Lab 3

Exercise 1

Explore DNS records (Not marked, No need to submit)
DNS servers use different record types for different purposes. For each type of DNS record, there is an associated type of DNS query. Check the following page (https://en.wikipedia.org/wiki/List_of_DNS_record_types) and find out what the following resource record types are used for:

- A - address record
- CNAME - alias for canonical name
- MX - mailserver
- NS - delegates a DNS zone to use the given authoritative name servers
- PTR - like CNAME, except DNS stops processing and just the canonical name is returned
- SOA - Specifies authoritative information about the DNS zone

Exercise 3

Digging into DNS (marked, include in the lab report)

In order to answer the following questions, you will make DNS queries using some of the query types you have encountered in the above exercise. Some questions require you to make multiple DNS queries. Before you proceed, read the manpage of dig (type man dig in the terminal). Make sure you understand how you can explicitly specify the following:

- nameserver to query
- type of DNS query to make (the default query types are those you saw in exercise 1)
- performing reverse queries

Note: Include the output of all the dig commands you have used in your answers.

To send a query to a particular name server (say x.x.x.x) you should use the following command:

```
dig @x.x.x.x hostname
```

Question 1. What is the IP address of www.eecs.berkeley.edu . What type of DNS query is sent to get this answer?

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig www.eecs.berkeley.edu

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> www.eecs.berkeley.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23400
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 4, ADDITIONAL: 7

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.eecs.berkeley.edu.         IN      A

;; ANSWER SECTION:
www.eecs.berkeley.edu.  5998    IN      CNAME   live-eecs.pantheonsite.io.
live-eecs.pantheonsite.io. 535  IN      CNAME   fe1.edge.pantheon.io.
fe1.edge.pantheon.io.   235     IN      A       23.185.0.1

;; AUTHORITY SECTION:
edge.pantheon.io.       235     IN      NS      ns-1213.awsdns-23.org.
edge.pantheon.io.       235     IN      NS      ns-644.awsdns-16.net.
edge.pantheon.io.       235     IN      NS      ns-233.awsdns-29.com.
edge.pantheon.io.       235     IN      NS      ns-2013.awsdns-59.co.uk.

;; ADDITIONAL SECTION:
ns-233.awsdns-29.com.   21409   IN      A       205.251.192.233
ns-644.awsdns-16.net.   87085   IN      A       205.251.194.132
ns-644.awsdns-16.net.   87085   IN      AAAA    2600:9000:5302:8400::1
ns-1213.awsdns-23.org.  84961   IN      A       205.251.196.189
ns-1213.awsdns-23.org.  65532   IN      AAAA    2600:9000:5304:bd00::1
ns-2013.awsdns-59.co.uk. 117313 IN      A       205.251.199.221

;; Query time: 0 msec
;; SERVER: 129.94.242.2#53(129.94.242.2)
;; WHEN: Sun Mar 13 15:15:05 AEDT 2022
;; MSG SIZE  rcvd: 397
```

23.185.0.1

Question 2. What is the canonical name for the eecs.berkeley webserver (i.e. www.eecs.berkeley.edu)? Suggest a reason for having an alias for this server.

live-eecs.pantheonsite.io.

Web hosting service

Can use same hostname for multiple services

Question 3. What can you make of the rest of the response (i.e. the details available in the Authority and Additional sections)?

Authority section shows the name servers

Additional section has IP address of name severs

Question 4. What is the IP address of the local nameserver for your machine?

SERVER: 129.94.242.2#53(129.94.242.2)

Question 5. What are the DNS nameservers for the “eecs.berkeley.edu.” domain (note: the domain name is eecs.berkeley.edu and not www.eecs.berkeley.edu . This is an example of what is referred to as the

apex/naked domain)? Find out their IP addresses? What type of DNS query is sent to obtain this information?

eecs.berkeley.edu is the naked domain

NS, a name server query

ns.CS.berkeley.edu.	14759	IN	A	169.229.60.61
ns.CS.berkeley.edu.	74376	IN	AAAA	2607:f140:8:1260::30
ns.eecs.berkeley.edu.	17969	IN	A	169.229.60.153
adns1.berkeley.edu.	2854	IN	A	128.32.136.3
adns1.berkeley.edu.	2464	IN	AAAA	2607:f140:ffff:fffe::3
adns2.berkeley.edu.	8191	IN	A	128.32.136.14
adns2.berkeley.edu.	2464	IN	AAAA	2607:f140:ffff:fffe::e
adns3.berkeley.edu.	10407	IN	A	192.107.102.142
adns3.berkeley.edu.	4818	IN	AAAA	2607:f140:a000:d::abc

```
; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> eecs.berkeley.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40689
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 5, ADDITIONAL: 10

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;eecs.berkeley.edu.             IN      A

;; ANSWER SECTION:
eecs.berkeley.edu.      9147    IN      A       23.185.0.1

;; AUTHORITY SECTION:
eecs.berkeley.edu.      11453   IN      NS      adns2.berkeley.edu.
eecs.berkeley.edu.      11453   IN      NS      ns.eecs.berkeley.edu.
eecs.berkeley.edu.      11453   IN      NS      ns.CS.berkeley.edu.
eecs.berkeley.edu.      11453   IN      NS      adns1.berkeley.edu.
eecs.berkeley.edu.      11453   IN      NS      adns3.berkeley.edu.

;; ADDITIONAL SECTION:
ns.CS.berkeley.edu.     15007   IN      A       169.229.60.61
ns.CS.berkeley.edu.     74624   IN      AAAA    2607:f140:8:1260::30
ns.eecs.berkeley.edu.   18217   IN      A       169.229.60.153
adns1.berkeley.edu.     3102    IN      A       128.32.136.3
adns1.berkeley.edu.     2712    IN      AAAA    2607:f140:ffff:fffe::3
adns2.berkeley.edu.     8439    IN      A       128.32.136.14
adns2.berkeley.edu.     2712    IN      AAAA    2607:f140:ffff:fffe::e
adns3.berkeley.edu.     10655   IN      A       192.107.102.142
adns3.berkeley.edu.     5066    IN      AAAA    2607:f140:a000:d::abc

;; Query time: 0 msec
;; SERVER: 129.94.242.2#53(129.94.242.2)
;; WHEN: Sun Mar 13 15:14:16 AEDT 2022
;; MSG SIZE  rcvd: 351
```

Question 6. What is the DNS name associated with the IP address 111.68.101.54? What type of DNS query is sent to obtain this information?

Before we gave hostname and got IP address, now we do the opposite.

Hostname:webserver.seecs.nust.edu.pk.

This is a reverse query (IP to hostname)

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig -x 111.68.101.54

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> -x 111.68.101.54
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7163
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;54.101.68.111.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
54.101.68.111.in-addr.arpa. 3600 IN     PTR     webserver.seecs.nust.edu.pk.

;; AUTHORITY SECTION:
101.68.111.in-addr.arpa. 53443  IN      NS      ns2.hec.gov.pk.
101.68.111.in-addr.arpa. 53443  IN      NS      ns1.hec.gov.pk.

;; Query time: 174 msec
;; SERVER: 129.94.242.2#53(129.94.242.2)
;; WHEN: Sun Mar 13 15:19:31 AEDT 2022
;; MSG SIZE  rcvd: 140
```

Question 7. Run dig and query the CSE nameserver (129.94.242.33) for the mail servers for Yahoo! Mail (again the domain name is yahoo.com, not www.yahoo.com). Did you get an authoritative answer? Why? (HINT: Just because a response contains information in the authoritative part of the DNS response message does not mean it came from an authoritative name server. You should examine the flags in the response to determine the answer)

No not authoritative, we didn't see aa flag. We queries CSE for Yahoo's hostname, CSE isn't the authority for Yahoo.

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @129.94.242.2 yahoo.com MX

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @129.94.242.2 yahoo.com MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18233
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 5, ADDITIONAL: 10

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;yahoo.com.                     IN      MX

;; ANSWER SECTION:
yahoo.com.              1800    IN      MX      1 mta6.am0.yahoodns.net.
yahoo.com.              1800    IN      MX      1 mta7.am0.yahoodns.net.
yahoo.com.              1800    IN      MX      1 mta5.am0.yahoodns.net.

;; AUTHORITY SECTION:
yahoo.com.              70092   IN      NS      ns3.yahoo.com.
yahoo.com.              70092   IN      NS      ns2.yahoo.com.
yahoo.com.              70092   IN      NS      ns1.yahoo.com.
yahoo.com.              70092   IN      NS      ns4.yahoo.com.
yahoo.com.              70092   IN      NS      ns5.yahoo.com.

;; ADDITIONAL SECTION:
ns1.yahoo.com.          83603   IN      A       68.180.131.16
ns1.yahoo.com.          41072   IN      AAAA    2001:4998:1b0::7961:686f:6f21
ns2.yahoo.com.          170787  IN      A       68.142.255.16
ns2.yahoo.com.          41716   IN      AAAA    2001:4998:1c0::7961:686f:6f21
ns3.yahoo.com.          262     IN      A       27.123.42.42
ns3.yahoo.com.          262     IN      AAAA    2406:8600:f03f:1f8::1003
ns4.yahoo.com.          74880   IN      A       98.138.11.157
ns5.yahoo.com.          19894   IN      A       202.165.97.53
ns5.yahoo.com.          12436   IN      AAAA    2406:2000:1d0::7961:686f:6f21

;; Query time: 234 msec
;; SERVER: 129.94.242.2#53(129.94.242.2)
```

Question 8. Repeat the above (i.e. Question 7) but use one of the nameservers obtained in Question 5. What is the result?

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @128.32.136.3 yahoo.com MX

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @128.32.136.3 yahoo.com MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: REFUSED, id: 58135
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1220
;; QUESTION SECTION:
;yahoo.com.                     IN      MX

;; Query time: 167 msec
;; SERVER: 128.32.136.3#53(128.32.136.3)
;; WHEN: Sun Mar 13 15:25:28 AEDT 2022
;; MSG SIZE  rcvd: 38
```

Response refused. Maybe the NS doesn't reply to DNS queries from outside the Berkeley network.

Question 9. Obtain the authoritative answer for the mail servers for Yahoo! Mail. What type of DNS query is sent to obtain this information?

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @68.180.131.16 yahoo.com MX

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @68.180.131.16 yahoo.com MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 13037
;; flags: qr aa rd; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1272
;; QUESTION SECTION:
;yahoo.com.                     IN      MX

;; ANSWER SECTION:
yahoo.com.              1800    IN      MX      1 mta6.am0.yahoodns.net.
yahoo.com.              1800    IN      MX      1 mta5.am0.yahoodns.net.
yahoo.com.              1800    IN      MX      1 mta7.am0.yahoodns.net.

;; Query time: 141 msec
;; SERVER: 68.180.131.16#53(68.180.131.16)
;; WHEN: Sun Mar 13 15:27:34 AEDT 2022
;; MSG SIZE  rcvd: 117
```

There is an aa flag, so this is an authoritative answer.

We sent an MX query to the authoritative server

;; ANSWER SECTION:

yahoo.com. 1800 IN MX 1 mta6.am0.yahoodns.net.

yahoo.com. 1800 IN MX 1 mta5.am0.yahoodns.net.

yahoo.com. 1800 IN MX 1 mta7.am0.yahoodns.net.

Question 10. In this exercise, you simulate the iterative DNS query process to find the IP address of your machine (e.g. lyre00.cse.unsw.edu.au). If you are using VLAB Then find the IP address of one of the following: lyre00.cse.unsw.edu.au, lyre01.cse.unsw.edu.au, drum00.cse.unsw.edu.au or drum01.cse.unsw.edu.au. First, find the name server (query type NS) of the "." domain (root domain). Query this nameserver to find the authoritative name server for the "au." domain. Query this second server to find the authoritative nameserver for the "edu.au." domain. Now query this nameserver to find the authoritative nameserver for "unsw.edu.au". Next query the nameserver of unsw.edu.au to find the authoritative name server of cse.unsw.edu.au. Now query the nameserver of cse.unsw.edu.au to find the IP address of your host. How many DNS servers do you have to query to get the authoritative answer?

First we dig the root

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig . NS

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> . NS
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5969
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;.                              IN      NS

;; ANSWER SECTION:
.                       238402  IN      NS      j.root-servers.net.
.                       238402  IN      NS      b.root-servers.net.
.                       238402  IN      NS      i.root-servers.net.
.                       238402  IN      NS      g.root-servers.net.
.                       238402  IN      NS      d.root-servers.net.
.                       238402  IN      NS      e.root-servers.net.
.                       238402  IN      NS      l.root-servers.net.
.                       238402  IN      NS      a.root-servers.net.
.                       238402  IN      NS      f.root-servers.net.
.                       238402  IN      NS      m.root-servers.net.
.                       238402  IN      NS      k.root-servers.net.
.                       238402  IN      NS      h.root-servers.net.
.                       238402  IN      NS      c.root-servers.net.

;; ADDITIONAL SECTION:
a.root-servers.net.     160492  IN      A       198.41.0.4
a.root-servers.net.     36855   IN      AAAA    2001:503:ba3e::2:30
b.root-servers.net.     116704  IN      A       199.9.14.201
b.root-servers.net.     47498   IN      AAAA    2001:500:200::b
```

Then, we dig one of the root nameserver for the comp I’m using

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @198.41.0.4 vx2.orchestra.cse.unsw.EDU.AU NS

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @198.41.0.4 vx2.orchestra.cse.unsw.EDU.AU NS
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34972
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 9
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;vx2.orchestra.cse.unsw.EDU.AU. IN      NS

;; AUTHORITY SECTION:
AU.                     172800  IN      NS      q.AU.
AU.                     172800  IN      NS      t.AU.
AU.                     172800  IN      NS      s.AU.
AU.                     172800  IN      NS      r.AU.

;; ADDITIONAL SECTION:
q.AU.                   172800  IN      A       65.22.196.1
q.AU.                   172800  IN      AAAA    2a01:8840:be::1
t.AU.                   172800  IN      A       65.22.199.1
t.AU.                   172800  IN      AAAA    2a01:8840:c1::1
s.AU.                   172800  IN      A       65.22.198.1
s.AU.                   172800  IN      AAAA    2a01:8840:c0::1
r.AU.                   172800  IN      A       65.22.197.1
r.AU.                   172800  IN      AAAA    2a01:8840:bf::1

;; Query time: 148 msec
;; SERVER: 198.41.0.4#53(198.41.0.4)
;; WHEN: Sun Mar 13 15:36:41 AEDT 2022
```

They don’t know the answer, but referred us to the .au NS. Let’s dig one of them

```
3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @65.22.196.1 vx2.orchestra.cse.unsw.EDU.AU NS

 <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @65.22.196.1 vx2.orchestra.cse.unsw.EDU.AU NS
 (1 server found)
; global options: +cmd
; Got answer:
; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59621
; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 6
; WARNING: recursion requested but not available

; OPT PSEUDOSECTION:
 EDNS: version: 0, flags:; udp: 1232
; QUESTION SECTION:
vx2.orchestra.cse.unsw.EDU.AU. IN      NS

; AUTHORITY SECTION:
nsw.edu.au.             900     IN      NS      ns2.unsw.edu.au.
nsw.edu.au.             900     IN      NS      ns1.unsw.edu.au.
nsw.edu.au.             900     IN      NS      ns3.unsw.edu.au.

; ADDITIONAL SECTION:
s1.unsw.edu.au.         900     IN      A       129.94.0.192
s2.unsw.edu.au.         900     IN      A       129.94.0.193
s3.unsw.edu.au.         900     IN      A       192.155.82.178
s1.unsw.edu.au.         900     IN      AAAA    2001:388:c:35::1
s2.unsw.edu.au.         900     IN      AAAA    2001:388:c:35::2

; Query time: 25 msec
; SERVER: 65.22.196.1#53(65.22.196.1)
; WHEN: Sun Mar 13 15:38:05 AEDT 2022
; MSG SIZE  rcvd: 227
```

The AU server knows UNSW

```
3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @129.94.0.192 vx2.orchestra.cse.unsw.EDU.AU NS

 <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @129.94.0.192 vx2.orchestra.cse.unsw.EDU.AU NS
 (1 server found)
; global options: +cmd
; Got answer:
; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39004
; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 2, ADDITIONAL: 5
; WARNING: recursion requested but not available

; OPT PSEUDOSECTION:
 EDNS: version: 0, flags:; udp: 4096
; QUESTION SECTION:
vx2.orchestra.cse.unsw.EDU.AU. IN      NS

; AUTHORITY SECTION:
se.unsw.EDU.AU.        300     IN      NS      maestro.orchestra.cse.unsw.EDU.AU.
se.unsw.EDU.AU.        300     IN      NS      beethoven.orchestra.cse.unsw.EDU.AU.

; ADDITIONAL SECTION:
eethoven.orchestra.cse.unsw.EDU.AU. 300 IN A   129.94.172.11
eethoven.orchestra.cse.unsw.EDU.AU. 300 IN A   129.94.208.3
eethoven.orchestra.cse.unsw.EDU.AU. 300 IN A   129.94.242.2
aestro.orchestra.cse.unsw.EDU.AU. 300 IN A     129.94.242.33

; Query time: 4 msec
; SERVER: 129.94.0.192#53(129.94.0.192)
; WHEN: Sun Mar 13 15:38:59 AEDT 2022
; MSG SIZE  rcvd: 168
```

Referred to some UNSW NS. Let’s query beethoven

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @129.94.172.11 vx2.orchestra.cse.unsw.EDU.AU A

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @129.94.172.11 vx2.orchestra.cse.unsw.EDU.AU A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18763
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;vx2.orchestra.cse.unsw.EDU.AU. IN      A

;; ANSWER SECTION:
vx2.orchestra.cse.unsw.EDU.AU. 3600 IN  A       129.94.242.115

;; AUTHORITY SECTION:
orchestra.cse.unsw.EDU.AU. 3600 IN      NS      maestro.orchestra.cse.unsw.EDU.AU.
orchestra.cse.unsw.EDU.AU. 3600 IN      NS      beethoven.orchestra.cse.unsw.EDU.AU.

;; ADDITIONAL SECTION:
maestro.orchestra.cse.unsw.EDU.AU. 3600 IN A    129.94.242.33
beethoven.orchestra.cse.unsw.EDU.AU. 3600 IN A  129.94.242.2

;; Query time: 0 msec
;; SERVER: 129.94.172.11#53(129.94.172.11)
;; WHEN: Sun Mar 13 15:40:46 AEDT 2022
;; MSG SIZE  rcvd: 152
```

We’ve found the authoritative answer (flag aa)!

We queried 5 nameservers to get an authoritative answer

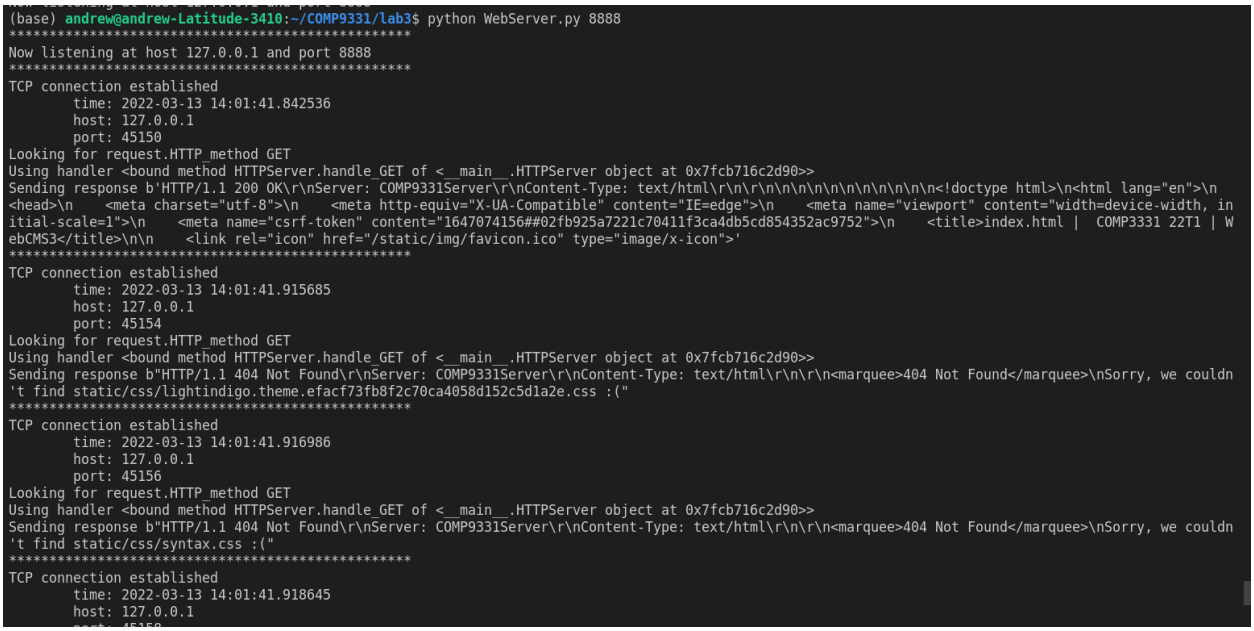
Question 11. Can one physical machine have several names and/or IP addresses associated with it?

Yes it can.

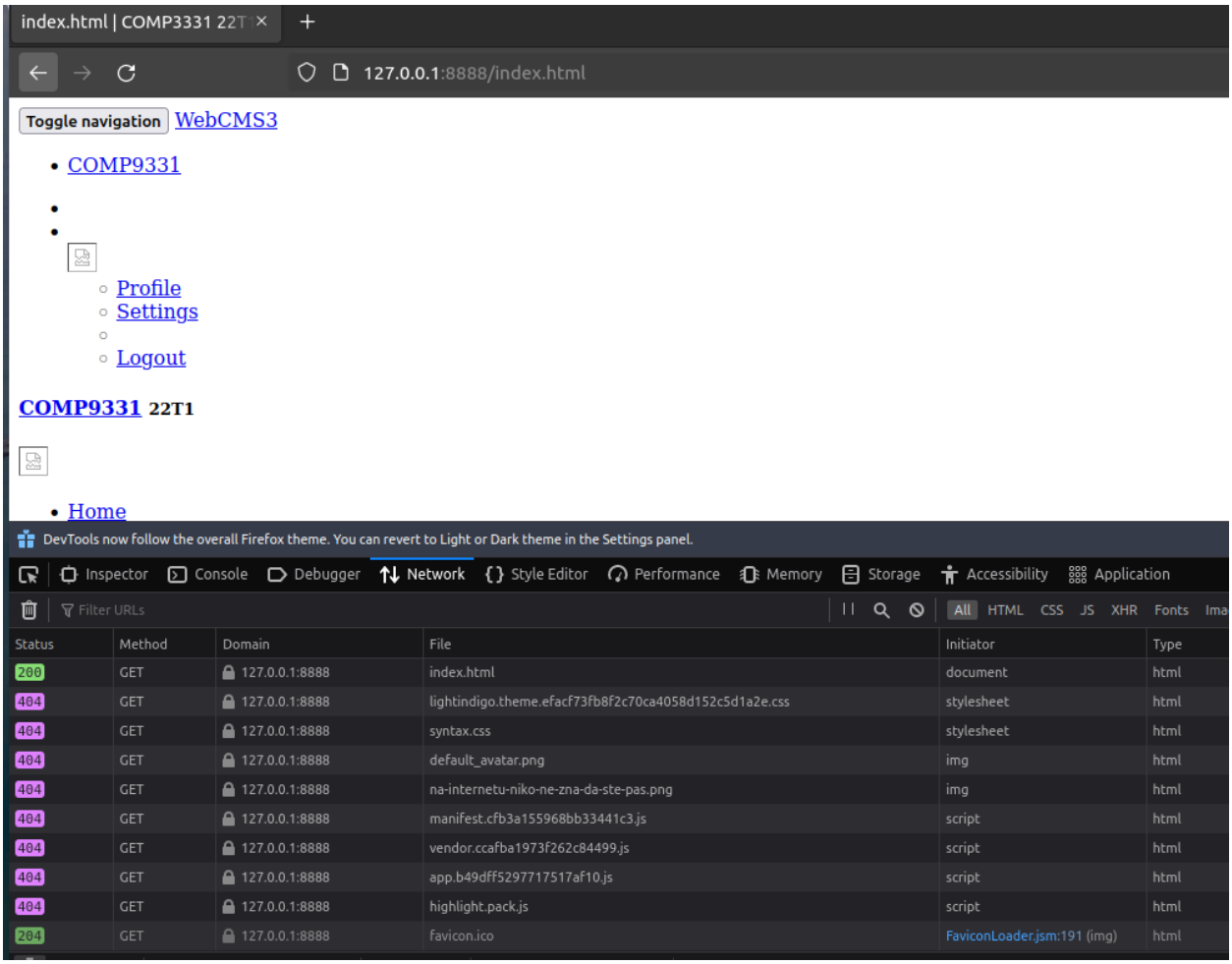
Exercise 4

Please see attached file `WebServer.py`

Question 1

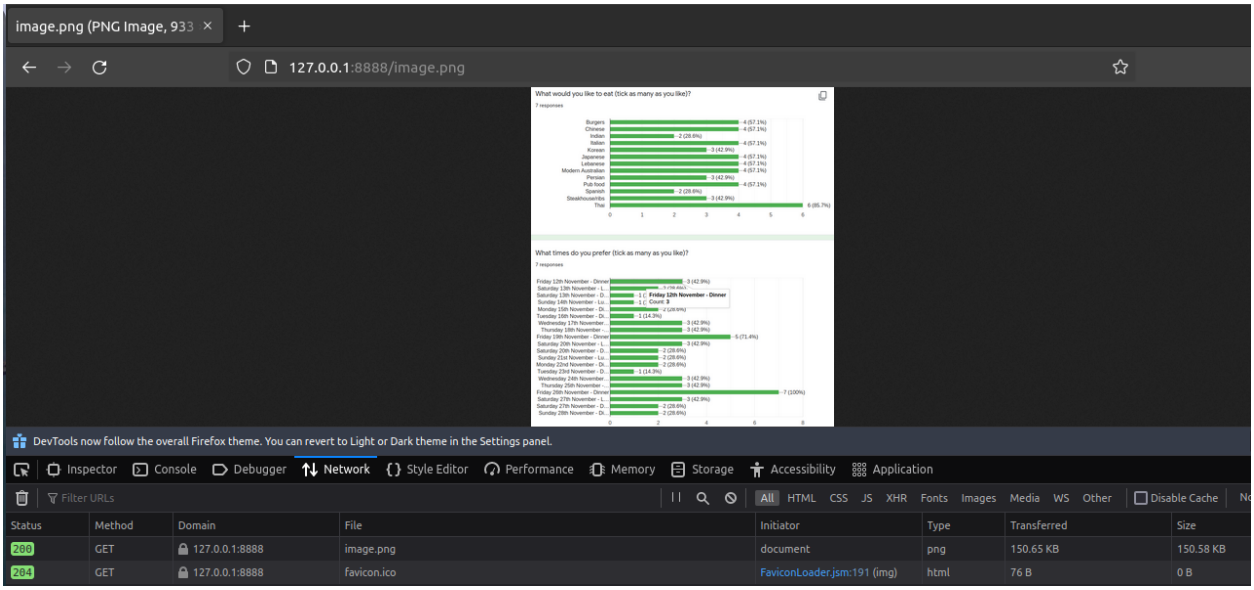


Terminal output on server side



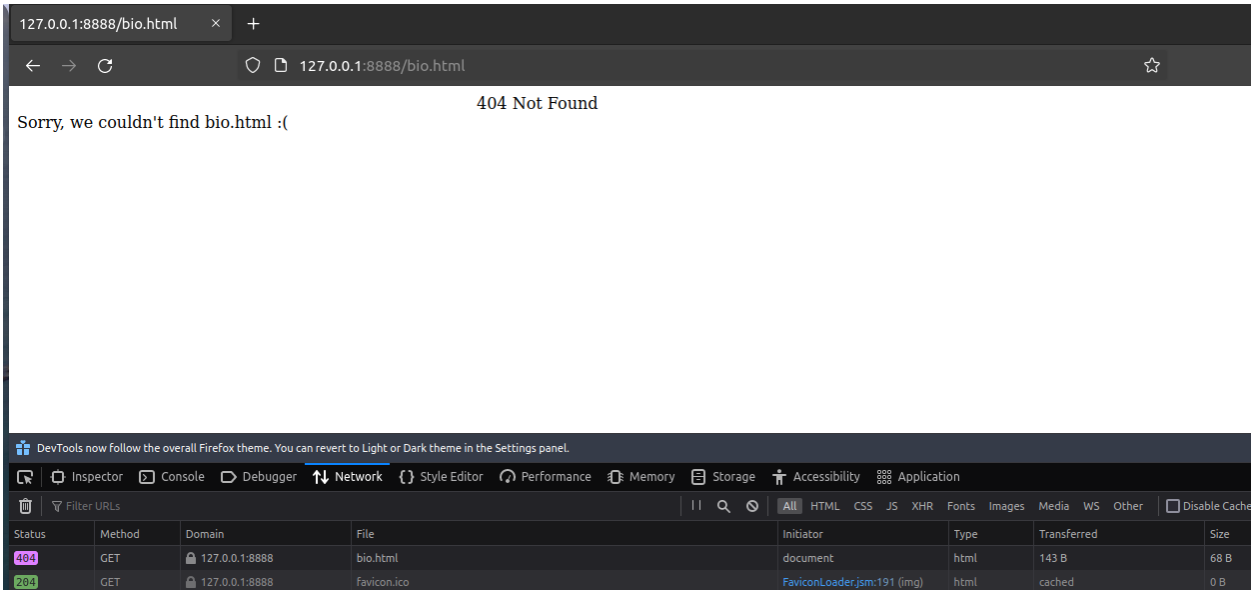
Client side HTTP request of “GET /index.html HTTP/1.1”.

Question 2



http://127.0.0.1:8888/image.png

Question 3



http://127.0.0.1:8888/bio.html and the 404 not found error

Code

```
"""
COMP9331 Lab 3
z3330164
Andrew Lau

Usage:
```

```
python3 WebServer.py [Port number, defaults to 5000]

Note that I have used the below reference as a starting point:
https://bhch.github.io/posts/2017/11/writing-an-http-server-from-scratch/
"""
import socket
import os
import mimetypes
import datetime
import sys

class TCPServer:
    """
    Basic TCP server upon which HTTP servers will build upon via inheritence
    This basic TCP server just echoes back any received messages
    """
    def __init__(self, host='127.0.0.1', port=5000):  # default to localhost and port 5000
        self.host = host
        self.port = port

    def start(self):
        # initiating socket object
        my_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        # setting socket host and port
        my_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        my_socket.bind((self.host, self.port))
        my_socket.listen(5)

        print("*" * 50, "\nNow listening at host", my_socket.getsockname()[0], 'and port', my_socket.getsockname()[1])

        while True:
            connection, address = my_socket.accept()  # connection is a new socket object
            print("*" * 50, "\nTCP connection established")
            print("\ttime:", str(datetime.datetime.now()))
            print("\thost:", address[0])
            print("\tport:", address[1])

            data_received = connection.recv(4096)  # buffer size 4096

            # request handling here, subsequent classes will implement more sophisticated HTTP GET request handling
            response = self.handle_request(data_received)

            # send response back
            connection.sendall(response)

            # close connection once done
            connection.close()

    def handle_request(self, data_received):
        """
        basic request handling (echo)
        this method is overwritten in the HTTPServer class' handle_request
        """
        print("Sending the below message to client:")
        print(data_received.decode())
        return data_received

class HTTPRequest:
    """
    Need to be able to parse HTTP requests like
        GET /image.png HTTP/1.1
    """
    def __init__(self, data_received):

        self.HTTP_method = None  # eg GET
        self.URL = None  # eg /index.html
        self.HTTP_version = None # eg HTTP/1.1

        # immediately parse HTTP request upon instantiation
        self.parse(data_received)
```

```
    def parse(self, data_received):
        lines = data_received.split(b"\r\n")  # each line in the HTTP request is delimited by a carriage return/newline

        request_line = lines[0]

        words = request_line.split(b" ")

        self.HTTP_method = words[0].decode() # bytes to string

        # if a URI is provided (rather than implicit request for index.html)
        if len(words) > 1:
            self.uri = words[1].decode() # call decode to convert bytes to str

        elif len(words) > 2:
            self.HTTP_version = words[2]

class HTTPServer(TCPServer):
    """
    HTTP Server class built on top of TCP class
    """
    def __init__(self, host_HTTP='127.0.0.1', port_HTTP=5000):
        super(HTTPServer, self).__init__(host=host_HTTP, port=port_HTTP)  # inherit from super class
        self.headers = {'Server': 'COMP9331Server', 'Content-Type': 'text/html'}  # header info
        self.status_codes = {200: 'OK', 204:'No Content', 404: 'Not Found', 501: 'Not Implemented'}  # HTTP status codes

    def handle_request(self, data_received):
        request = HTTPRequest(data_received)

        print('Looking for request.HTTP_method', request.HTTP_method)

        try:
            handler = getattr(self, f'handle_{request.HTTP_method}')
        except AttributeError:  # we don't have a handler for that method
            print(request.HTTP_method, "not found, 501 error")
            handler = self.HTTP_501_handler

        print("Using handler", handler)
        response = handler(request)
        print("Sending response", response[:500])
        return response

    def HTTP_501_handler(self, request):
        """
        When the HTTP method hasn't been implemented
        """
        response_line = self.response_line(status_code=501)

        response_headers = self.response_headers()

        blank_line = b"\r\n"

        response_body = b"<marquee>501 Not Implemented</marquee>\nSorry, this HTTP method has not been implemented"

        return b"".join([response_line, response_headers, blank_line, response_body])

    def handle_GET(self, request):
        """
        HTTP GET request method
        """
        # strip forward slash to get the filename the client is trying to GET
        filename = request.uri.strip('/')

        response_headers = self.response_headers()
        response_body = b""
        blank_line = b"\r\n"

        # handle GET favicon.ico
        if filename.endswith('favicon.ico'):
            response_line = self.response_line(status_code=204)
```

```
        elif os.path.exists(filename):  # see if that file exists in our directory
            response_line = self.response_line(status_code=200)  # if we can find it, all good send back a 200

            # guess a file's MIME type
            content_type = mimetypes.guess_type(filename)[0]

            # if guess failed (returns none, then set to TEXT HTML)
            if not content_type:
                content_type = 'text/html'

            extra_headers = {'Content-Type': content_type}
            response_headers = self.response_headers(extra_headers)  # add context type as an extra header

            with open(filename, 'rb') as f:
                response_body = f.read()
        else:
            response_line = self.response_line(status_code=404)

            response_body = f"<marquee>404 Not Found</marquee>\nSorry, we couldn't find {filename} :("
            response_body = response_body.encode()

        # concat the bytes and return the response
        return b"".join([response_line, response_headers, blank_line, response_body])


    def response_line(self, status_code):
        """
        Response line with HTTP/1.1 and the mapped status code and reason"""
        reason = self.status_codes[status_code]
        line = "HTTP/1.1 %s %s\r\n" % (status_code, reason)

        return line.encode() # call encode to convert str to bytes

    def response_headers(self, extra_headers=None):
        """
        Generates header line of HTTP response
        """
        headers_copy = self.headers.copy() # make a local copy of headers

        if extra_headers:
            headers_copy.update(extra_headers)

        headers = ""

        for h in headers_copy:
            headers += "%s: %s\r\n" % (h, headers_copy[h])

        return headers.encode() # call encode to convert str to bytes

if __name__ == '__main__':
    # allow command line arguments, otherwise take default port as 5000
    if len(sys.argv) == 2:
        server_port = int(sys.argv[1])
    else:
        server_port = 5000  #change this port number if required

    server = HTTPServer(port_HTTP=server_port)
    # server = TCPServer()
    server.start()
```

```
z3330164@vx2:/tmp_amd/ravel/export/ravel/2/z3330164/Desktop$ dig @129.94.172.11 vx2.orchestra.cse.unsw.EDU.AU A

; <<>> DiG 9.9.5-9+deb8u19-Debian <<>> @129.94.172.11 vx2.orchestra.cse.unsw.EDU.AU A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18763
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;vx2.orchestra.cse.unsw.EDU.AU. IN      A

;; ANSWER SECTION:
vx2.orchestra.cse.unsw.EDU.AU. 3600 IN  A       129.94.242.115

;; AUTHORITY SECTION:
orchestra.cse.unsw.EDU.AU. 3600 IN      NS      maestro.orchestra.cse.unsw.EDU.AU.
orchestra.cse.unsw.EDU.AU. 3600 IN      NS      beethoven.orchestra.cse.unsw.EDU.AU.

;; ADDITIONAL SECTION:
maestro.orchestra.cse.unsw.EDU.AU. 3600 IN A    129.94.242.33
beethoven.orchestra.cse.unsw.EDU.AU. 3600 IN A  129.94.242.2

;; Query time: 0 msec
;; SERVER: 129.94.172.11#53(129.94.172.11)
;; WHEN: Sun Mar 13 15:40:46 AEDT 2022
;; MSG SIZE  rcvd: 152
```