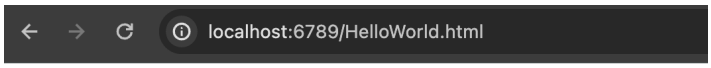
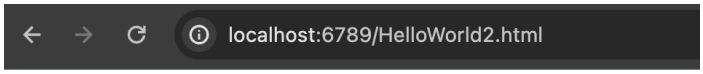
Persistent and Non-Persistent Web Servers

1.



Hello world!



404 Not Found

**HelloWorld2.html does not exist on the server, so 404 error is given*
Used localhost for testing on browser, with html file HelloWorld.html, WebServerOriginal.py

2. Before testing, make sure to place all text and image files in same directory as web server

Persistent HTTP (WebServerP.py)

WebServer:

```
MacBook-Pro-96:ece158b_hw2_sp24 praveenswaminathan$ python3 WebServerP.py
Ready to serve...
b'/BaseFile.html'
b'/dino.jpg'
b'/car.jpg'
b'/tiger.jpg'
b'/lion.jpg'
b'/tree.jpg'
b'/smile.jpg'
b'/rat.jpg'
b'/moth.jpg'
b'/food.jpg'
b'/cat.jpg'
b'/f1.txt'
b'/f2.txt'
b'/f3.txt'
b'/f4.txt'
b'/f5.txt'
b'/f6.txt'
b'/f7.txt'
b'/f8.txt'
b'/f9.txt'
b'/f10.txt'
no message received
Ready to serve...
```

Client:

```
MacBook-Pro-96:ece158b_hw2_sp24 praveenswaminathan$ python3 HTTPClient.py 127.0.0.1 6810 BaseFile.html
dino.jpg done
car.jpg done
tiger.jpg done
lion.jpg done
tree.jpg done
smile.jpg done
rat.jpg done
moth.jpg done
food.jpg done
cat.jpg done
f1.txt done
f2.txt done
f3.txt done
f4.txt done
f5.txt done
f6.txt done
f7.txt done
f8.txt done
f9.txt done
f10.txt done
Total Time: 1.2977397441864014s
```

Wireshark:

```
192019 1.265831   127.0.0.1   127.0.0.1   HTTP   131   64 GET /f3.txt HTTP/1.1
192020 1.265837   127.0.0.1   127.0.0.1   TCP     56   64 6810 → 60720 [ACK] Seq=896364 Ack=1077 Win=145856 L
192021 1.266020   127.0.0.1   127.0.0.1   HTTP    74   64 Continuation
192022 1.266038   127.0.0.1   127.0.0.1   TCP     56   64 60720 → 6810 [ACK] Seq=1077 Ack=896382 Win=131072 L
192023 1.266043   127.0.0.1   127.0.0.1   HTTP    67   64 Continuation
192024 1.266046   127.0.0.1   127.0.0.1   TCP     56   64 60720 → 6810 [ACK] Seq=1077 Ack=896393 Win=131072 L
192025 1.267199   127.0.0.1   127.0.0.1   HTTP   131   64 GET /f4.txt HTTP/1.1
192026 1.267204   127.0.0.1   127.0.0.1   TCP     56   64 6810 → 60720 [ACK] Seq=896393 Ack=1152 Win=145792 L
192027 1.267431   127.0.0.1   127.0.0.1   HTTP    74   64 Continuation
192028 1.267447   127.0.0.1   127.0.0.1   TCP     56   64 60720 → 6810 [ACK] Seq=1152 Ack=896411 Win=131072 L
192029 1.267452   127.0.0.1   127.0.0.1   HTTP    64   64 Continuation
192030 1.267455   127.0.0.1   127.0.0.1   TCP     56   64 60720 → 6810 [ACK] Seq=1152 Ack=896419 Win=131072 L
192031 1.268599   127.0.0.1   127.0.0.1   HTTP   131   64 GET /f5.txt HTTP/1.1
192032 1.268604   127.0.0.1   127.0.0.1   TCP     56   64 6810 → 60720 [ACK] Seq=896419 Ack=1227 Win=145728 L
192033 1.268808   127.0.0.1   127.0.0.1   HTTP    74   64 Continuation
```

Non-Persistent HTTP(WebServerNP.py)

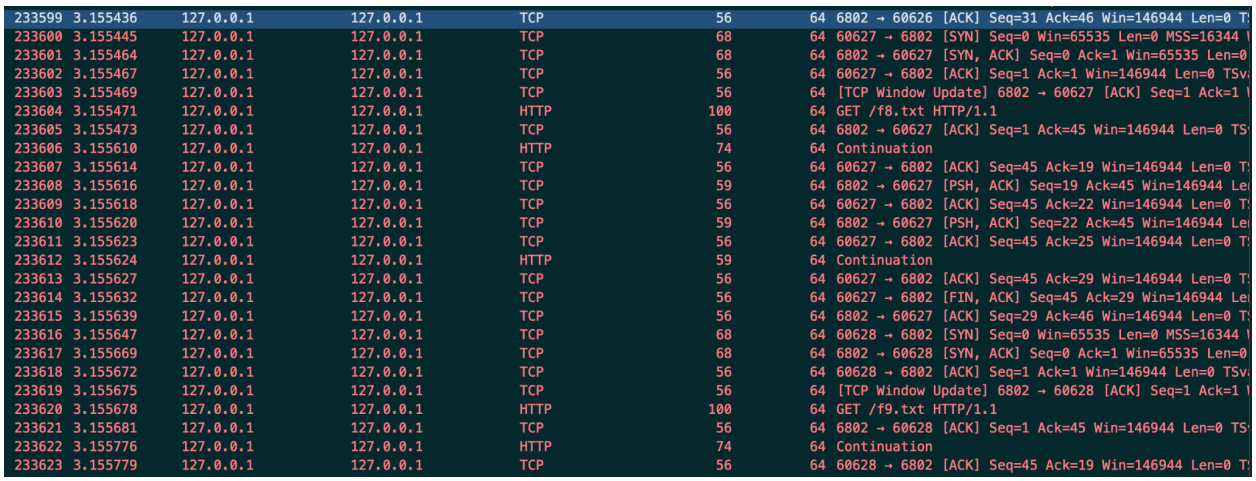
WebServer:

```
MacBook-Pro-96:ece158b_hw2_sp24 praveenswaminathan$ python3 WebServer.py
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
Ready to serve...
```

Client:

```
MacBook-Pro-96:ece158b_hw2_sp24 praveenswaminathan$ python3 HTTPClient.py 127.0.0.1 6802 BaseFile.html
dino.jpg done
car.jpg done
tiger.jpg done
lion.jpg done
tree.jpg done
smile.jpg done
rat.jpg done
moth.jpg done
food.jpg done
cat.jpg done
f1.txt done
f2.txt done
f3.txt done
f4.txt done
f5.txt done
f6.txt done
f7.txt done
f8.txt done
f9.txt done
f10.txt done
Total Time: 1.2818861007690435
```

Wireshark:



233599	3.155436	127.0.0.1	127.0.0.1	TCP	56	64	6802 → 60626 [ACK] Seq=31 Ack=46 Win=146944 Len=0 T
233600	3.155445	127.0.0.1	127.0.0.1	TCP	68	64	60627 → 6802 [SYN] Seq=0 Win=65535 Len=0 MSS=16344
233601	3.155464	127.0.0.1	127.0.0.1	TCP	68	64	6802 → 60627 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
233602	3.155467	127.0.0.1	127.0.0.1	TCP	56	64	60627 → 6802 [ACK] Seq=1 Ack=1 Win=146944 Len=0 TSv
233603	3.155469	127.0.0.1	127.0.0.1	TCP	56	64	[TCP Window Update] 6802 → 60627 [ACK] Seq=1 Ack=1
233604	3.155471	127.0.0.1	127.0.0.1	HTTP	100	64	GET /f8.txt HTTP/1.1
233605	3.155473	127.0.0.1	127.0.0.1	TCP	56	64	6802 → 60627 [ACK] Seq=1 Ack=45 Win=146944 Len=0 TS
233606	3.155610	127.0.0.1	127.0.0.1	HTTP	74	64	Continuation
233607	3.155614	127.0.0.1	127.0.0.1	TCP	56	64	60627 → 6802 [ACK] Seq=45 Ack=19 Win=146944 Len=0 T
233608	3.155616	127.0.0.1	127.0.0.1	TCP	59	64	6802 → 60627 [PSH, ACK] Seq=19 Ack=45 Win=146944 Le
233609	3.155618	127.0.0.1	127.0.0.1	TCP	56	64	60627 → 6802 [ACK] Seq=45 Ack=22 Win=146944 Len=0 T
233610	3.155620	127.0.0.1	127.0.0.1	TCP	59	64	6802 → 60627 [PSH, ACK] Seq=22 Ack=45 Win=146944 Le
233611	3.155623	127.0.0.1	127.0.0.1	TCP	56	64	60627 → 6802 [ACK] Seq=45 Ack=25 Win=146944 Len=0 T
233612	3.155624	127.0.0.1	127.0.0.1	HTTP	59	64	Continuation
233613	3.155627	127.0.0.1	127.0.0.1	TCP	56	64	60627 → 6802 [ACK] Seq=45 Ack=29 Win=146944 Len=0 T
233614	3.155632	127.0.0.1	127.0.0.1	TCP	56	64	60627 → 6802 [FIN, ACK] Seq=45 Ack=29 Win=146944 Le
233615	3.155639	127.0.0.1	127.0.0.1	TCP	56	64	6802 → 60627 [ACK] Seq=29 Ack=46 Win=146944 Len=0 T
233616	3.155647	127.0.0.1	127.0.0.1	TCP	68	64	60628 → 6802 [SYN] Seq=0 Win=65535 Len=0 MSS=16344
233617	3.155669	127.0.0.1	127.0.0.1	TCP	68	64	6802 → 60628 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
233618	3.155672	127.0.0.1	127.0.0.1	TCP	56	64	60628 → 6802 [ACK] Seq=1 Ack=1 Win=146944 Len=0 TSv
233619	3.155675	127.0.0.1	127.0.0.1	TCP	56	64	[TCP Window Update] 6802 → 60628 [ACK] Seq=1 Ack=1
233620	3.155678	127.0.0.1	127.0.0.1	HTTP	100	64	GET /f9.txt HTTP/1.1
233621	3.155681	127.0.0.1	127.0.0.1	TCP	56	64	6802 → 60628 [ACK] Seq=1 Ack=45 Win=146944 Len=0 TS
233622	3.155776	127.0.0.1	127.0.0.1	HTTP	74	64	Continuation
233623	3.155779	127.0.0.1	127.0.0.1	TCP	56	64	60628 → 6802 [ACK] Seq=45 Ack=19 Win=146944 Len=0 T

For both tests, my wireshark capture traffic is too large of a file (still included in folder). For ease of representation, I included a snippet of the wireshark traffic (to show difference between persistent and non-persistent), as well as an alternative way to display download time. In my client code, I made a timer, which ends once all the requests are sent and objects are received, outputting the total time of the implementation. All testing was done with the loopback address, with the base html file, and all objects being present in the same directory as the WebServer. As seen on the Wireshark screenshots, on the non-persistent HTTP, there is a 3-way handshake before each GET request, and a FIN message at the end of the transmission. For persistent HTTP, there are no new TCP connections between consecutive GET requests. Due to the reason that I used the loopback address (which has virtually no delay when making connections), the times for both HTTP implementations, persistent and non-persistent, are extremely similar. However, in a practical setting, the difference between the two would be much more noticeable, with persistent being much faster than non-persistent HTTP.