


Load, Automation, Security & Web Optimization

Task 1: Load Test Using Locust

```
locustfile.py x
1 from locust import HttpUser, task, between
2
3 class WebsiteUser(HttpUser):
4     wait_time = between(1, 3) # Simulates realistic user behavior
5
6     @task(5) # Highest priority (5x more likely)
7     def load_homepage(self):
8         self.client.get("/")
9
10    @task(3) # Higher weight (3x more likely to be tested)
11    def load_training_list(self):
12        self.client.get("/careers")
13
14    @task(2) # Lower weight (2x)
15    def load_fair_list(self):
16        self.client.get("/contact")
17
18
19
20
```

```
C:\Windows\System32\cmd.e x + v
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

D:\Business_Automation_Internship\Assignments\New folder (2)>python -m locust -f locustfile.py --host=https://daraz.com
[2025-03-11 22:45:51,640] DESKTOP-8V93IVK/INFO/locust.main: Starting Locust 2.33.1
[2025-03-11 22:45:51,641] DESKTOP-8V93IVK/INFO/locust.main: Starting web interface at http://localhost:8089, press enter
to open your default browser.
[2025-03-11 22:46:49,386] DESKTOP-8V93IVK/INFO/locust.runners: Ramping to 200 users at a rate of 10.00 per second
[2025-03-11 22:47:08,480] DESKTOP-8V93IVK/INFO/locust.runners: All users spawned: {"WebsiteUser": 200} (200 total users)
```


 **LOCUST**

HOST
https://daraz.com

STATUS
READY

RPS
0

FAILURES
0%



Start new load test

Number of users (peak concurrency) *

200

Ramp up (users started/second) *

10

Host

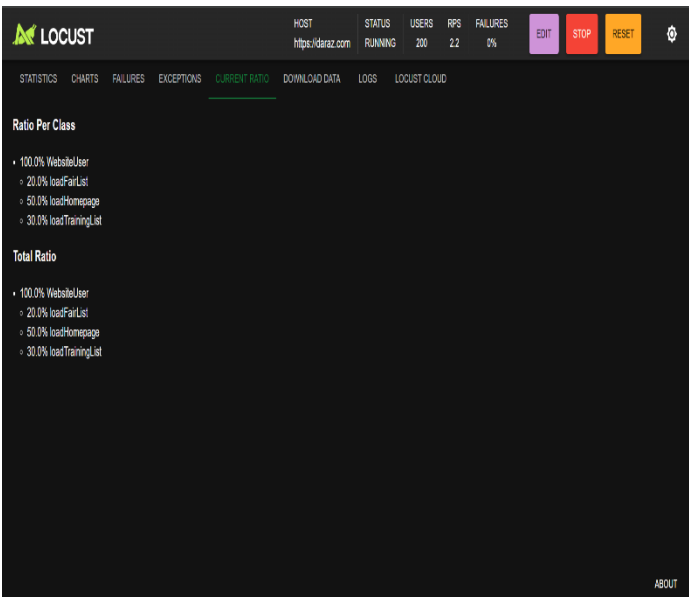
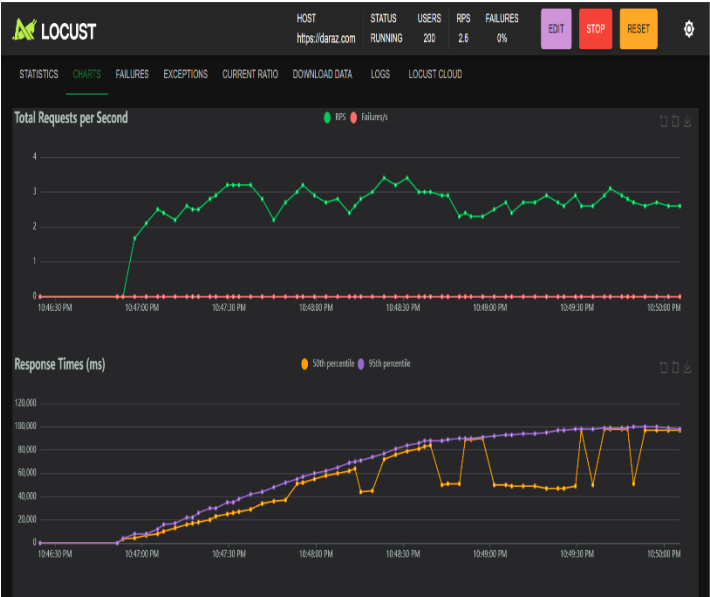
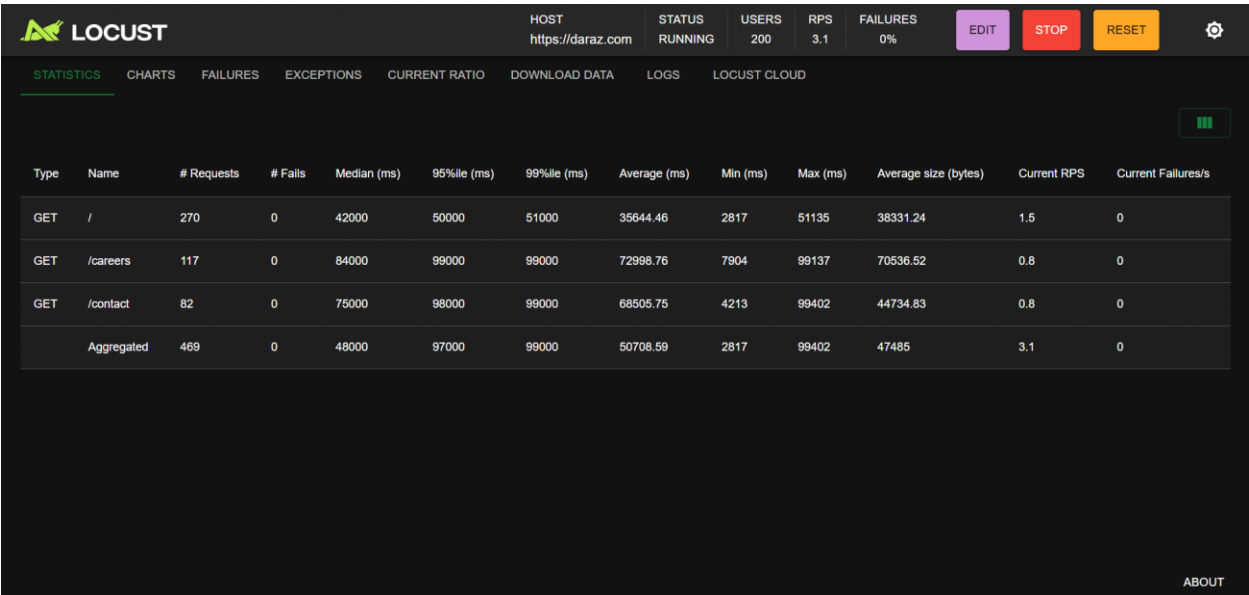
https://daraz.com

Advanced options

▼

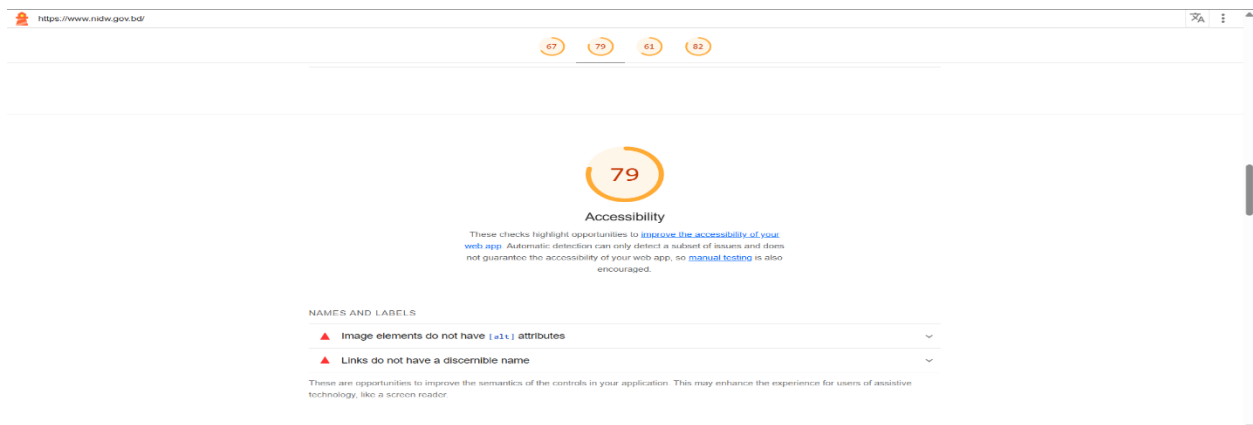
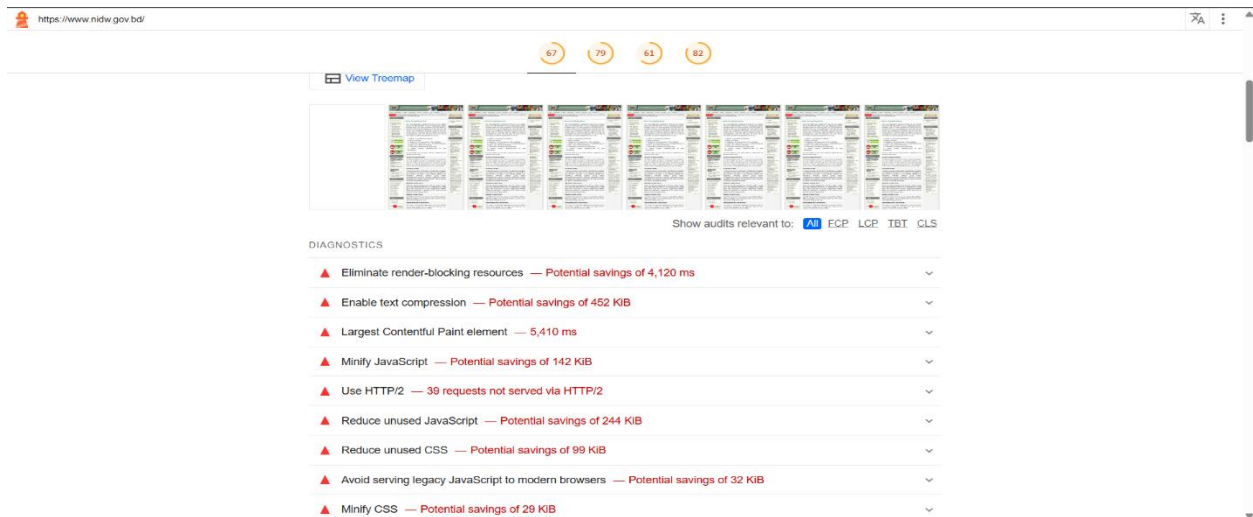
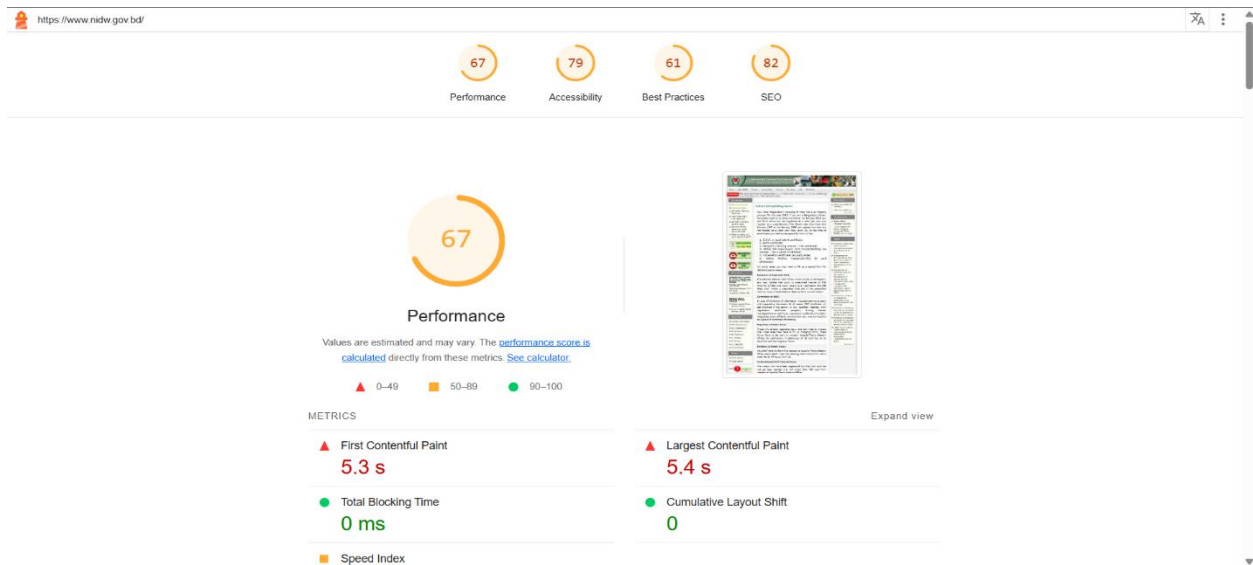
START

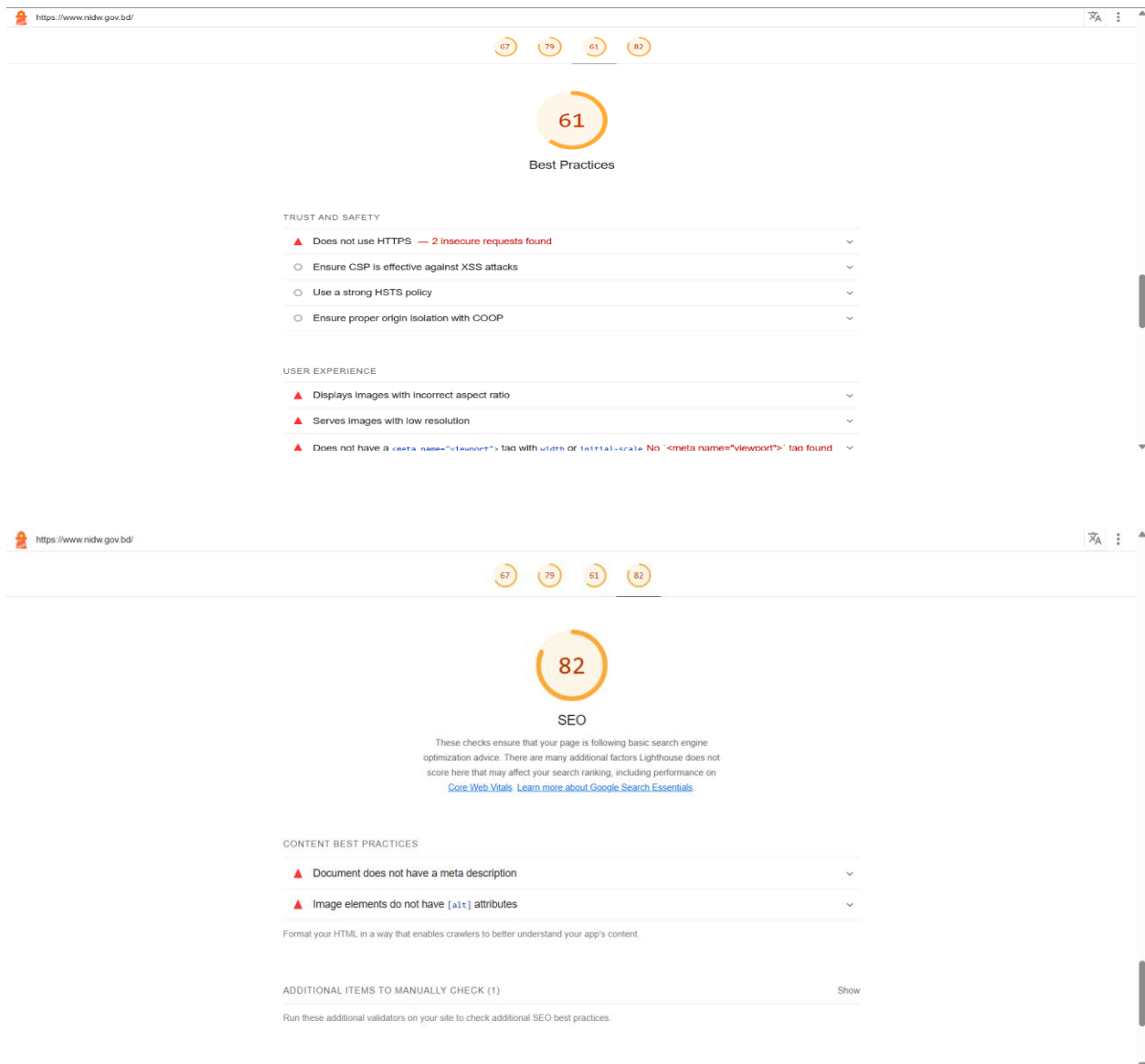
[ABOUT](#)



Summary: I am choosing website Daraz BD. The Locust load test results show high response times, with /careers route having a median of 84s and /contact route at 75s. The 99th percentile response time reaches 99s, indicating severe latency. Despite 0 failures, the requests per second (RPS) is low, with a total of 3.1 RPS. The max response time is nearly 99s, suggesting backend or database bottlenecks. Possible causes include slow queries, high server load, or network latency. Performance optimizations like caching, database indexing, and load balancing are recommended.

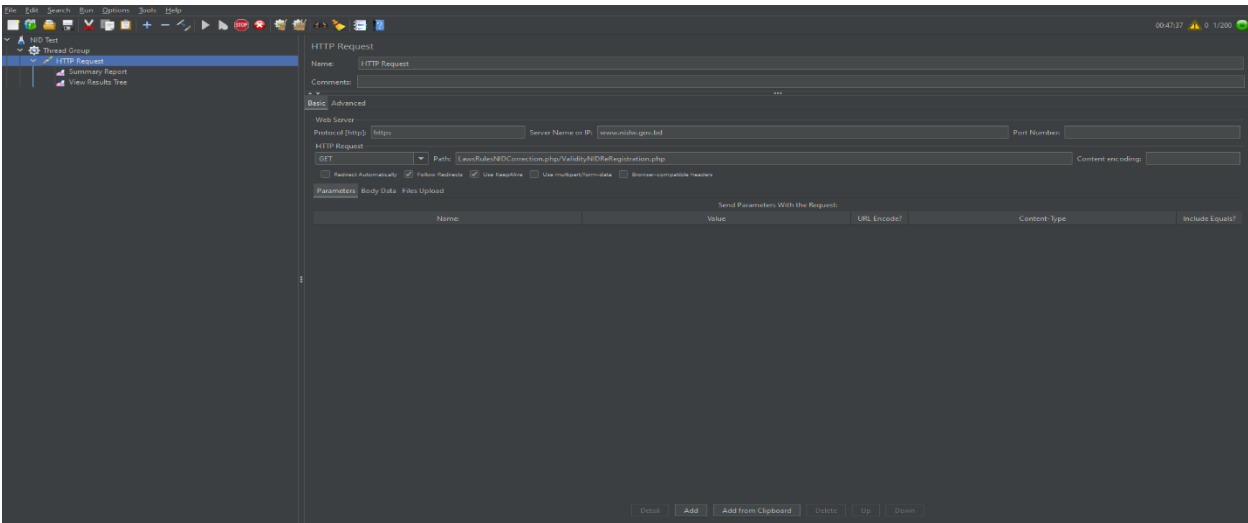
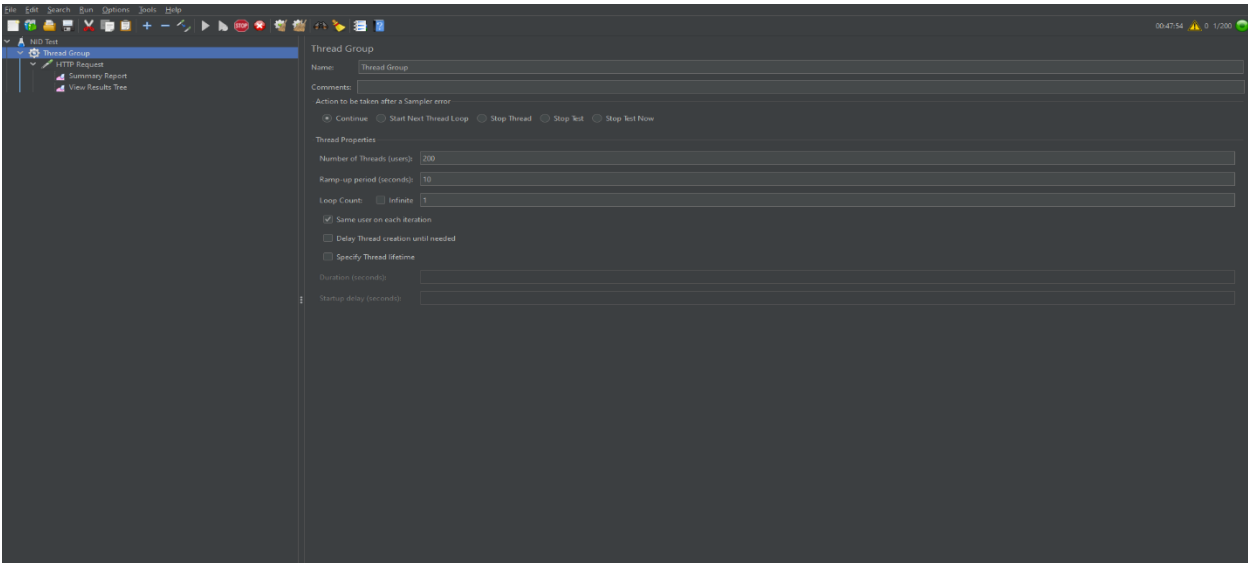
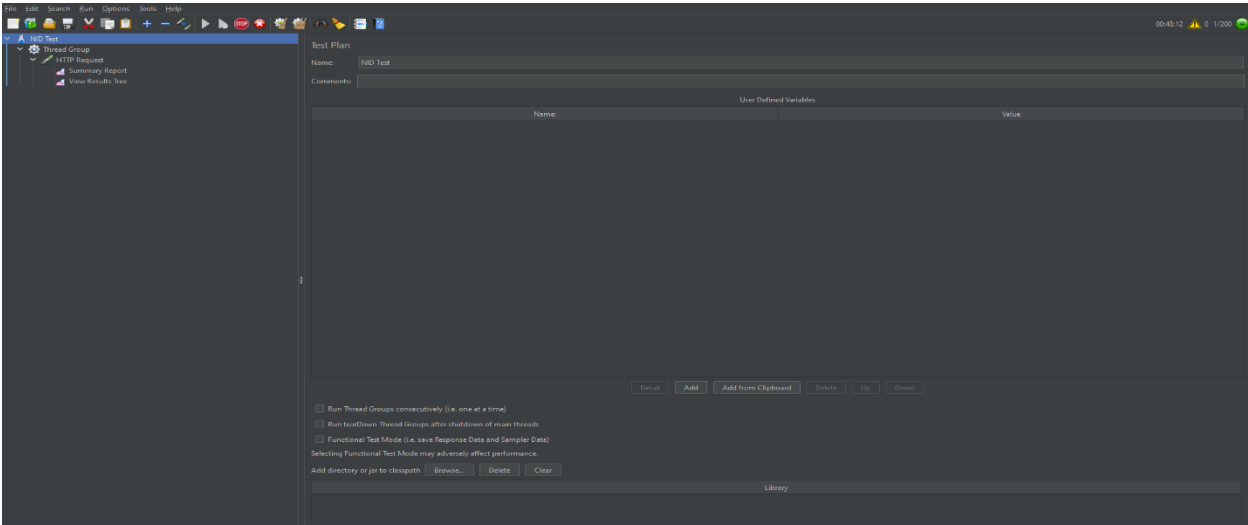
Task 2: Google Lighthouse Performance Report





Summary: I am choosing website Bangladesh Election Commission. The website has a performance score of 67, indicating moderate efficiency but room for improvement. The First Contentful Paint (FCP) takes 5.3 seconds, and the Largest Contentful Paint (LCP) is 5.4 seconds, both of which are relatively slow and can negatively impact user experience. However, Total Blocking Time (0ms) and Cumulative Layout Shift (0) are excellent, ensuring no delays due to scripts and a stable layout. The SEO score of 82 is decent, but the Best Practices score of 61 suggests areas that need enhancement. To improve performance, the website should optimize image sizes and implement lazy loading to reduce load times. Server-side caching and a Content Delivery Network (CDN) can further enhance speed and reliability. Minifying CSS, JavaScript, and HTML will help streamline rendering. Additionally, enabling compression techniques like Gzip or Brotli can reduce file sizes for faster delivery. Lastly, improving server response times will significantly enhance the overall loading speed.

Task 3: Load Testing Using JMeter



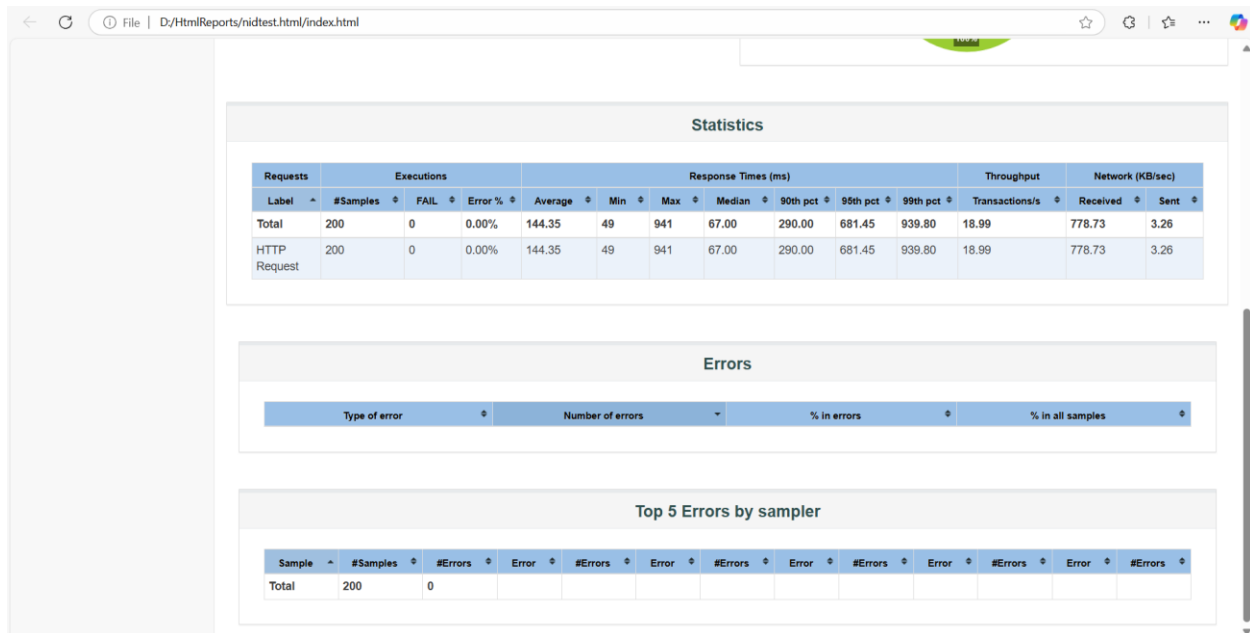
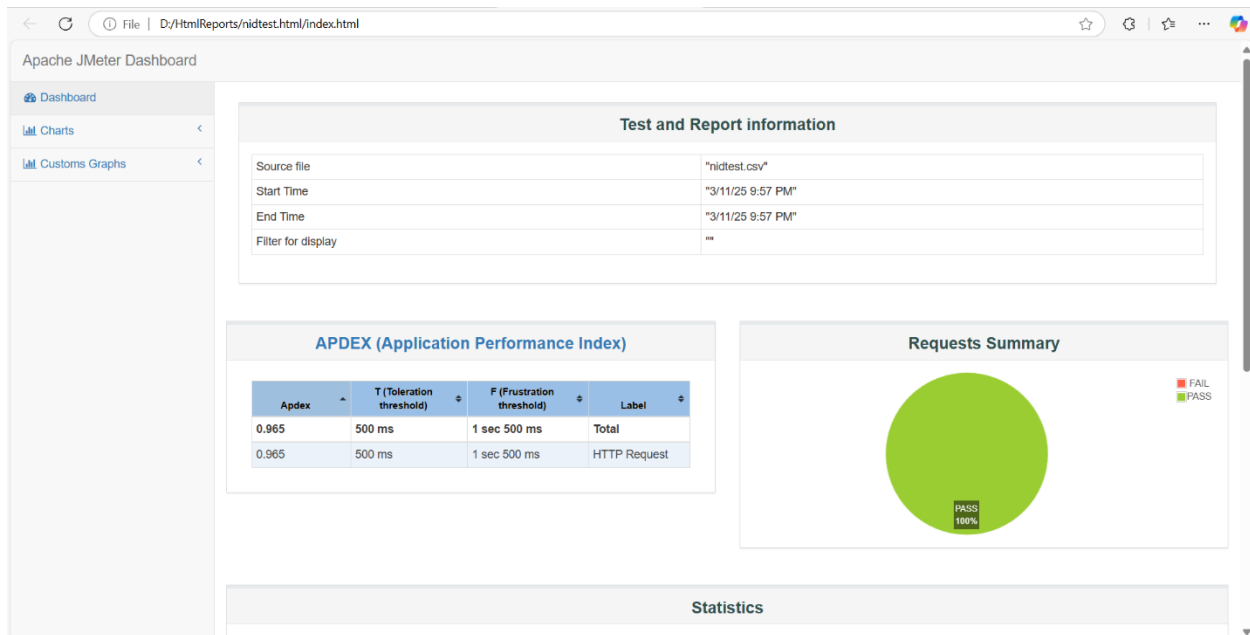
The screenshot shows the Burp Suite interface. On the left, the 'Project Tree' shows a hierarchy: 'HTTP Request' is selected under 'Thread Group'. The main 'View Results Tree' panel displays a list of 25 'HTTP Request' items, each with a green checkmark icon. The 'Test' dropdown is set to 'Sampler result'. The bottom status bar shows 'Raw' and 'Parsed' tabs.

```
C:\Windows\System32\cmd.exe + -
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aminur Rahman\Downloads\apache-jmeter-5.6.3\apache-jmeter-5.6.3\bin>jmeter -n -t "NID Test.jmx" -l D:\CSVReport
s\nidtest.csv -e -o D:\HtmlReports\nidtest.html
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
Mar 11, 2025 9:57:33 PM java.util.prefs.WindowsPreferences <init>
WARNING: Could not open/create prefs root node Software\JavaSoft\Prefs at root 0x80000002. Windows RegCreateKeyEx(...) r
eturned error code 5.
Creating summariser <summary>
Created the tree successfully using NID Test.jmx
Starting standalone test @ March 11, 2025 9:57:33 PM BDT (1741708653953)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +      1 in 00:00:01 =    0.9/s Avg:   575 Min:   575 Max:   575 Err:    0 (0.00%) Active: 15 Started: 21 Finis
hed: 6
summary +    199 in 00:00:10 =   20.7/s Avg:   142 Min:    49 Max:   941 Err:    0 (0.00%) Active: 0 Started: 200 Finis
hed: 200
summary +    200 in 00:00:11 =   18.7/s Avg:   144 Min:    49 Max:   941 Err:    0 (0.00%)
Tidying up ... @ March 11, 2025 9:57:44 PM BDT (1741708664960)
... end of run

C:\Users\Aminur Rahman\Downloads\apache-jmeter-5.6.3\apache-jmeter-5.6.3\bin>
```

HTML Reports



[illegible]

Summary: I am choosing website Bangladesh Election Commission. The load test executed a total of 199 HTTP requests, with an average response time of 234 milliseconds. The response times varied, with a minimum of 46 milliseconds and a maximum of 3172 milliseconds, indicating occasional performance spikes. Notably, there were no errors recorded, resulting in a 0.00% error rate, which suggests that all requests were processed successfully. The throughput was measured at 20.0 requests per second, ensuring a consistent request-handling capability. Additionally, the data transfer rates were recorded at 819.34 KB/sec for received data and 3.43 KB/sec for sent data. The "View Results Tree" confirms that all requests were successfully executed without failures. While the overall performance appears stable, the high maximum response time indicates potential slowdowns that may need further investigation.