy.

**Alternative Solution : Spring Mvc suggests to create a single class with annotation @ControllerAdvice and put all the @ExceptionHandler methods under this class .this method is applicable for all the Controller class.**

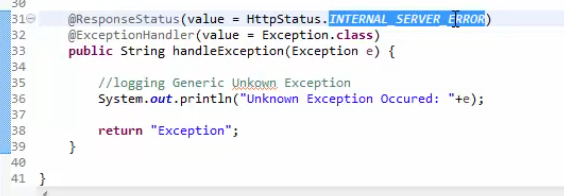


**Note:- The exact alternative way of above java code is by changing in the configuration file as shown below ----**



**How We get appropriate Status code on hitting for a url :**

**Normally there are some List of Status code and its exact meaning is mentioned below. But we can customize them by using @ResponseStatus annotation with the required value .**



**It means if we call hadleException(Exception e) method it’s returned status code would be 500.if we don’t put @ResponseStatus annotation it will return 200 status code which is wrong**

**Http Status code :**

