Week 6: collaboration tools

NRSC 7657 Workshop in Advanced Programming for Neuroscientists

course business

- We’re going to start group programming time during class this week.
- Questions for later:

(1) what is the purpose of your project

(2) what are you currently working/stuck on.

Collaboration

- As we'll all learn this afternoon, everyone is writing great code for their projects already (right?)

Collaboration

- As we'll all learn this afternoon, everyone is writing great code for their projects already (right?)
- If you want to share it with your lab-mates, or include it with your paper, or you've made a stand-alone tool that would be useful to the field, there are several ways to do so.

Collaboration

- As we'll all learn this afternoon, everyone is writing great code for their projects already (right?)
- If you want to share it with your lab-mates, or include it with your paper, or you've made a stand-alone tool that would be useful to the field, there are several ways to do so.
  - The best way (cost/benefit) depends on on your code and your audience (lab, other code-savvy users, non-coders, etc)

Collaboration

- As we'll all learn this afternoon, everyone is writing great code for their projects already (right?)
- If you want to share it with your lab-mates, or include it with your paper, or you've made a stand-alone tool that would be useful to the field, there are several ways to do so.
  - The best way (cost/benefit) depends on on your code and your audience (lab, other code-savvy users, non-coders, etc)
  - **No matter what, documentation is a key to effective code sharing**

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

docker

jupyterhub

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

docker

jupyterhub

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

Collaboration

Running code *not* on your computer

```python
class ThingWeMade:
    def __init__(self):
        self.thing='neuron'
        self.start()

    def start(self):
        self.thing2='neuron'

    def fire(self):
        self.spike = self.thing+' fires an action potential'
```

Collaboration

Running code *not* on your computer

- Easiest way: share your scripts/notebooks. This includes .m files.
  - Github: version control, so people can stay up to date if others update the code.
- Con: need to manage version control (python, MATLAB) and dependency management (python)
- Con: hardware limited to target machine, which may not be able or be efficient at running your code.
- Pro: hardware is owned by the user, so it is managed by them and not limited.

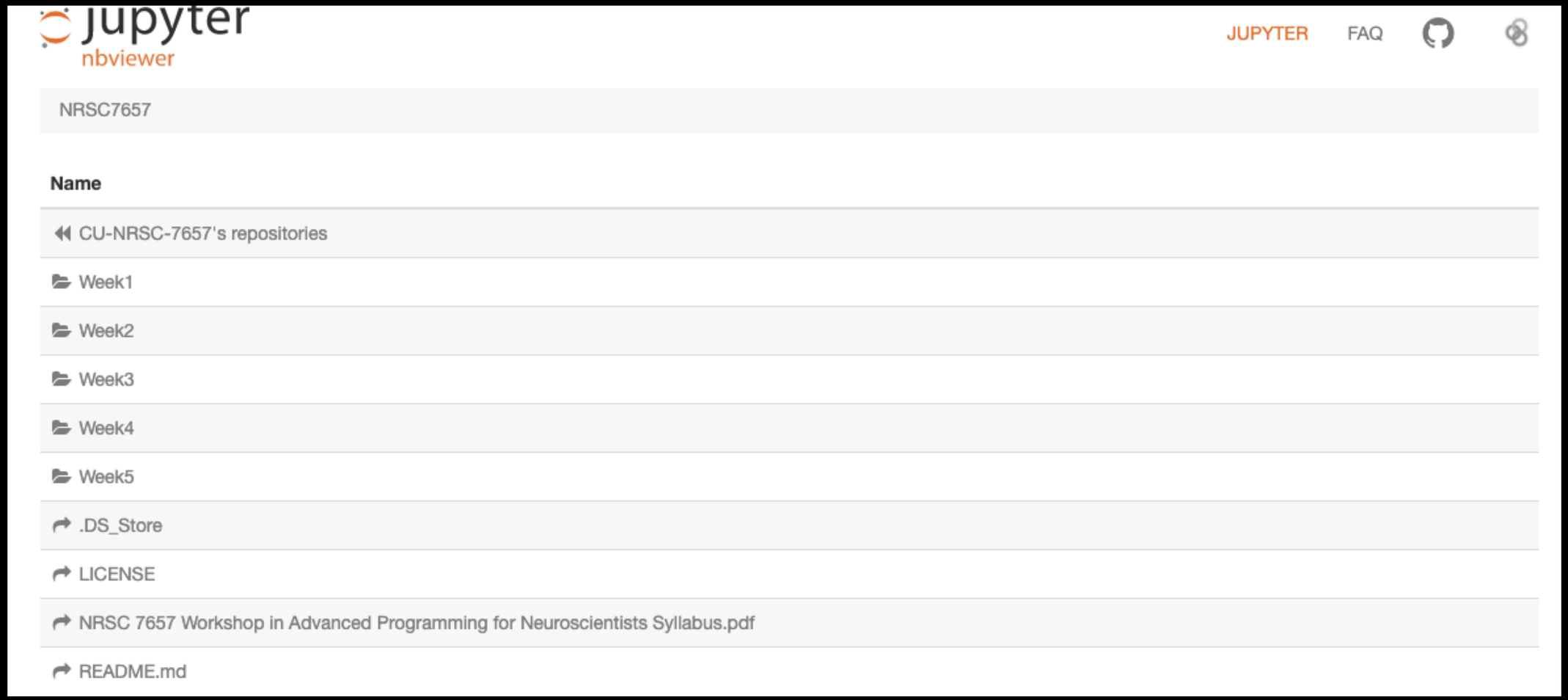
Collaboration

python sharing tools

- jupyter notebooks are rendered by GitHub automatically
- nbviewer.jupyter.org is another, more feature rich way to render notebooks over the internet (not on your localhost: like a regular jupyter server)
  - This is still viewing, not executing the notebook.

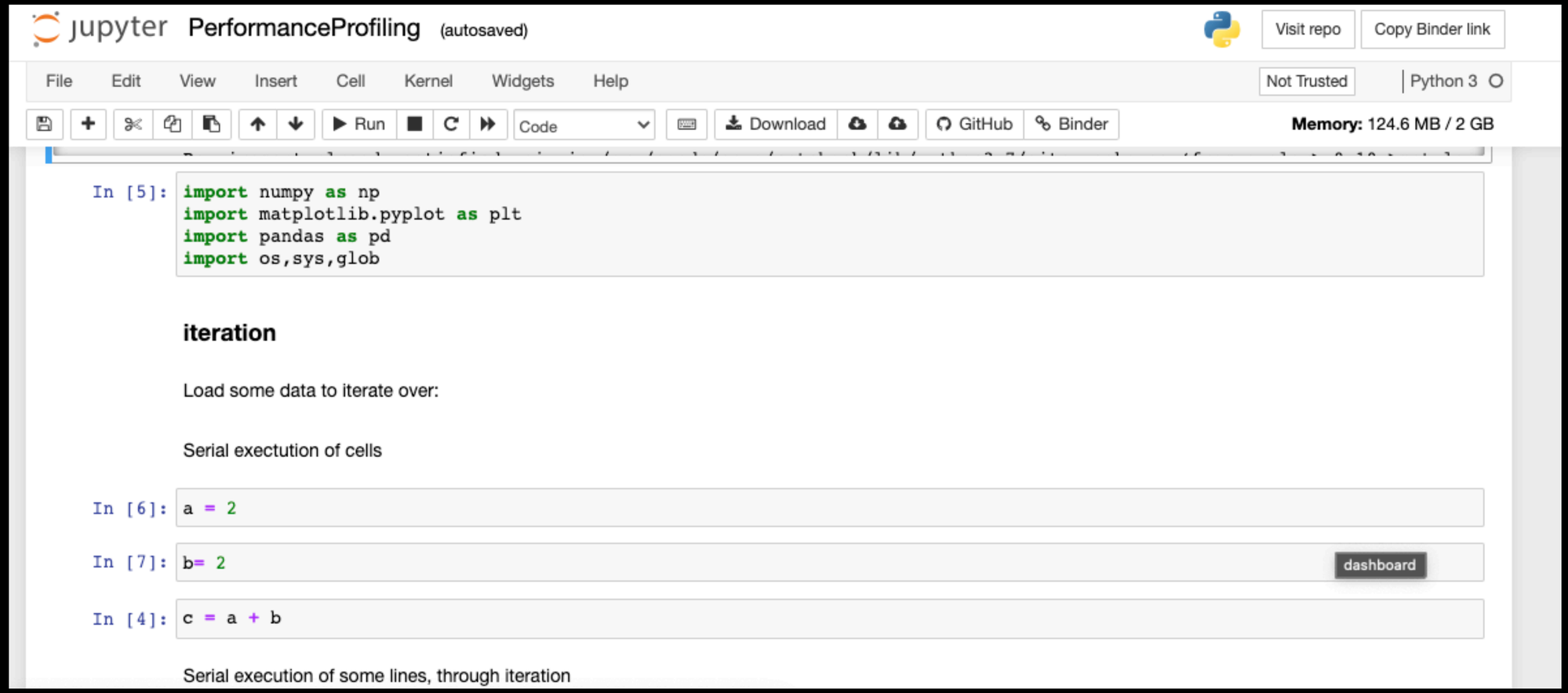
Collaboration

nbviewer



Collaboration

- mybinder.org



Collaboration

Containerization: docker

- Both nbviewer and mybinder.org use docker containers and virtual machines
  - nbviewer for creating a machine to render the code
  - binder uses docker to create a virtual machine that can run the code
- Because python (and jupyter) use tools that are integrated with the OS (python kernels), cloud servers (running server OSs) can more easily (and automatically) be configured for python

Collaboration

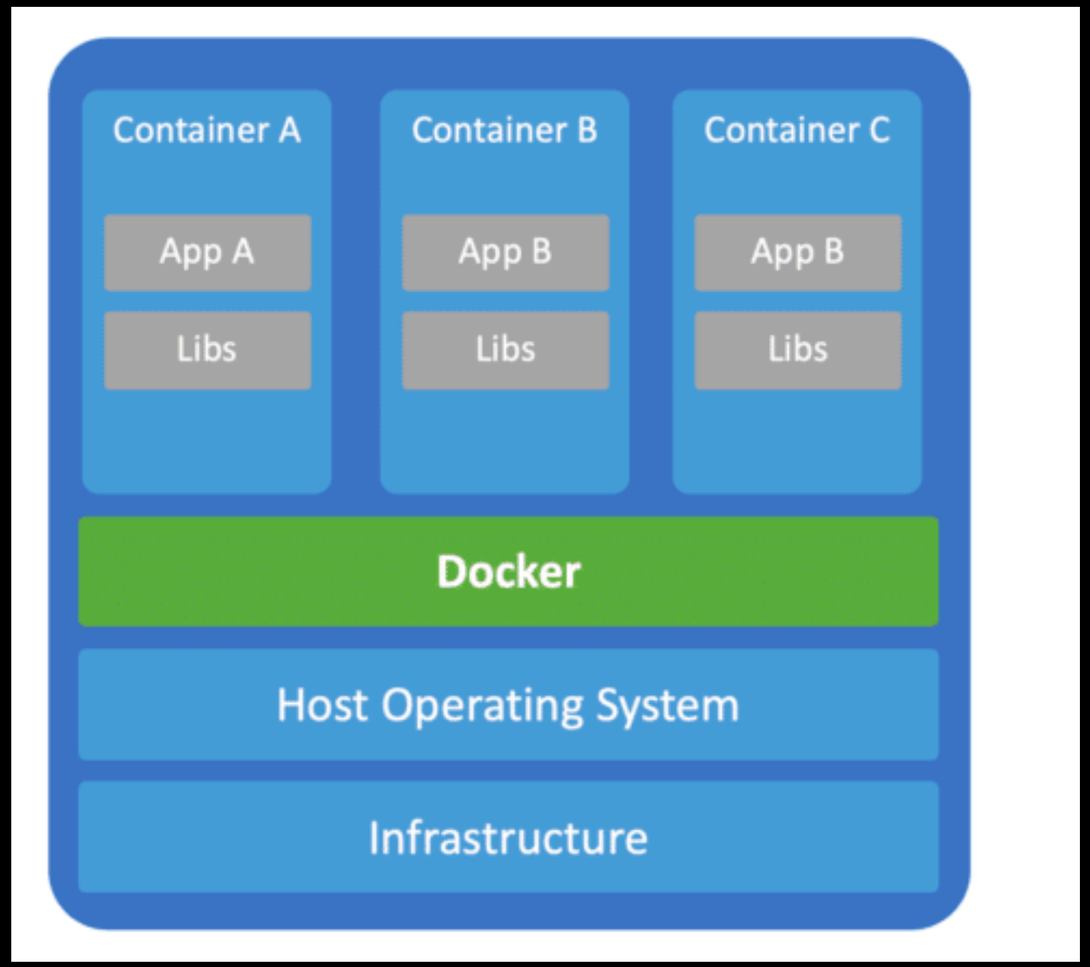
Containerization: docker

- Both nbviewer and mybinder.org use docker containers and virtual machines
  - nbviewer for creating a machine to render the code
  - binder uses docker to create a virtual machine that can run the code
- Because python (and jupyter) use tools that are integrated with the OS (python kernels), cloud servers (running server OSs) can more easily (and automatically) be configured for python

Collaboration

Containerization: docker

- What is Docker?
  - A platform to deliver self-sufficient packages of code from a host to a client.
  - Platform as a Service (PaaS)
  - A package (called an image) contains *all* of what is needed - binaries, libraries, config, runtime that can run your code as one process.

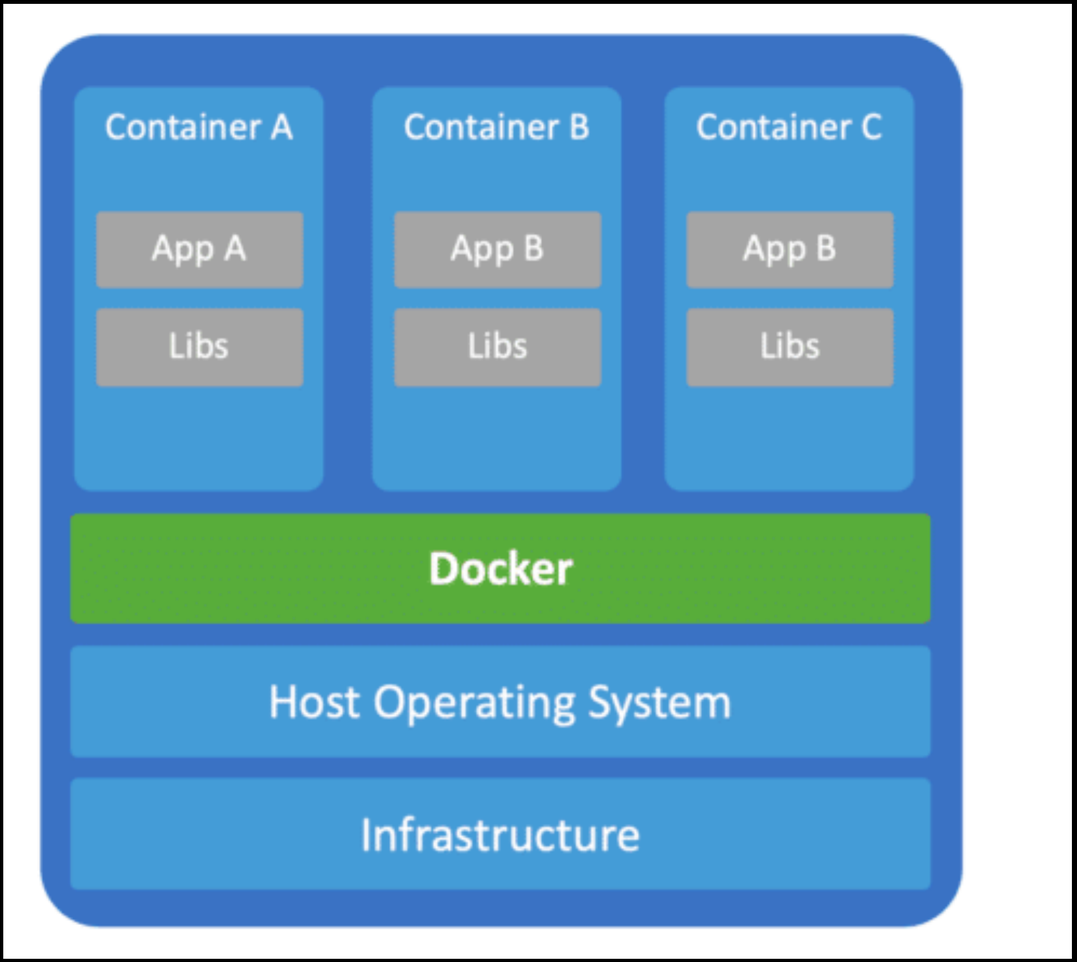


https://codingthesmartway.com/docker-beginners-guide-part-1-images-containers/

Collaboration

Containerization: docker

- What is Docker?
  - A platform to deliver self-sufficient packages of code from a host to a client.
  - Platform as a Service (PaaS)
  - A package (called an image) contains *all* of what is needed - binaries, libraries, config, runtime that can run your code as one process.



https://codingthesmartway.com/docker-beginners-guide-part-1-images-containers/

Collaboration

Containerization: docker + binder

https://github.com/alan-turing-institute/the-turing-way/blob/master/workshops/boost-research-reproducibility-binder/workshop-presentations/zero-to-binder-python.md#whats-happening-in-the-background---part-1

While you wait, BinderHub (the backend of Binder) is:

- Fetching your repo from GitHub
- Analysing the contents
- Creating a Docker file based on your repo
- Launching that Docker image in the Cloud
- Connecting you to it via your browser

Collaboration

Containerization: docker + binder

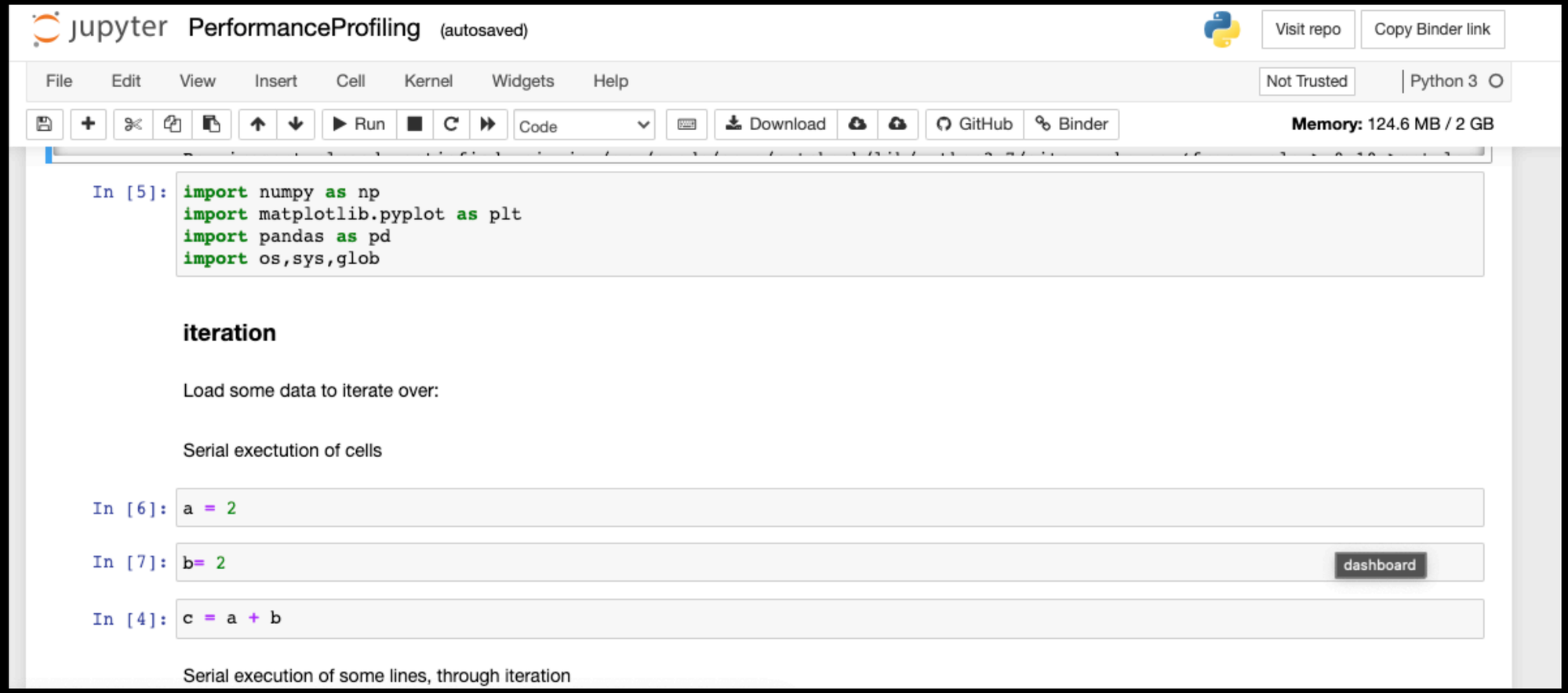
https://github.com/alan-turing-institute/the-turing-way/blob/master/workshops/boost-research-reproducibility-binder/workshop-presentations/zero-to-binder-python.md#whats-happening-in-the-background---part-1

While you wait, BinderHub (the backend of Binder) is:

- Fetching your repo from GitHub
- Analysing the contents
- Creating a Docker file based on your repo
- Launching that Docker image in the Cloud
- Connecting you to it via your browser

Collaboration

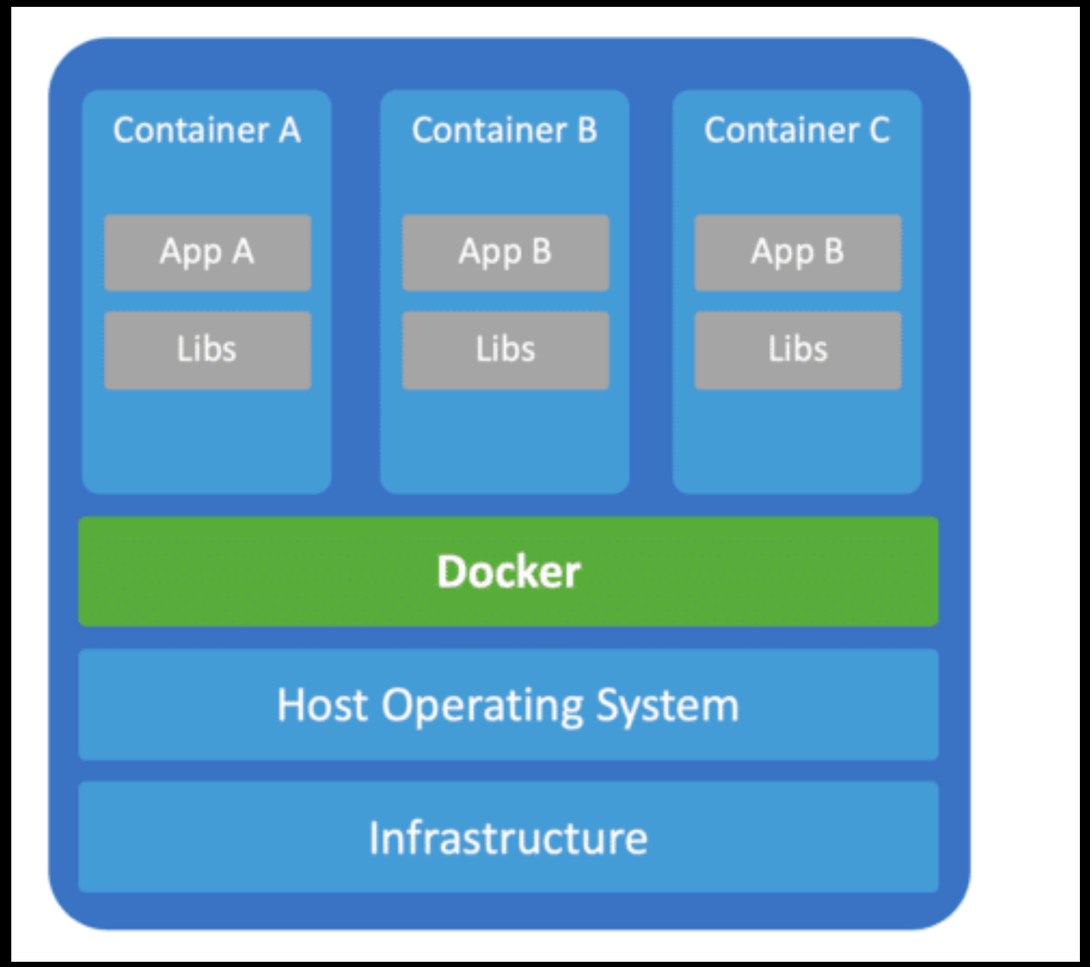
- mybinder.org



Collaboration

Containerization: docker + binder

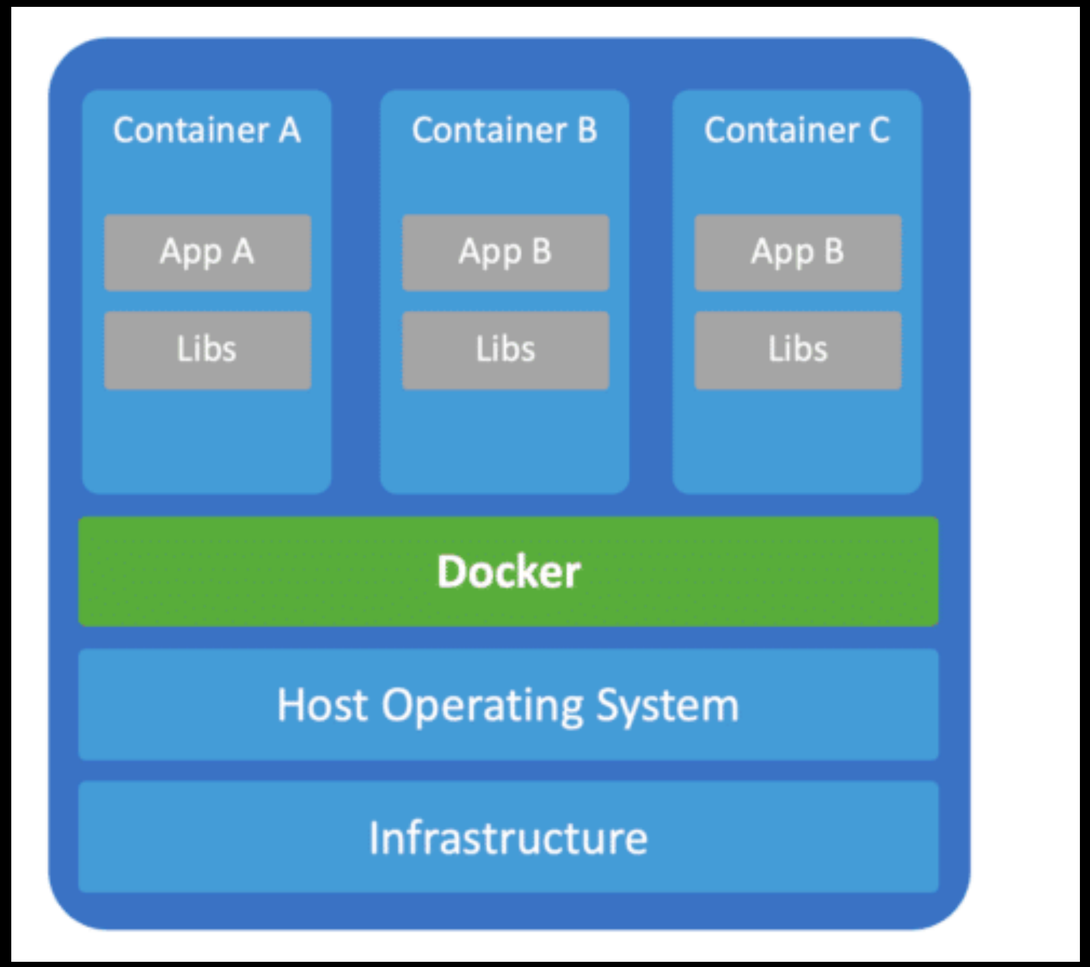
- What is the hardware



Collaboration

Containerization: docker + binder

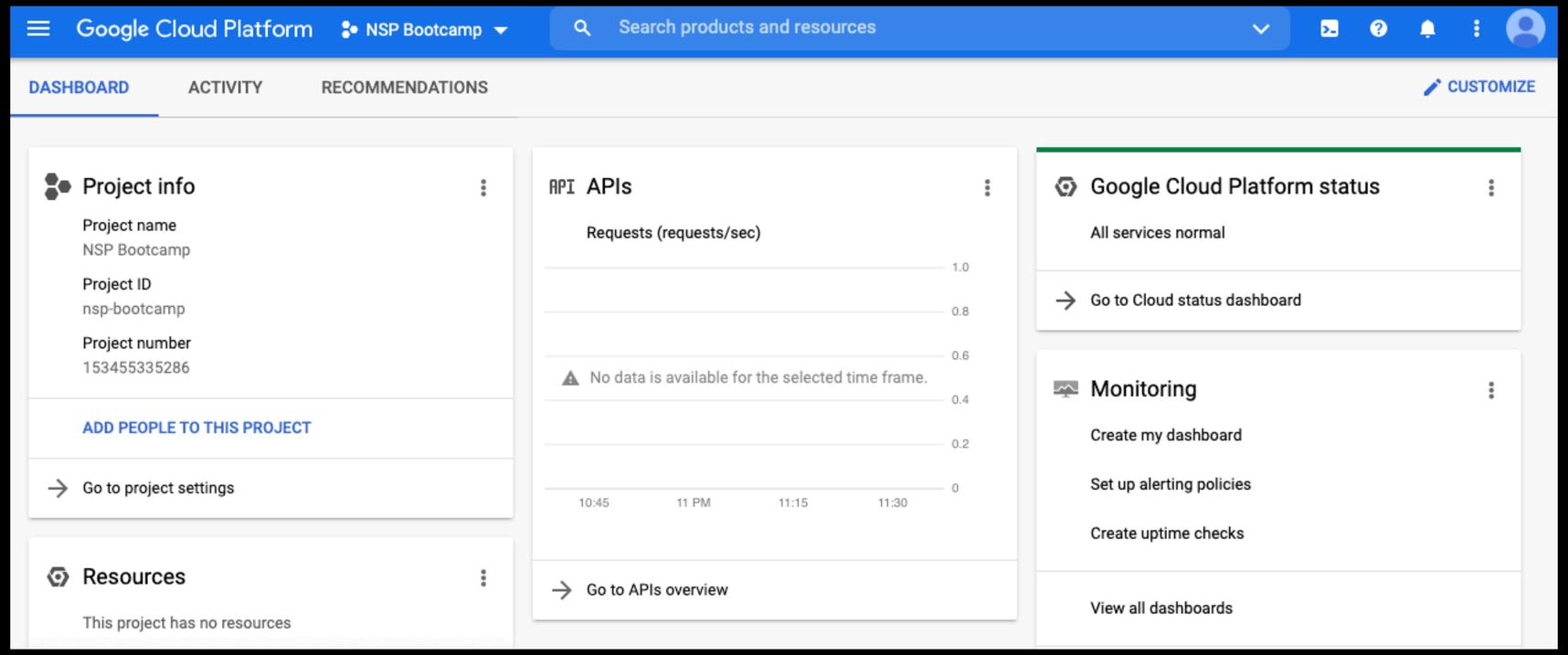
- What is the hardware



Collaboration

GCP

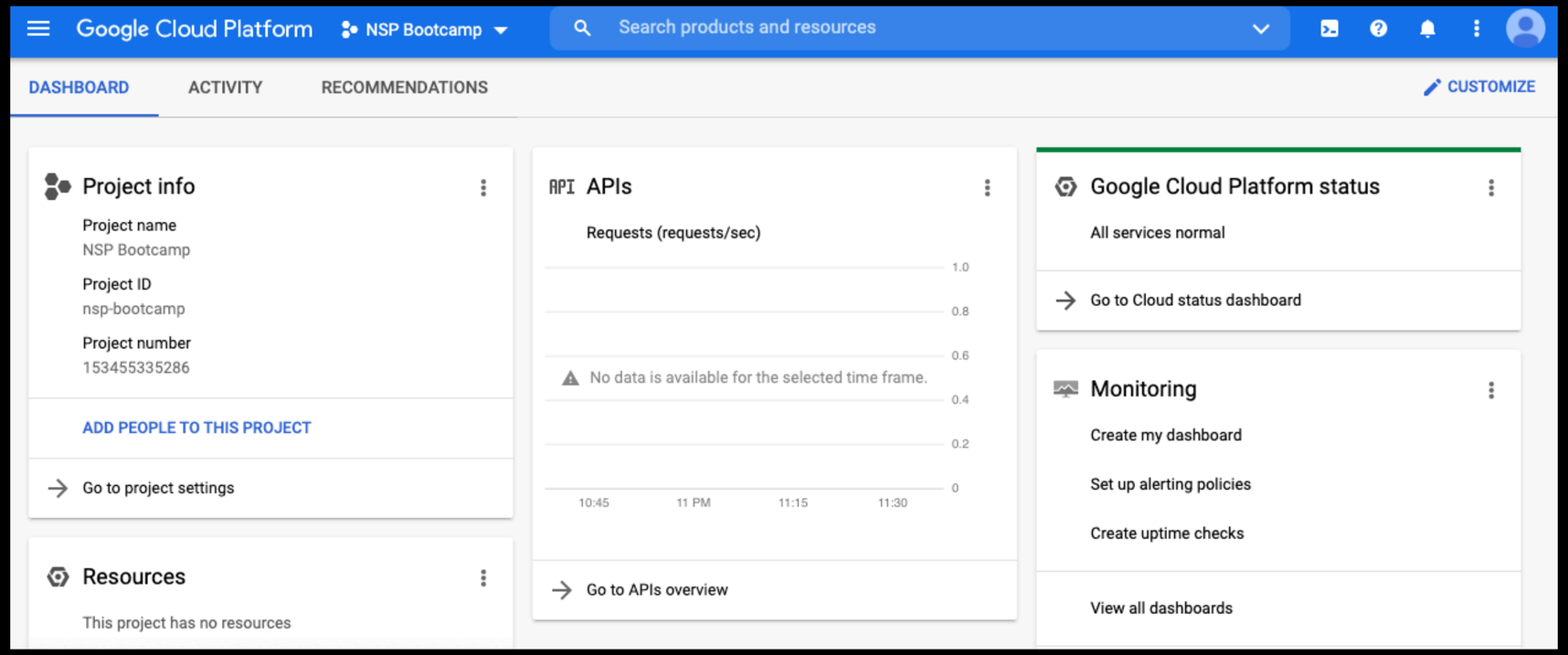
- Google Cloud Platform
- Access to Google's compute resources



Collaboration

GCP

- Google Cloud Platform
- Access to Google's compute resources



Collaboration

Other platforms

Google Cloud Platform	Amazon Web Services[12]	Microsoft Azure[13]	Oracle Cloud[14]
Google Compute Engine	Amazon EC2	Azure Virtual Machines	Oracle Cloud Infra OCI
Google App Engine	AWS Elastic Beanstalk	Azure App Services	Oracle Application Container
Google Kubernetes Engine	Amazon Elastic Kubernetes Service	Azure Kubernetes Service	Oracle Kubernetes Service
Google Cloud Bigtable	Amazon DynamoDB	Azure Cosmos DB	Oracle NoSQL Database
Google BigQuery	Amazon Redshift	Azure Synapse Analytics	Oracle Autonomous Data Warehouse
Google Cloud Functions	AWS Lambda	Azure Functions	Oracle Cloud Fn
Google Cloud Datastore	Amazon DynamoDB	Azure Cosmos DB	Oracle NoSQL Database
Google Cloud Storage	Amazon S3	Azure Blob Storage	Oracle Cloud Storage OCI

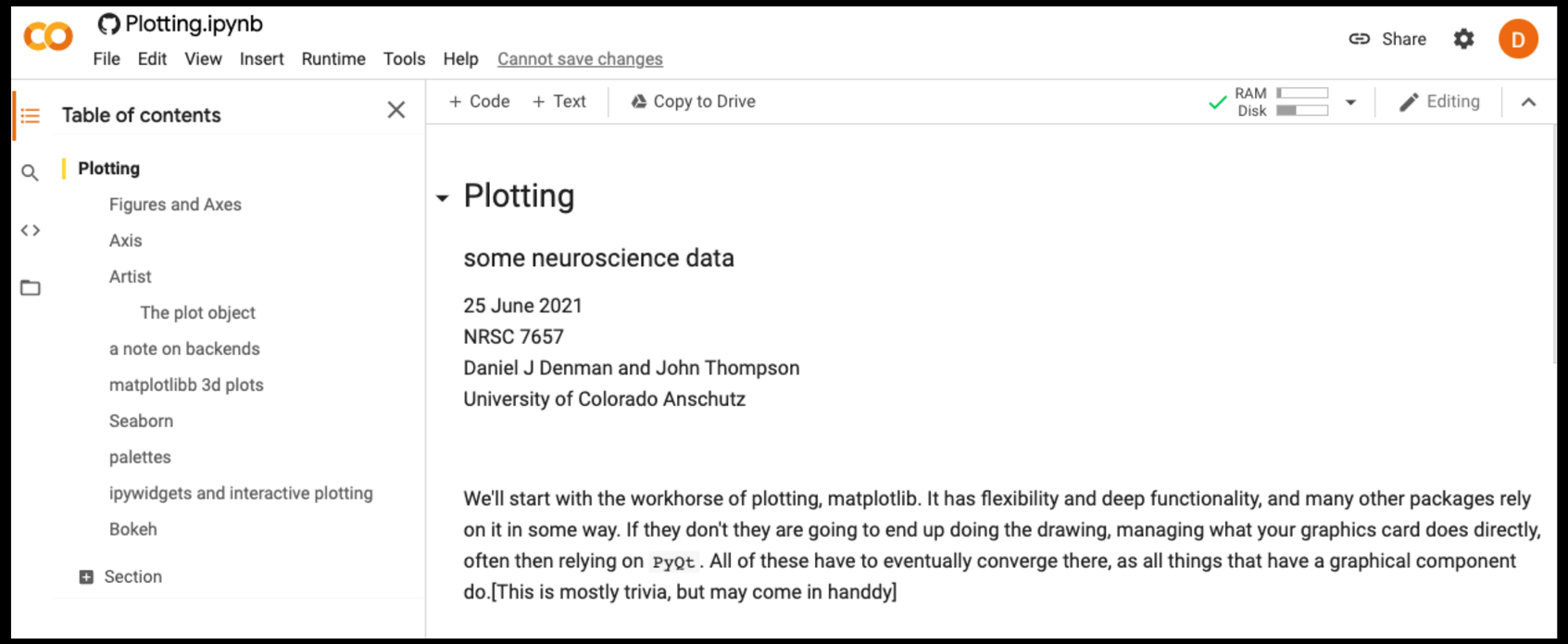
Collaboration

Other platforms

Google Cloud Platform	Amazon Web Services[12]	Microsoft Azure[13]	Oracle Cloud[14]
Google Compute Engine	Amazon EC2	Azure Virtual Machines	Oracle Cloud Infra OCI
Google App Engine	AWS Elastic Beanstalk	Azure App Services	Oracle Application Container
Google Kubernetes Engine	Amazon Elastic Kubernetes Service	Azure Kubernetes Service	Oracle Kubernetes Service
Google Cloud Bigtable	Amazon DynamoDB	Azure Cosmos DB	Oracle NoSQL Database
Google BigQuery	Amazon Redshift	Azure Synapse Analytics	Oracle Autonomous Data Warehouse
Google Cloud Functions	AWS Lambda	Azure Functions	Oracle Cloud Fn
Google Cloud Datastore	Amazon DynamoDB	Azure Cosmos DB	Oracle NoSQL Database
Google Cloud Storage	Amazon S3	Azure Blob Storage	Oracle Cloud Storage OCI

Collaboration

Google colab



Collaboration

Google colab

- there are time limits (at)
- Resource (GPU, memory, disk space on the VM)
- Time - 12 - 24 hours and then it will kick you off. Not for hardcore number crunching.

Collaboration

Deepnote

- Similar idea to Colab, but with project organization
- Most importantly: like a google doc, everyone viewing a Deepnote notebook sees the code and the results of execution live in their own browser.
- Not sure what the hardware is! (GCP, AWS, Azure??)

https://deepnote.com/project/NRSC7610-Python-Activities-JayFWIp7SGmIW-JFkMKIbg/%2FNRSC7610_ReceptiveField.ipynb

Collaboration

Why all of this cloud stuff?

- Your computer works…and many have a computer or 17 in the lab…

Collaboration

Why all of this cloud stuff?

1. Working with other people, especially not locally (colab, deepnote)

Collaboration

Why all of this cloud stuff?

1. Working with other people, especially not locally (colab, deepnote)
2. Publishing code along with manuscripts. Reproducability!

Collaboration

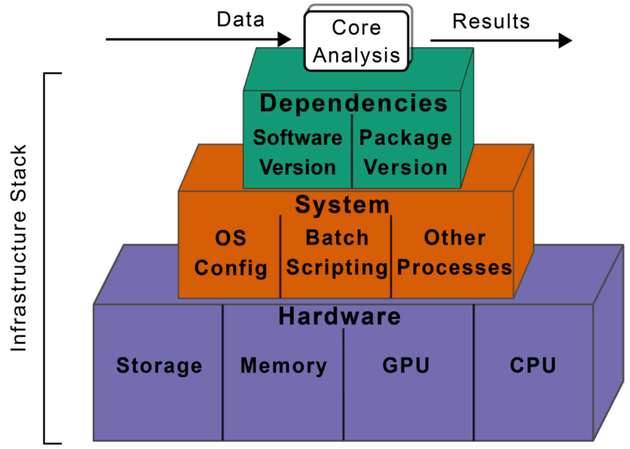
Why all of this cloud stuff?

1. Working with other people, especially not locally (colab, deepnote)
2. Publishing code along with manuscripts. Reproducability!
3. SaaS - a future direction for some analysis, which runs on the cloud?

Collaboration

Software as a Service - SaaS

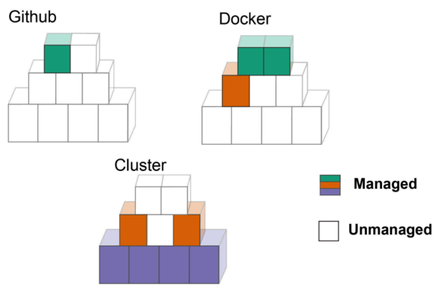
- http://neurocaas.org/

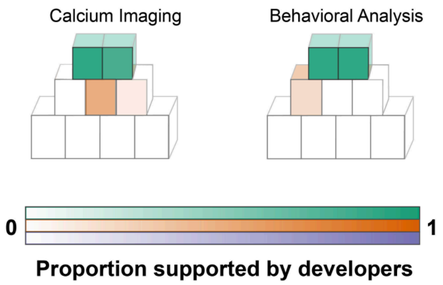


User-Side	Developer-Side
• Pipeline restructuring around software updates • Time and knowledge cost of installation • Manual scaling to multiple machines • Interruptions from system updates • Data loss due to human error • Compatibility issues with local hardware • Insufficient resources for dataset volume • Delays due to resource sharing with other users	• Volatile dependencies • Coordinated development across multiple operating systems • Stability/efficiency at scale • Difficult to test in relevant environment • Accomodation to hardware limitations of end user

C. Generic management tools

D. Field management standards



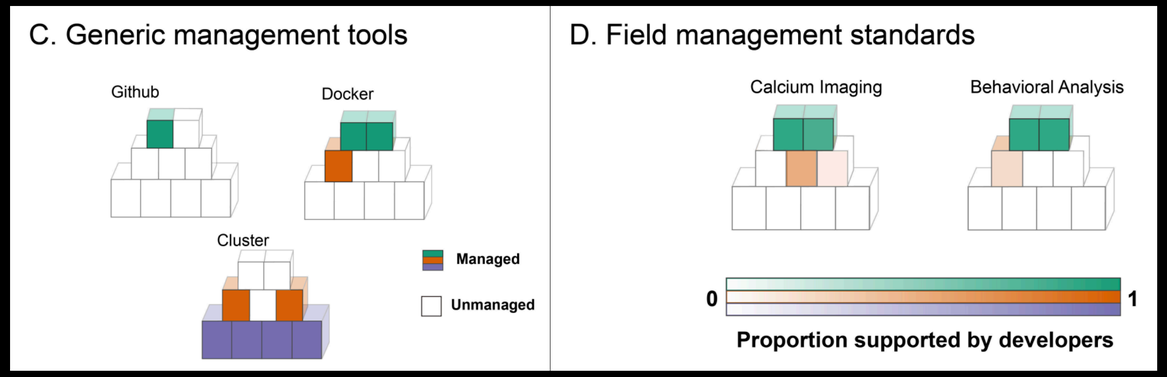


Collaboration

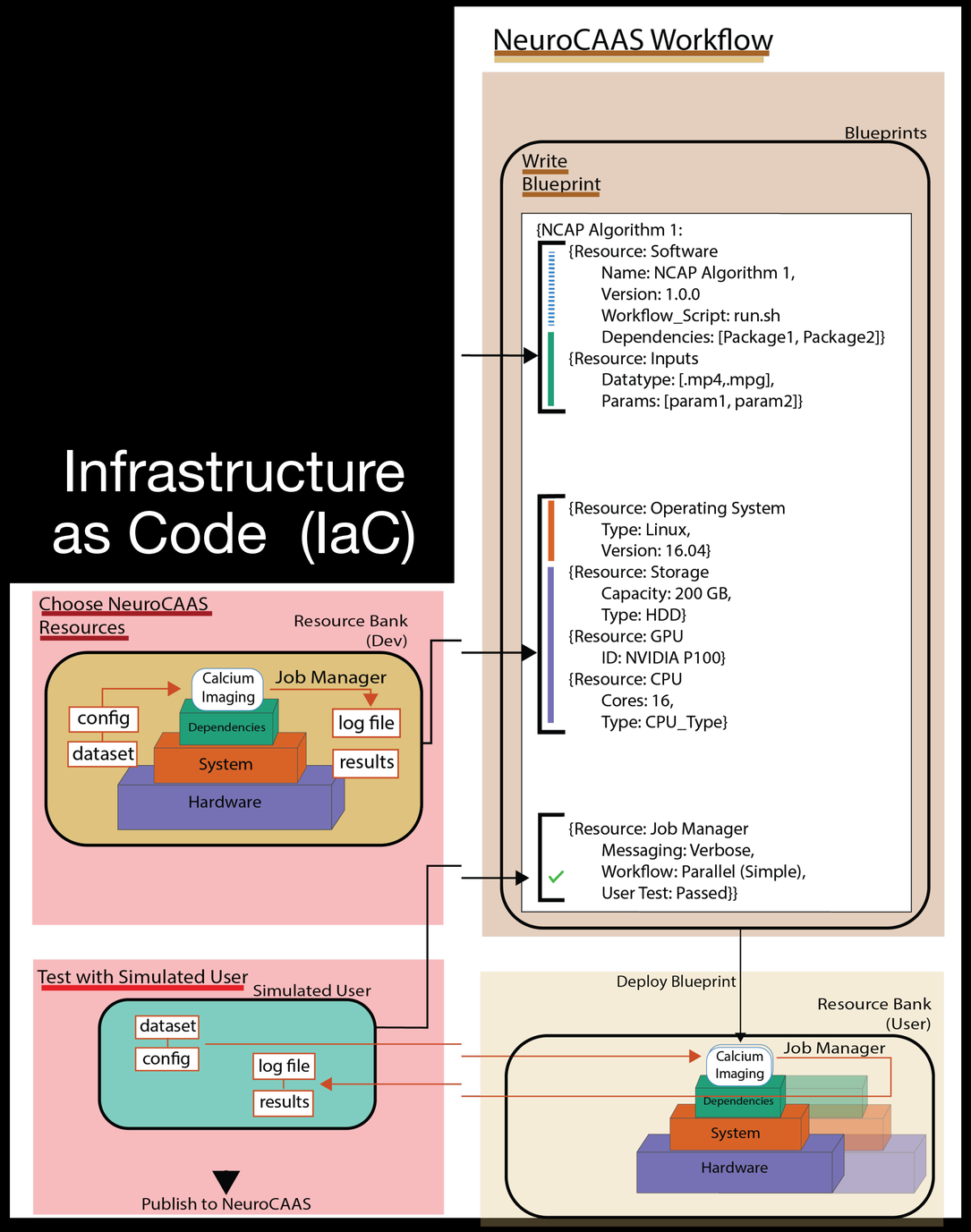
Software as a Service - SaaS

- http://neurocaas.org/

Infrastructure as Graduate Student (IaGS)



Infrastructure as Code (IaC)



Collaboration

http://neurocaas.org/

- Make it easier to use these cloud stacks for neurscientists
  - (So you don’t have to do as much of, say, what we’ll do next week)
- (1) data + (2) configuration file.

https://github.com/cunningham-lab/neurocaas/blob/master/experiments/DLC/batch_10.json

Collaboration

http://neurocaas.org/

- https://www.biorxiv.org/content/10.1101/2020.06.11.146746v2

Cold Spring Harbor Laboratory

bioRxiv

THE PREPRINT SERVER FOR BIOLOGY

bioRxiv posts many COVID19-related papers. A reminder: they have not been formally peer-reviewed and should not guide health-related behavior or be reported in the press as conclusive.

New Results

Neuroscience Cloud Analysis As a Service

Taiga Abe, Ian Kinsella, Shreya Saxena, E. Kelly Buchanan, Joao Couto, John Briggs, Sian Lee Kitt, Ryan Glassman, John Zhou, Liam Paninski, John P. Cunningham

doi: https://doi.org/10.1101/2020.06.11.146746