Stress Testing Chameleon Website

Logo

Description automatically generated

Contents

[What is stress testing? 3](#_Toc143884767)

[Tools Used 3](#_Toc143884768)

[Results – single page 3](#_Toc143884769)

[30 concurrent users for 5 minutes 3](#_Toc143884770)

[50 Concurrent users for 5 minutes 3](#_Toc143884771)

[100 Concurrent users for 5 Minutes 4](#_Toc143884772)

[300 concurrent users for 5 minutes 5](#_Toc143884773)

[Results – multiple pages 5](#_Toc143884774)

[30 Concurrent users for 5 minutes 6](#_Toc143884775)

[50 Concurrent users for 5 minutes 6](#_Toc143884776)

[100 Concurrent users for 5 minutes 6](#_Toc143884777)

[300 Concurrent users for 5 minutes 7](#_Toc143884778)

[Results Comparison 7](#_Toc143884779)

[Conclusion 8](#_Toc143884780)

# What is stress testing?

Stress testing is a method of evaluating the performance, stability, and reliability of a website. This method of evaluation will introduce a larger load then its generally expected to receive, This can be important to test as it allows us to see how a website will handle unexpected surges in user traffic and if there is any need to upgrade the current setup in order to avoid bottlenecking or to see how scalable the website needs to be. In the case of the Chameleon website, this test will serve as a method of comparison once the new website gets built, if the new website has similar performance to this current website then the suggested improvements towards the end of the document will be valid and less research will be required for the web development team.

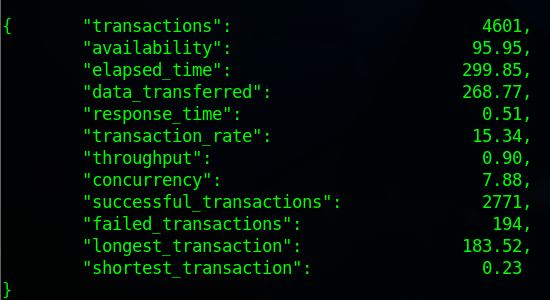
# Tools Used

* ParrotOS
* Siege
* ChatGPT

# Results – single page

## 30 concurrent users for 5 minutes





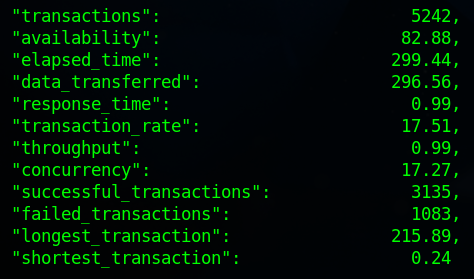
## 50 Concurrent users for 5 minutes





## 100 Concurrent users for 5 Minutes





## 300 concurrent users for 5 minutes

A computer screen shot of text

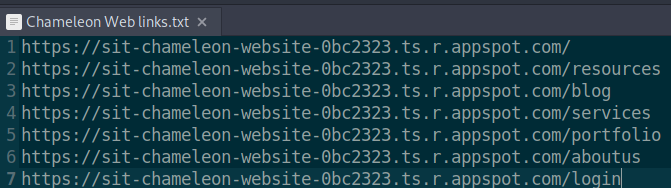
Description automatically generated

As we can see the more users that start using this website the more this website starts to fail. After checking with ChatGPT when at what point we should be worried regarding the availability dropping and normally anything below is 95% but since chameleon is a smaller organization, I gave it a bit of leeway and decided that anything below 80% availability isn’t ideal.

As we can see here with 300 users all on the home page it becomes quite slow when it comes to responding to the users.

# Results – multiple pages

Part of simulating an increased amount of traffic is making sure multiple pages are being attacked. In order to do this I had to gather the pages that are within the chameleon website as shown below



The way the attacks will work for this section onward is that instead of all users being sent to one page causing possible instability, it is going to divide the users randomly, this will most likely lead to a decrease in actual transactions but will allow us to see how many users the whole website can handle rather than a single page.

## 30 Concurrent users for 5 minutes



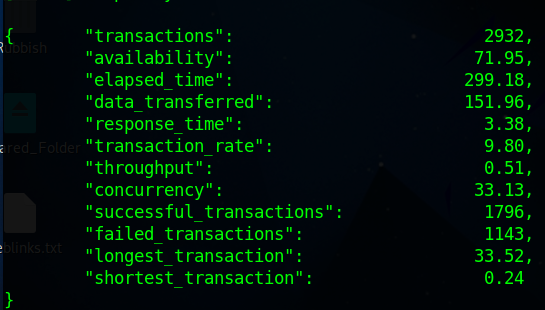
## 50 Concurrent users for 5 minutes



## 100 Concurrent users for 5 minutes



## 300 Concurrent users for 5 minutes



# Results Comparison

|  |  |  |
| --- | --- | --- |
| **Major Differences** | **Similarities** | **Improvement suggestions** |
| * Longer transaction times when all the users are on one page * Less failed transactions when the users are spread out | * Both tests show that the website can only hold roughly 300 users before bottlenecking occurs * Tests were able to run to their completion (server wasn’t overloaded) | * Increasing server resources to better handle traffic * Optimize page(s) with longer response time * Utilize caching and content delivery networks |

# Conclusion

During the stress testing of the Chameleon website, I have successfully managed to assess the performance of the current website under varying loads. The results have shown that as user numbers increase the availability decreases, emphasizing the importance of having set limits for an experience that provides minimum frustrations. Distributing the users across various pages demonstrated a reduction in failed transactions but fewer transactions overall. To improve the performance of the chameleon website can be improved through optimization tactics such as improving load times, using caching and content delivery networks, and ensuring that there are enough resources to handle the increasing user base. The insights for the tests conducted today can be used as a comparison point for when the new chameleon website gets built, and if needed they can come back to these documents for optimizations that can benefit the website.