

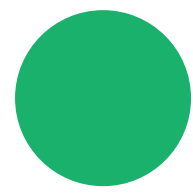


#CLOUDENGINEERINGSUMMIT

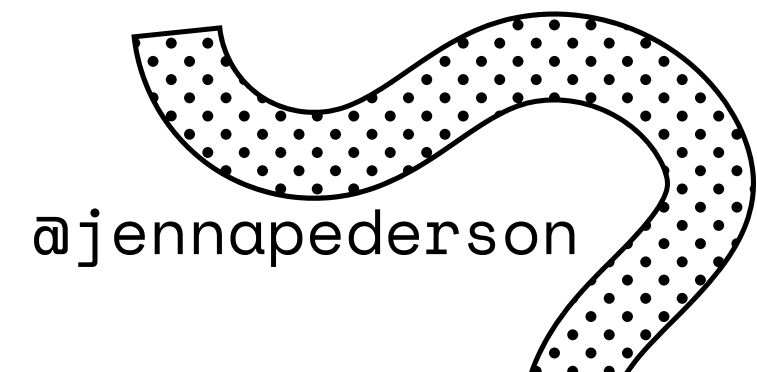
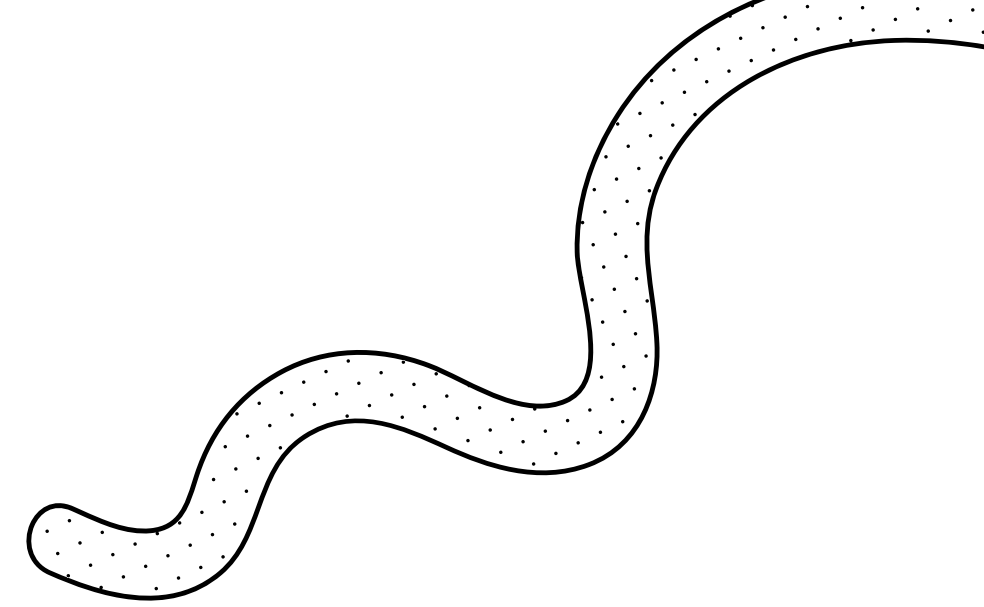
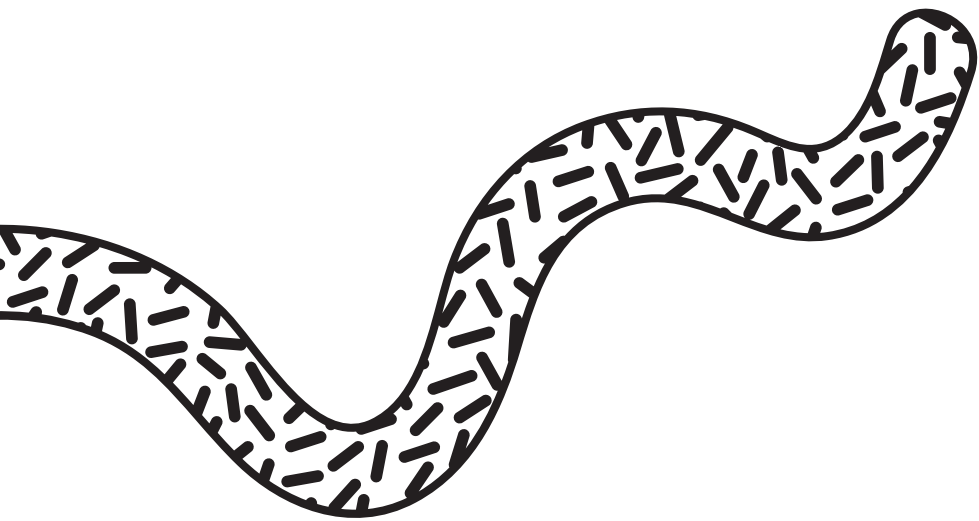
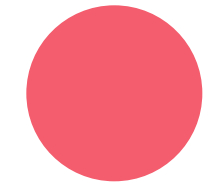
Bringing software  
development practices  
to your **infrastructure**



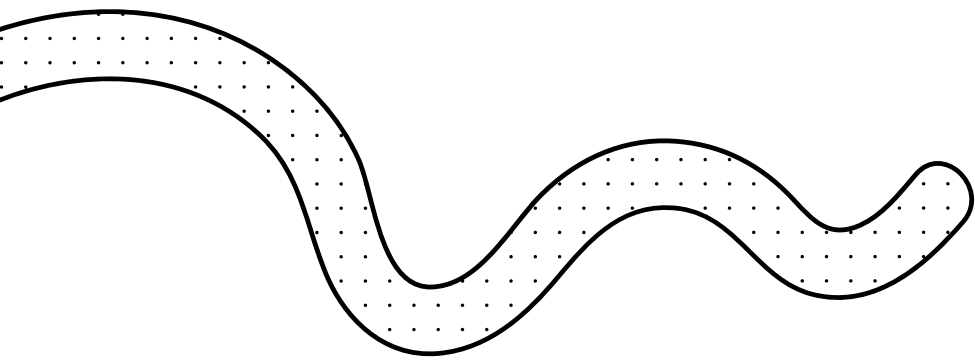
@jennapederson



# The awesomeness of Infrastructure as Code



@jennapederson



# Infrastructure as Code

## IS Code

```
2  constructor(scope: Construct, id: string, props: HttpListenerProps) {
3    super(scope, id, props);
4
5    const taskDefinition = new ecs.FargateTaskDefinition(this, 'TaskDe
6      memoryLimitMiB: 1024,
7      cpu: 256,
8    });
9
10   const image = props.image || new ecs.AssetImage(path.join(__dirname,
11   const container = taskDefinition.addContainer("WebServer", {
12     image,
13   });
14   container.addPortMappings({containerPort: 80});
15
16   const service = new ecs.FargateService(this, 'Service', {
17     cluster: props.cluster,
18     taskDefinition,
19   });
20
21   const lb = new elbv2.ApplicationLoadBalancer(this, 'LB', {
22     vpc: props.vpc,
23     internetFacing: true,
24   });
25
26   const listener = lb.addListener('HttpListener', {
27     port: 80,
28   });
29 }
```

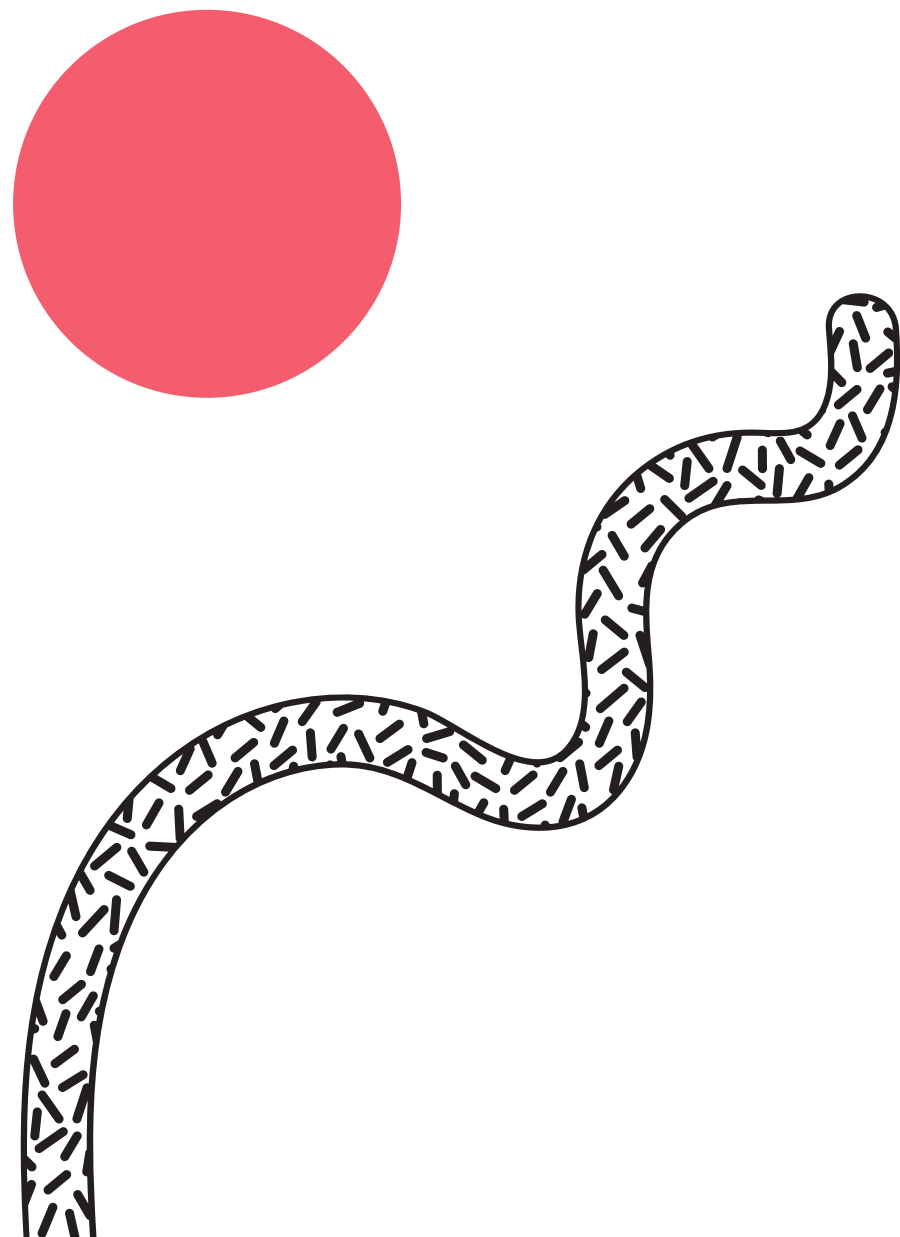


@jennapederson



(or account or region)

# Agenda



## **Different Types of Testing**

Using the right type at the right time

## **Using Test Driven Development**

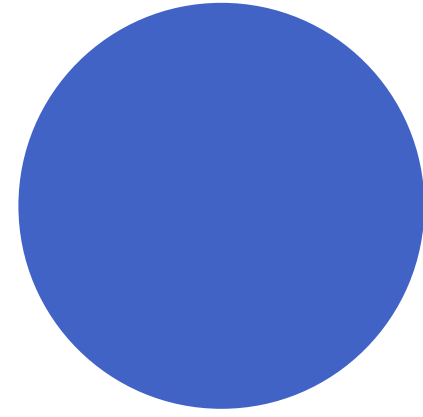
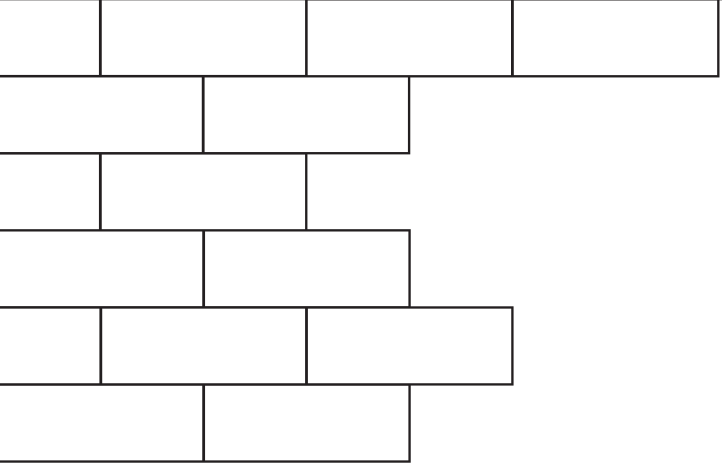
Build what you need and only what you need

## **Testing Your Infrastructure Directly**

Making sure it was created correctly and hasn't drifted

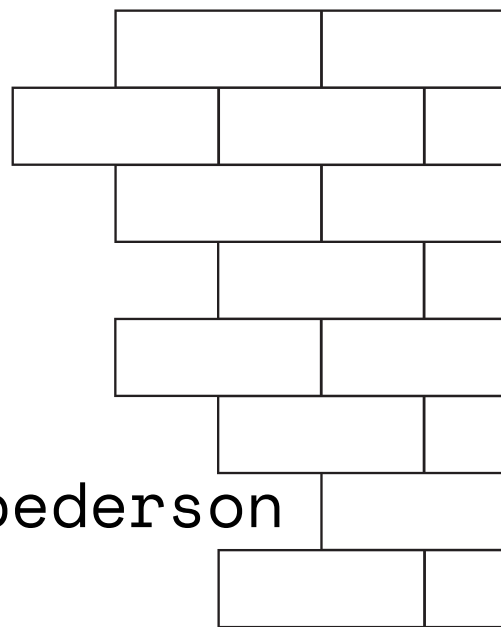
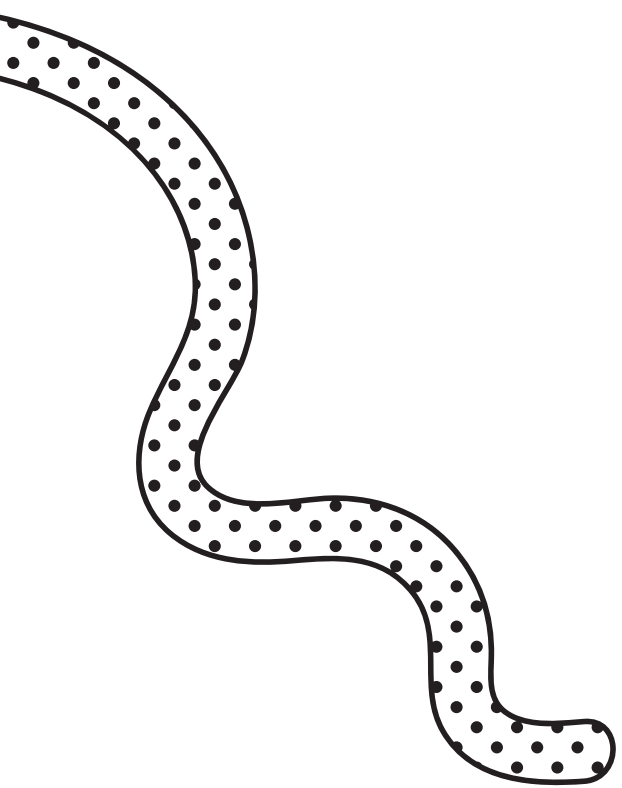
## **Using a CI/CD Pipeline**

Run tests in the real world and isolate issues quicker

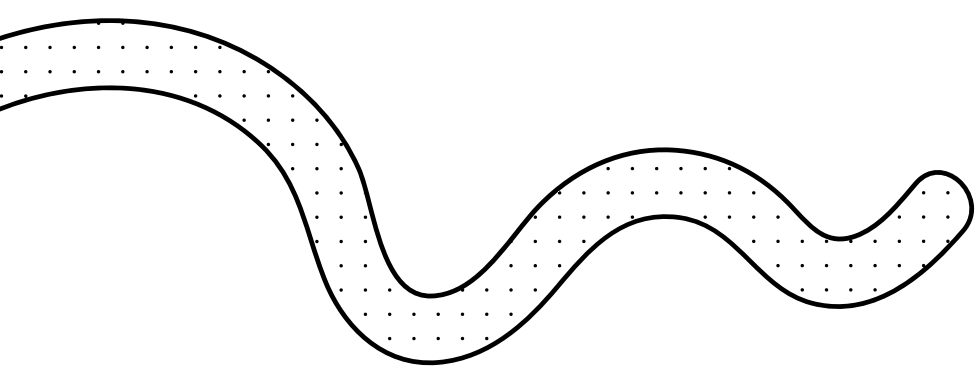


# Why Test Infrastructure?

The cloud makes it easier and quicker to provision infrastructure, but there is complexity with that scale.

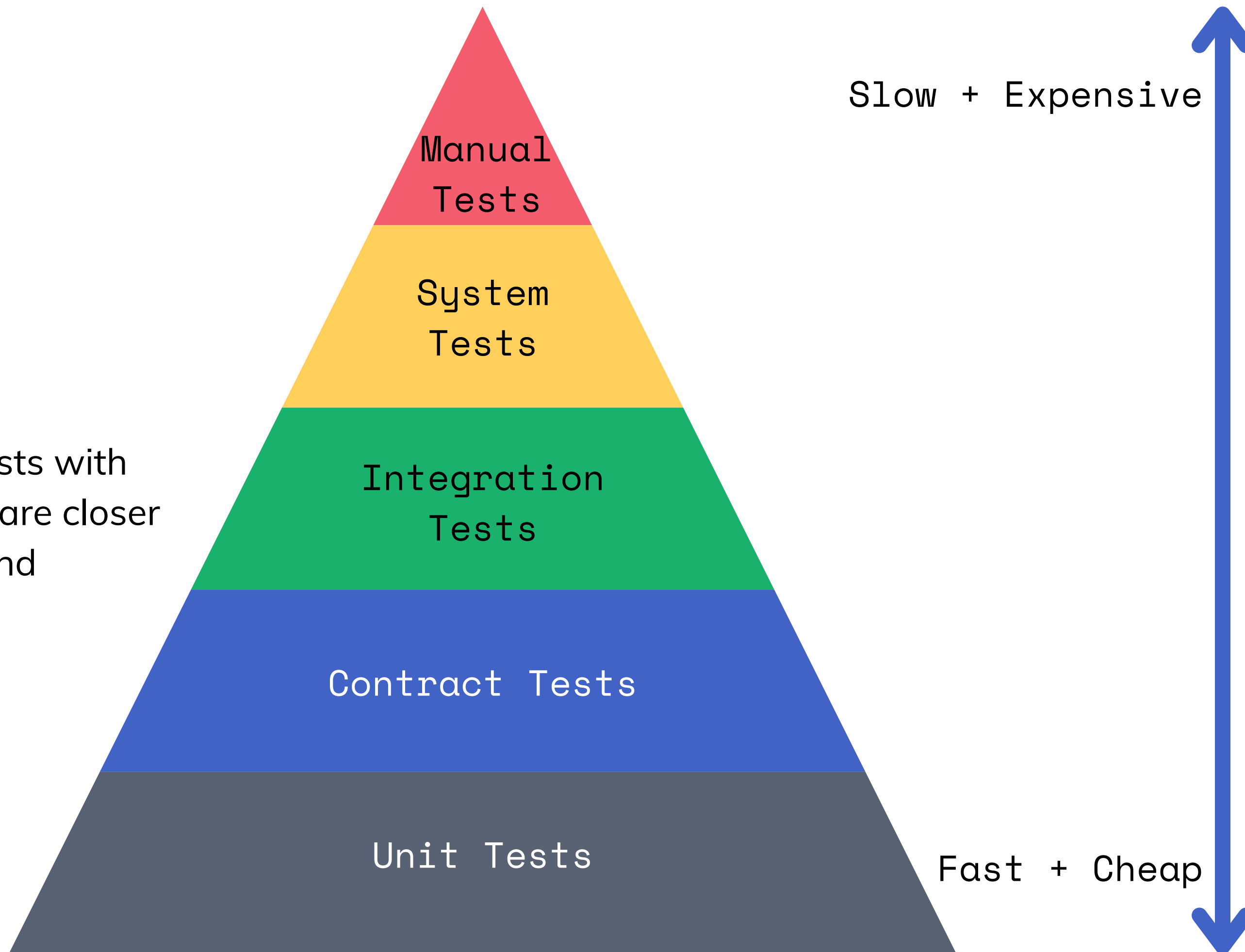
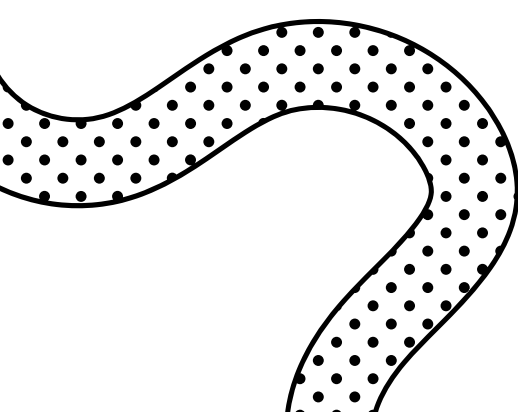


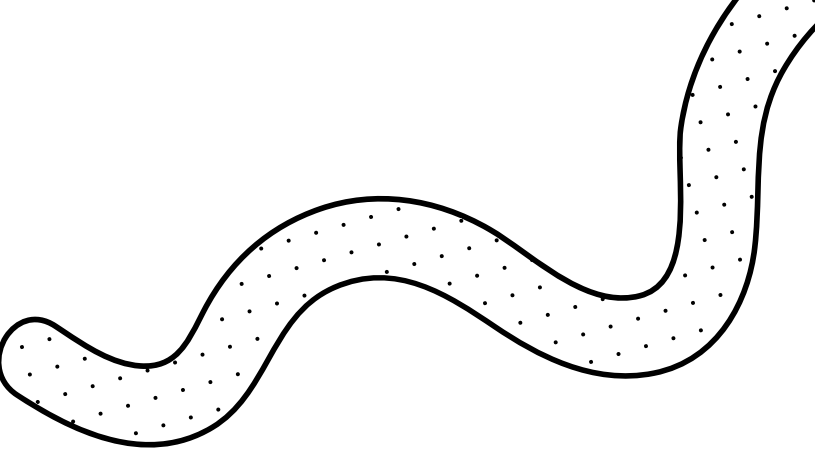
@jennapederson



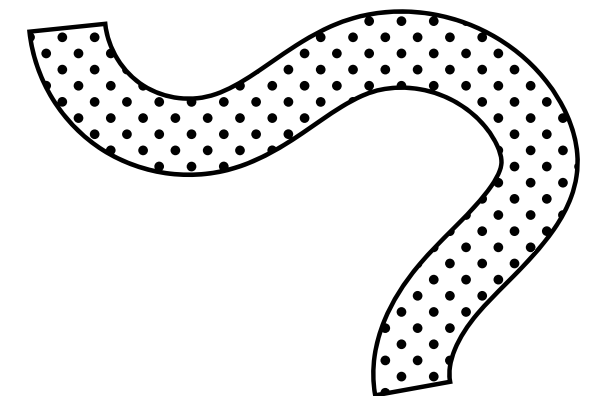
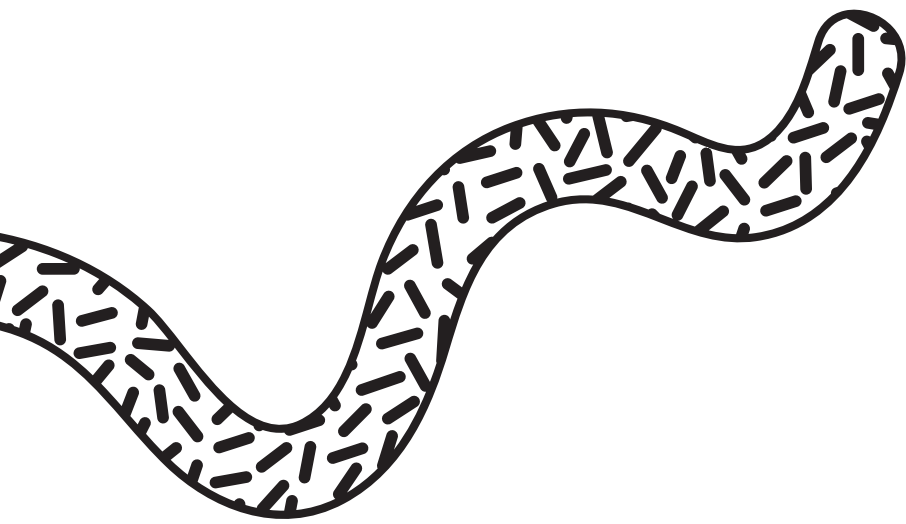
## Failing Fast

Balance fast and cheap tests with more expensive tests that are closer to the real infrastructure and production environment.





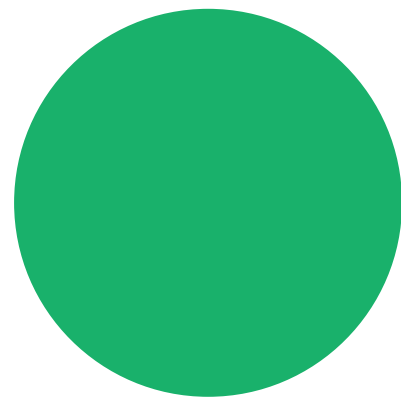
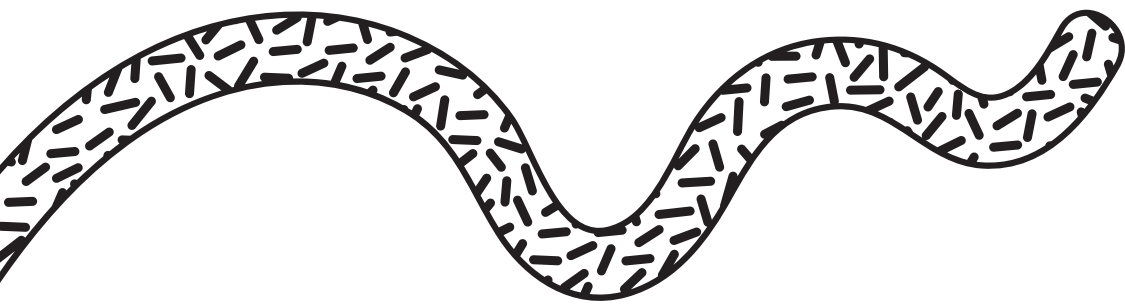
If you're TDDing your **application code**,  
why not do the same for your  
**infrastructure code**?



@jennapederson

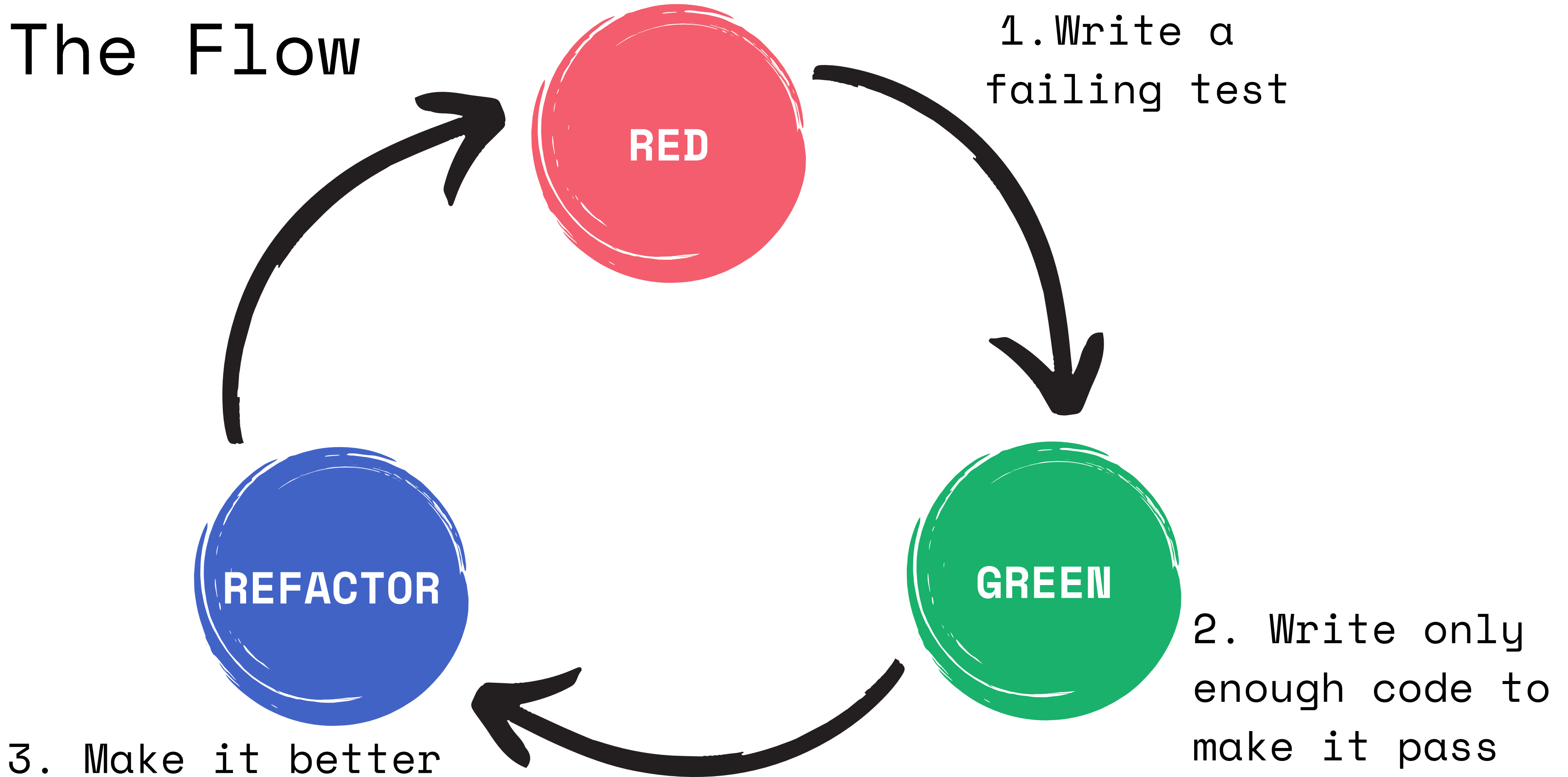


# Benefits of TDD

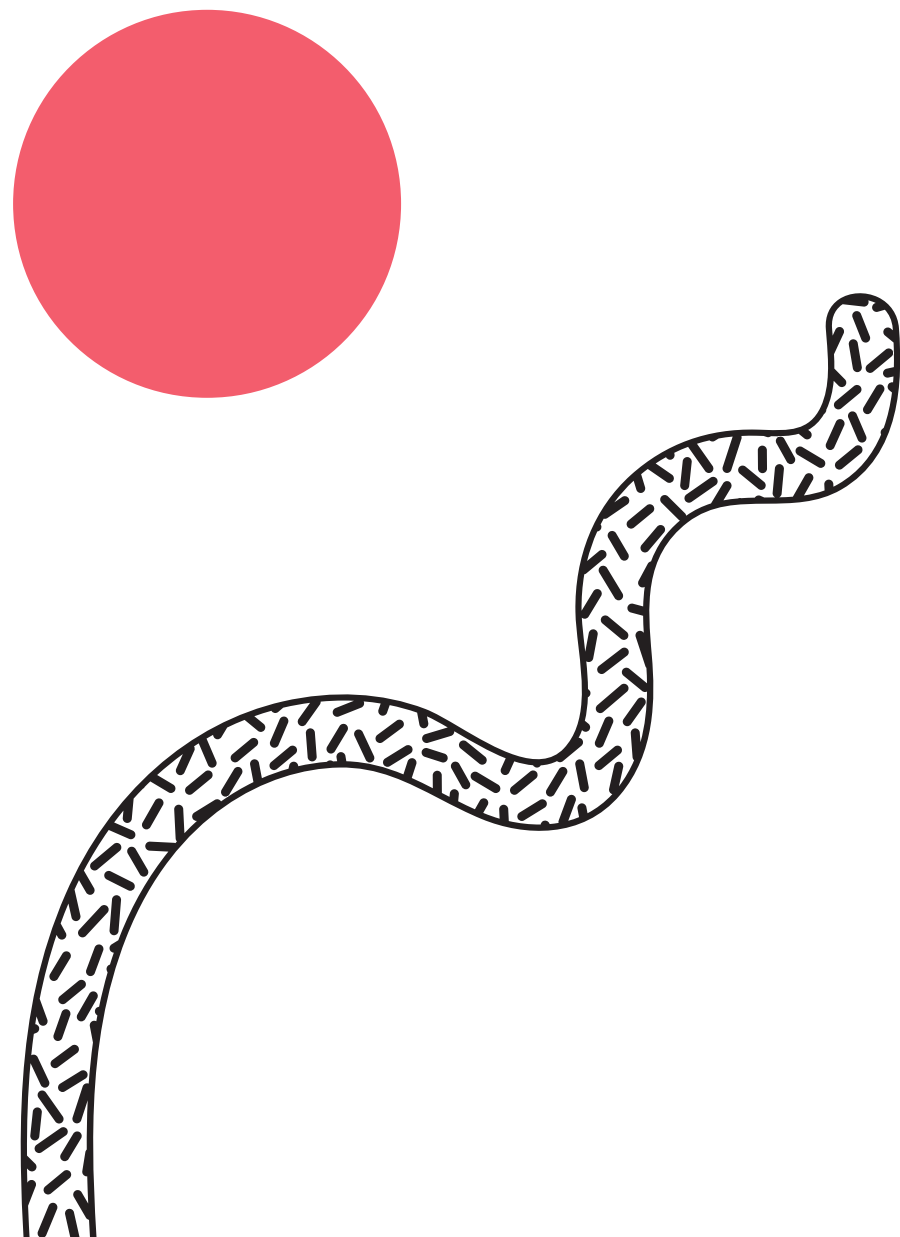


- Reduced defect rates
- Improve the overall design
- Focused on requirements
- Focused on small chunks
- Serves as documentation
- Confidence

# The Flow

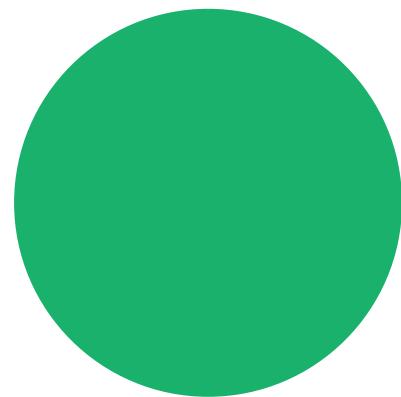
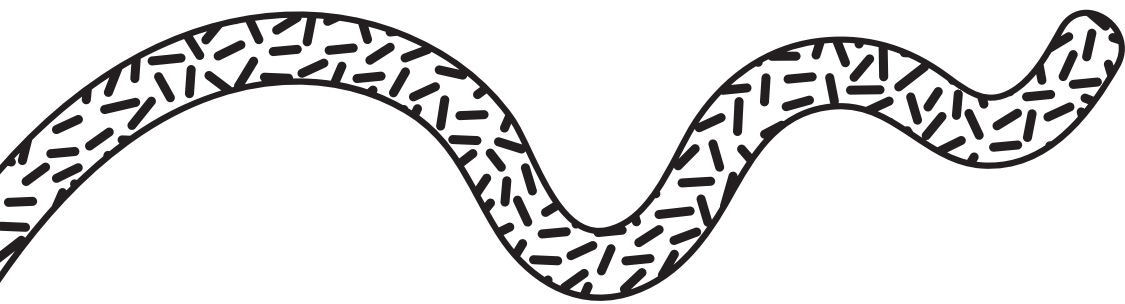


# What is a unit test?



- Exercises a small part of your application, one unit, and verifies that it's correct.
- Get feedback early on to shorten the feedback loop between changes
- Serves as documentation
- Can be run in your CI/CD tool
- Isolated from other resources and external APIs

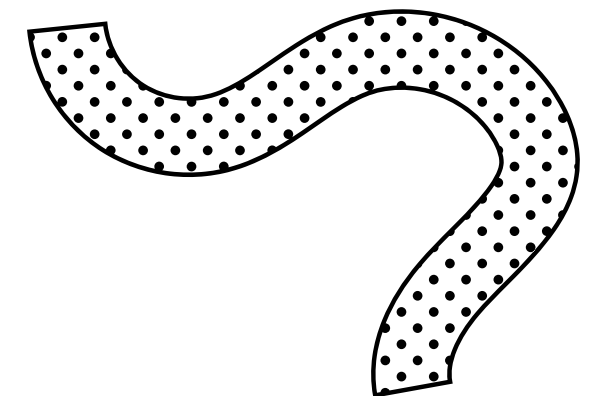
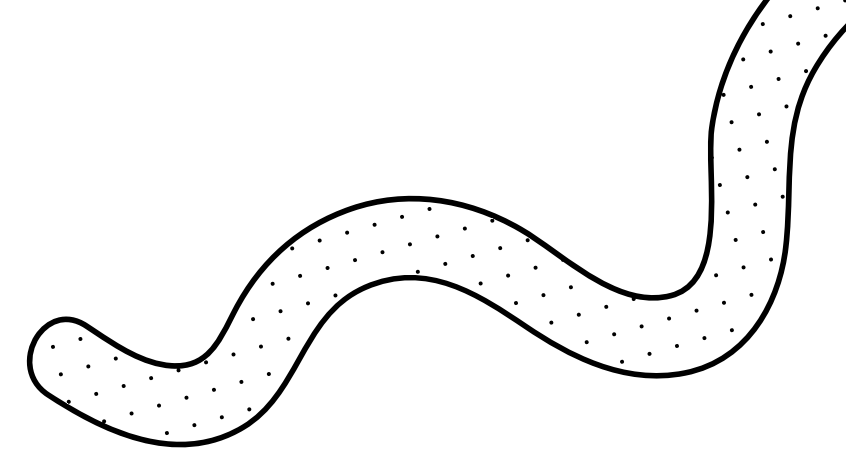
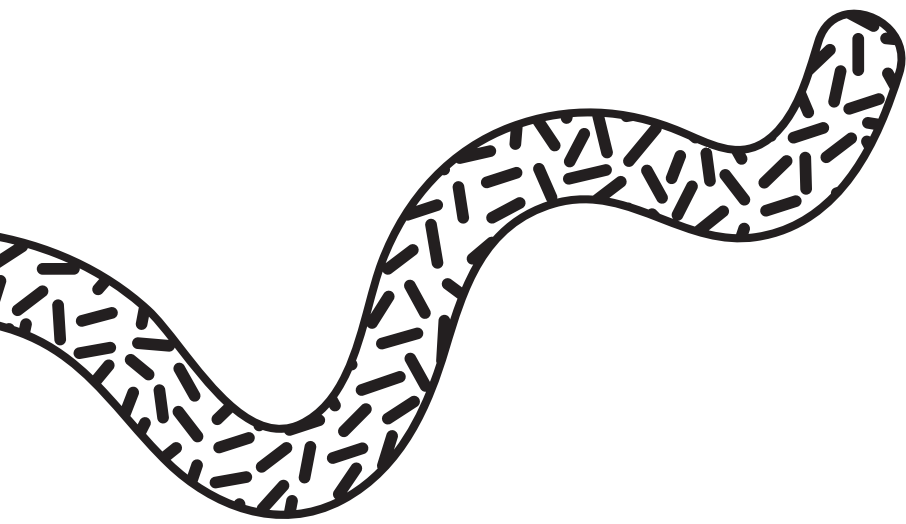
# Unit testing infrastructure code



- If a resource will be created with the correct configuration
- The correct number of resources will be created
- Dependencies between resources are correct
- Interpolated values are correct

# Demo

S3 + CDK + Jest



@jennapederson

EXPLORER

INFRASTRUCTUR... [Icons]

- > bin
- > **cdk.out**
- > inspec
- > node\_modules

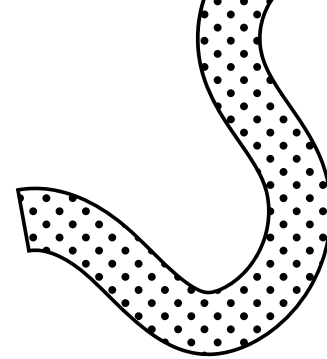
**> test**

- ◆ .gitignore
- npm .npmignore
- { } cdk.json
- JS jest.config.js
- 📄 LICENSE
- { } package-lock.json
- { } package.json
- 📄 README.md
- TS tsconfig.json

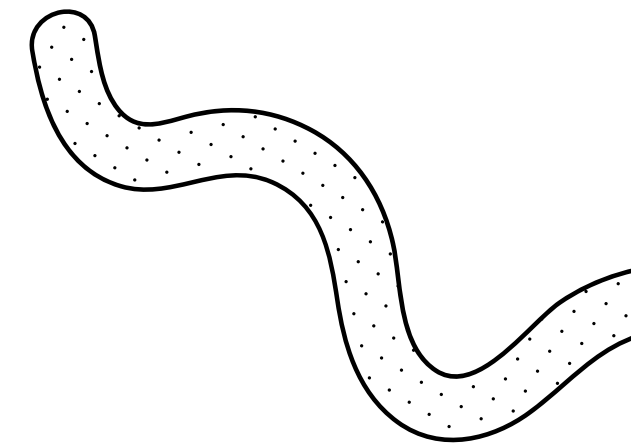
- > OUTLINE
- > TIMELINE
- > NPM SCRIPTS



- Show All Commands ⌘ P
- Go to File ⌘ P
- Find in Files ⌘ ⌘ F
- Start Debugging F5
- Toggle Terminal ^ `



# How do we go from code to infrastructure?



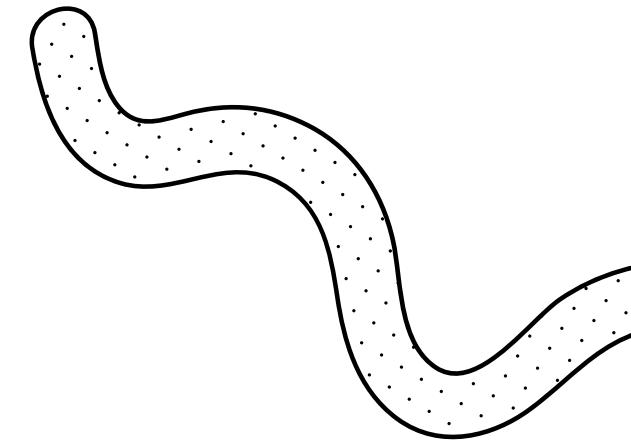
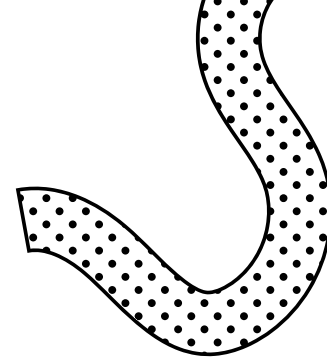
```
1  super(scope, id, props);
2
3  super(scope, id, props);
4
5  const taskDefinition = new ecs.FargateTaskDefinition(this, 'TaskDe
6    memoryLimitMiB: 1024,
7    cpu: 256,
8  });
9
10 const image = props.image || new ecs.AssetImage(path.join(__dirnam
11 const container = taskDefinition.addContainer("WebServer", {
12   image,
13 });
14 container.addPortMappings({containerPort: 80});
15
16 const service = new ecs.FargateService(this, 'Service', {
17   cluster: props.cluster,
18   taskDefinition,
19 });
20
21 const lb = new elbv2.ApplicationLoadBalancer(this, 'LB', {
22   vpc: props.vpc,
23   internetFacing: true,
24 });
25
26 const listener = lb.addListener('HttpListener', {
27   port: 80
```

## InfrastructureTestExamplesStack

[Delete](#)[Update](#)[Sta](#)[Stack info](#)[Events](#)[Resources](#)[Outputs](#)[Parameters](#)[Template](#)[Change sets](#)

### Events (100+)

Timestamp	Logical ID	Status	Status reason
2021-07-17 20:41:34 UTC-0500	InfrastructureTestExampl esStack	✔ UPDATE_COMPLETE	-
2021-07-17 20:41:33 UTC-0500	InfrastructureTestExampl esStack	ⓘ UPDATE_COMPLETE_C LEANUP_IN_PROGRES S	-
2021-07-17 20:41:05 UTC-0500	InfrastructureTestExampl esStack	ⓘ UPDATE_IN_PROGRES S	User Initiated
2021-07-17 20:01:16 UTC-0500	InfrastructureTestExampl esStack	✔ UPDATE_COMPLETE	-
2021-07-17 20:01:16 UTC-0500	InfrastructureTestExampl esStack	ⓘ UPDATE_COMPLETE_C LEANUP_IN_PROGRES	-

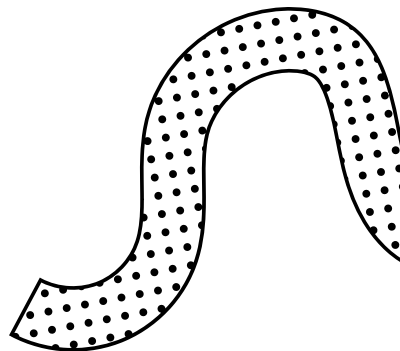
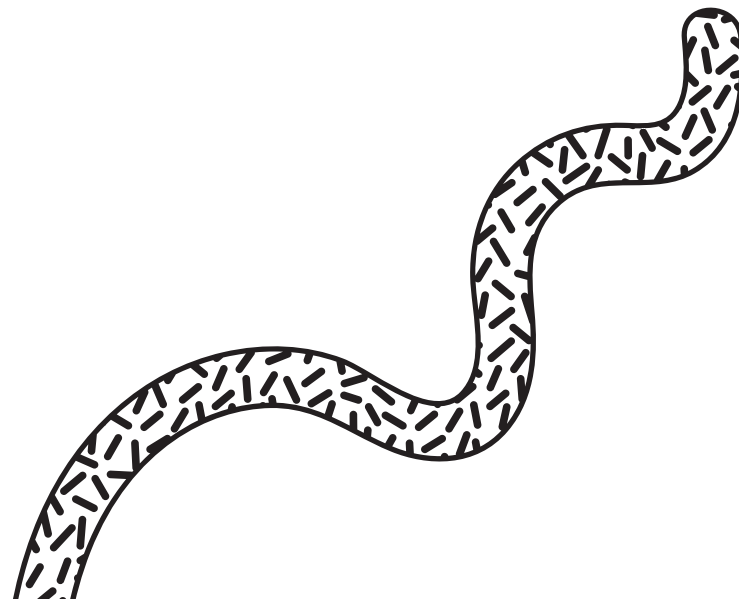


## What is an Integration Test?

Tests the interactions across different units or modules, or in the case of infrastructure testing, across cloud resources.

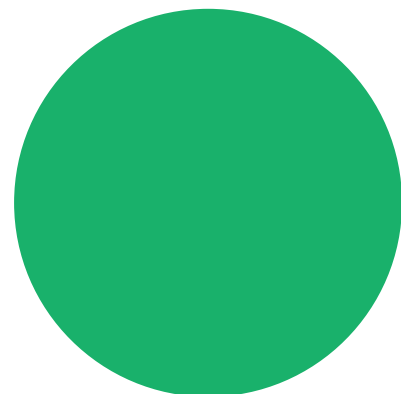
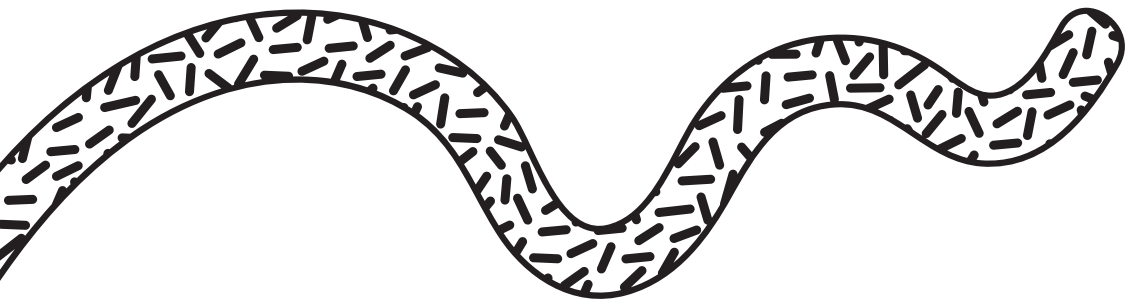
Verifies your provisioned cloud resources are created and configured as you expect them to be.

Gives you confidence in infrastructure at scale and at velocity.





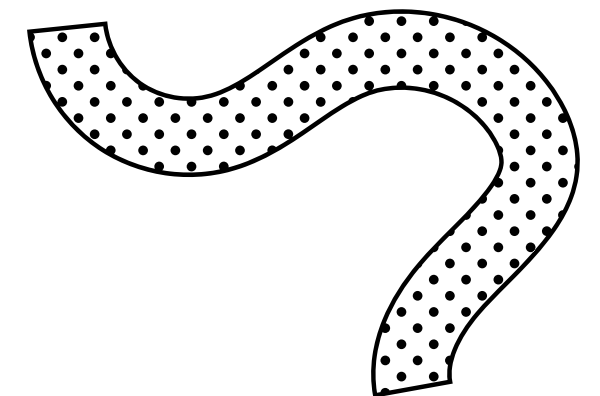
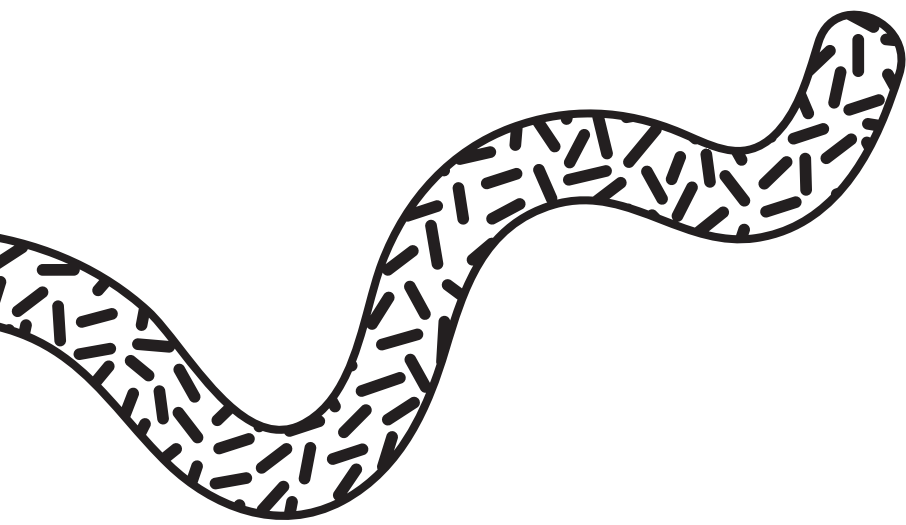
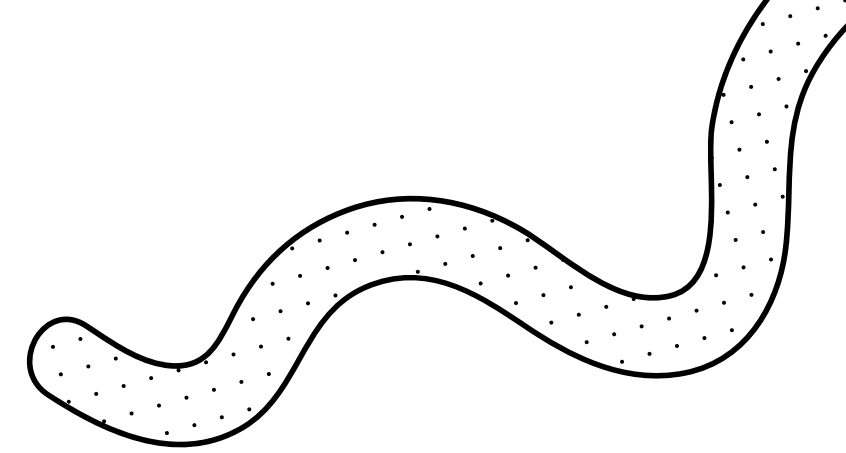
# Chef InSpec



- Open-source framework to test and audit cloud resources IN the cloud
- Tests are written with a DSL
- Test resources that are managed manually or with code
- Can be used across teams

# Demo

EC2 + RDS + CDK + InSpec



@jennapederson

EXPLORER

- INFRASTRUCTURE-TEST-EXAMPLES
  - bin
  - cdk.out
  - inspec
  - lib
  - node\_modules
  - test
- .gitignore
- .npmignore
- cdk.json
- jest.config.js
- LICENSE
- package-lock.json
- package.json
- README.md
- tsconfig.json

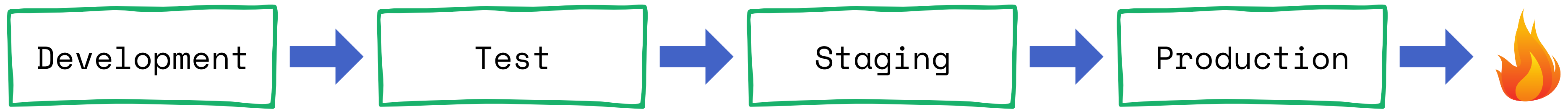
OUTLINE

NPM SCRIPTS



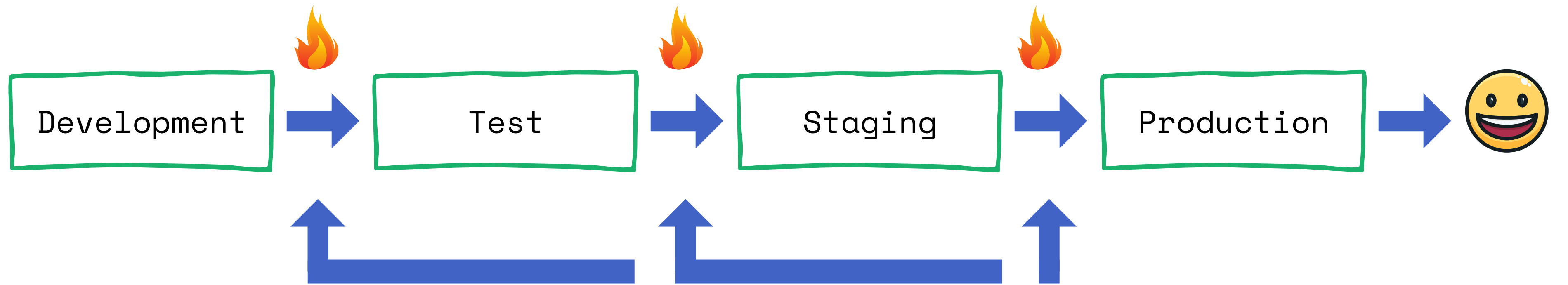
- Show All Commands ⇧ ⌘ P
- Go to File ⌘ P
- Find in Files ⇧ ⌘ F
- Start Debugging F5
- Toggle Terminal ^ `

# Without CI/CD



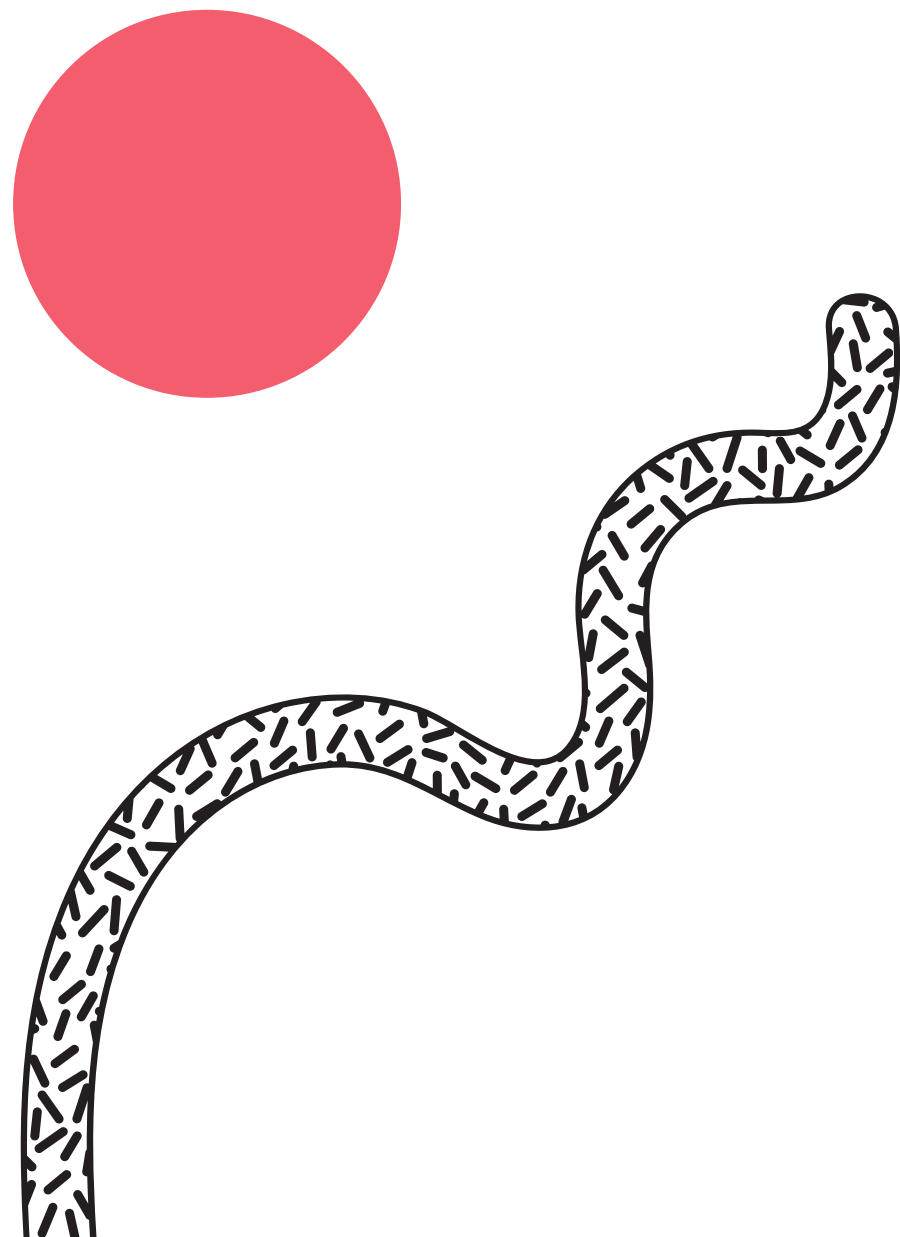
@jennapederson

# With CI/CD



@jennapederson

# Wrapping Up



Infrastructure code is like any other code, treat it as such.

Testing is never done, even once you reach production.

It's cheaper to detect broken code early.

**Thank you!**



@jennapederson



/in/jennapederson



jennapederson



<https://jenna.link/hq7>



Feedback

