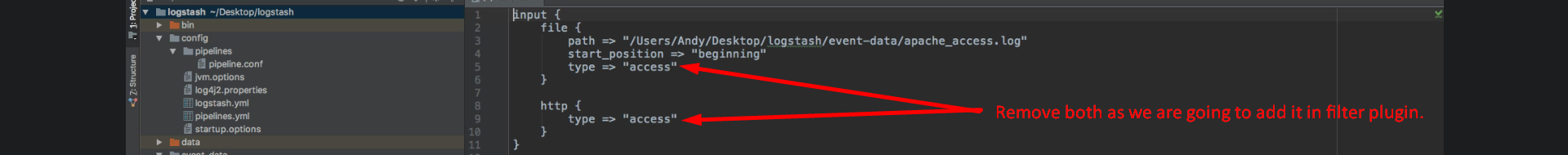
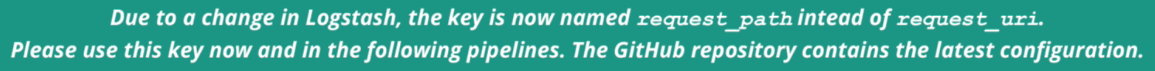
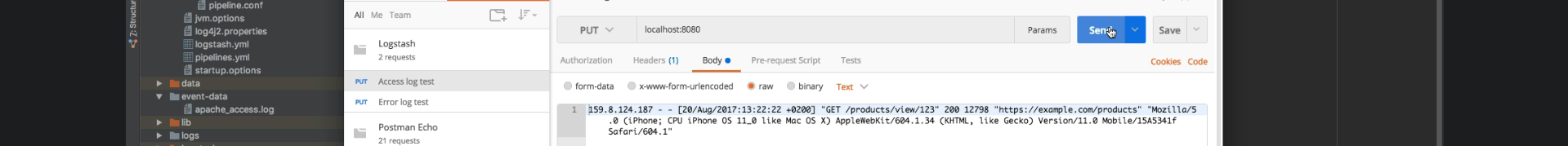
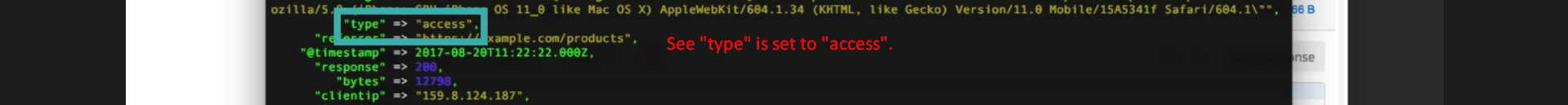
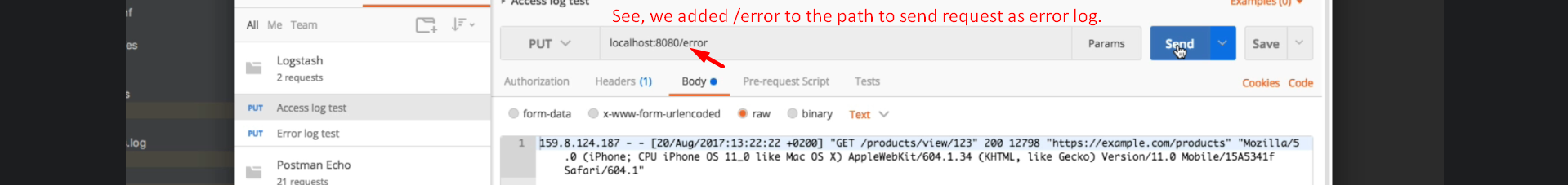
1. **Resource**:  
   <https://www.elastic.co/guide/en/logstash/current/glob-support.html>
2. 
3. Let’s gradually prepare a pipeline to handle both Access and Error log using conditional statements.
4. **Conditional statement can be used only at the root level of input, filter, and output blocks not within any plugins.**
5. The 1st step is to set the “type” field based on what kind of log we’re receiving either access or error.  
   For this, we need conditional statement to help us resolve which type we’re dealing with at runtime.
6. Since we are using the file and http input plugins, we can check the file name and request path.
7. Let’s get started by handling the two event types (Access log and Error Log) from the http input plugin.  
   And then get back to the file input afterwards.
8. As we already discussed, the conditional statements can be used at the root level of 3 parts (input, filter, output) of the pipeline.  
   This means that we can’t add conditional statement within the two input plugins like  
   A screen shot of a computer

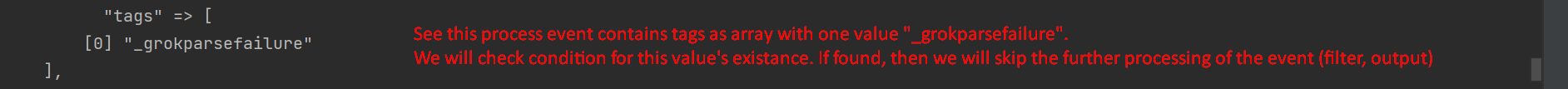
   Description automatically generated with medium confidence  
   Why we can’t set “type” in input block itself.  
   **There are two reasons**:
   1. Conditional statements can’t be used within plugins but at the root level.
   2. But the “type” field can’t be used at the root level of input block but within input plugin.  
      So, adding field “type” outside of input plugin is not allowed.
9. Basically, this leaves us two choices.
   1. Setting “type” field in the input plugin and then conditionally replace the value within the filter object or
   2. Just add the “type” field in the filter object conditional. (We will do this)
10. Let’s 1st remove the “type” from the input block from both plugins file and http.  
    
    1. Mentor will use **“request\_uri”** but we need to use **“request\_path”** which is change in Logstash.  
         
       A picture containing text, screenshot, multimedia software, graphics software

       Description automatically generated
11. The following is the final pipeline configuration:  
    A screenshot of a computer program

    Description automatically generated with medium confidence
12. Let’s see this in action now.
13. Sending 1st Access log (We will come back to error log to see how to send error log)  
    **Output**:  
      
      
      
    Sending Error Log:  
      
    **Output**:  
    
14. So far so good. But we still have work to do.  
    Our conditional statement works only for events sent to the HTTP input right now.  
    That is great as long as we’re testing but eventually we will want to switch over to using file input.
15. Before adjusting the conditional statement, we need to adjust the path, the file input is watching.  
    As right now, it is watching only a single file.  
    What we can do is to add wild card in the path to make the file name flexible with the purpose of watching both access and error logs at the same time.  
    A screen shot of a computer

    Description automatically generated with medium confidence
16. **GLOB Patterns Example**s:  
    A picture containing text, font, screenshot

    Description automatically generated
17. Handling the type of log from the file input plugin.  
    A screenshot of a computer program

    Description automatically generated with medium confidence
18.   
      
    A picture containing text, screenshot, font

    Description automatically generated  
      
    A screenshot of a computer

    Description automatically generated with medium confidence  
      
    