

COMPUTER NETWORKS

SEMESTER 2018

ASSIGNMENT 2

Members:

Nirvan Singhania - 20161070

Nikhil Bansal - 20161065

Objectives:

1. To build a proxy server between the client and the web server .
2. The proxy server will forward the client's request to the web server. The web server will then generate a response message and deliver it to the proxy server, which in turn sends it to the client

Features:

1. Threading in proxy server to handle multiple clients
2. Mutex locks to restrict multiple client accessing the same file
3. File is added to the cache when file is at least requested for the 2nd time because adding it the first time will overload the server and there is no guarantee that the object will be requested again.
4. Code along with error handling
5. Modular code with options of specifying buffer size , limit on the number of occurrences for cache
6. Used classes and private variables.

Server:

- Hosted at localhost/20000 or 127.0.0.1/20000
- To run: python server.py

Proxy Server:

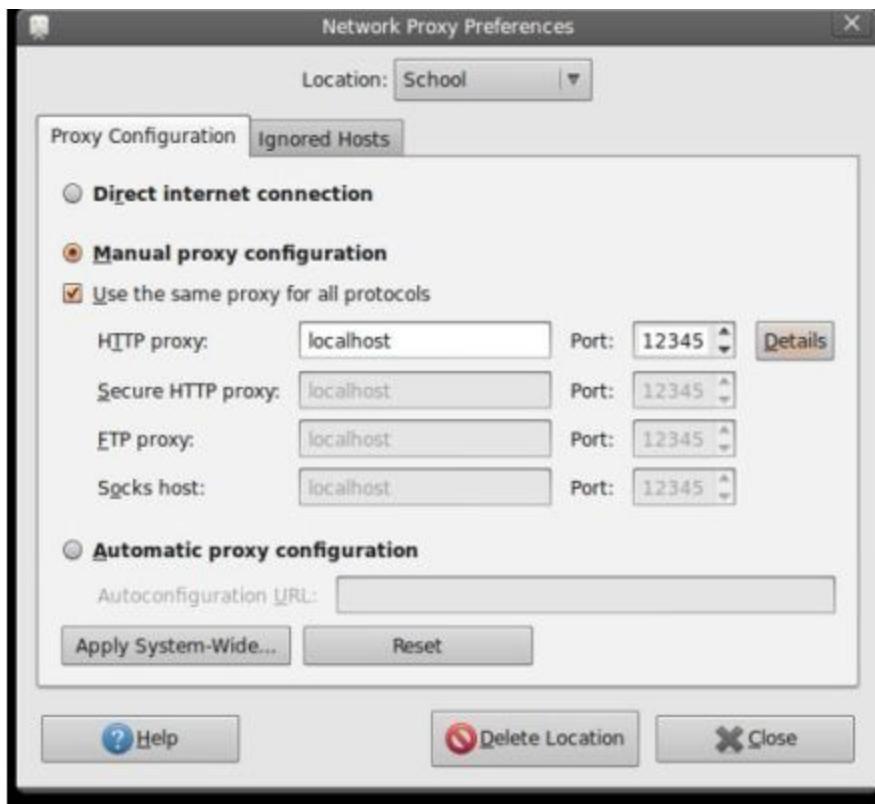
- Hosted at localhost/12345 or 127.0.0.1/12345
-

- To run: python proxy.py

Client:

The following 2 ways could be used to retrieve the files from the client side.

- Using curl commands
 - curl -x http://localhost:12345 http://127.0.0.1:20000/filename.txt
- Using browser:
 - Set proxy on your browser:



- Now all the proxy requests will pass through the proxy server including ones from already started webpages.
- Type the full path i.e. localhost:20000/filename.txt to get the file

Assignment Details:

1. Language Used: Python 2.7

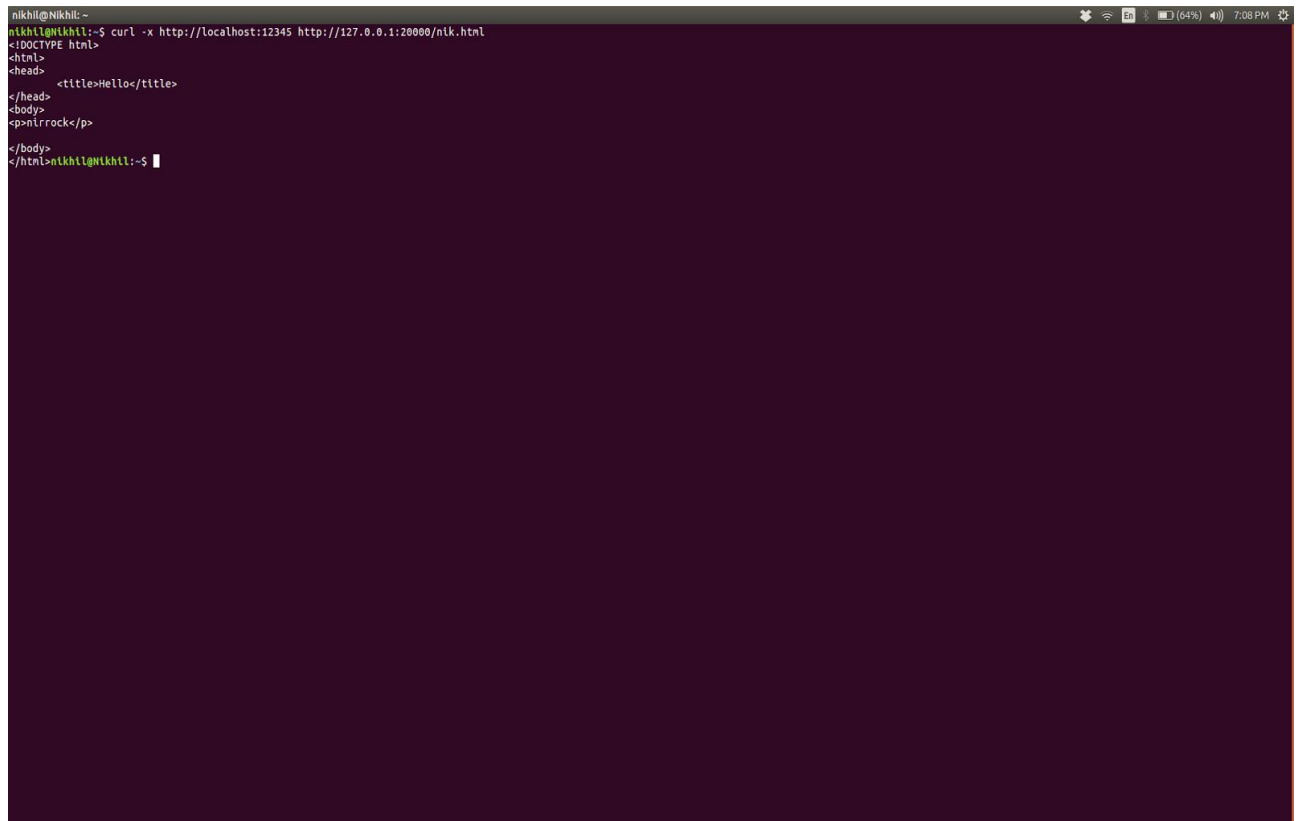
Running the assignment:

- Server: python server.py
- Proxy: python proxy.py
- curl -x http://localhost:12345 http://127.0.0.1:20000/file_name

Brief description :

- There are two classes - 1. LockSystem and 2. Proxyserver
- The main controller for the server is responsible for taking action dependent on whether the requested file is in cache.
- The cacheSpaceHandler and cachedetailHandler are responsible for dealing with the empty space in the cache and the other for extracting details .
- The parser is the main manipulator of data between the server and the client .
- The replacement policy is according to Least Recently Modified policy

Screenshots for the assignment:



```
nikhil@nikhil:~$ curl -x http://localhost:12345 http://127.0.0.1:20000/nik.html
<!DOCTYPE html>
<html>
<head>
</head>
<title>Hello</title>
</head>
<body>
<p>nirrock</p>
</body>
</html>nikhil@nikhil:~$
```

```
nikhil@Nikhil: ~/Downloads
nikhil@Nikhil:~/Downloads$ python new.py
Serving proxy on 127.0.0.1 port 12345 ...
('127.0.0.1', 35628) - [2018-02-14 19:09:21.080610] "GET http://127.0.0.1:20000/nik.html HTTP/1.1"
without caching serving ./cache/http:____127.0.0.1:20000__nik.html to ('127.0.0.1', 35628)
('127.0.0.1', 35628) closed
('127.0.0.1', 35632) - [2018-02-14 19:09:21.784644] "GET http://127.0.0.1:20000/nik.html HTTP/1.1"
caching file while serving ./cache/http:____127.0.0.1:20000__nik.html to ('127.0.0.1', 35632)
('127.0.0.1', 35632) closed
('127.0.0.1', 35636) - [2018-02-14 19:09:22.437644] "GET http://127.0.0.1:20000/nik.html HTTP/1.1"
returning cached file from ./cache/http:____127.0.0.1:20000__nik.html to ('127.0.0.1', 35636)
('127.0.0.1', 35636) closed
('127.0.0.1', 35640) - [2018-02-14 19:09:23.157995] "GET http://127.0.0.1:20000/nik.html HTTP/1.1"
returning cached file from ./cache/http:____127.0.0.1:20000__nik.html to ('127.0.0.1', 35640)
('127.0.0.1', 35640) closed
```

```
nikhil@Nikhil: ~/Downloads/server
nikhil@Nikhil:~/Downloads/server$ python server.py
Serving on port 20000
127.0.0.1 - - [14/Feb/2018 20:24:41] "GET /nik.html HTTP/1.1" 200 -
127.0.0.1 - - [14/Feb/2018 20:24:41] "GET /nik.html HTTP/1.1" 200 -
127.0.0.1 - - [14/Feb/2018 20:24:42] "GET /nik.html HTTP/1.1" 304 -
127.0.0.1 - - [14/Feb/2018 20:24:43] "GET /nik.html HTTP/1.1" 304 -
```