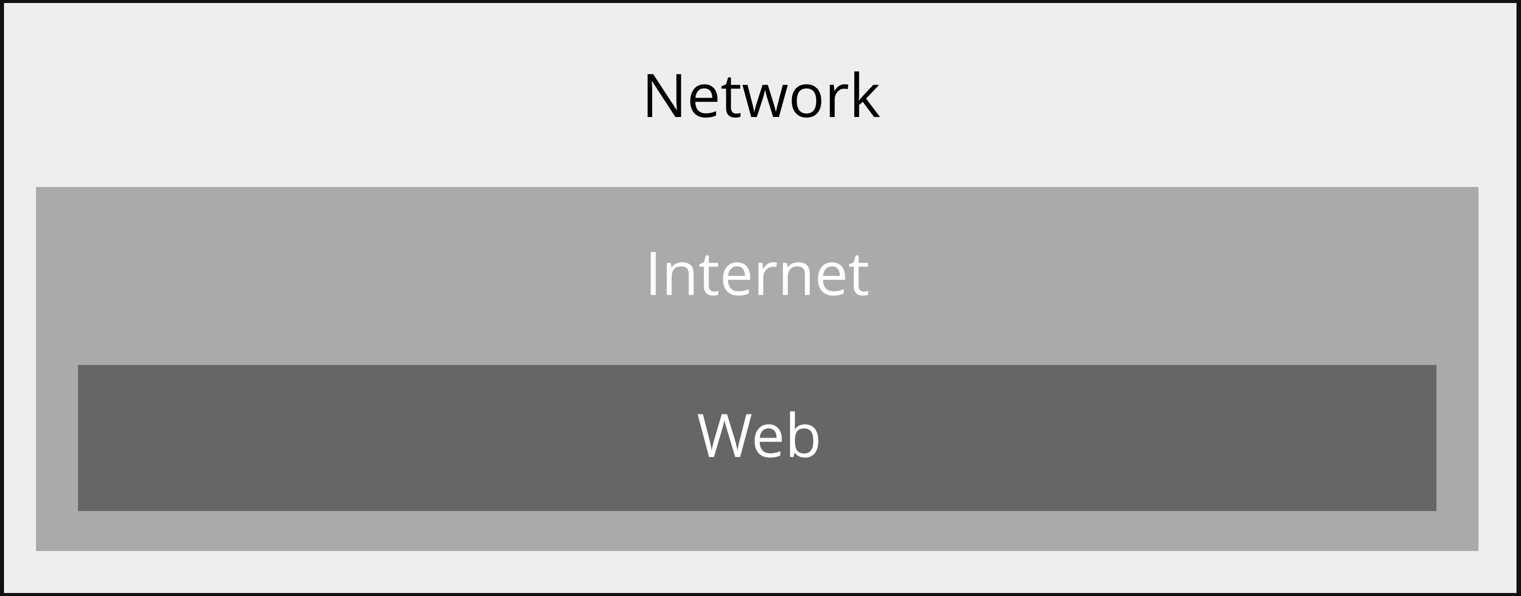
COMP1531

4.1 - HTTP, Flask

First Things

- Feedback form will be sent out for the course
- Sometimes our lectures won't cover everything:
  - Teach yourself
  - Help others
  - Wait until we teach it
- Some lecture code won't be fully pylint compliant due to screen size restrictions
  - In particular "docstring"

Computer Networks



The network

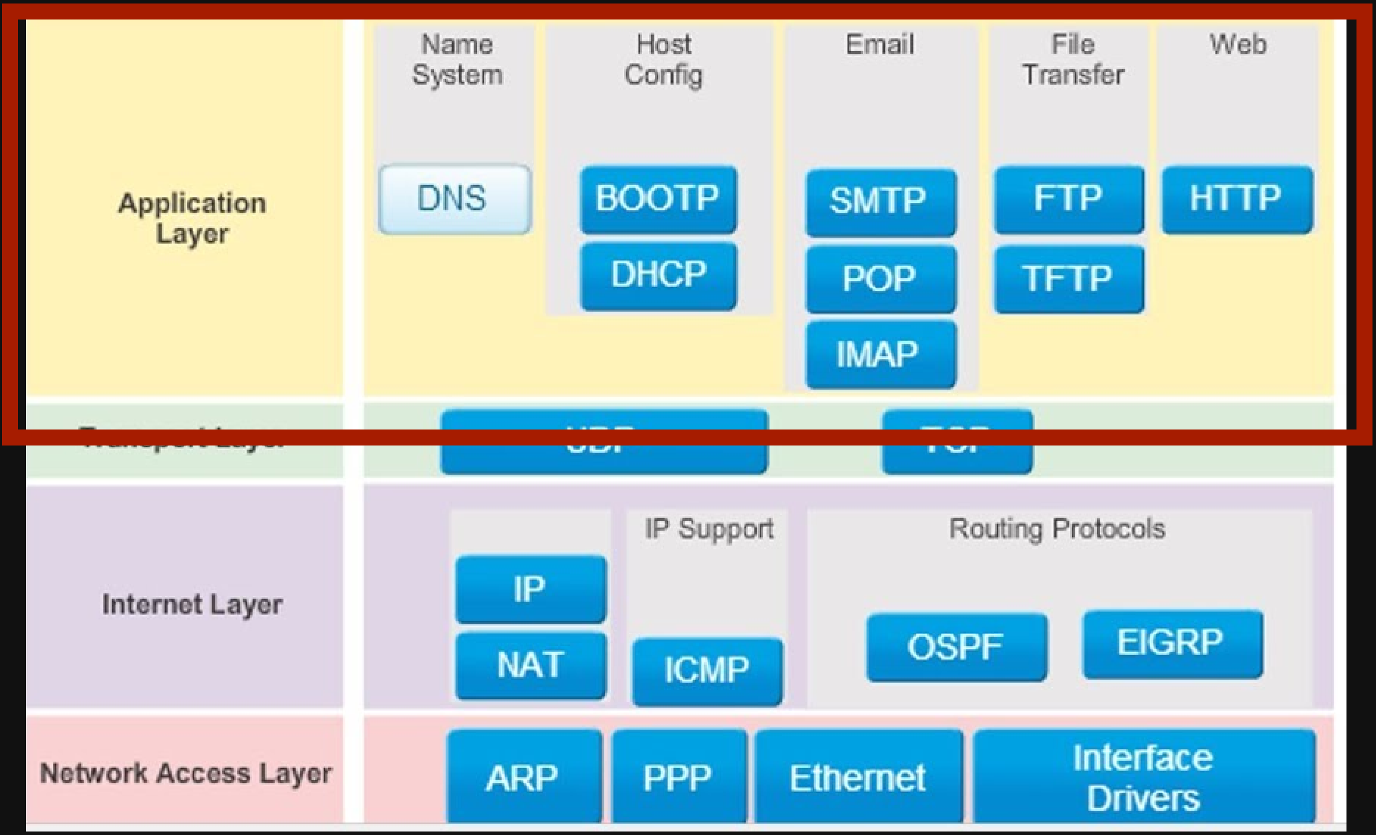
This is not a networking course:

- **Network**: A group of interconnected computers that can communicate
- **Internet**: A global infrastructure for networking computers around the entire world together
- **World Wide Web**: A system of documents and resources linked together, accessible via URLs

Network Protocols

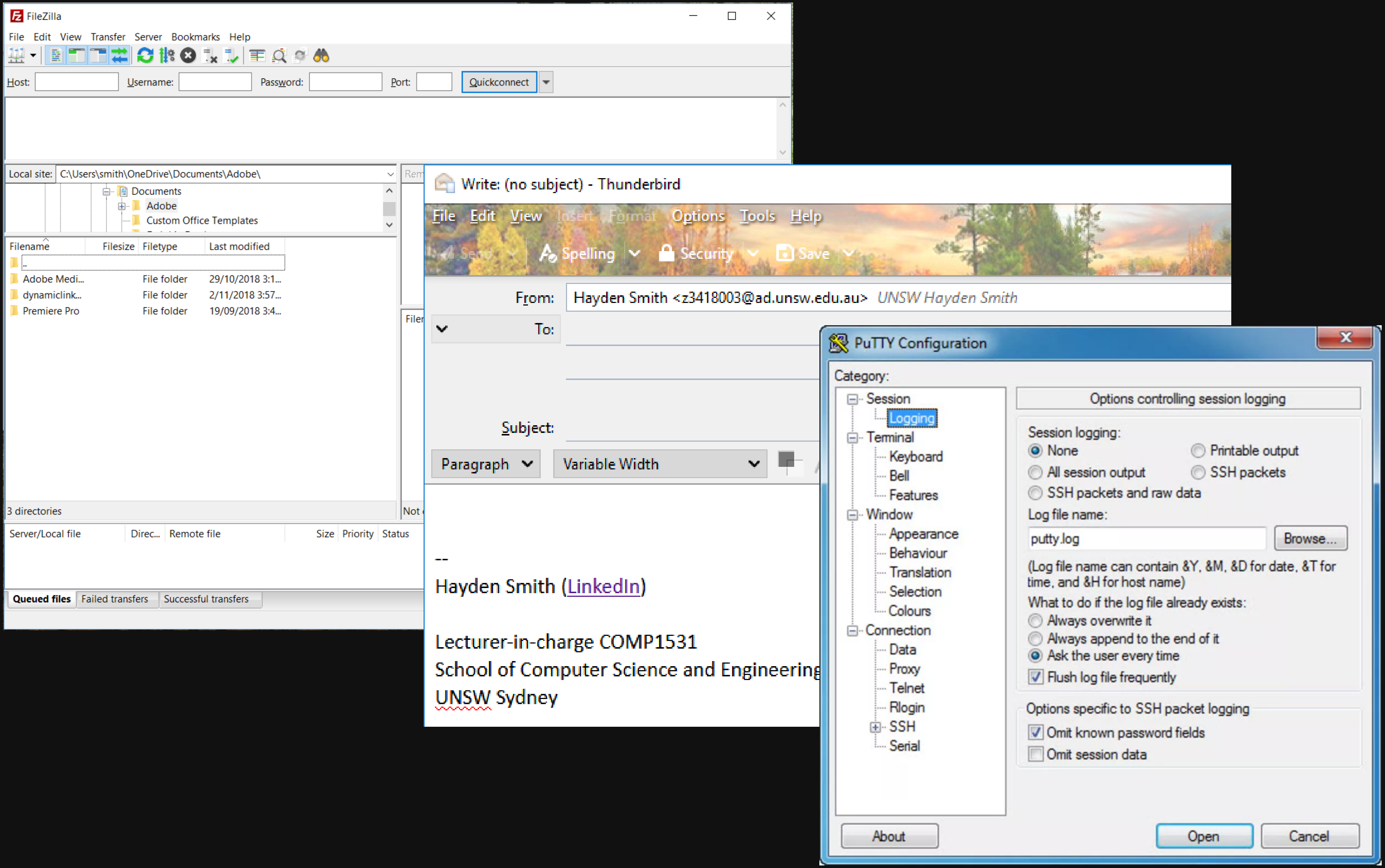
- Communication over networks must have a certain "structure" so everyone can understand
- Different "structures" (protocols) are used for different types of communication

Network Protocols



Source

Examples?



Host: Username: Password: Port:

Local site: C:\Users\smith\OneDrive\Documents\Adobe\

Documents
Adobe
Custom Office Templates

Filename	Filesize	Filetype	Last modified
..			
Adobe Medi...		File folder	29/10/2018 3:1...
dynamiclink...		File folder	2/11/2018 3:57...
Premiere Pro		File folder	19/09/2018 3:4...

3 directories

Server/Local file	Direc...	Remote file	Size	Priority	Status

From: Hayden Smith <z3418003@ad.unsw.edu.au> *UNSW Hayden Smith*

To:

Subject:

--
Hayden Smith (LinkedIn)

Lecturer-in-charge COMP1531
School of Computer Science and Engineering
UNSW Sydney

Category:
- Session
  - Logging
- Terminal
  - Keyboard
  - Bell
  - Features
- Window
  - Appearance
  - Behaviour
  - Translation
  - Selection
  - Colours
- Connection
  - Data
  - Proxy
  - Telnet
  - Rlogin
  - SSH
  - Serial

Options controlling session logging

Session logging:
[✓ None] Printable output
All session output SSH packets
SSH packets and raw data

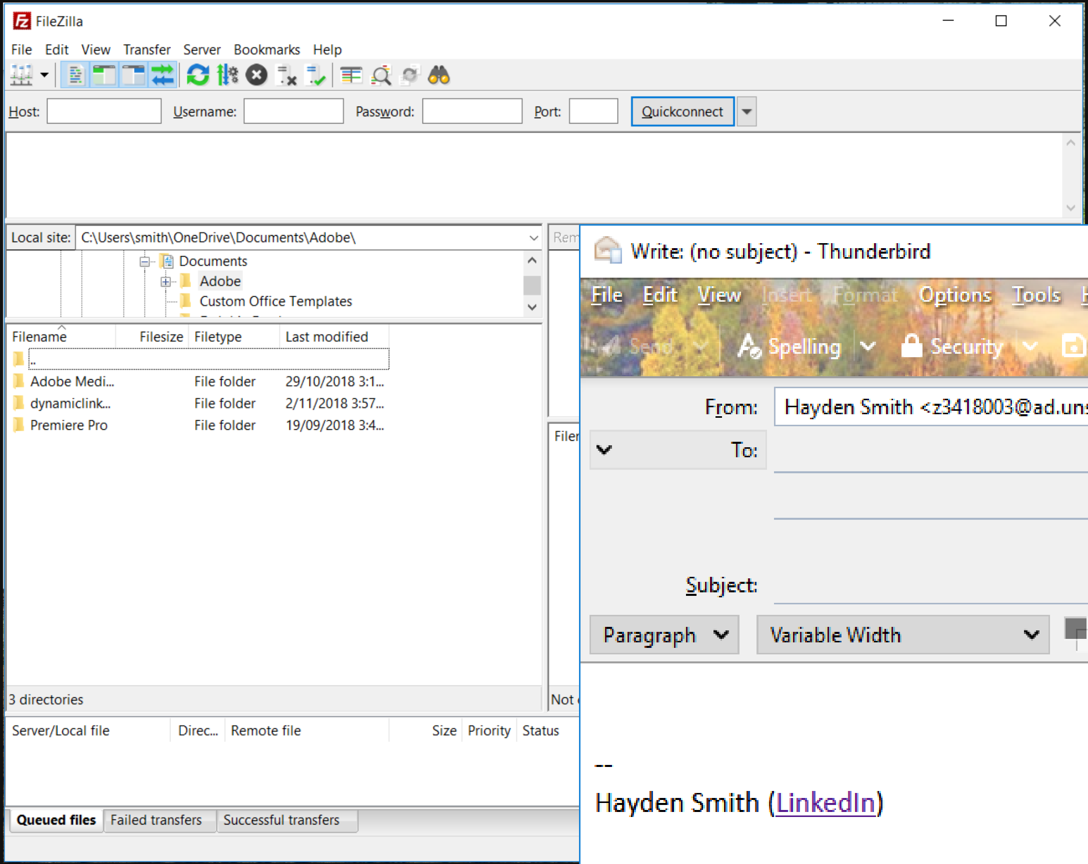
Log file name:
putty.log

(Log file name can contain &Y, &M, &D for date, &T for time, and &H for host name)

What to do if the log file already exists:
Always overwrite it
Always append to the end of it
[✓ Ask the user every time]
[✓ Flush log file frequently]

Options specific to SSH packet logging
[✓ Omit known password fields]
Omit session data

Examples?


FileZilla
File Edit View Transfer Server Bookmarks Help
Host: Username: Password: Port: Quickconnect
Queued files | Failed transfers | Successful transfers

Local site: C:\Users\smith\OneDrive\Documents\Adobe\

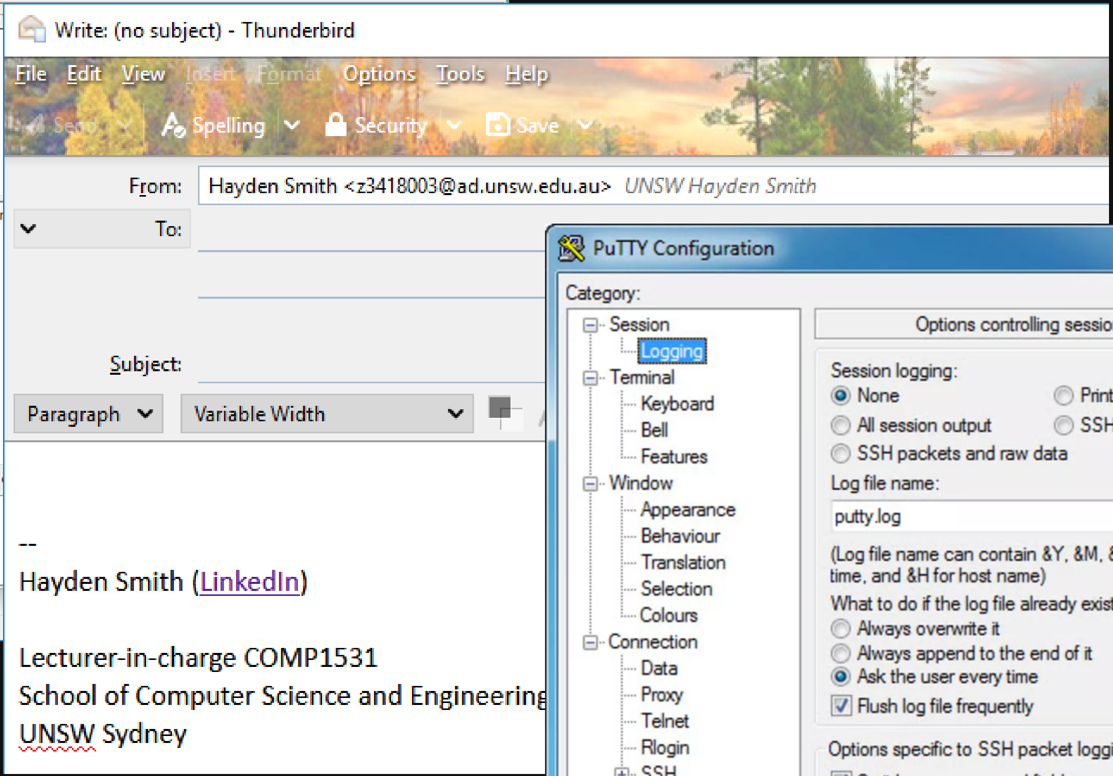
Documents
Adobe
Custom Office Templates

Filename	Filesize	Filetype	Last modified
..			
Adobe Medi...		File folder	29/10/2018 3:1...
dynamiclink...		File folder	2/11/2018 3:57...
Premiere Pro		File folder	19/09/2018 3:4...

3 directories

Server/Local file	Direc...	Remote file	Size	Priority	Status

FTP


Write: (no subject) - Thunderbird
File Edit View Insert Format Options Tools Help
Send Spelling Security Save
Paragraph
Variable Width

From: Hayden Smith <z3418003@ad.unsw.edu.au> *UNSW Hayden Smith*

To:

Subject:

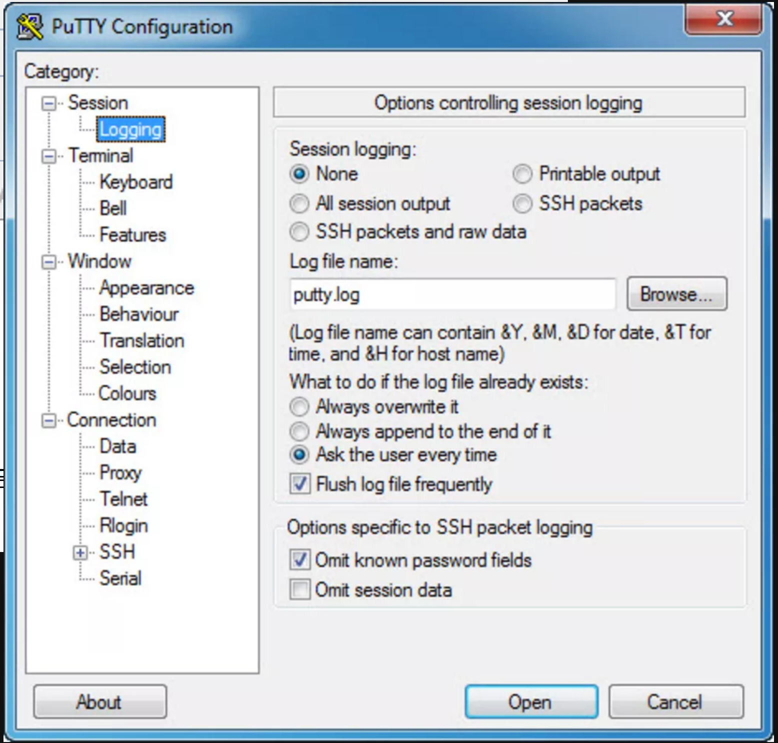
--

Hayden Smith (LinkedIn)

Lecturer-in-charge COMP1531
School of Computer Science and Engineering
UNSW Sydney

IMAP

SSH



Options controlling session logging

Session logging:
[● None] ○ Printable output
○ All session output ○ SSH packets
○ SSH packets and raw data

Log file name: putty.log

(Log file name can contain &Y, &M, &D for date, &T for time, and &H for host name)

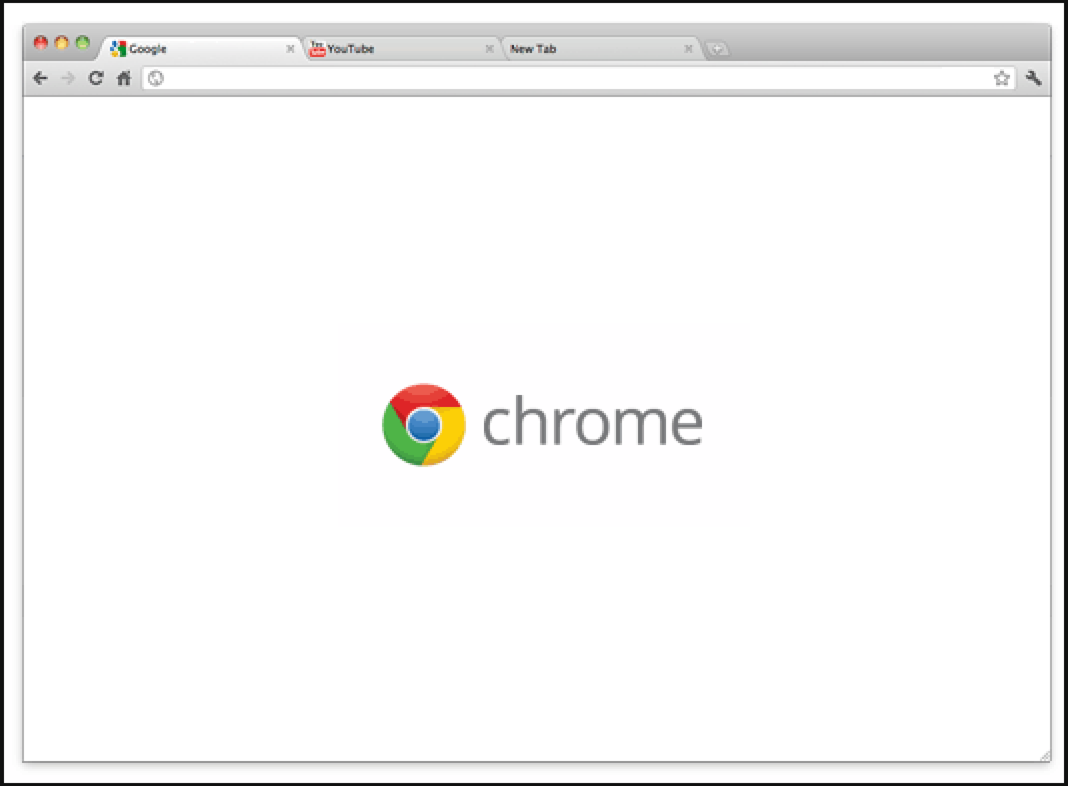
What to do if the log file already exists:
○ Always overwrite it
○ Always append to the end of it
[● Ask the user every time]
[✓] Flush log file frequently

Options specific to SSH packet logging
[✓] Omit known password fields
[ ] Omit session data

HTTP

HTTP: Hypertext Transfer Protocol

I.E. Protocol for sending and receiving HTML documents (nowadays much more)



Request →

Response ←

HTTP Request & Response

HTTP Request

```
GET /hello HTTP/1.1
Host: 127.0.0.1:5000
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Ge
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;
Sec-Fetch-Site: none
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
```

HTTP Response

```
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 12
Server: Werkzeug/0.16.0 Python/3.5.3
Date: Wed, 09 Oct 2019 13:21:51 GMT

Hello world!
```

Flask

Lightweight HTTP web server built in python

flask1.py

```python
from flask import Flask
APP = Flask(__name__)

@APP.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    APP.run()
```

```
$ python3 flask1.py
```

Server an image

Time to serve an image via a flask server...

flask2.py

```
from flask import Flask, send_file
APP = Flask(__name__)

@APP.route("/img")
def img():
    return send_file('./cat.jpg', mimetype='image/jpg')

if __name__ == "__main__":
    APP.run()
```

```
$ python3 flask2.py
```

Flask Reloading

Lightweight HTTP web server built in python

flask1.py

```python
from flask import Flask
APP = Flask(__name__)

@APP.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    APP.run()
```

```
$ FLASK_APP=flask1.py
$ FLASK_DEBUG=1
$ flask run
```

Learn More

Some tutorials include:

1. https://pythonspot.com/flask-web-app-with-python/
2. https://blog.miguelgrinberg.com/post/designing-a-restful-api-with-python-and-flask

When it comes to online tutorials, note that:

- Each "tutorial" may be using different python versions
- Each "tutorial" may have different aims in mind

Talking to Flask

How can we talk to flask?

Ideas?

Talking to Flask

How can we talk to flask?

1. cURL
2. API client
3. Web Browser

Curl (cURL)

- cURL Stands for "Client URL"
- Is a common line tool for making network requests to particular URLs
    - We will be using it only for HTTP
- curl also has bindings to a range of libraries that allow it to be used within languages like C, python

```bash
# A bash HTTP GET example
$ curl 'http://127.0.0.1:5000/hello'
```

API Client (Postman)

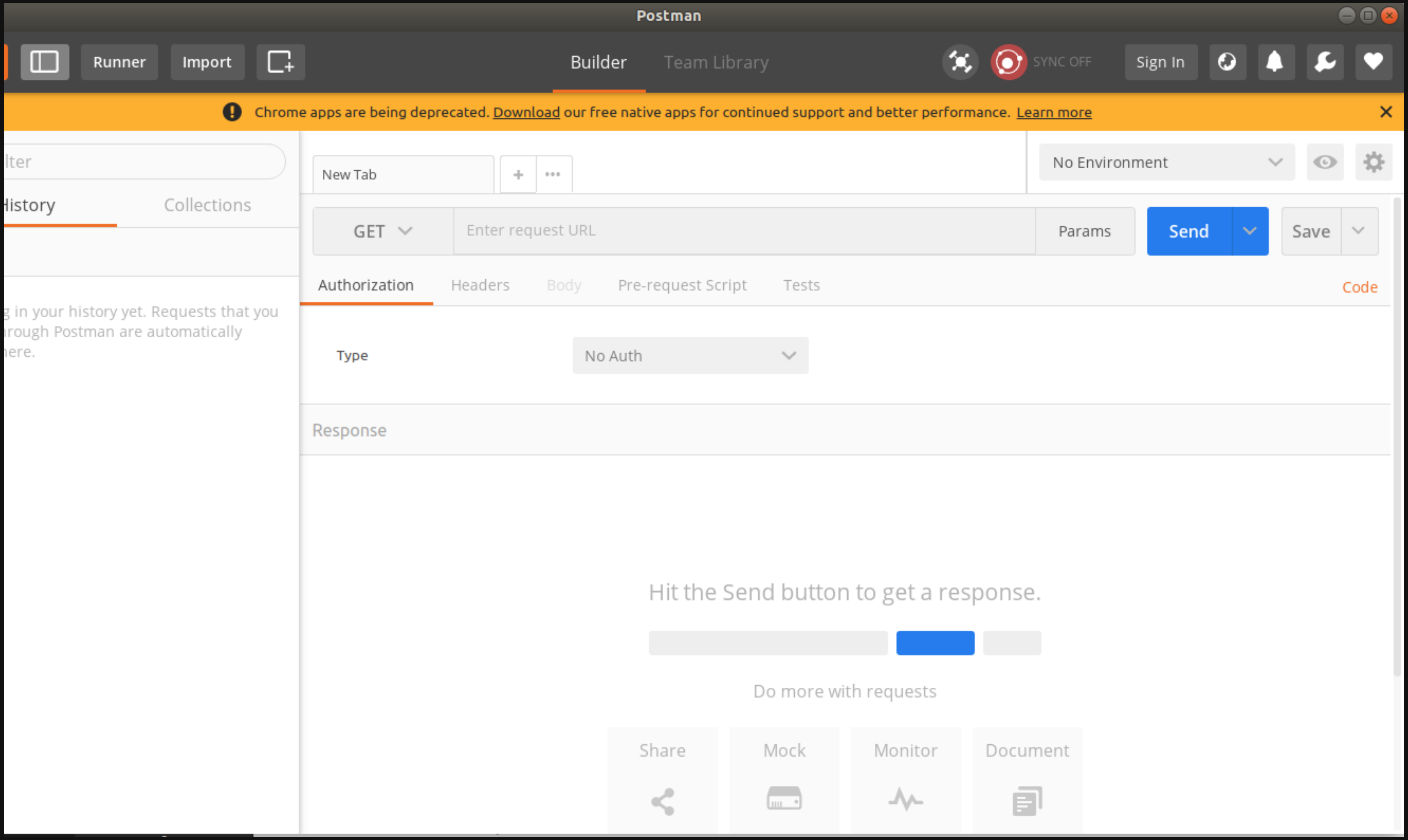
How to download/install postman:

- Open google chrome
- Google "postman chrome addon"
- Install the addon and open it
- Follow the demo in the lectures

Other notes:

- There are many of these types of apps
- Think of it like "a GUI for cURL"
- You may be expected to use postman (or equivalent) in a final exam

API Client (Postman)


Chrome apps are being deprecated. Download our free native apps for continued support and better performance. Learn more

lter

istory Collections

g in your history yet. Requests that you
rough Postman are automatically
ere.

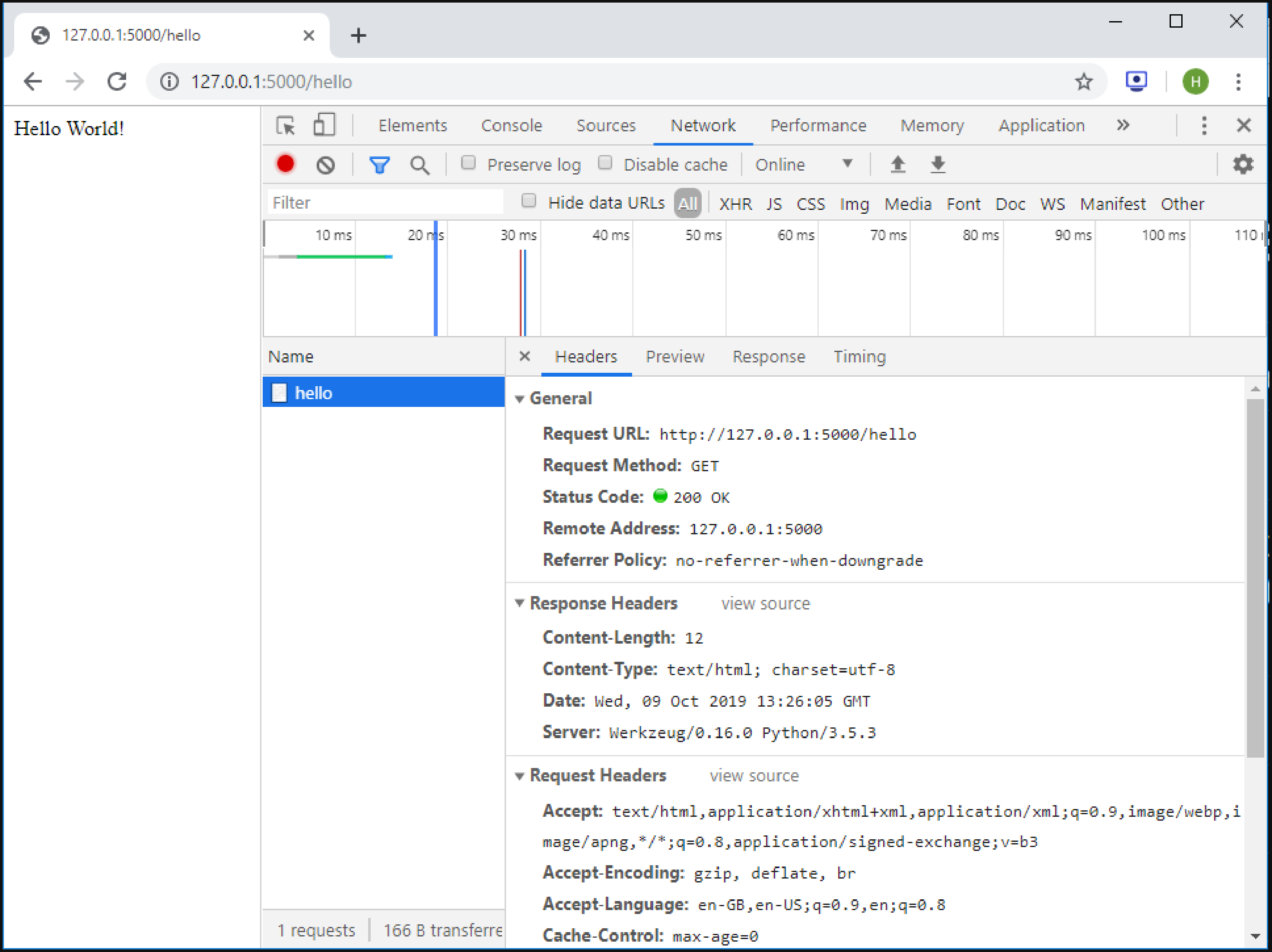
Type No Auth

Response

Hit the Send button to get a response.

Do more with requests

Web Browser



Hello World!

hello

General

Request URL: `http://127.0.0.1:5000/hello`
Request Method: `GET`
Status Code: ● `200 OK`
Remote Address: `127.0.0.1:5000`
Referrer Policy: `no-referrer-when-downgrade`

Response Headers view source

Content-Length: `12`
Content-Type: `text/html; charset=utf-8`
Date: `Wed, 09 Oct 2019 13:26:05 GMT`
Server: `Werkzeug/0.16.0 Python/3.5.3`

Request Headers view source

Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3`
Accept-Encoding: `gzip, deflate, br`
Accept-Language: `en-GB,en-US;q=0.9,en;q=0.8`
Cache-Control: `max-age=0`

Restful API & "CRUD"

A *RESTful API* is an application program interface (*API*) that uses HTTP requests to GET, PUT, POST and DELETE data

GET, PUT, POST, DELETE are HTTP Methods

Method	**Operation**
POST	**C**reate
GET	**R**ead
PUT	**U**pdate
DELETE	**D**elete

Using CRUD and state

Task:

Create a web server that uses CRUD to allow you to create, update, read, and delete a point via HTTP requests

Use a global variable to manage the state.

Iteration 2

Iteration 2 is now out

Key points of iteration 2:

- Implementing the backend via a flask server
- Using good coverage for tests
- Practicing good team and project methodologies (e.g. agile, stories)

Iteration 2

- **Notes**:
    - Please keep an eye on merge requests we send
        - Look at the git diff on gitlab to see what happened

- **Iteration-relevant content will be taught in lectures**:
    - State, authentication, authorisation, timers, will be covered in week 5~
    - Front-end will be covered in week 6/7~
        - The front-end will be released in week 6

A cool study

search.py

```python
def search_fn(token, query_str):
    return {
        'messages' : [
            'Hello ' + token + ' ' + query_str,
            # Not the right structure
        ]
    }
```

server.py

```python
from json import dumps
from flask import Flask, request

from search import search_fn

APP = Flask(__name__)

@APP.route('/search', methods=['GET'])
def search():
    return dumps(search_fn(request.args.get('token'), request.args.get('query_str')))

if __name__ == '__main__':
    APP.run()
```

An interesting question

How do companies track whether or not you've read an email they've sent you?