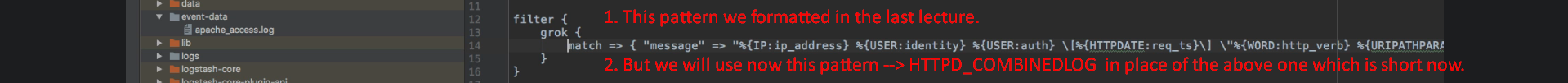
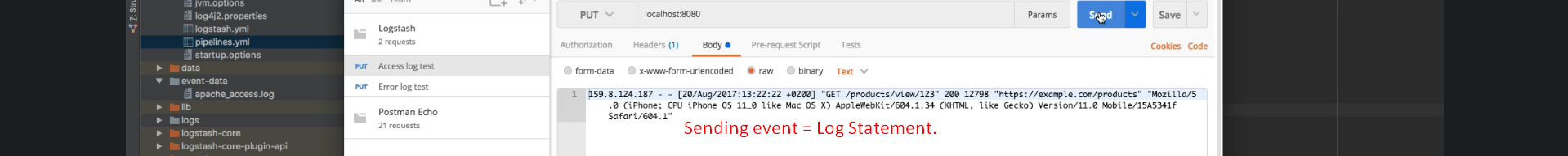
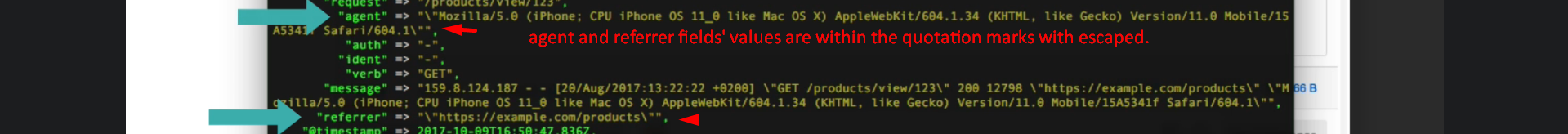
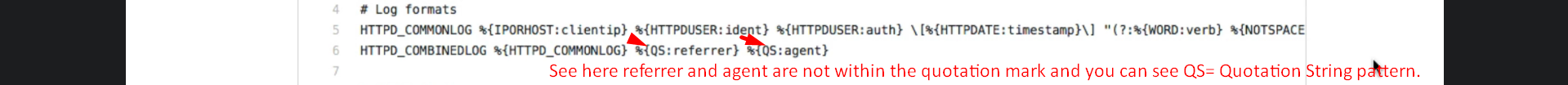
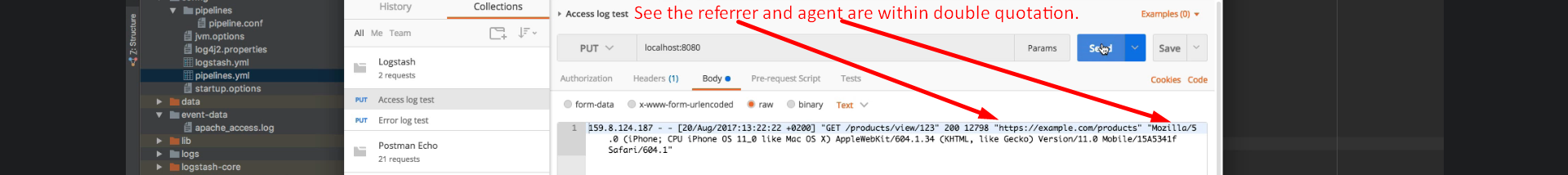
1. 
2. Now we know the basics of GROK Pattern.
3. Let’s finish our filter before moving on to doing some other processing.
4. Until now, we were matching the various parts of the Apache Access Log Request by using GROK Patterns.
5. This was the great way of learning the basics of the GROK but it takes quite a bit of typing.
6. Fortunately, there is an easy way because passing a Apache Web Logs is such a common thing with Logstash.
7. There is actually, a predefined GROK Pattern for this.   
   At least when using a popular Log Formats.
8. Open this Github Link: Http**d** : D for **Daemon**.  
   <https://github.com/logstash-plugins/logstash-patterns-core/blob/main/patterns/legacy/httpd>  
   A picture containing text, font, receipt, screenshot

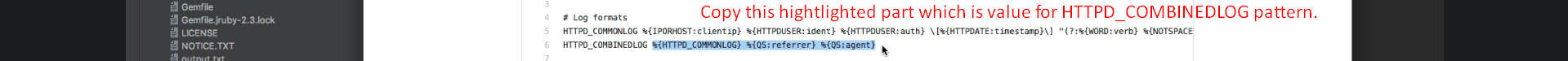
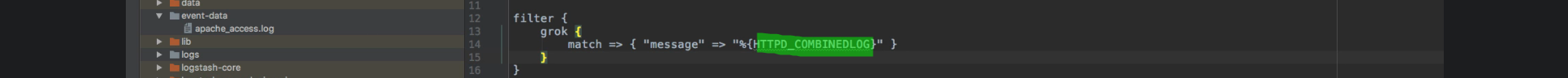
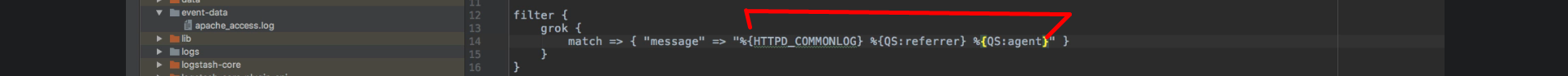
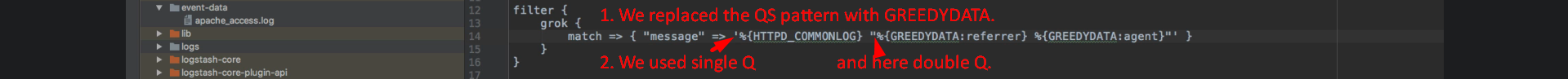
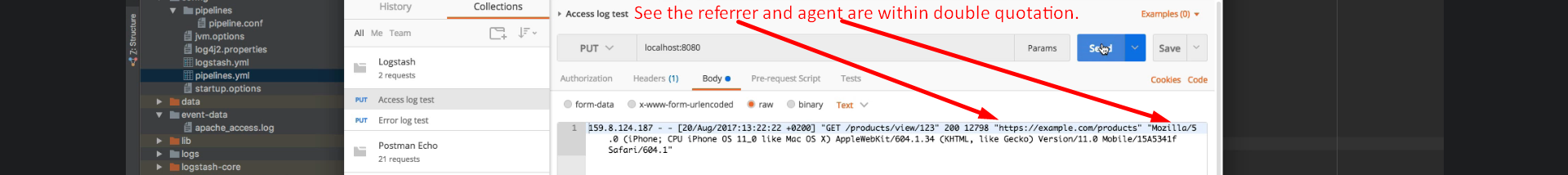
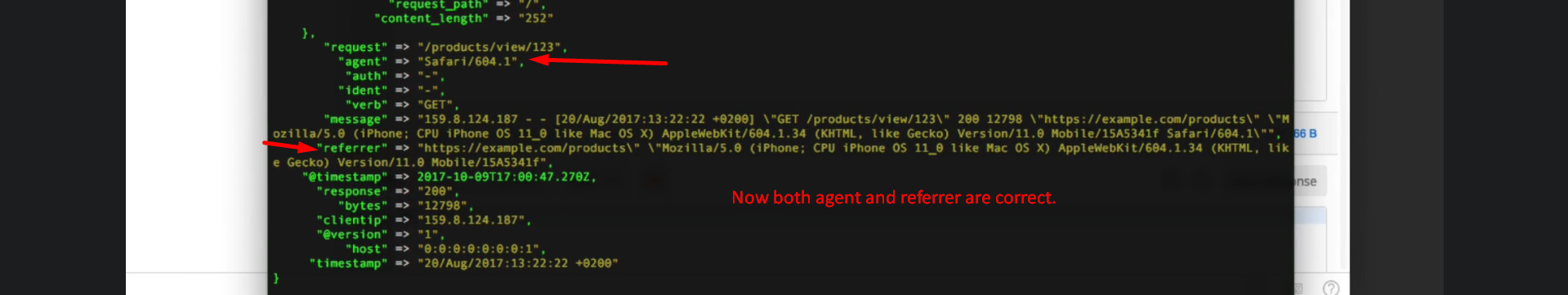
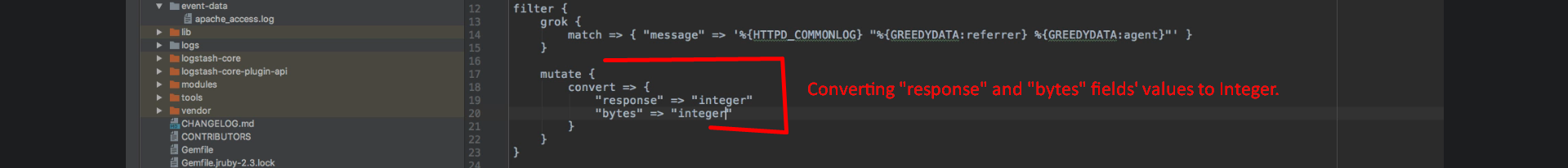
   Description automatically generated
9. This file httpd contains the patterns for Apache Web Server.
10. Here we have two patterns of our interest one for each popular Apache log format.
    1. **Common Log Format**:
    2. **Combined Log Format**:
       1. **Difference**: Same as Common along with including referrer and User Agent HTTP headers.
       2. We will be using this GROK Pattern throughout the course.
11. 
12. A screen shot of a computer

    Description automatically generated with medium confidence
13. 
14. Let’s look at the processed event that was output within the terminal, we can see that the request was processed correctly by the Logstash.  
    A screenshot of a computer

    Description automatically generated with medium confidence
15. We have a number of fields corresponding various parts of the request = event (sent by postman) including **referrer and user agent**.
16. But we have a problem though
    1. But both the referrer and user fields’ values begin and end with double quotation marks.  
       

1. Let’s look at the definition of the pattern 🡺 HTTPD\_COMBINEDLOG  
   Before that notice that referrer and user agent that we are passing in the POSTMAN is within the quotation marks.  
   Let us look at the definition of the pattern.  
     
   That means QS will also capture the quotation marks.
2. There are two ways to solve it (We will look at both).
   1. Modifying the Pattern.
   2. Replacing the quotation marks within the fields.
3. **Replacing the Quotation Mark First**:
   1. We can use Mutate plugin in filter with an option called GSUB.
   2. Mutate plugin is used to convert to one data type to another.
   3. **Pipeline Configuration**:  
      A screenshot of a computer program

      Description automatically generated with medium confidence
   4. **POSTMAN Event Request**: 
   5. **Processed Event on Console:**A screenshot of a computer program

      Description automatically generated with medium confidence
4. **2nd Way 🡺 Modifying the Pattern**:
   1. Instead of using HTTPD\_COMBINEDLOG GROK pattern, we will use our own and include the quotation mark within it.
   2. 
   3. Earlier we used the following:  
        
      Now:  
        
      Just change the QS at two places.   
        
      After Changing the QS:  
      
   4. **The POSTMAN Request**:  
      
   5. **Console Output:**  
      
   6. **But note that fields “response” and “bytes” are no longer integer after using our custom GROK Pattern with the predefined pattern.**  
      
   7. **Pipeline Configuration**:  
      
   8. Now both fields “response” and “bytes” are integer.  
      