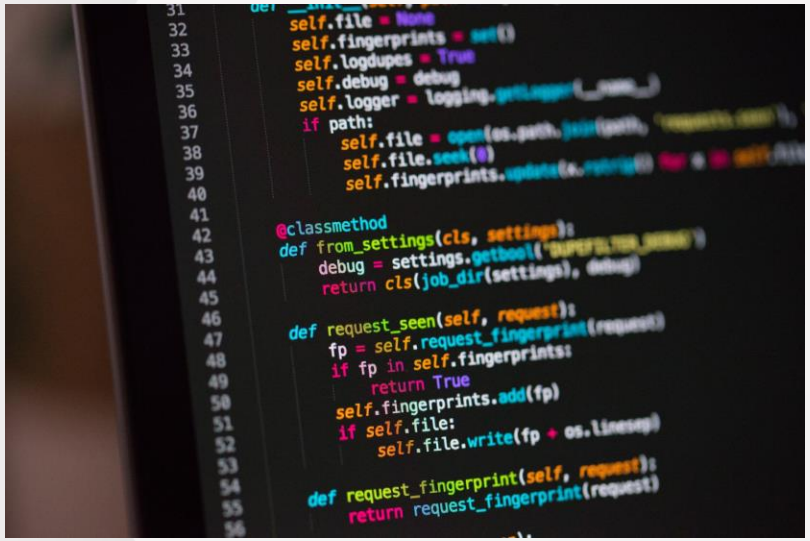
Level 5 Data Engineer
Module 3 Topic 8

Practical programming and module consolidation



L5 Data Engineer Higher Apprenticeship
Module 3 / 12 (**“Programming and Scripting Essentials”**)
Topic 9 / 9

Ice breaker: Discussion

A bit of fun to start…

Which of the following fictional characters best represents your work style?

A. Hermione Granger (meticulous and organised)

B. Tony Stark (innovative and tech-savvy)

C. Michael Scott (enthusiastic but chaotic)

D. Gandalf (wise and strategic)

Submit your responses to the chat or turn on your microphone

Session aim and objectives

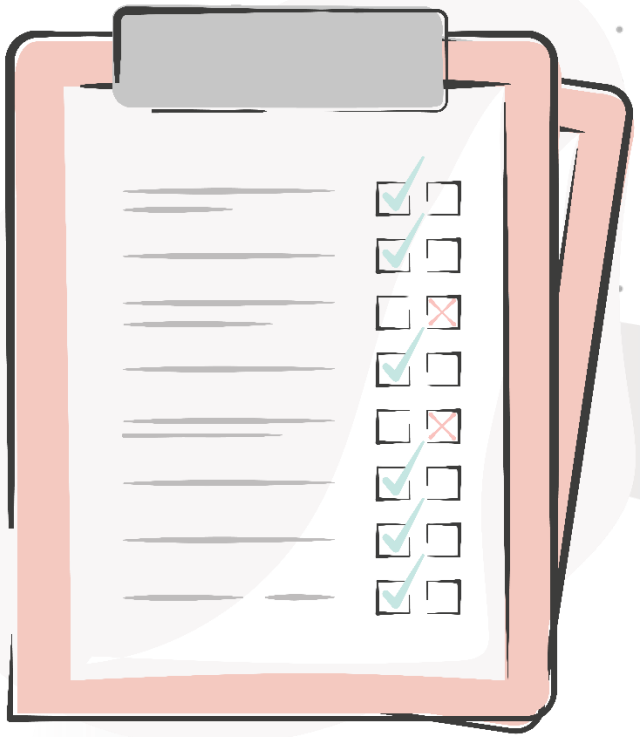
Completion of this topic supports the following outcomes:

- **Employ** software development tools and techniques for designing, deploying and maintaining secure data products and pipelines
- **Construct** algorithms that correctly and efficiently handle data at scale whilst mitigating risks
- **Demonstrate** knowledge of the steps needed to prepare the code for production

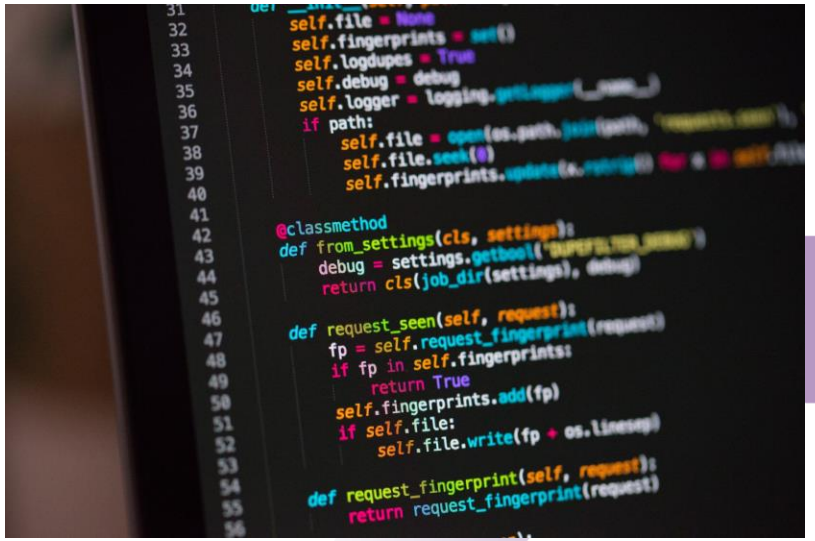
Webinar Agenda

This webinar will include the following:

- DevOps pipelines
- Continuous integration
- Continuous delivery
- Continuous deployment
- Containerisation
- Docker
- Consolidation of the knowledge from this module – a little friendly competition!



DevOps Pipelines and Continuous Integration



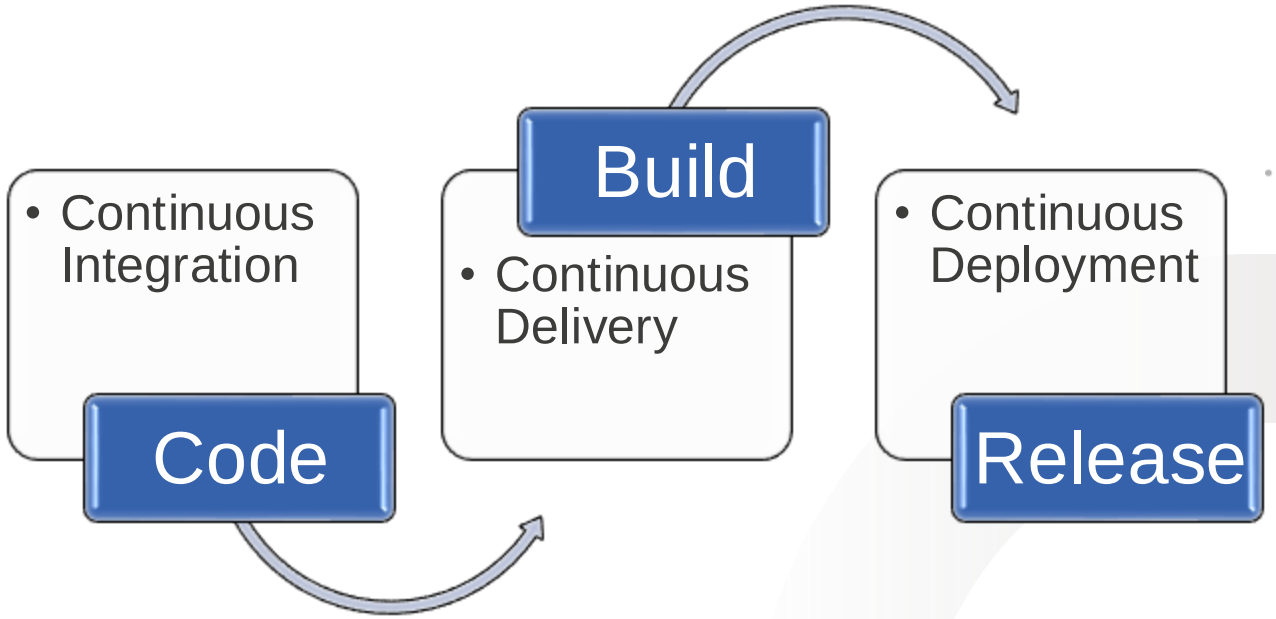
BPP

Introduction

What do DevOps pipelines comprise of and what are the benefits?

The benefits:

- Reduced risk
- Shorter review time
- Better code quality
- Faster bug fixes
- Measurable progress
- Faster feedback loops
- Increased collaboration



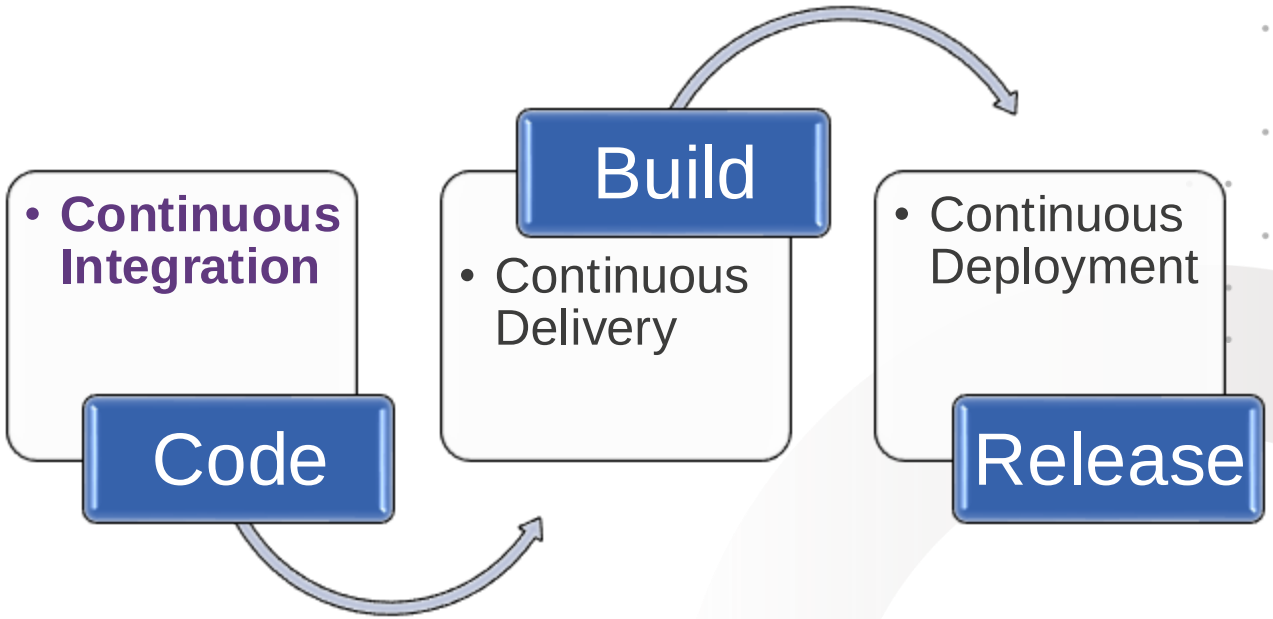
The key components of a DevOps pipeline

Continuous Integration

What does this mean?

The benefits:

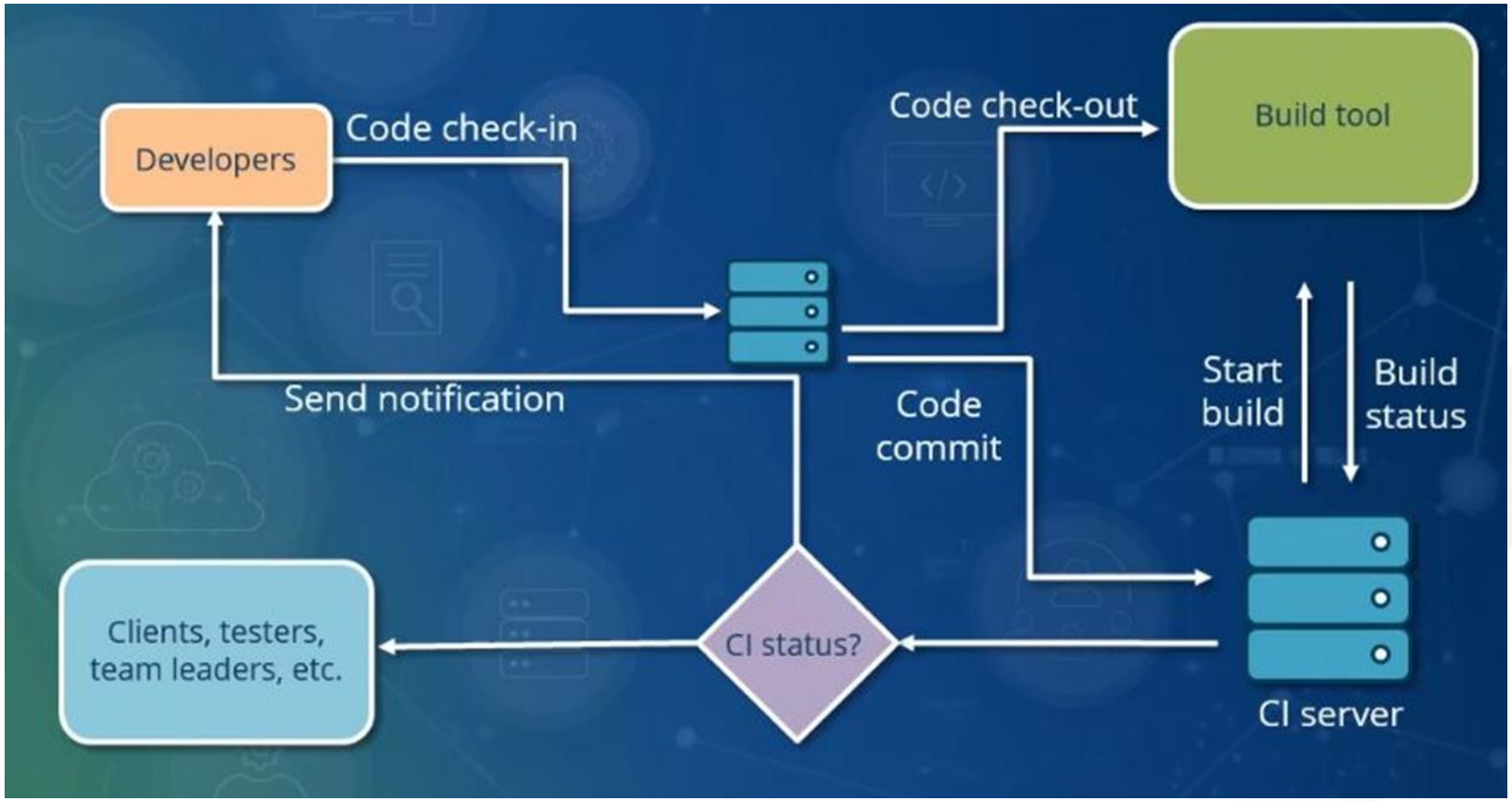
- Merging all programmers' code frequently
- Automated process
  - Checks all code compiles
  - Runs tests and quality checks
  - Reports on failures



The key components of a DevOps pipeline

Continuous Integration

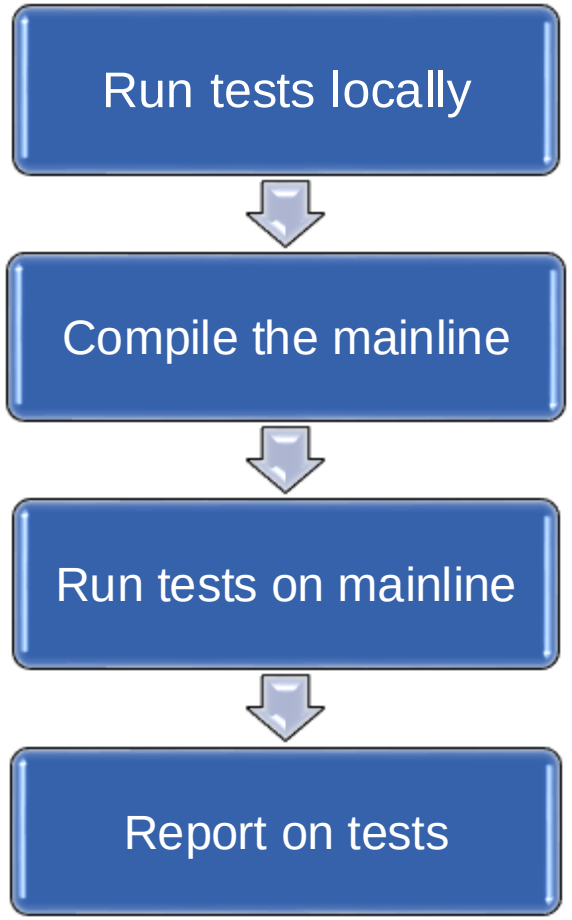
A spotlight on the process…



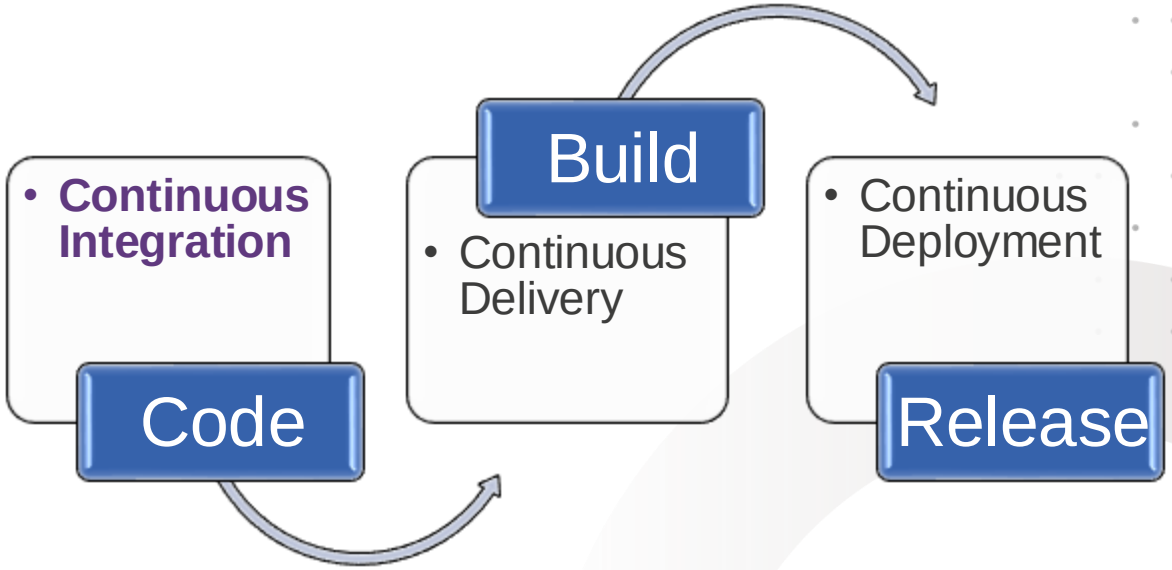
The key touchpoints of the Continuous Integration (CI) process

Continuous Integration

Workflow



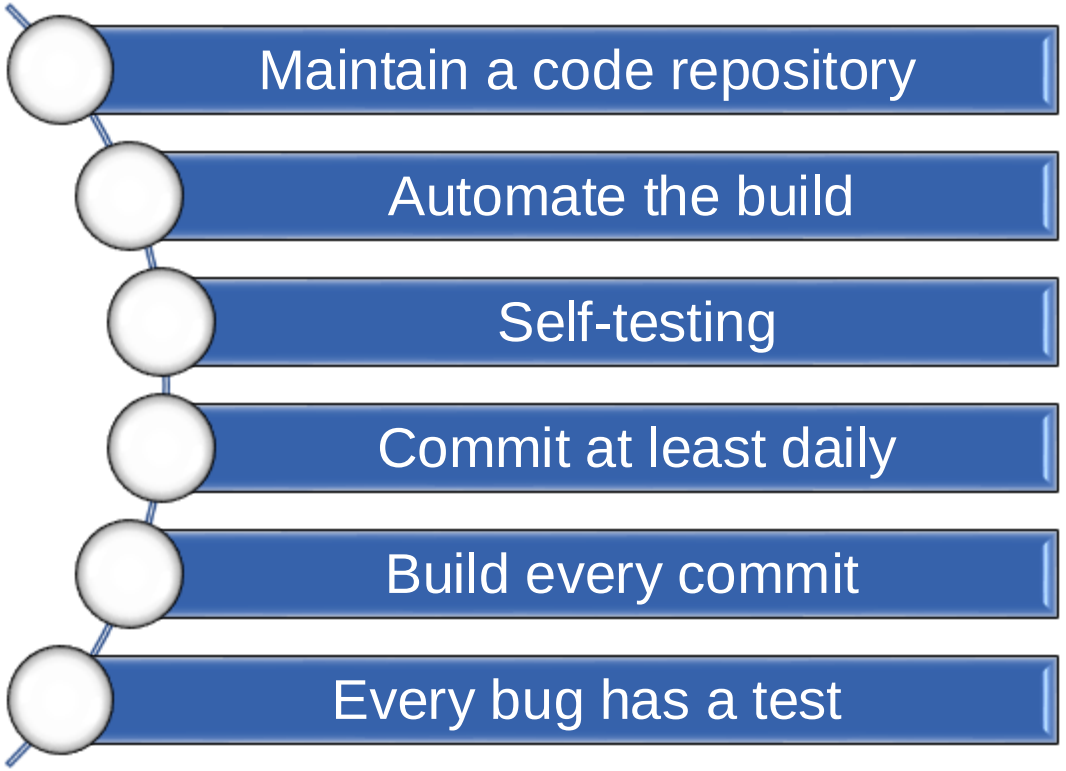
The Continuous Integration (CI) workflow



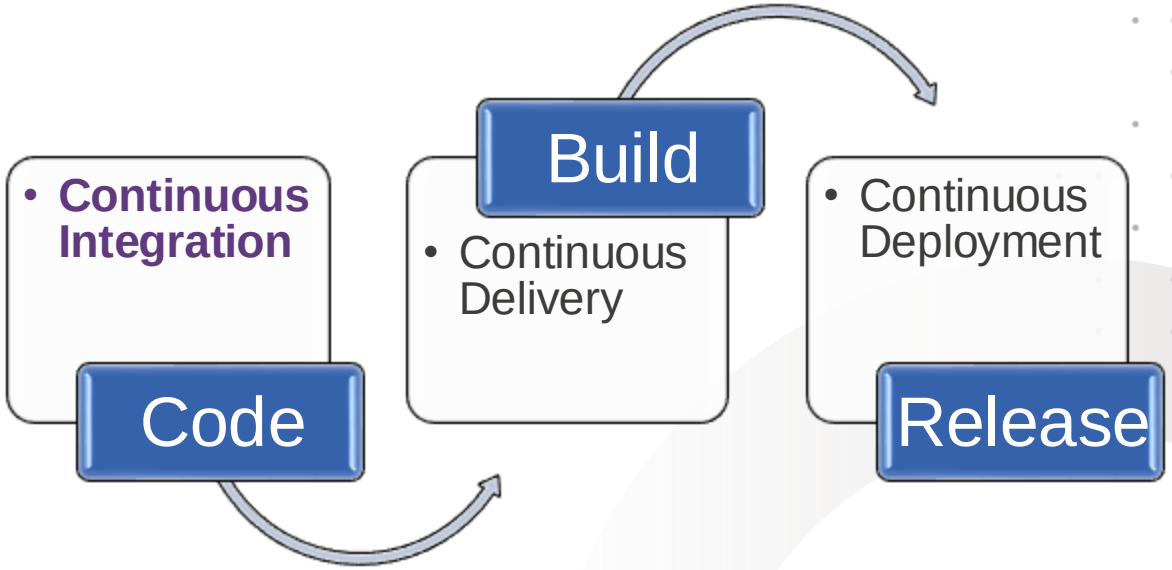
The key components of a DevOps pipeline

Continuous Integration

Best practice



Best practice guidance for CI



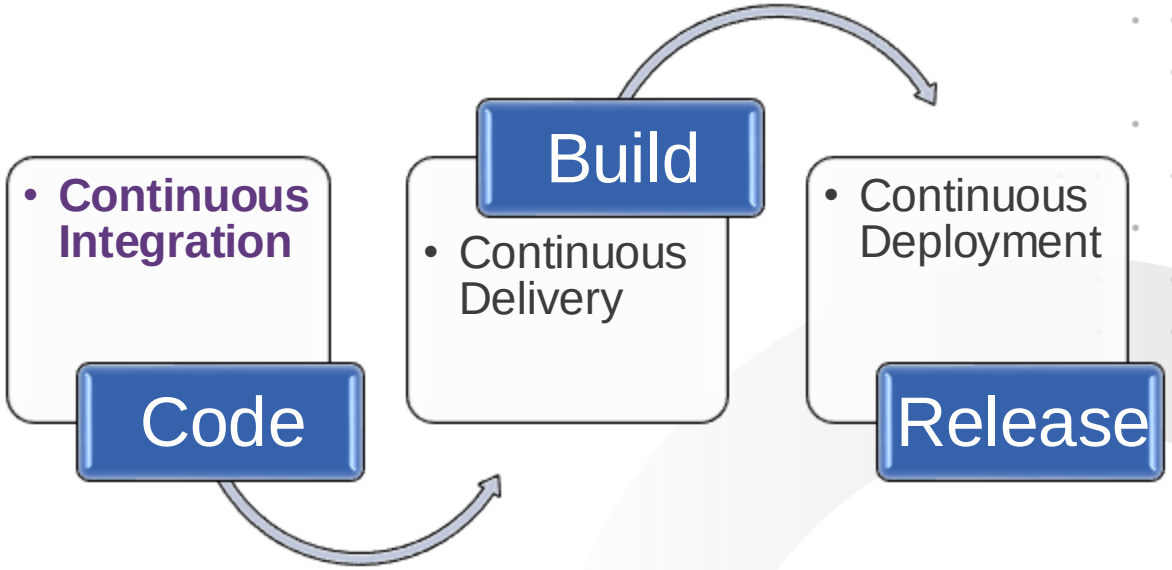
The key components of a DevOps pipeline

Continuous Integration

Best practice (cont)

- Builds should be fast
- Life-like test environments
- Easy to get deliverables
- Transparent build results
- Automate deployment

Best practice guidance for CI



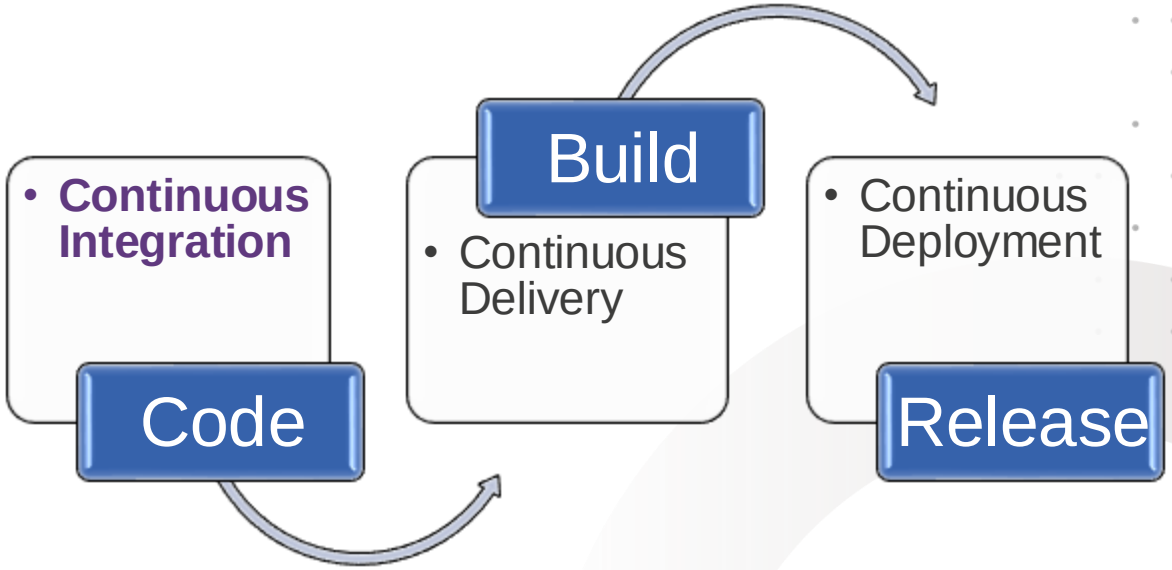
The key components of a DevOps pipeline

Continuous Integration

The benefits

- Integration bugs are found early
- Frequent commits mean easier to locate issues
- ‘Current’ build is always testable
- Rigour around testing
- Early access to metrics eg static analysis
- Early feedback on changes

The benefits of CI



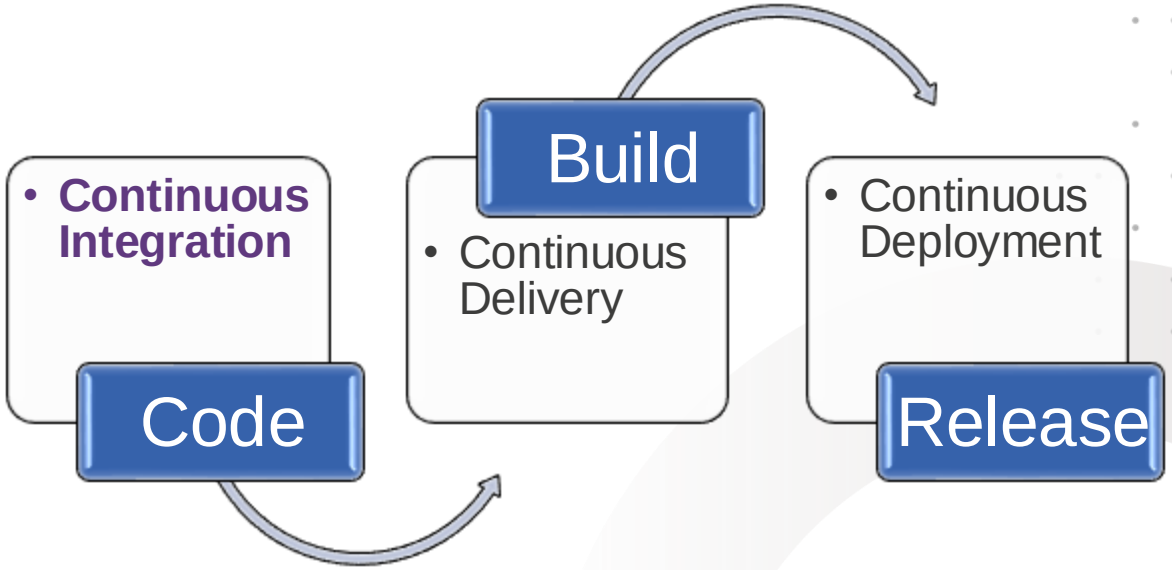
The key components of a DevOps pipeline

Continuous Integration

The disadvantages

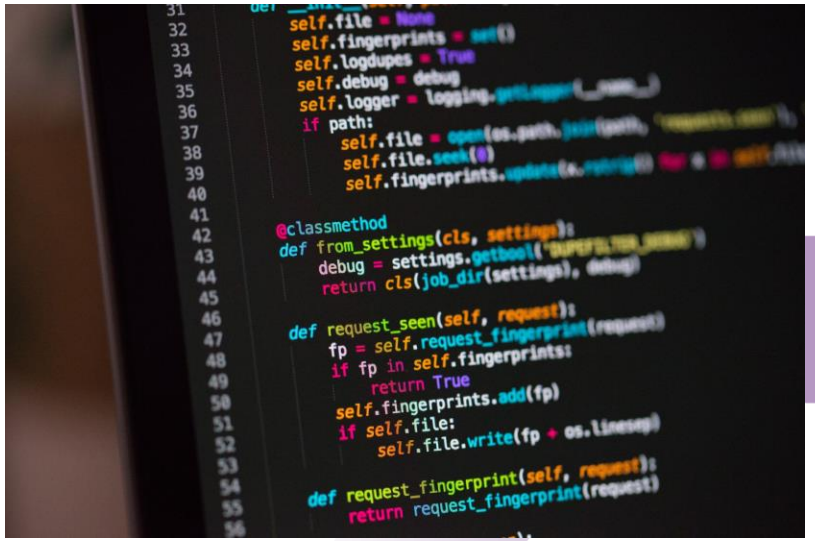
- Creating tests is time consuming
- Build system needs to be maintained
- Value depends on the quality of tests
- Many commits can mean a hold up on build que

The disadvantages of CI



The key components of a DevOps pipeline

Continuous Delivery



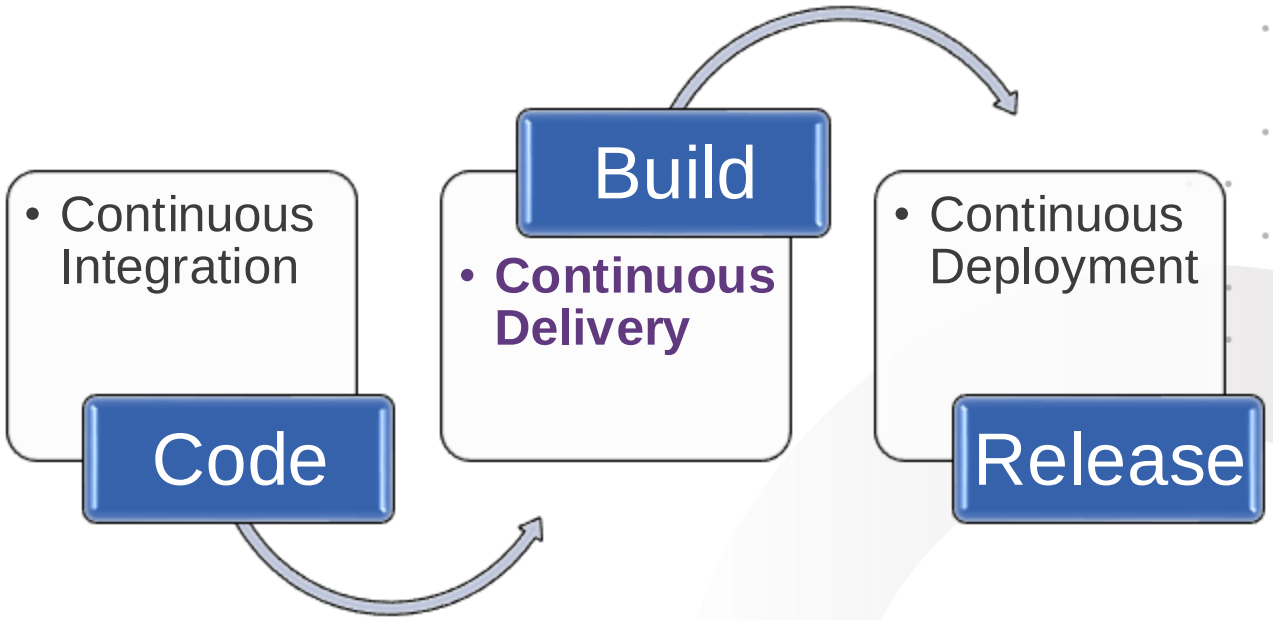
Continuous Delivery

What does this mean?

CD means always having a potentially releasable product.

This means:

- Developing in short cycles
- Ensuring quality of release candidates
- Always having built and tested software

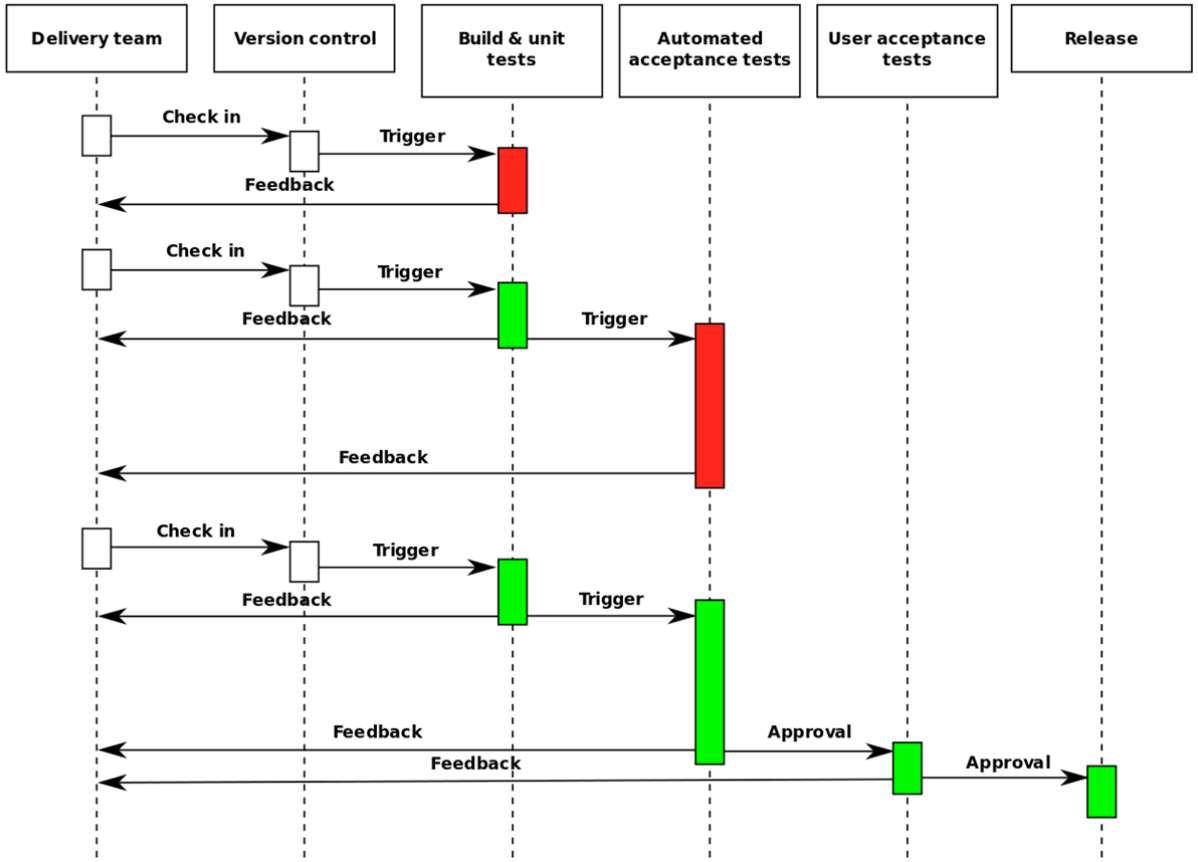


The key components of a DevOps pipeline

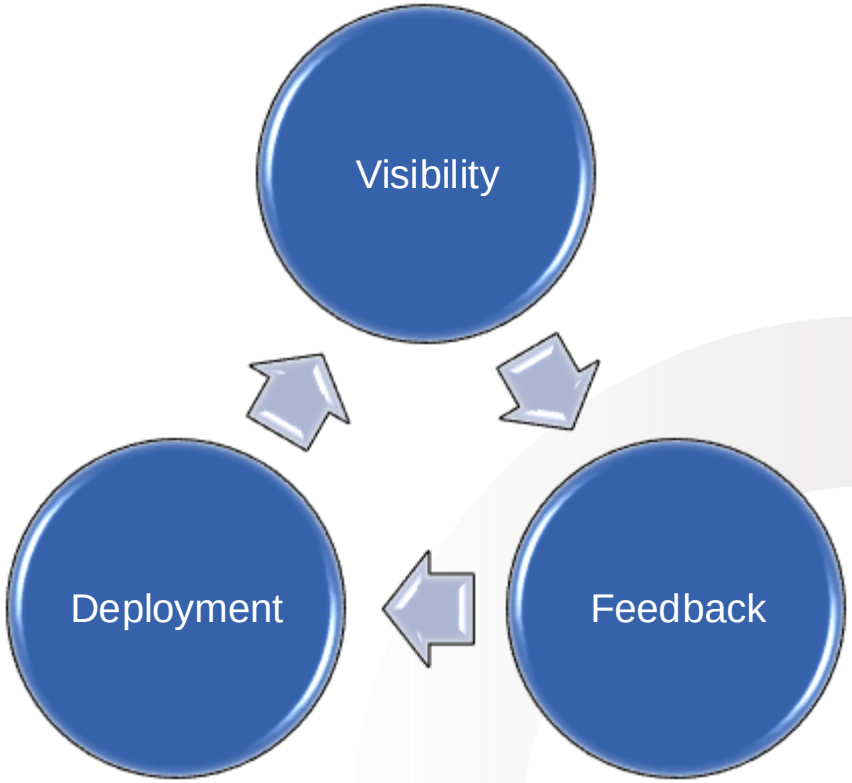
CD relies on CI to ensure this.

Continuous delivery and integration

A spotlight on deployment…



The key touchpoints of the Continuous Integration (CD) process



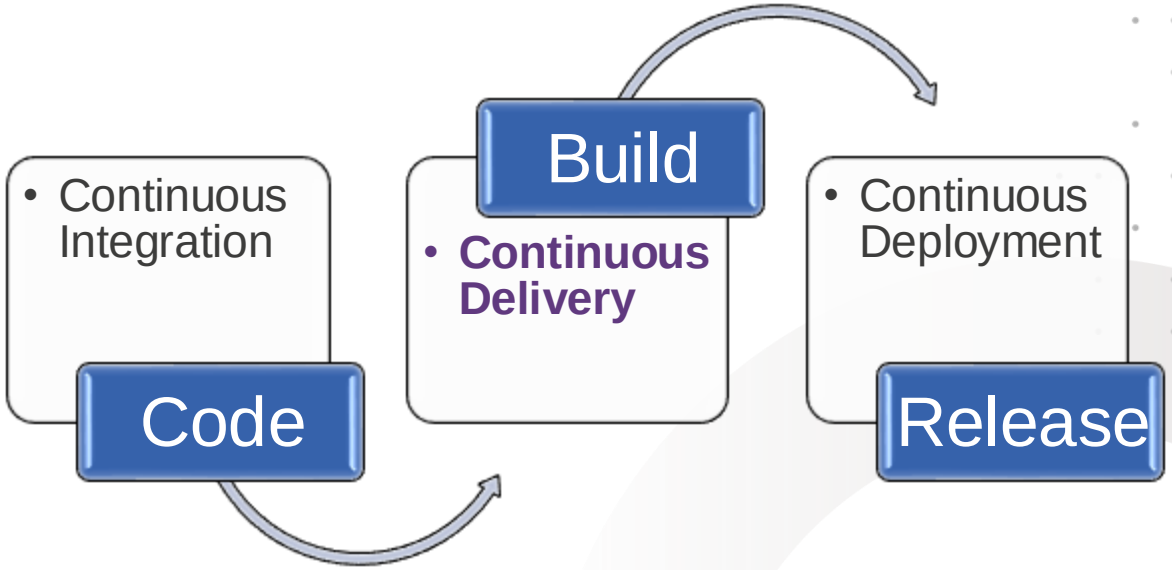
The deployment pipeline

Continuous Delivery

The benefits

- Faster time to market
- Build the right product
- Improved efficiency and productivity
- Reduced release risk
- Improved quality
- Improved customer satisfaction

The benefits of CD



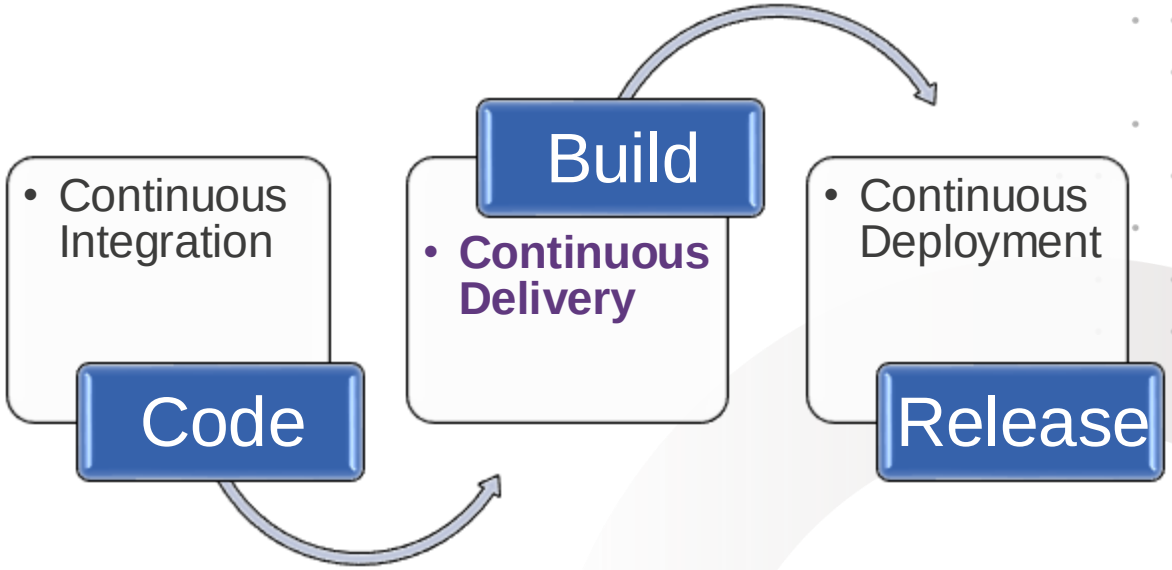
The key components of a DevOps pipeline

Continuous Delivery

The disadvantages

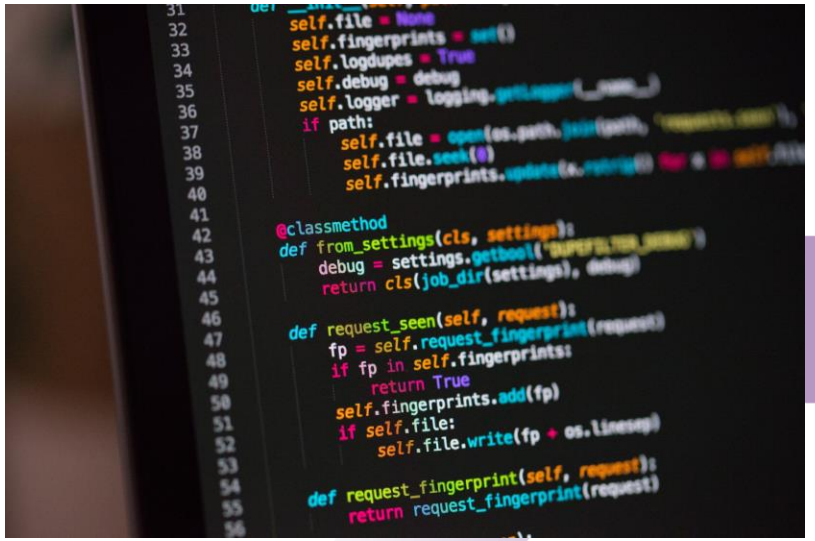
- Effort in maintaining pipeline
- Relies on test quality
- Not everything can be tested automatically
- Some tests can be slow
- Old systems are difficult or impossible to test automatically

The disadvantages of CD



The key components of a DevOps pipeline

Continuous Deployment



BPP

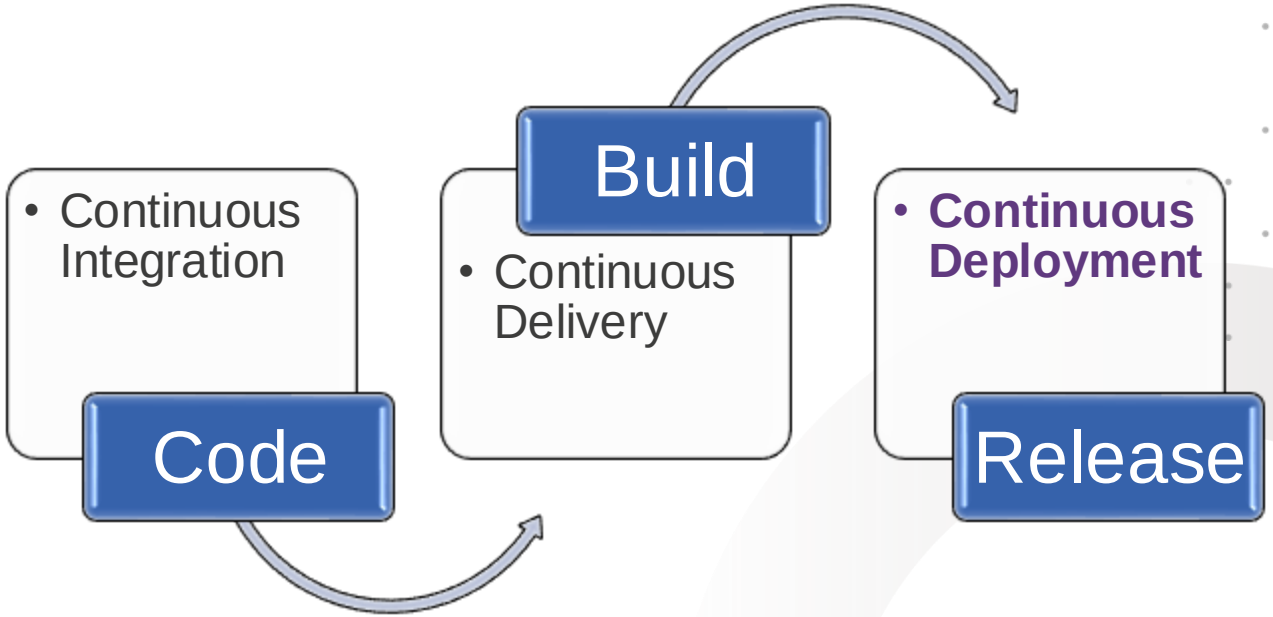
Continuous Deployment

What does this mean?

Also called CD.

- Continuous Deployment is essentially automated continuous delivery
- Continuous delivery relies on a human approval system before a release can take place
- Continuous deployment does not

CD does not rely on a human approval system.



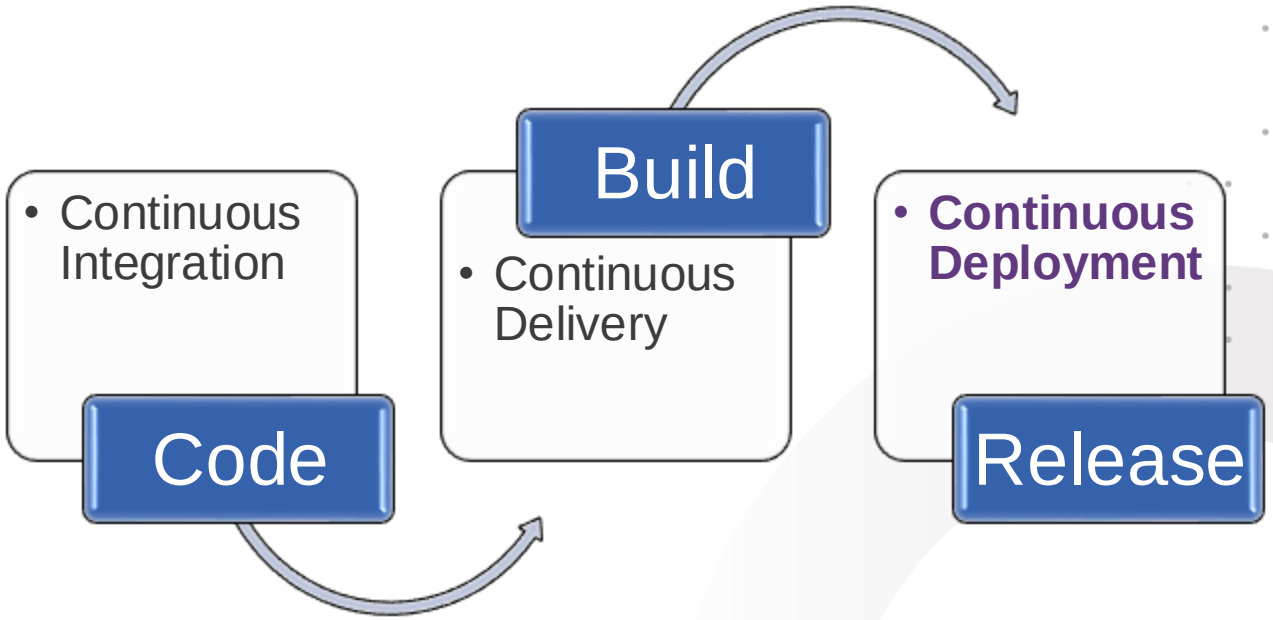
The key components of a DevOps pipeline

Continuous Deployment

The benefits

- **Automated release process**
- **Bugs created recently are easier to fix**
- **Faster time to market**

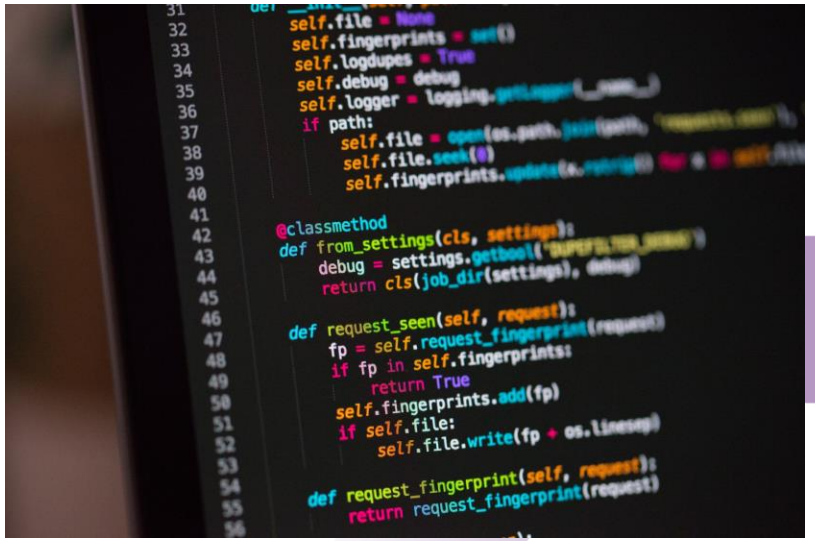
The benefits of CD



The key components of a DevOps pipeline

CD Relies on CI and CD (delivery).

Containerisation



Containerisation

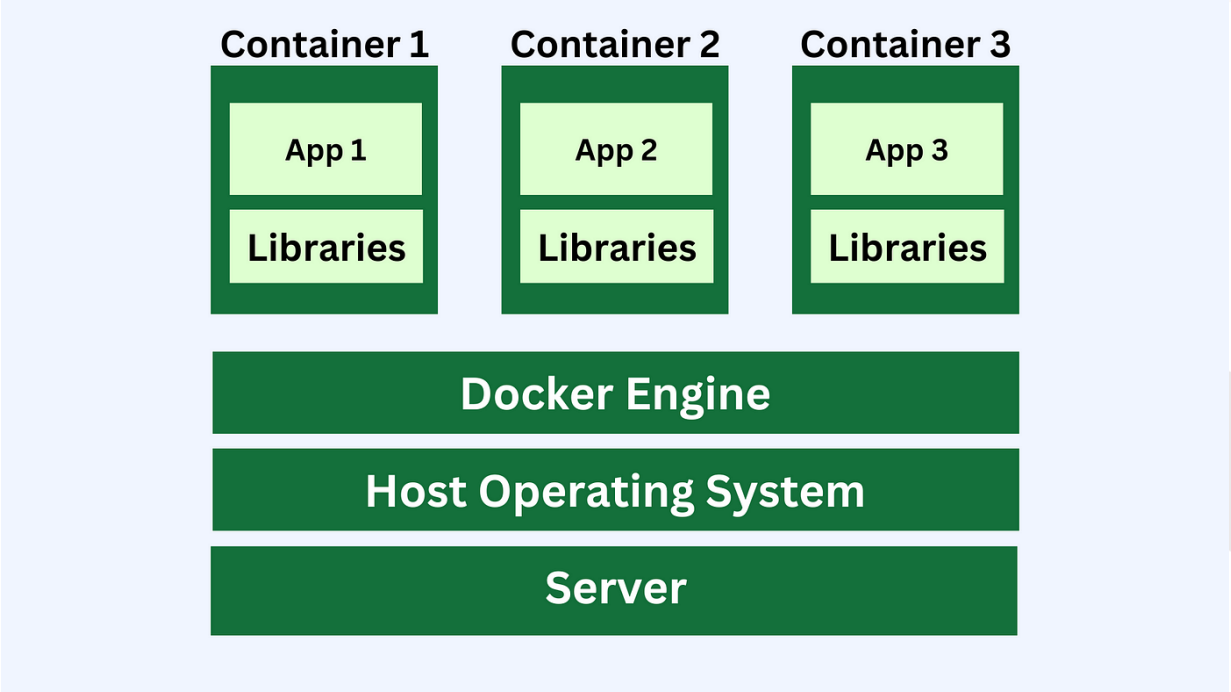
What does this mean?

Packaging software so it is totally self-contained.

This means it must contain:

- The actual code
- Libraries
- Runtime environment
- OS kernel

This is called a container.



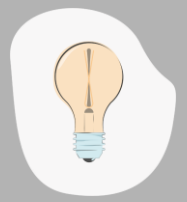
The key components of containerisation

Containerisation

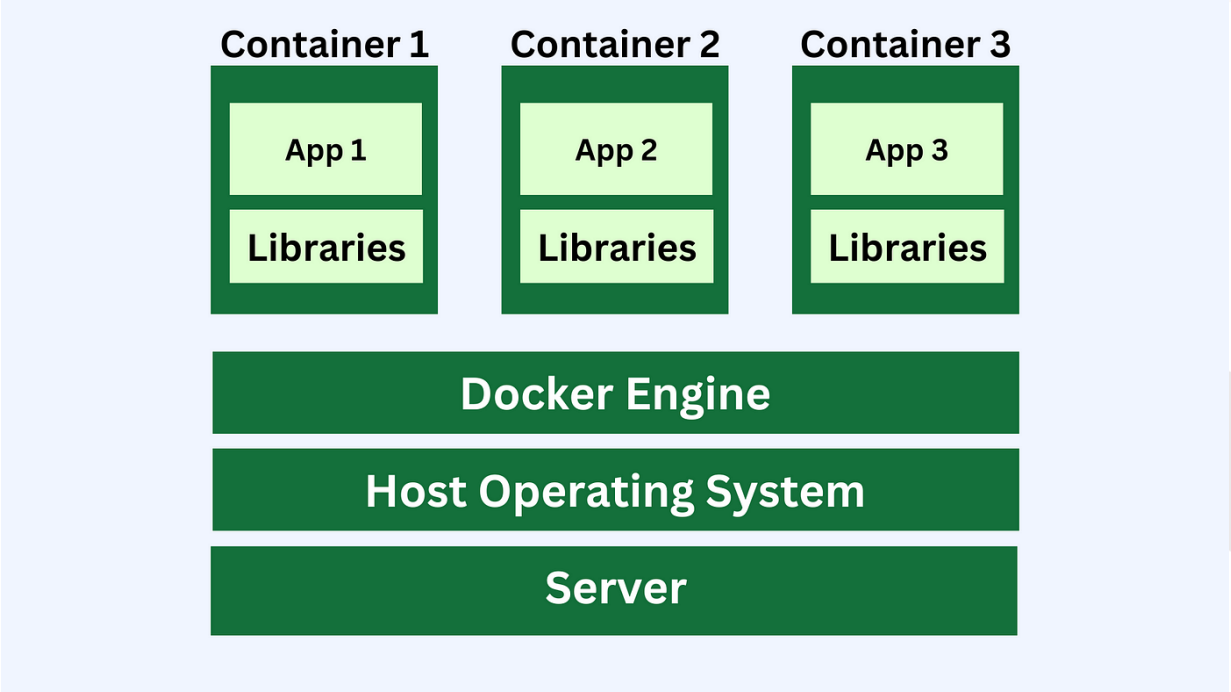
What does this mean?

Containers are designed to be:

- OS agnostic
- Write once, run anywhere (or at least more so)
- Lightweight
- Isolated



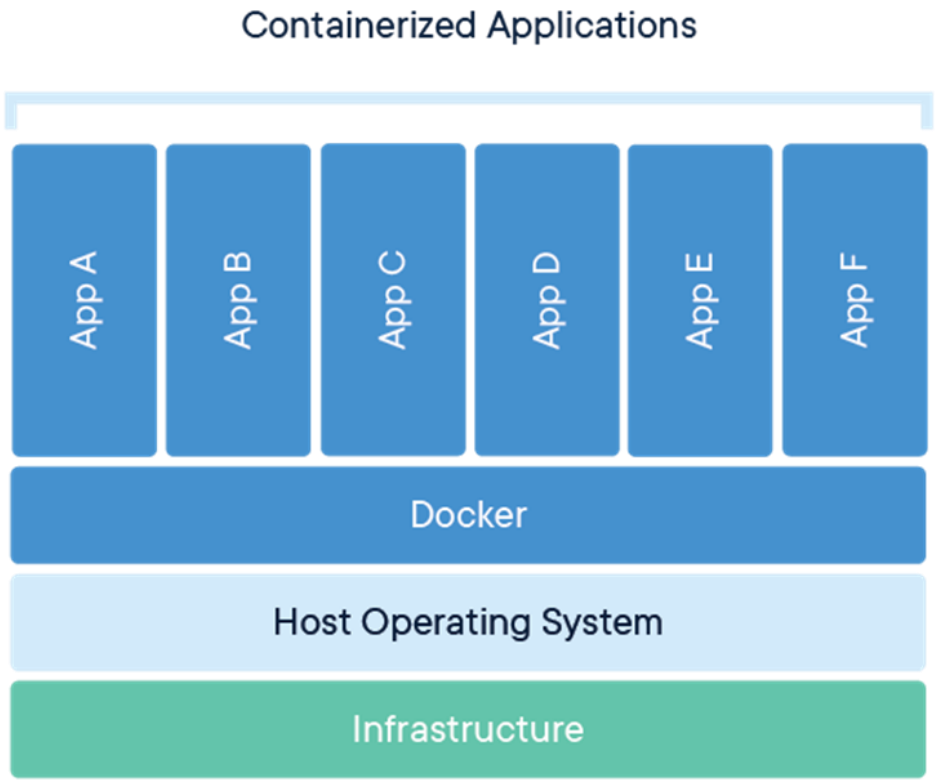
Similar to Virtual Machines (VMs) but better.

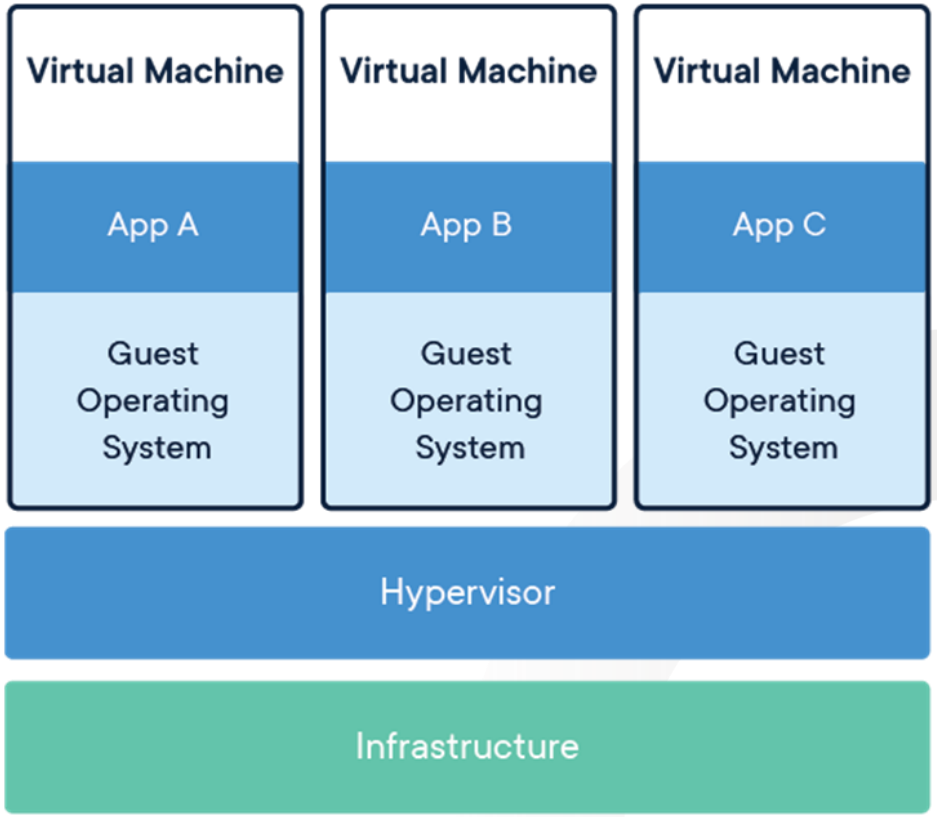


The key components of containerisation

Containerisation

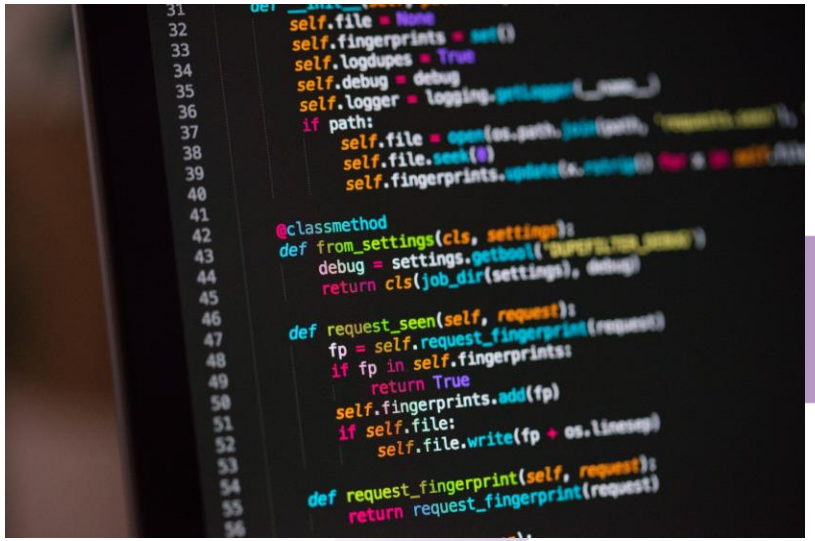
A spotlight on the architecture…





Containerisation architecture

Infrastructure as Code



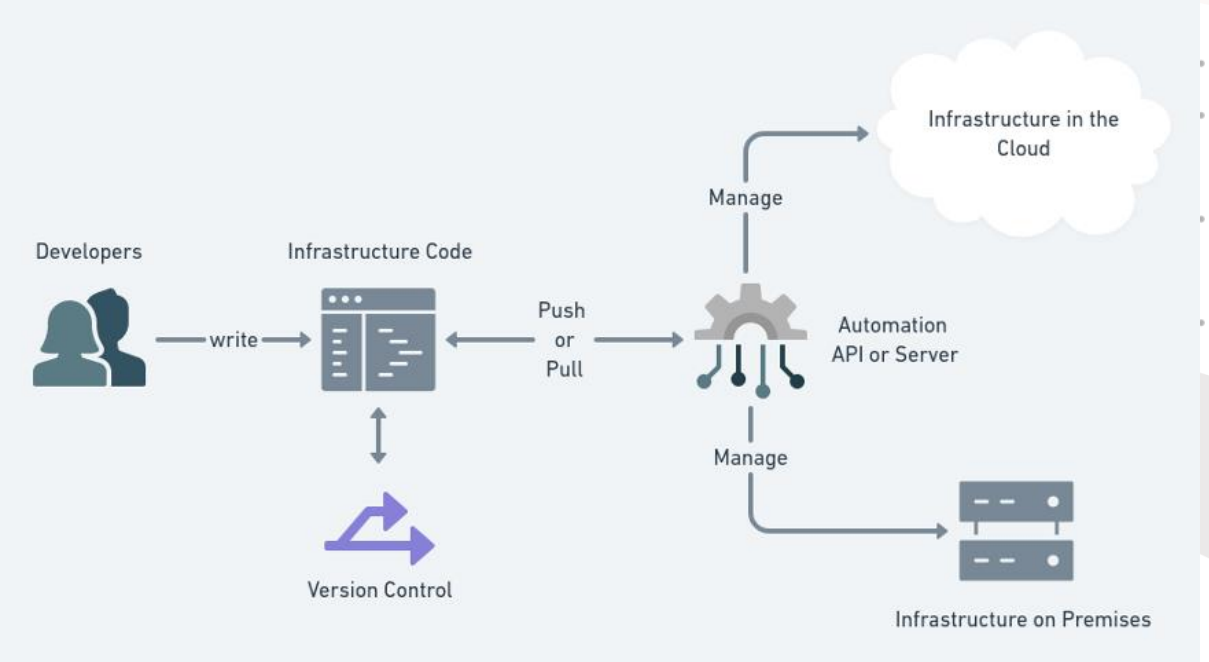
Infrastructure as Code

What does this mean?

Treating infrastructure in the same way we would code.

- Can be source controlled
- Can be configured at will
- Can be automated

Tools exist for managing this.

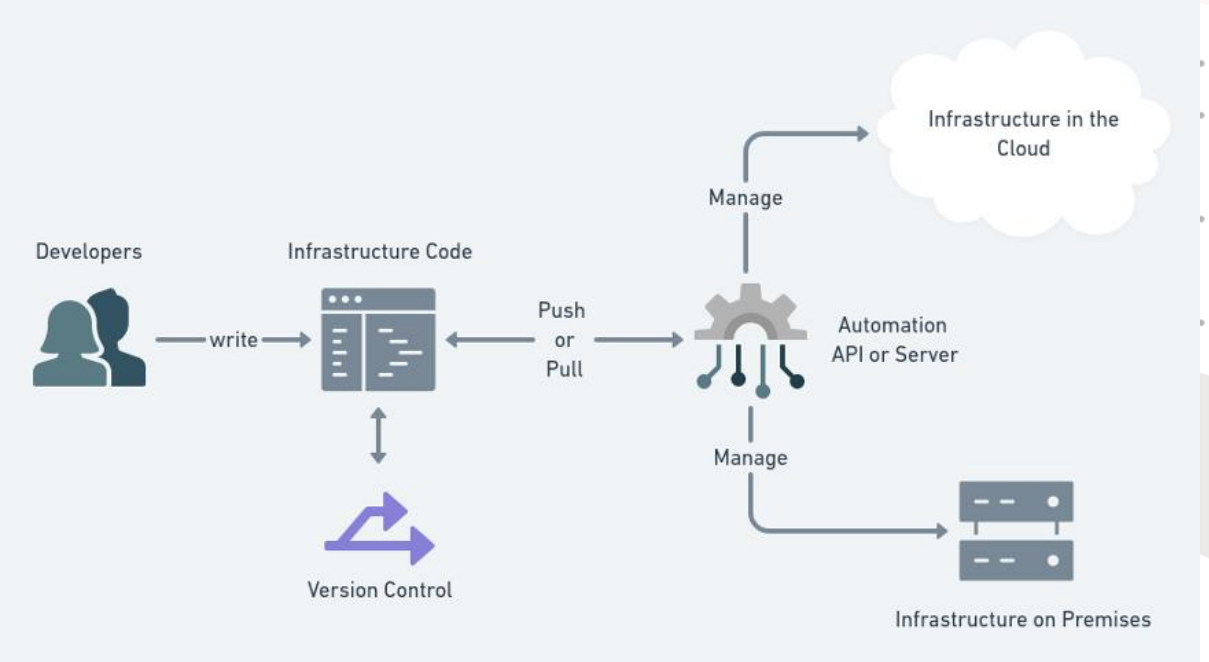


An illustration of Infrastructure as Code (IaC)

Infrastructure as Code

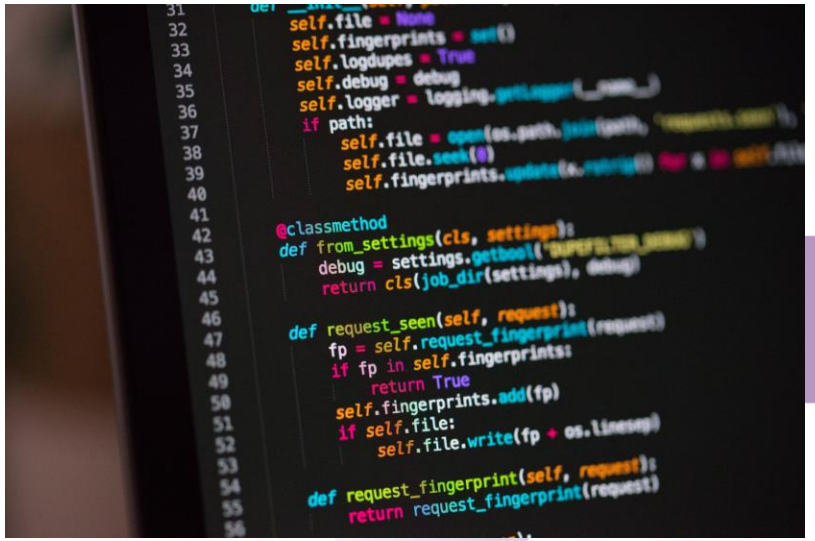
The benefits

The benefits of IaC



An illustration of Infrastructure as Code (IaC)

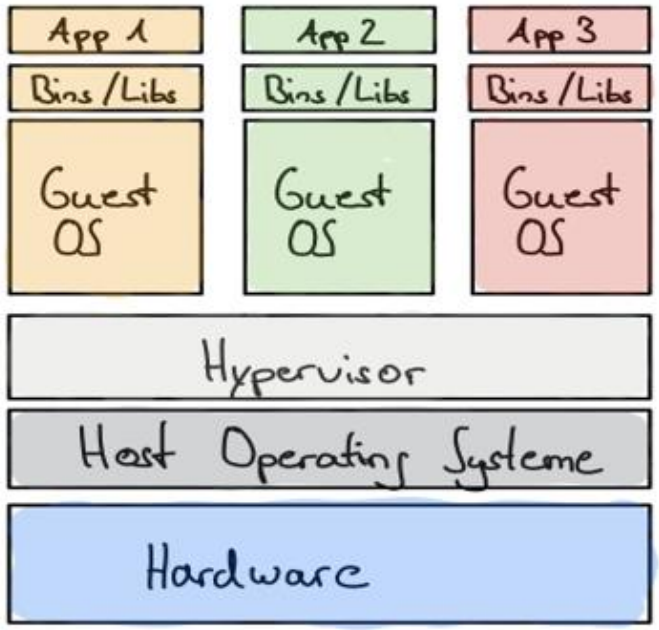
Docker


self.file = None
self.fingerprints = set()
self.logdupes = True
self.debug = debug
self.logger = logging.getLogger(__name__)
if path:
self.file = open(os.path.join(path, 'requests.seen'), 'a+')
self.file.seek(0)
self.fingerprints.update(x.rstrip() for x in self.file)
@classmethod
def from_settings(cls, settings):
debug = settings.getbool('DUPEFILTER_DEBUG')
return cls(job_dir(settings), debug)
def request_seen(self, request):
fp = self.request_fingerprint(request)
if fp in self.fingerprints:
return True
self.fingerprints.add(fp)
if self.file:
self.file.write(fp + os.linesep)
def request_fingerprint(self, request):
return request_fingerprint(request)

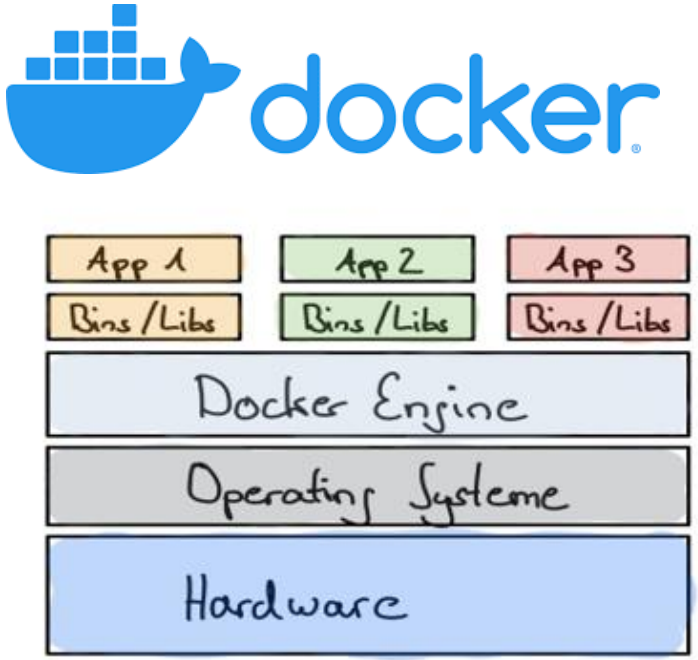
BPP

Docker

What is it and what does it do?



Virtual Machines



Containers

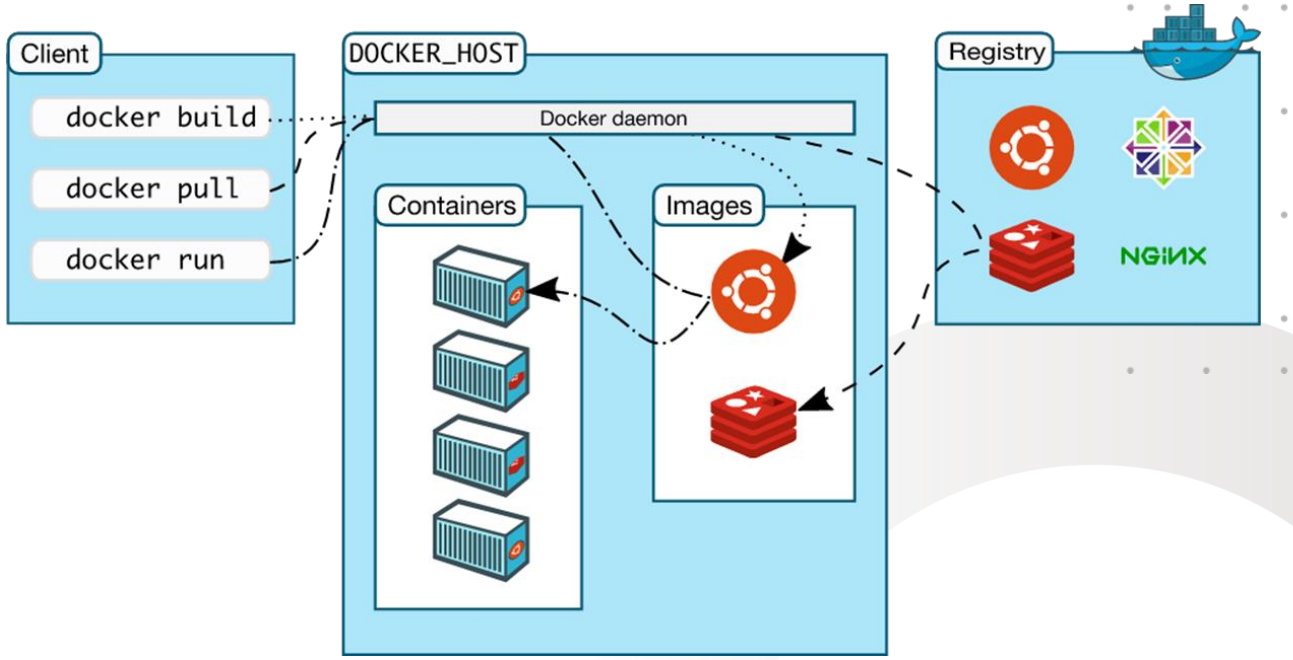
Docker is a Platform as a Service (PaaS)

Docker

Why Docker?

It simplifies…

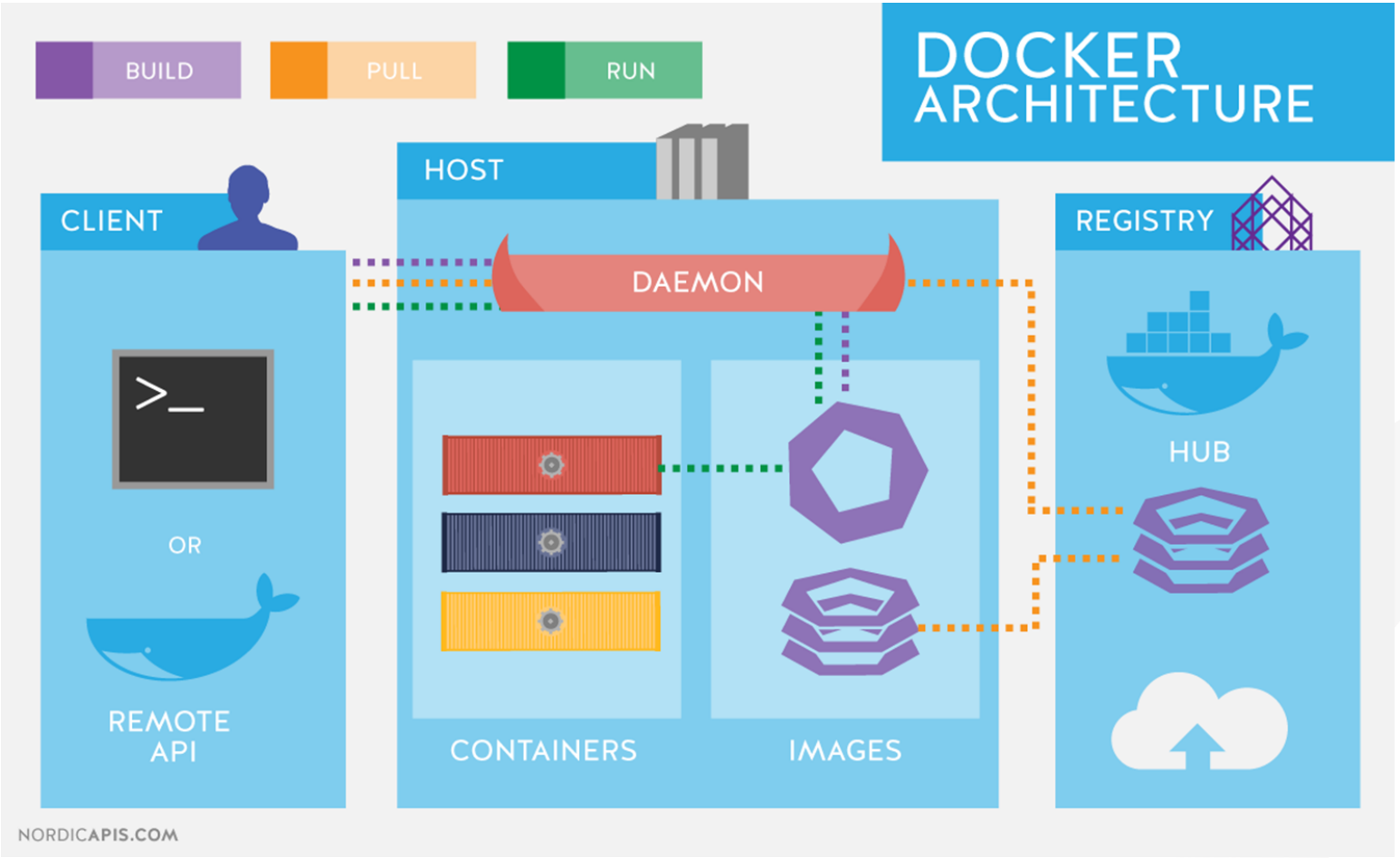
- **Environment Setup:** Quick and consistent
- **Rapid Deployment:** Get up and running with a few keystrokes
- **Code Distribution:** Access code, data, and libraries for analysis
- **Collaboration:** Work together seamlessly
- **Simplicity:** Easier than dealing with virtual machines (VMs)



We can create Docker containers and run them on a Docker 'daemon'

Docker architecture

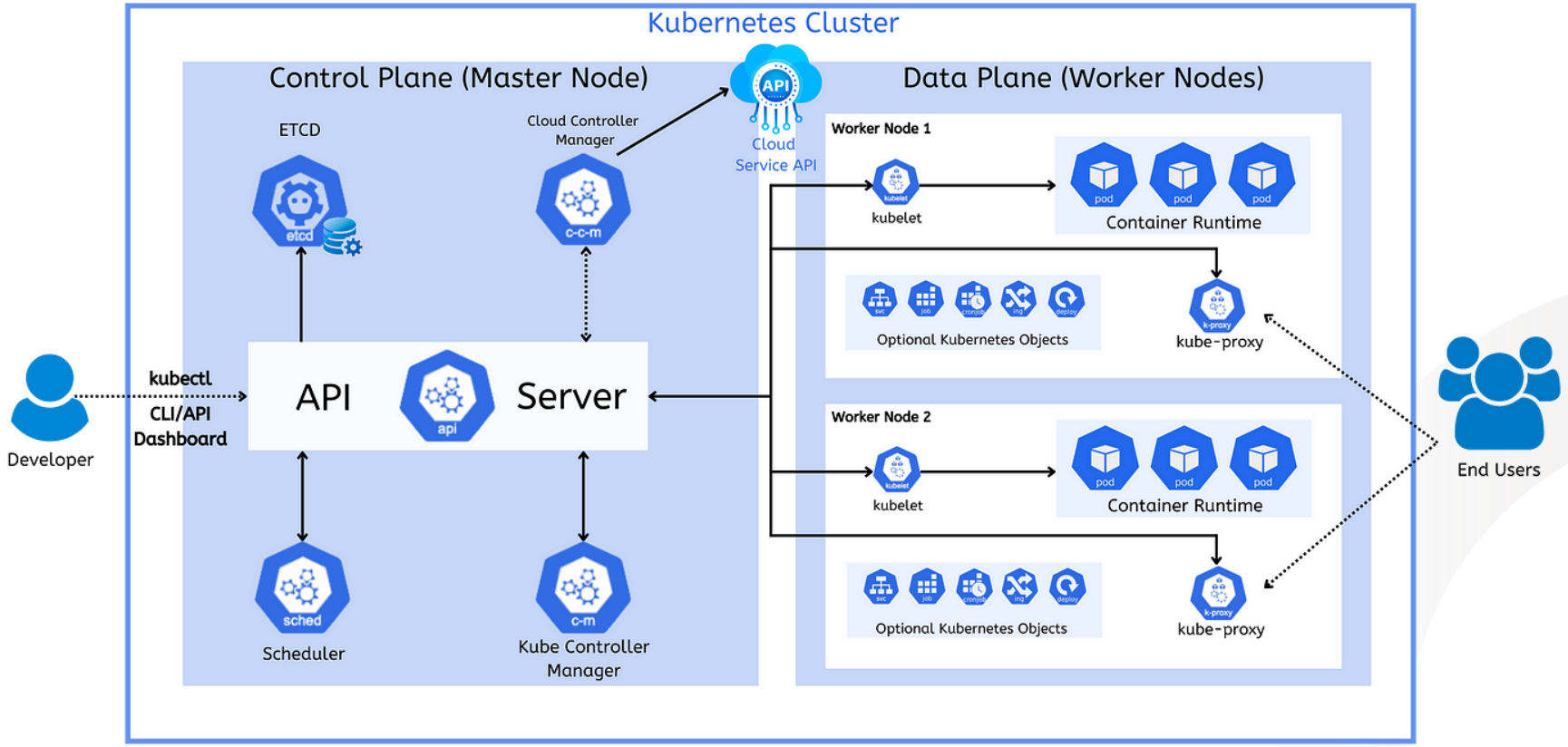
What does this look like?



A high-level overview of Docker architecture

Kubernetes

What are they?



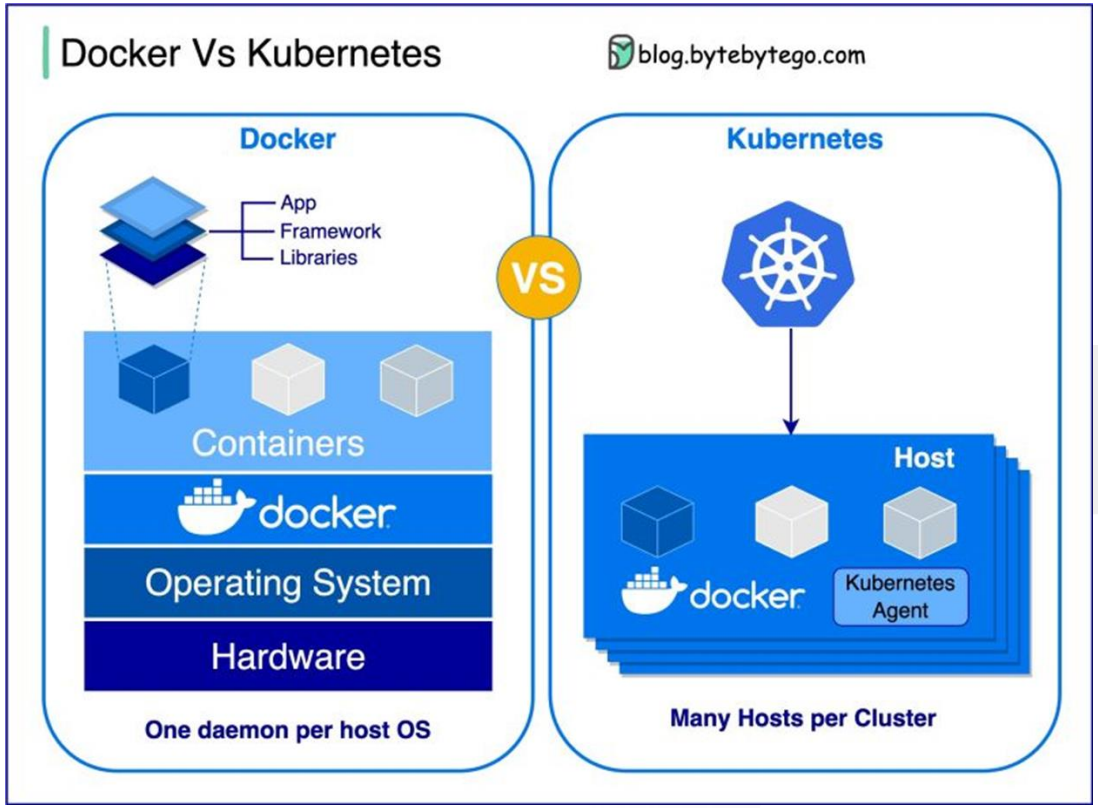
Kubernetes (K8) architecture

Docker vs Kubernetes

How do they compare?

Docker key features	Kubernetes
Isolation	Scalability
Resource management	API-Driven
Popular choice	Container runtime
	Control over resources
	Complexity

Key features of Docker and Kubernetes

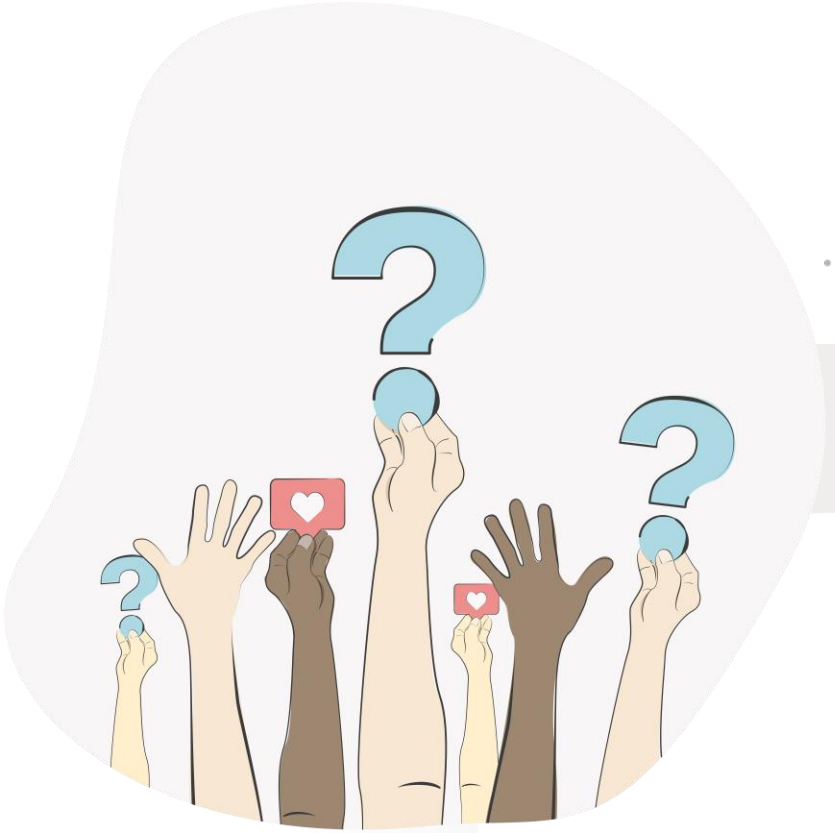


An illustration comparing Docker and Kubernetes architecture

Module consolidation

Part 1

1	Linux for Data Engineers	1. Explain Linux filesystem and structure for Big Data
		2. Show familiarity with key Linux utilities
		3. Justify the use of Linux for job scheduling
		4. Analyse log files and scripting in Linux
2	Version Control	1. Evaluate the use of Git in version control
		2. Argue the benefits of CI/CD in data projects
		3. Appraise strategies for branching and merging
		4. Conduct a code review using GitHub
3	Python for Data Engineers	1. Demonstrate Python syntax and coding conventions
		2. Analyse control flow and function usage
		3. Appreciate recursion and test-driven development
		4. Construct basic software applications

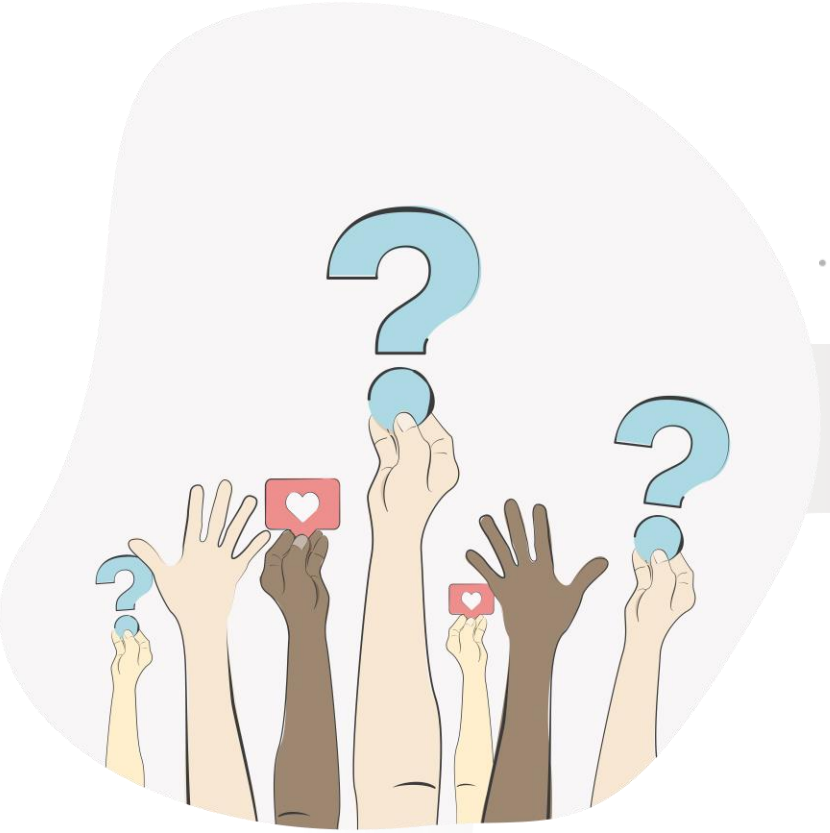


Are there any areas from this section of the module you would like to revisit?

Module consolidation

Part 2

4	Data Structures and OOP	1. Evaluate different collections and data serialisation methods
		2. Show familiarity with OOP concepts and design patterns
		3. Justify the use of specific debugging techniques
		4. Develop and test OOP-based applications
5	Data Manipulation	1. Manipulate data using Pandas DataFrames
		2. Handle common errors and perform error logging
		3. Use regex and APIs for data processing
		4. Appreciate the role of key libraries in data manipulation
6	Algorithmic Thinking	1. Develop algorithms for searching and sorting
		2. Evaluate algorithmic complexity and mitigate risks
		3. Analyse graph and machine learning algorithms
		4. Document algorithms clearly and effectively

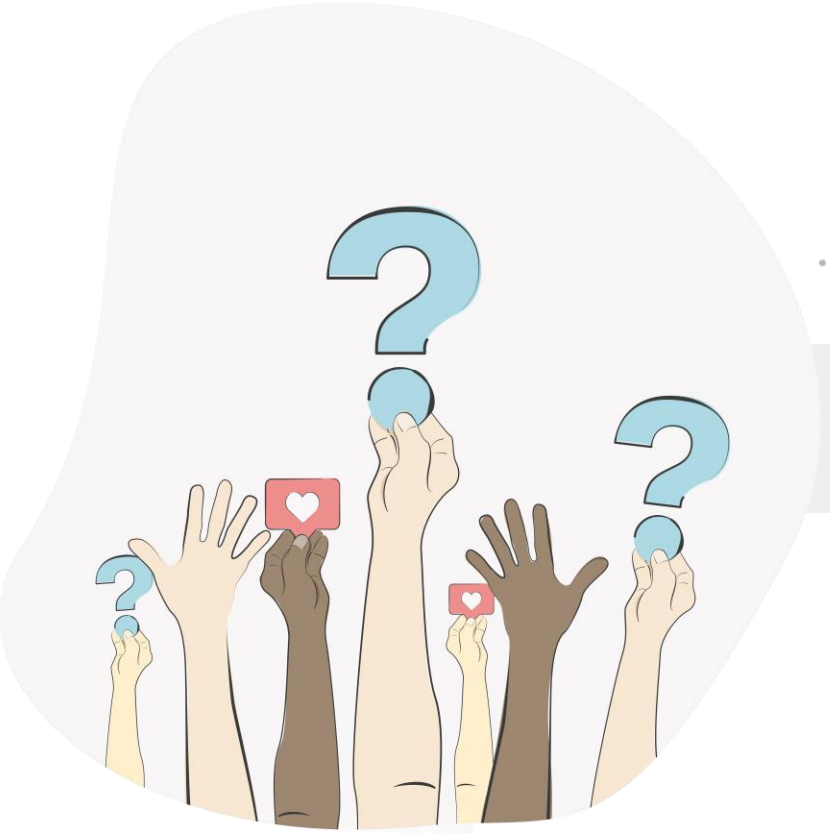


Are there any areas from this section of the module you would like to revisit?

Module consolidation

Part 3

7	Parallel Programming	1. Explain the concepts of concurrency and parallelism
		2. Utilise Databricks and PySpark for distributed computing
		3. Develop RDD algorithms
		4. Analyse performance implications of parallel algorithms
8	Spark for Data Engineers	1. Construct data pipelines using SparkSQL and Spark Streaming
		2. Evaluate the use of Spark clusters for data processing
		3. Appreciate the essentials of data pipelines
		4. Justify the deployment strategies for Spark applications
9	Practical Programming Skills	1. Utilise Docker for environment management
		2. Create data visualisations
		3. Write and perform integration testing
		4. Argue the importance of security considerations in software development



Are there any areas from this section of the module you would like to revisit?

Module quiz challenge

Welcome to the quiz challenge!

Are you ready to put your knowledge to the test? 🤔💡

The Rules of the Game:

You'll now face a series of 5 brain-teasing questions related to this module.

Each correct answer earns you points— 10 points per each correct question!

Let's see who can remember the most from the module!

Good luck!

Quiz challenge

Quiz challenge 1 of 5

In Python testing, which of the following statements is true?

A) Unit tests are used to test the system as a whole

B) Test planning is not necessary in Python testing

C) The unittest module in Python is used for constructing and running tests

D) Integration tests are used to test individual components of software

Submit your responses to the chat!

Quiz challenge 1 of 5

In Python testing, which of the following statements is true?

A) Unit tests are used to test the system as a whole

B) Test planning is not necessary in Python testing

C) The unittest module in Python is used for constructing and running tests

D) Integration tests are used to test individual components of software

Feedback C – The unittest module in Python provides a rich set of tools for constructing and running tests, making it a key part of Python testing**.**

Got it correct?! Bag yourself 10 points!

Submit your responses to the chat!

Quiz challenge 2 of 5

Which command would you use to remove the directory containing all the files in Linux?

a. mkdir

b. ls

c. rm –r

d. rmdir

Submit your responses to the chat!

Quiz challenge 2 of 5

Which command would you use to remove the directory containing all the files in Linux?

a. mkdir

b. ls

c. rm –r

d. rmdir

Feedback: C – In Linux, rmdir and rm -r are both commands that remove directories, but rmdir requires an empty directory to delete.

Got it correct?! Bag yourself 10 points! 💰

Submit your responses to the chat!

Quiz challenge 3 of 5

In Python testing, which of the following statements is true?

A) Unit tests are used to test the system as a whole

B) Test planning is not necessary in Python testing

C) The unittest module in Python is used for constructing and running tests

D) Integration tests are used to test individual components of software

Submit your responses to the chat!

Quiz challenge 3 of 5

In Python testing, which of the following statements is true?

A) Unit tests are used to test the system as a whole

B) Test planning is not necessary in Python testing

C) The unittest module in Python is used for constructing and running tests

D) Integration tests are used to test individual components of software

Feedback: C – The unittest module in Python provides a rich set of tools for constructing and running tests, making it a key part of Python testing.

Got it correct?! Bag yourself 10 points!

Submit your responses to the chat!

Quiz challenge 4 of 5

Which Pandas function allows you to join two DataFrames by matching the indexes row-wise?

A) concat()

B) append()

C) concat()

D) Join ()

Submit your responses to the chat!

Quiz challenge 4 of 5

Which Pandas function allows you to join two DataFrames by matching the indexes row-wise?

A) concat()

B) append()

C) concat()

D) Join ()

Feedback: D - The pandas .join() method joins DataFrames by matching the index labels row-wise, similar to a SQL left join.

Got it correct?! Bag yourself 10 points!

Submit your responses to the chat!

BPP

Quiz challenge 5 of 5

What is the primary purpose of Kubernetes?

A) Source code management

B) Automated testing

C) Container orchestration

D) Configuration management

Submit your responses to the chat!

Quiz challenge 5 of 5

What is the primary purpose of Kubernetes?

A) Source code management

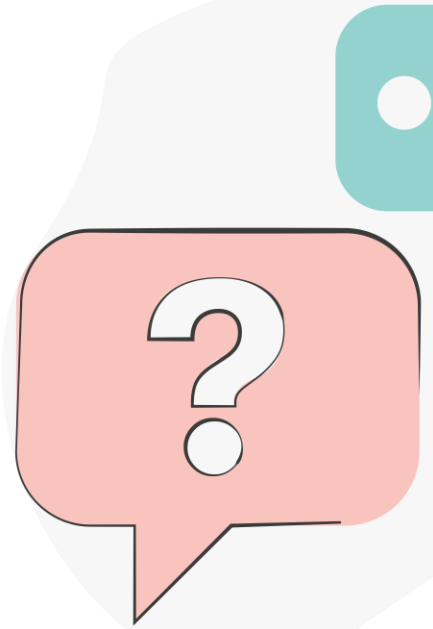
B) Automated testing

C) Container orchestration

D) Configuration management

Feedback: C - Kubernetes is a powerful open-source platform designed for automating, managing, and orchestrating containerised applications.

Got it correct?! Bag yourself 10 points! 💰



Submit your responses to the chat!

Well-done!

And the winner is…!

Congratulations - you are this week's data science champion!

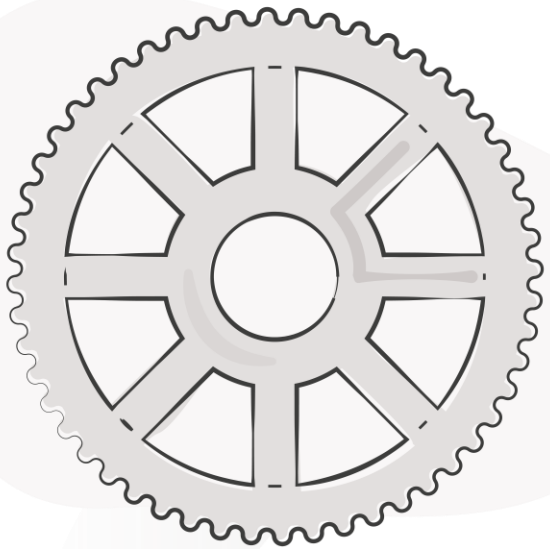
Practical application

Tutorial walkthrough

Your tutor will now walk you through a notebook creating a data pipeline that follows programming best practices and results in a data visualisation.

The link is available here:

Python notebook



Practical application

Thank you

Do you have any questions, comments, or feedback?

How confident do you now feel about your knowledge of practical programming and this module following this webinar?

- **A:** Very confident
- **B:** More confident than before this webinar
- **C:** What was this webinar session even about?!

Submit your responses to the chat!