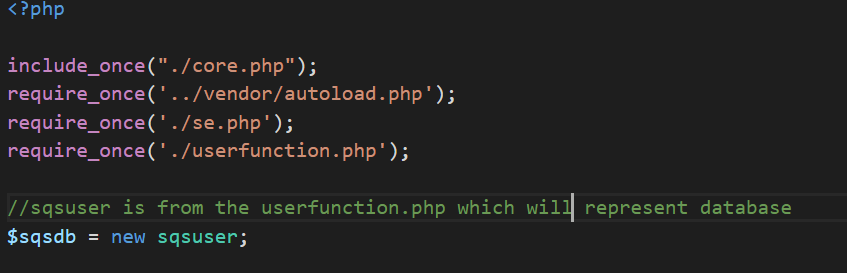
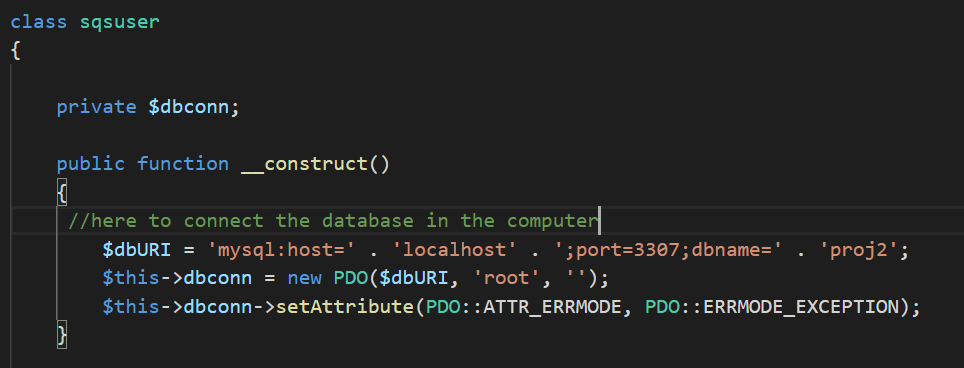
1. Denote where and explain why you instantiated the database and session objects in that location

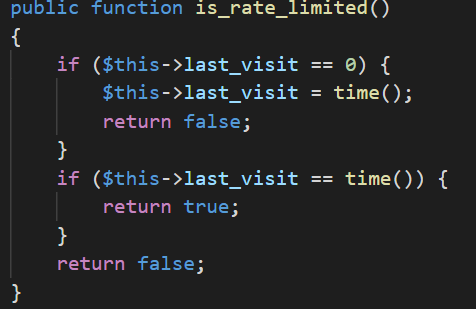
I use userfunction.php to connect the database ,and userfunction.php include the command functions which control all the data transport.

The reasons are in the code line **//**

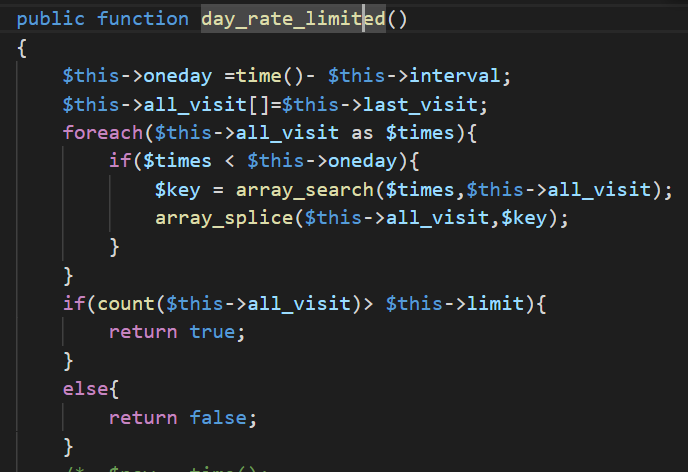
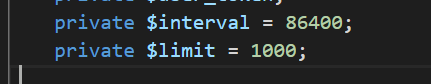




1. Explain the mathematics of either of your two rate limiting code.



is\_rate\_limited:it controls the per second per user. If last\_visit is 0 ,it will announce last\_visit to current time.If it is not 0 ,it will return false.Last\_visit should be 0 in the first time session,if it equal current time which is mean it already has a session ,and it can not have two session in one second.

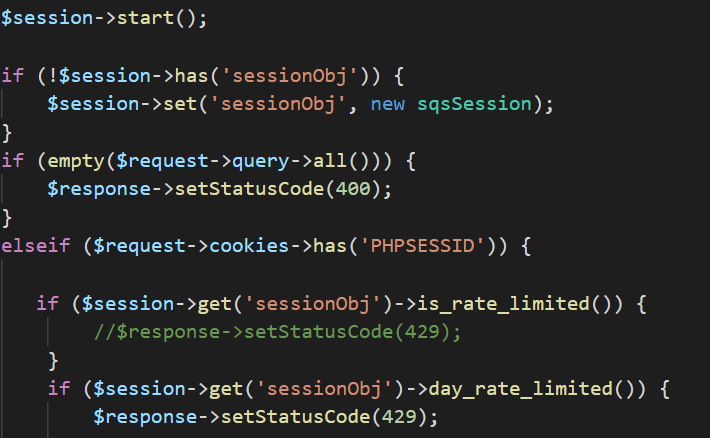


day\_rate\_limited: First of all, time() returns the current time when the user triggers this function,and oneday variable will record the yesterday current time which value is from current time decrease seconds in a day which are 86400 seconds.

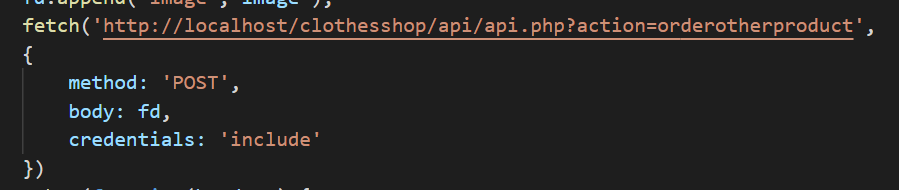
Second, the all\_visit array will record each last\_vist time from is\_rate\_limited function ,and if all\_visit time is smaller than oneday variable it means it is an error ,so the function will search this time data and remove it.

Third, if today's rate limit is over 1000 ,the function will return true to tell the user ,today's visit times are too many.

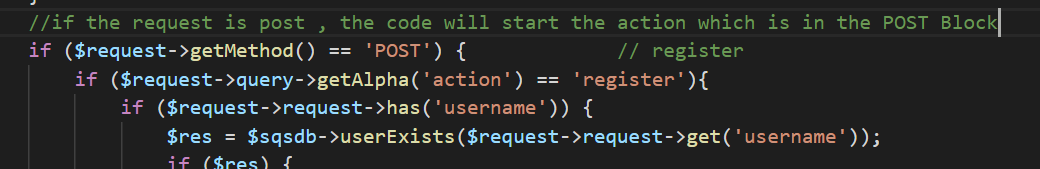
1. Note where you are checking if a session pre-exists, what are you doing if it does.

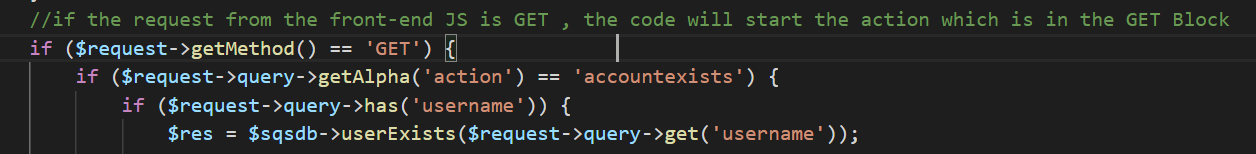
If a session already exists ,the code will skip the creating session part ,and start the api.php main code.

1. Explain the code structure that checks all of the GET/POST structures



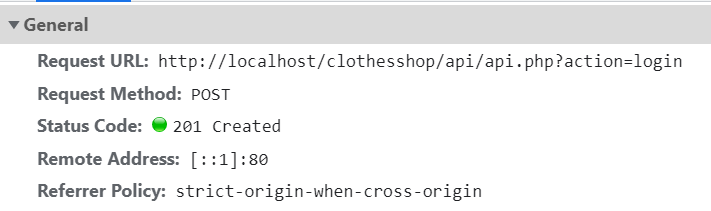
Javascript will send the fetch to the backend , and the backend will send the data to different functions depending on transfer method.



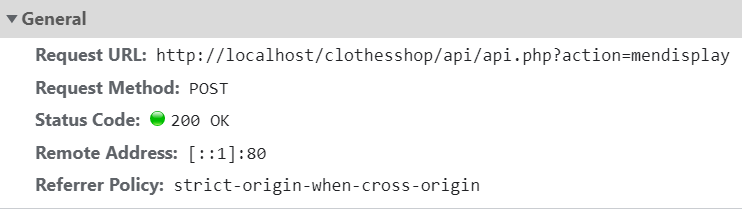


1. Write a README file that explains how to setup and configure Web Service
2. Write a test script that interacts with the web service to test all the known GET and POST requests as a part of Unit testing

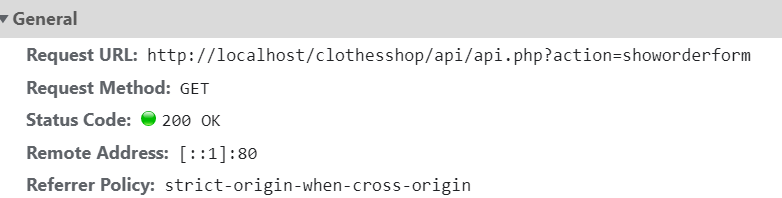
Login (POST)



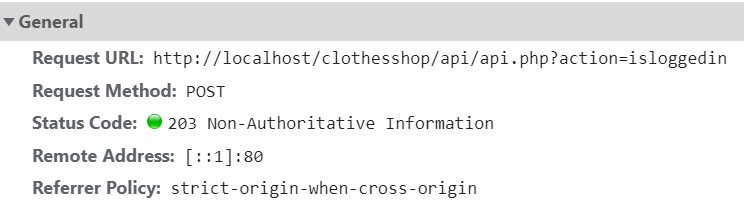
Display products(POST)



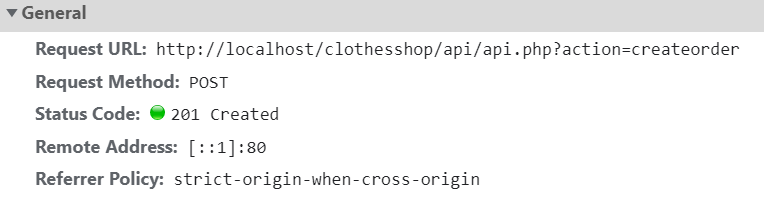
Show ordering products(GET)



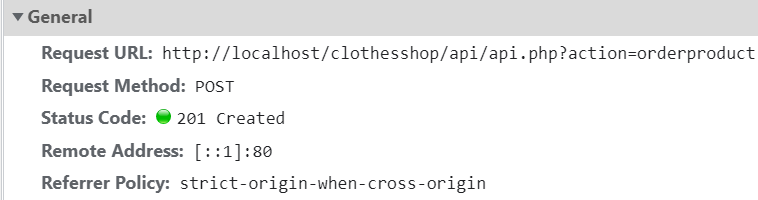
check loggin(POST)



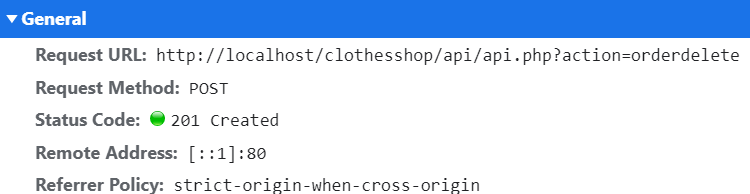
Create order (POST)



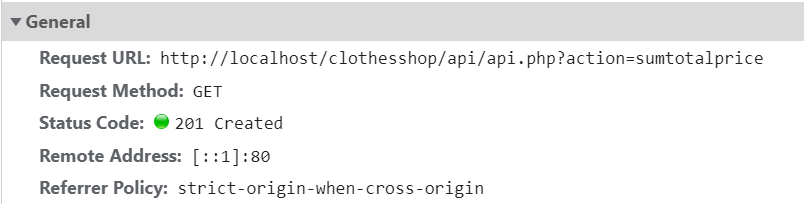
add product to order chart(POST)



delete order product (POST)

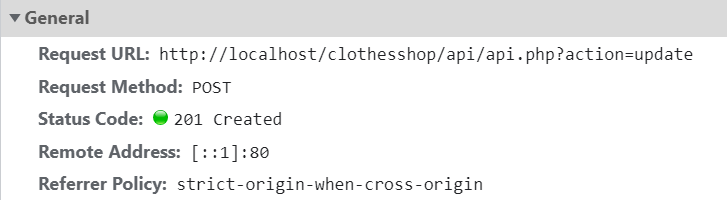


Confirm order and sumtotal(GET)



register which will create new user(POST)

Update user info(POST)



Logout(GET)