



# Deep Dive: Infrastructure as Code

Chetan Dandekar, Senior Product Manager, AWS



# You are on-board ...

Business

needs to experiment, innovate, reduce risk

---

Continuous  
Delivery

of services and applications

---

DevOps

culture, automation, measurement,  
sharing

---

Cloud

infrastructure-as-code

---



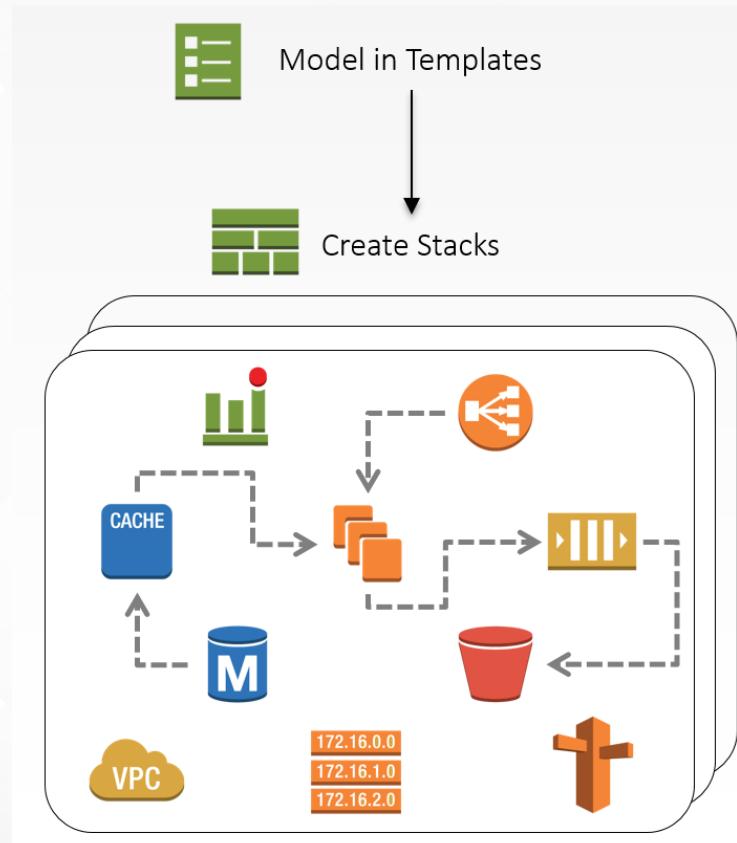
---

# AWS CloudFormation

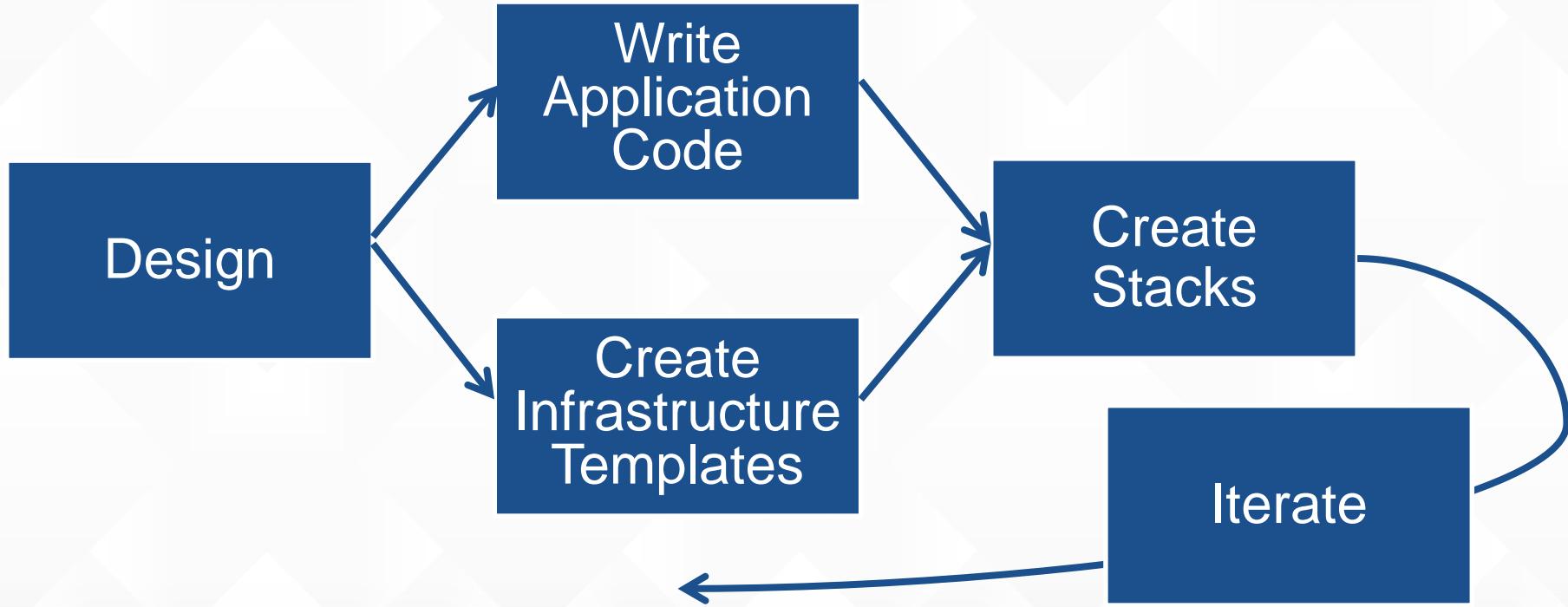


# AWS CloudFormation

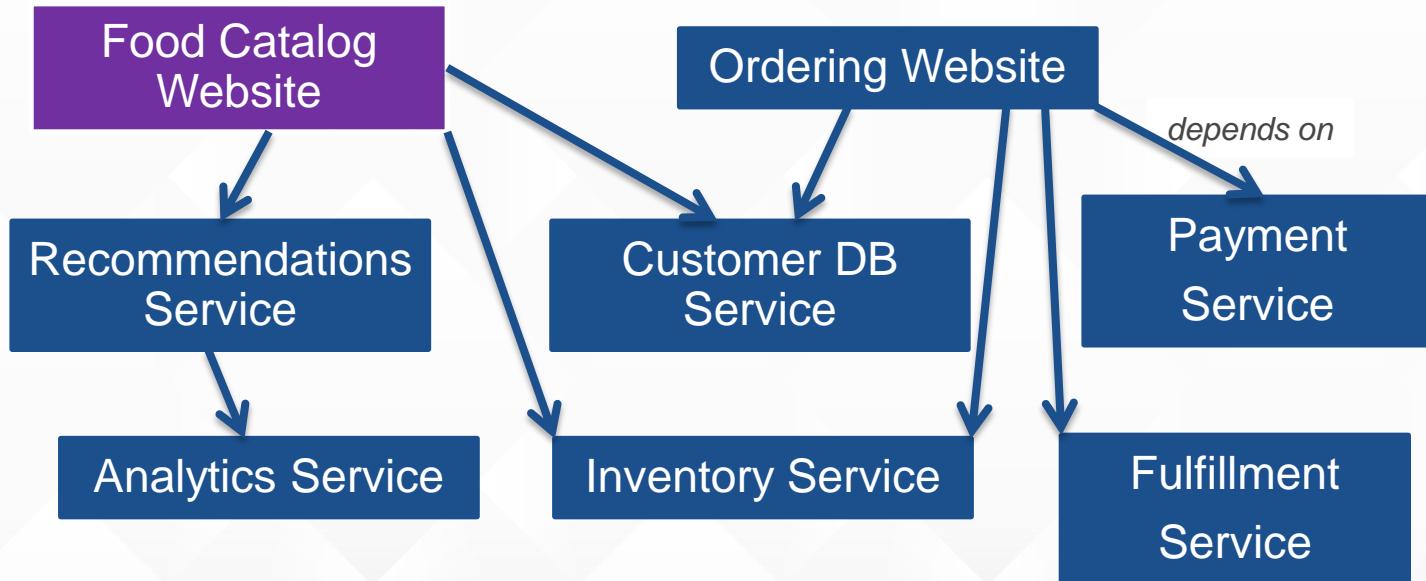
- Create templates of the infrastructure and applications you want to run on AWS
- Have the CloudFormation service automatically provision the required AWS resources and their relationships from the templates
- Easily version control, replicate or update the infrastructure and applications using the templates
- Integrates with other development, CI/CD, and management tools.



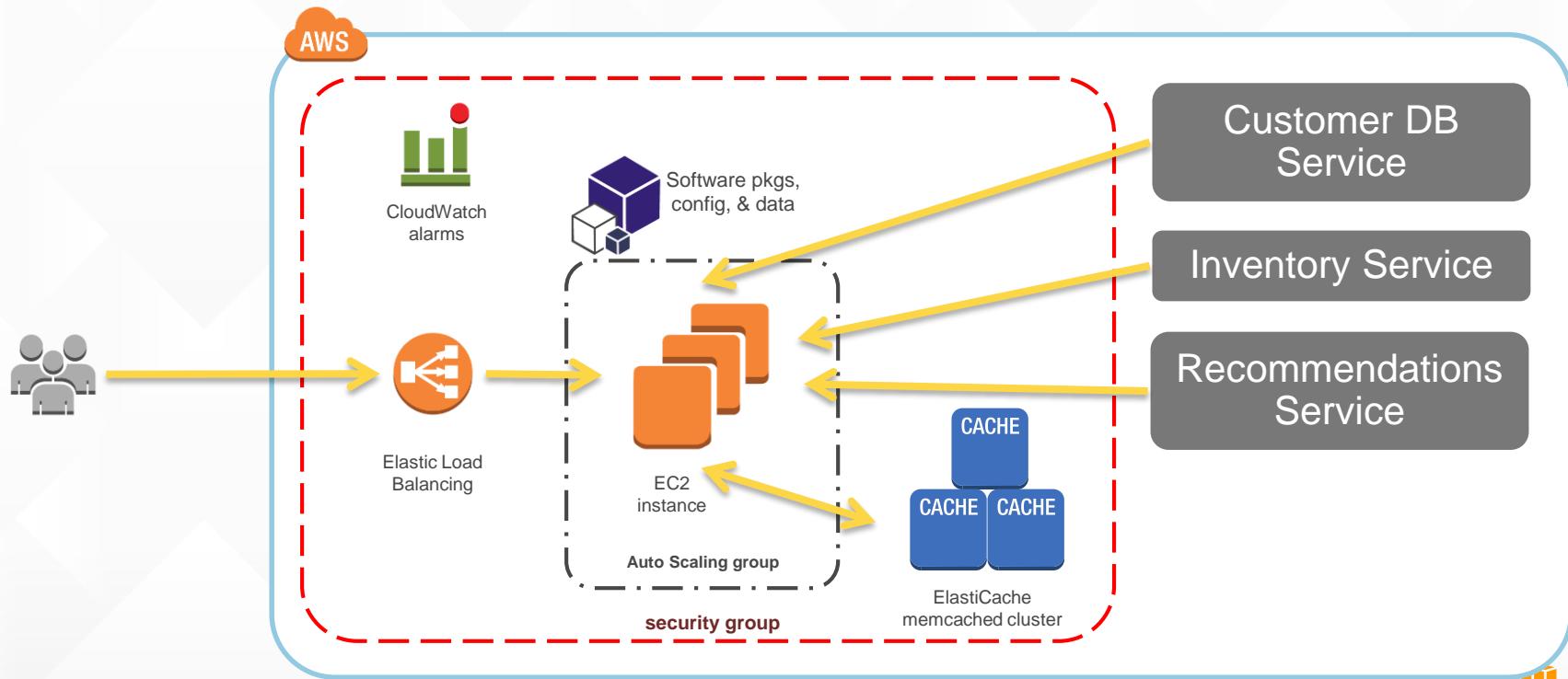
# Basic workflow



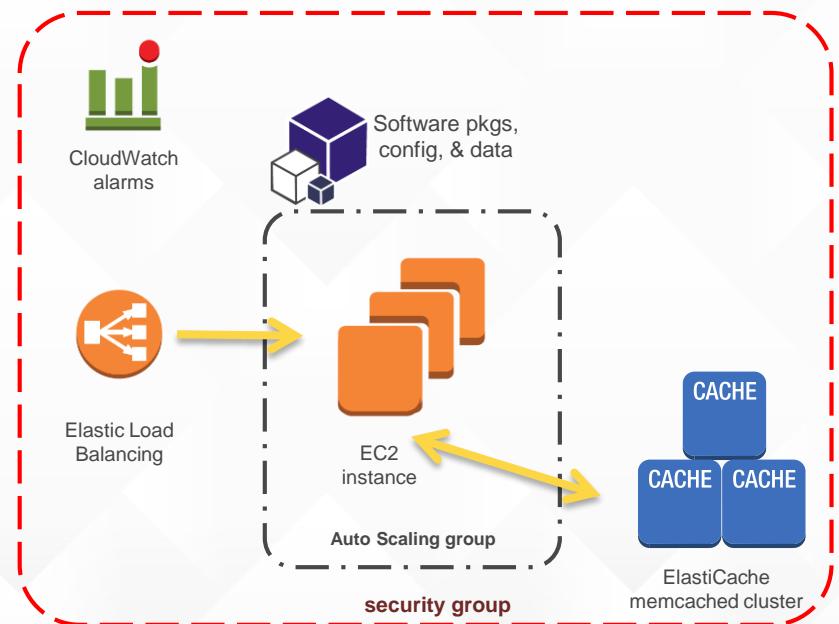
# Design - Imagine building a food ordering service



# Create template – For example, for the food catalog website



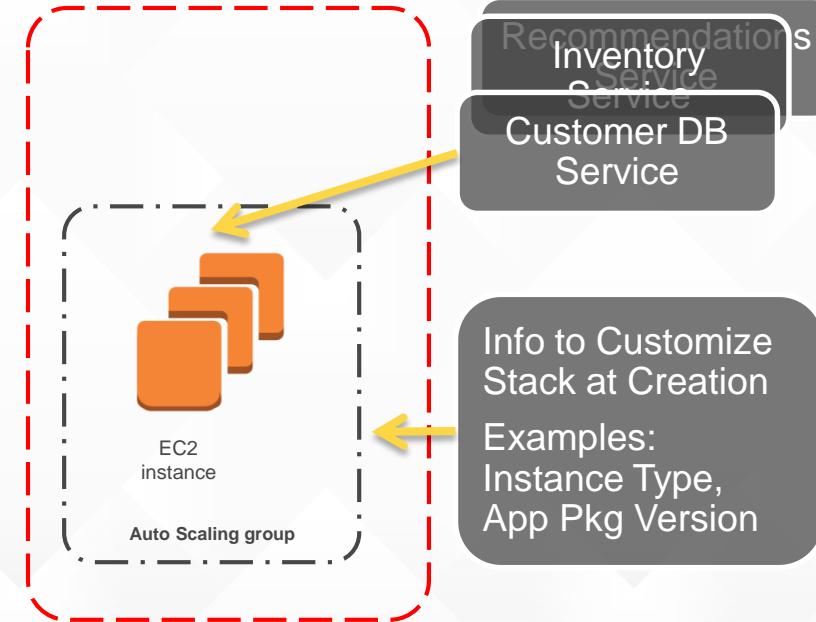
# Create template – Resources



CloudFormation Template

```
"Resources" : {  
    "SecurityGroup" : {},  
    "WebServerGroup" : {  
        "Type" : "AWS::AutoScaling::AutoScalingGroup",  
        "Properties" : {  
            "MinSize" : "1",  
            "MaxSize" : "3",  
            "LoadBalancerNames" : [ { "Ref" :  
                "LoadBalancer" } ],  
            ...  
        }  
    },  
    "LoadBalancer" : {},  
    "CacheCluster" : {},  
    "Alarm" : {}  
},
```

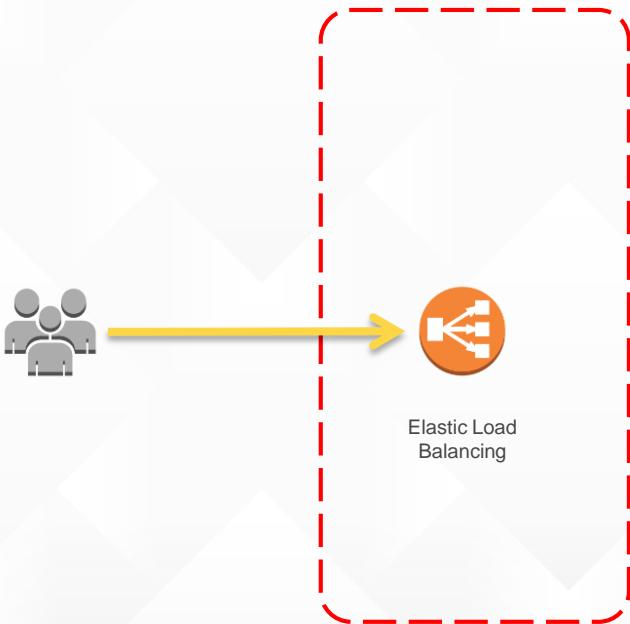
# Create template – Parameters



## CloudFormation Template

```
"Parameters" : {  
    "CustomerDBServiceEndPoint" : {  
        "Description" : "URL of the Customer DB Service",  
        "Type" : "String"  
    },  
    "CustomerDBServiceKey" : {  
        "Description" : "API key for the Customer DB Service",  
        "Type" : "String",  
        "NoEcho" : "true"  
    },  
    "InstanceType" : {  
        "Description" : "WebServer EC2 instance type",  
        "Type" : "String",  
        "Default" : "m3.medium",  
        "AllowedValues" :  
            ["m3.medium", "m3.large", "m3.xlarge"],  
        "ConstraintDescription" : "Must be a valid instance type"  
    }  
}
```

# Create template – Outputs

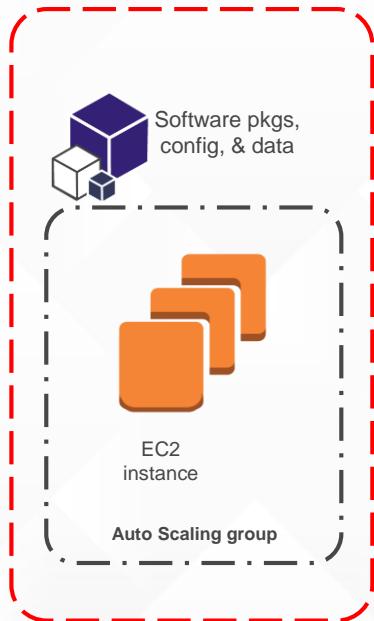


## CloudFormation Template

```
"Resources" : {  
    "LoadBalancer" : {},  
    ...  
},  
"Outputs" : {  
    "WebsiteDNSName" : {  
        "Description" : "The DNS name of the website",  
        "Value" : {  
            "Fn::GetAtt" : [ "LoadBalancer", "DNSName" ]  
        }  
    }  
}
```

# Create template – Deploy and configure software

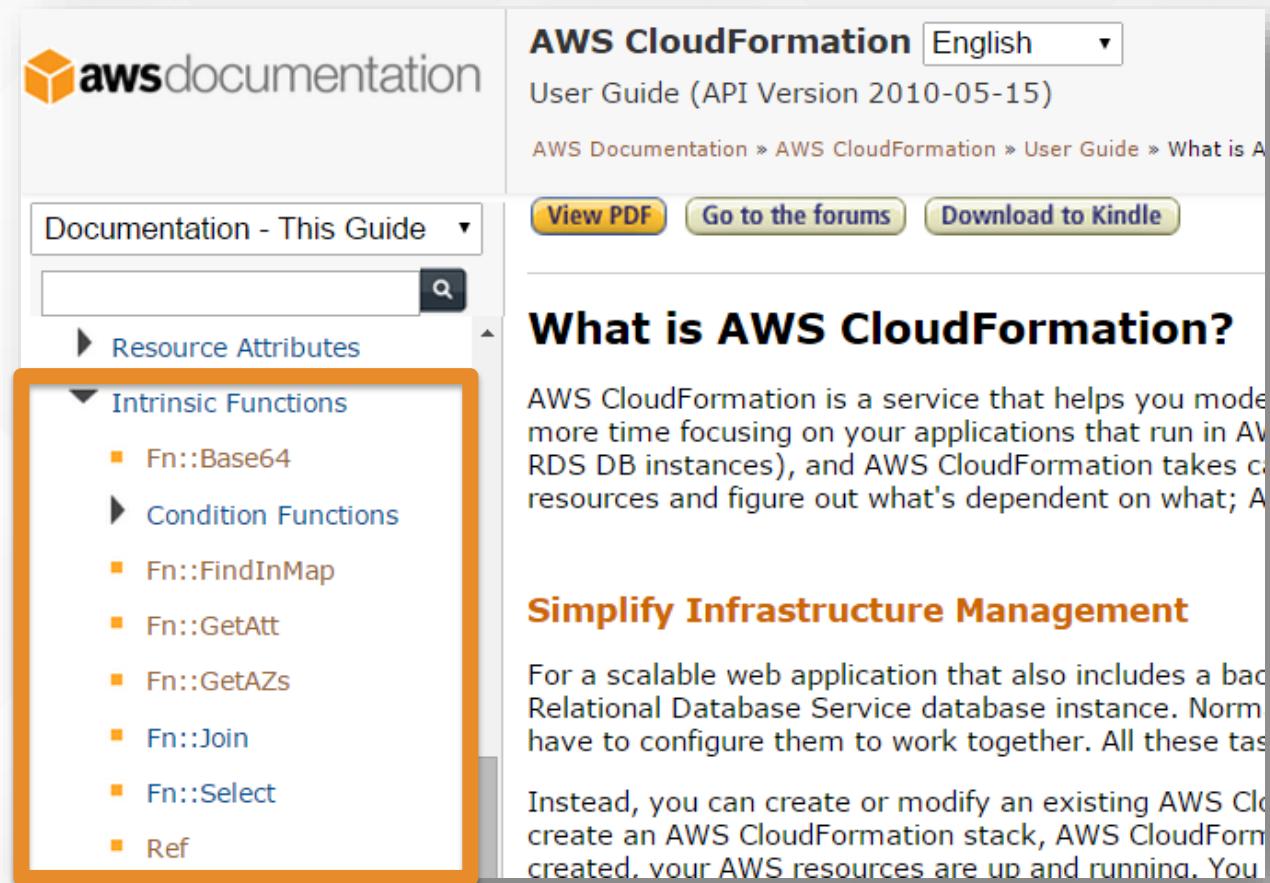
## CloudFormation Template



```
"AWS::CloudFormation::Init": {  
    "webapp-config": {  
        "packages" : {}, "sources" : {}, "files" : {},  
        "groups" : {}, "users" : {},  
        "commands" : {}, "services" : {}  
    },  
  
    "chef-config" : {}  
}
```

- ✓ Declarative
- ✓ Debug-able
- ✓ Updatable
- ✓ Highly Secure
- ✓ BIOT™ Bring In Other Tools

# Create template – Language features



The screenshot shows the AWS CloudFormation User Guide page. At the top, it says "AWS CloudFormation English ▾" and "User Guide (API Version 2010-05-15)". Below that is a breadcrumb trail: "AWS Documentation » AWS CloudFormation » User Guide » What is A". There are three buttons at the top right: "View PDF", "Go to the forums", and "Download to Kindle". On the left, there's a sidebar with "Documentation - This Guide ▾" and a search bar. The main content area has a heading "What is AWS CloudFormation?" followed by a detailed description. Below that is a section titled "Simplify Infrastructure Management" with a paragraph of text. The "Intrinsic Functions" section in the sidebar is highlighted with an orange border. It contains a list of intrinsic functions:

- ▶ Intrinsic Functions
  - Fn::Base64
  - ▶ Condition Functions
    - Fn::FindInMap
    - Fn::GetAtt
    - Fn::GetAZs
    - Fn::Join
    - Fn::Select
    - Ref

# Create stack

The screenshot shows the AWS CloudFormation 'Create stack' wizard interface. On the left, a sidebar lists services under 'Deployment & Management' (CloudFormation, IAM). The main area is titled 'Select Template'.

**Select Template**

Specify a stack name and then select the template that describes the stack that you want to create.

**Stack**

An AWS CloudFormation stack is a collection of related resources that you provision and update as a single unit.

**Name**

**Template**

A template is a JSON-formatted text file that describes your stack's resources and their properties. AWS CloudFormation stores the stack's template in an Amazon S3 bucket. [Learn more](#).

**Source**

Select a sample template

Upload a template to Amazon S3  No file chosen

Specify an Amazon S3 template URL

# Operate stack

Create Stack   Update Stack   Delete Stack  

Filter: Active ▾ By Name:  Loaded 1

	Name	Created	Status	Description
<input checked="" type="checkbox"/>	my-test-stack	2013-12-17 17:16:21 UTC-0800	CREATE_COMPLETE	AWS CloudFormation Sample Tem...

[Overview](#)   [Outputs](#)   [Resources](#)   [Events](#)   [Template](#)   [Parameters](#)   [Tags](#)   [Policy](#)

**Stack Name:** my-test-stack

**Stack ID:** arn:aws:cloudformation:XXXXXXXXXX:stack/my-test-stack/00fcbe20-6782-11e3-92c4-5088487ec896

**Status:** CREATE\_COMPLETE

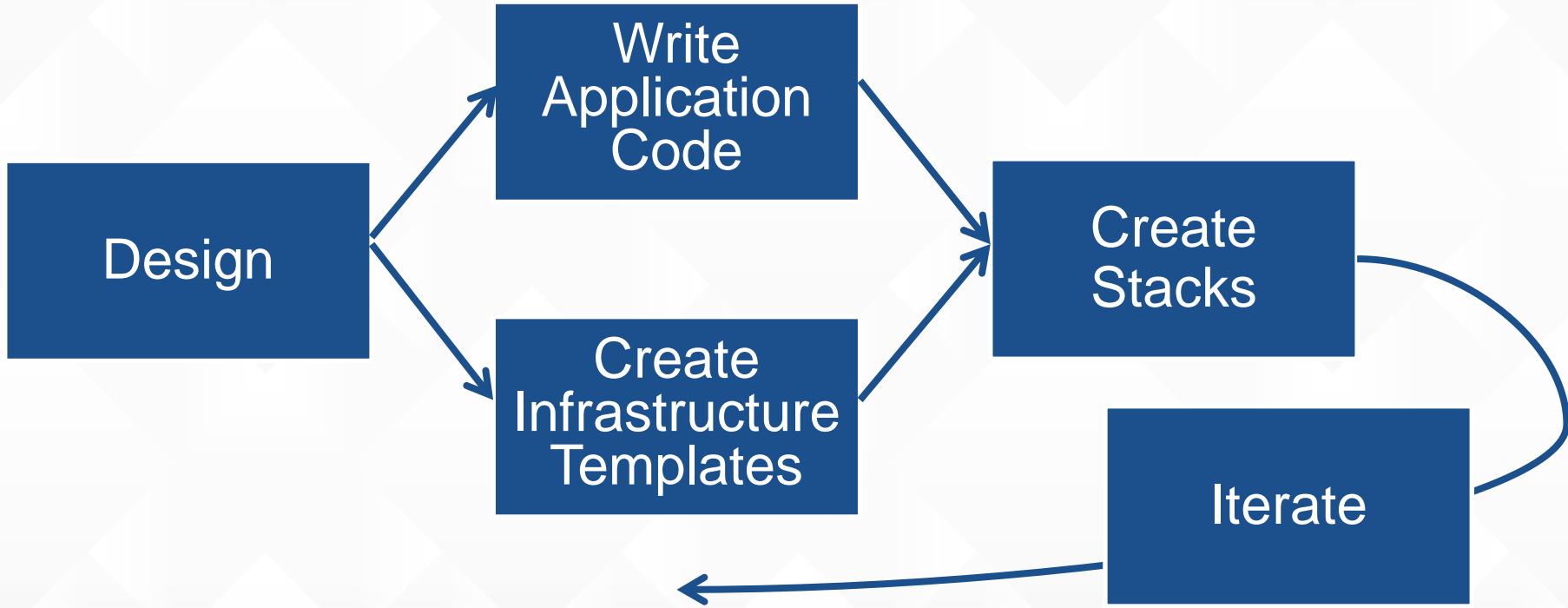
**Status (Reason):**

**Description:** AWS CloudFormation Sample Template LAMP\_Single\_Instance: Create a LAMP stack using a single EC2 instance and a local MySQL database for storage. This template demonstrates using the AWS CloudFormation bootstrap scripts to install the packages and files necessary to deploy the Apache web server, PHP and MySQL at instance launch time. \*\*WARNING\*\* This template creates an Amazon EC2 instance. You will be billed for the AWS resources used if you create a stack from this template.

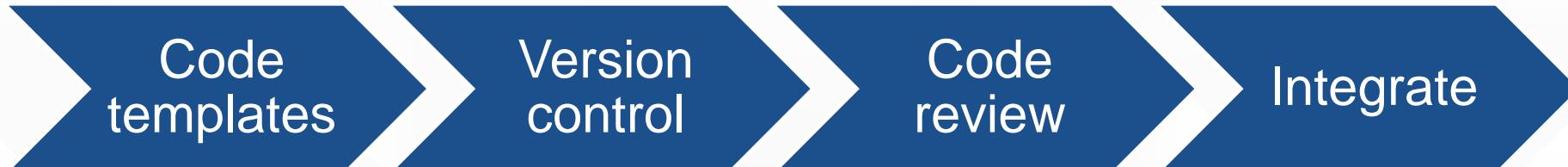
# Use a wide range of AWS services

- ✓ Auto Scaling
- ✓ Amazon CloudFront
- ✓ AWS CloudTrail
- ✓ Amazon CloudWatch
- ✓ Amazon DynamoDB
- ✓ Amazon EC2
- ✓ AWS Elastic Beanstalk
- ✓ Amazon ElastiCache
- ✓ Elastic Load Balancing
- ✓ Amazon Kinesis
- ✓ IAM
- ✓ AWS OpsWorks
- ✓ Amazon RDS
- ✓ Amazon Redshift
- ✓ Amazon Route 53
- ✓ Amazon S3
- ✓ Amazon SimpleDB
- ✓ Amazon SNS
- ✓ Amazon SQS
- ✓ Amazon VPC

# Basic workflow



# Infrastructure-as-code workflow



Code  
templates

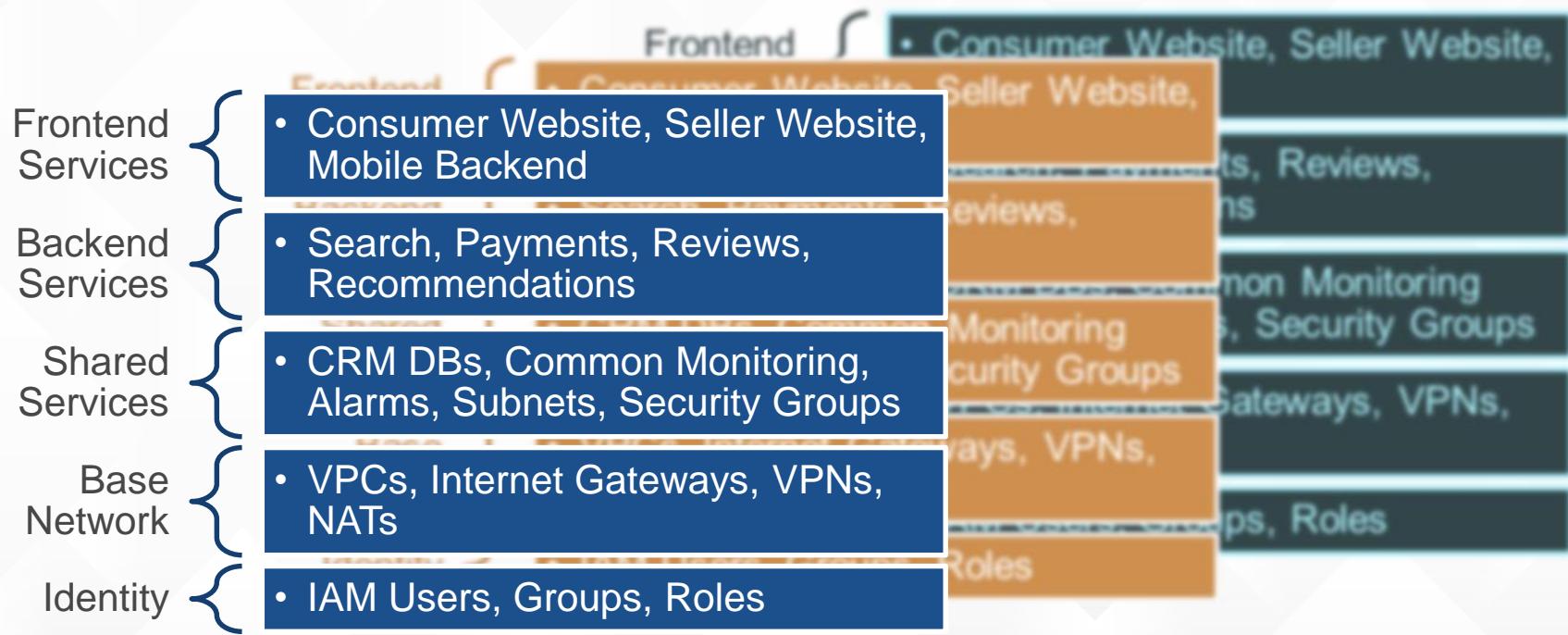
Version  
control

Code  
review

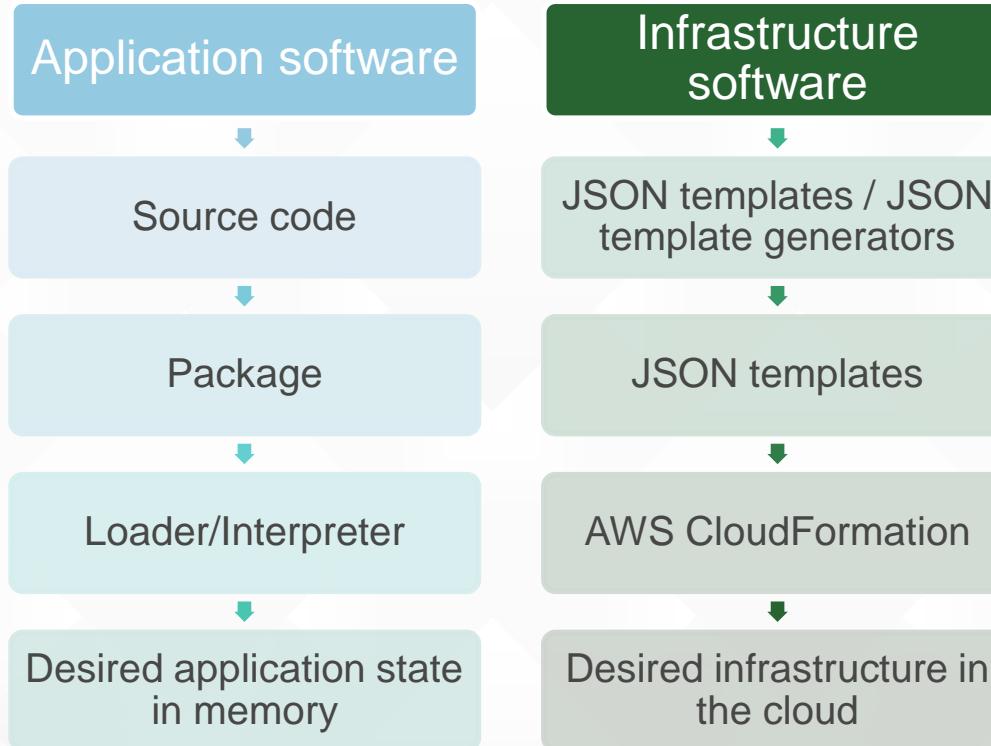
Integrate

**“It’s all software”**

# “It’s all software” – organize like it’s software



# “It’s all software” – build and operate like it’s software





---

# Iterate on infrastructure

# Update stack

## In-place



Faster

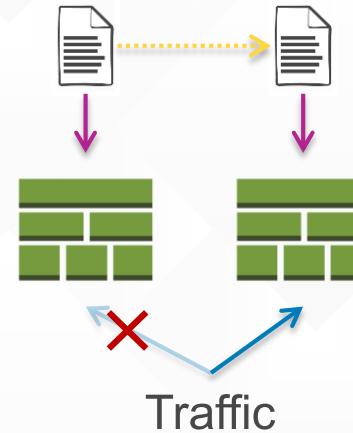
Cost-efficient

Simpler state and  
data migration

Templates

Stacks

## Blue-Green



Traffic

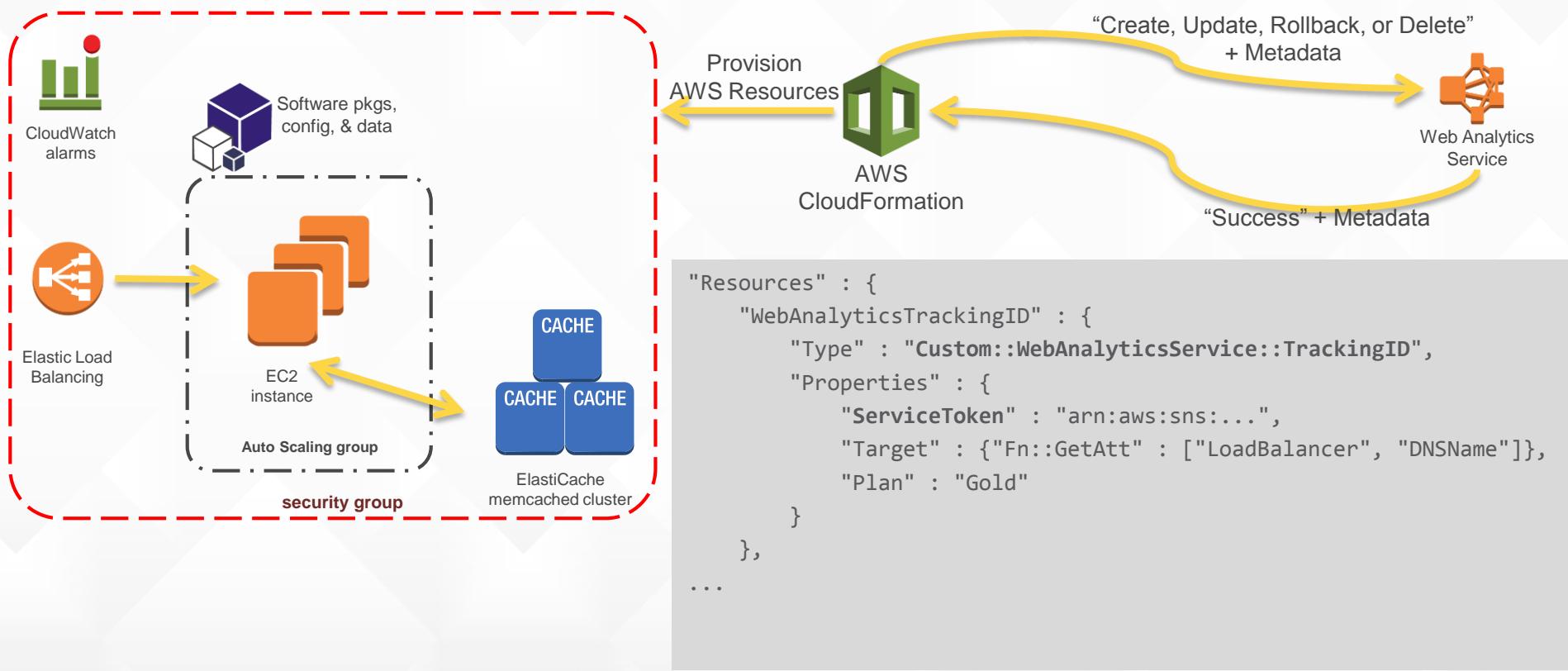
Working stack  
not touched



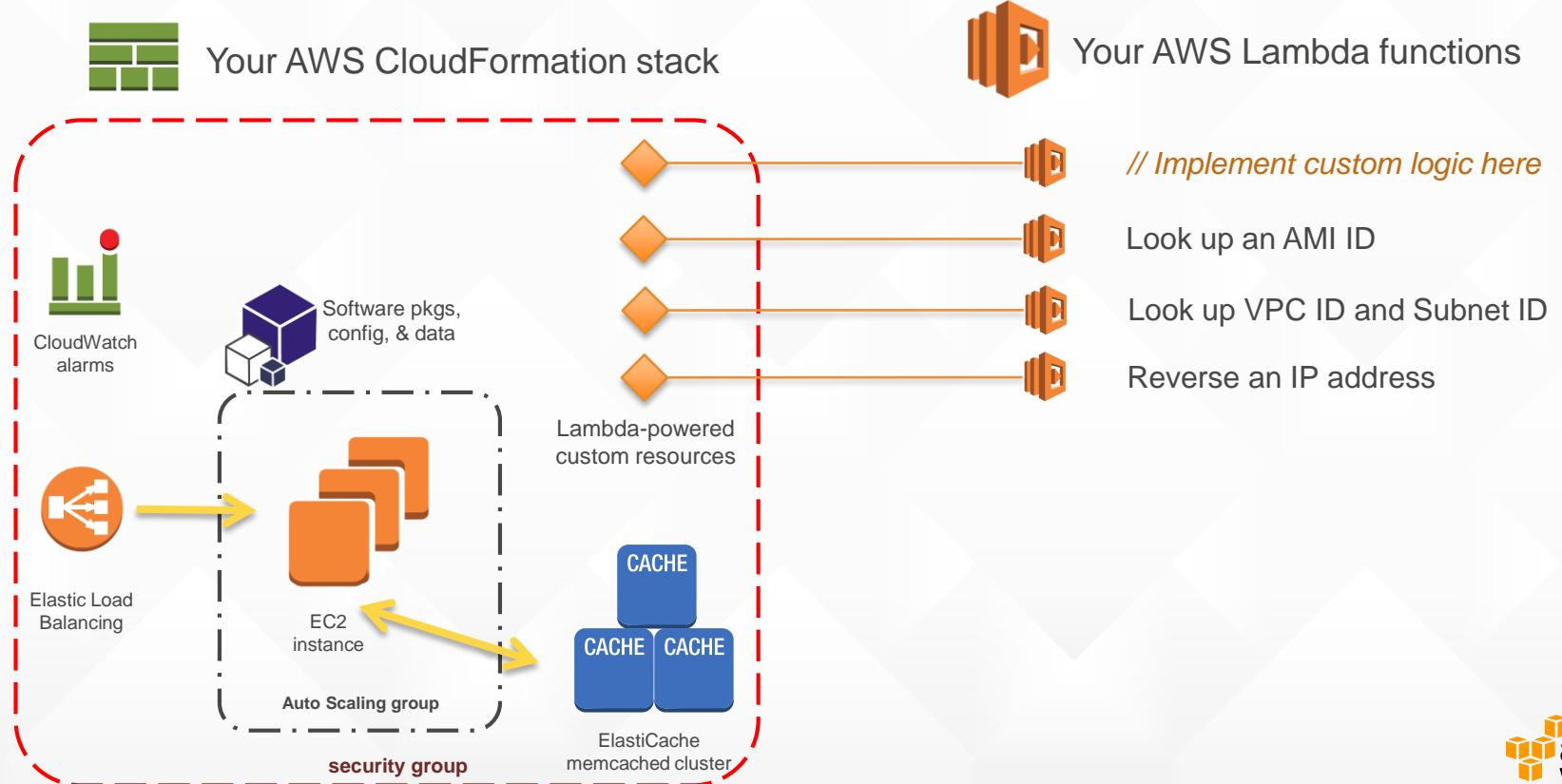
---

# Extending AWS CloudFormation

# Extend with Custom Resources



# Lambda-backed custom resources





---

# Application-deployment-as-code

# Infrastructure Provisioning

## CloudFormation

- Templatize
- Replicate
- Automate

EC2

SQS, SNS,  
Amazon Kinesis, etc.

Databases

VPC

IAM

# Application Deployment

Download Packages,  
Install Software,  
Configure Apps,  
Bootstrap Apps, Update  
Software, Restart Apps,  
etc.

# Application-deployment-as-code inside a CloudFormation template

Chef, Puppet,  
CodeDeploy, ...

Chef

CloudFormation::Init

OpsWorks

Amazon Machine Images

# AWS::CloudFormation::Init

```
"MyInstance": {  
    "Type": "AWS::EC2::Instance",  
    "Metadata" : {  
        "AWS::CloudFormation::Init" : {  
            "webapp-config": {  
                "packages" : {}, "sources" : {}, "files" : {},  
                "groups" : {}, "users" : {},  
                "commands" : {}, "services" : {}  
            }  
        }  
    }  
}
```

- ✓ Declarative
- ✓ Reusable
- ✓ Grouping & Ordering
- ✓ Debug-able
- ✓ Updatable
- ✓ Highly Secure
- ✓ BIOT™ (Bring In Other Tools)

# AWS::CloudFormation::Init

## Declarative

```
"AWS::CloudFormation::Init": {
    "webapp-config": {
        "packages" : {}, "sources" : {}, "files" : {},
        "groups" : {}, "users" : {},
        "commands" : {}, "services" : {}
    }
}
"sources" : {
    "/etc/myapp" :
    "https://s3.amazonaws.com/mybucket/myapp.tar.gz"
}
```

# AWS::CloudFormation::Init

## Debug-able

The screenshot shows the AWS CloudWatch Logs interface. The left sidebar navigation bar includes links for Dashboard, Alarms, ALARM, INSUFFICIENT, OK, Billing, Logs (which is selected), Metrics, Selected Metrics, DynamoDB, EBS, EC2, RDS, SNS, and Custom Metrics... A message bar at the top indicates 0 unreviewed items.

The main content area displays a log group hierarchy: Log Groups > Streams for my-web-app-stack-6-CloudFor... > Events for i-e1deaccd/cfn-init.log. A search bar at the top right contains the identifier i-e1deaccd/cfn-init.log.

A timestamp selector at the top allows jumping to specific dates and times. The current view shows events from July 19, 2014, at 00:05 UTC-7. The table lists the Creation Time and Event Data for each log entry. One event is highlighted in blue, indicating it is currently selected or being viewed.

Creation Time	Event Data
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:04:19,885 [INFO] Running configSets: default
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:04:19,886 [INFO] Running configSet default
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:04:19,963 [INFO] Running config cfn-logs-config
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:02,297 [INFO] Command install-logs-agent succeeded
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:02,342 [INFO] Running config cfn-hup-config
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:02,437 [INFO] enabled service cfn-hup
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:03,638 [INFO] Restarted cfn-hup successfully
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:03,704 [INFO] Running config application-config
2014-07-19 00:05:14 UTC-7	▶ 2014-07-19 07:05:14,092 [INFO] Yum installed [u'httpd', u'php']
2014-07-19 00:05:14 UTC-7	▶ 2014-07-19 07:05:14,243 [INFO] enabled service httpd
2014-07-19 00:05:15 UTC-7	▶ 2014-07-19 07:05:14,603 [INFO] Started httpd successfully
2014-07-19 00:05:15 UTC-7	▶ 2014-07-19 07:05:14,846 [INFO] disabled service sendmail
2014-07-19 00:05:15 UTC-7	▶ 2014-07-19 07:05:14,974 [INFO] ConfigSets completed

# AWS::CloudFormation::Init

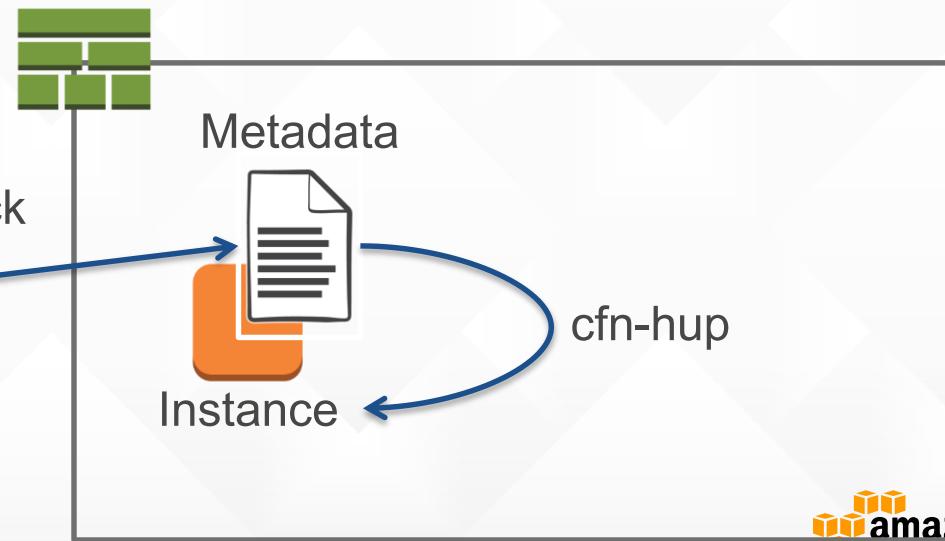
## Supports updates

1. Update instance metadata  
in the template



2. UpdateStack

3. CloudFormation daemon  
updates configuration



# AWS::CloudFormation::Init

Flexibility to bring in other tools such as AWS CodeDeploy and Chef

```
"install_chef" : {},  
  
"install_wordpress" : {  
    "commands" : {  
        "01_get_cookbook" : {}, ...,  
        "05_configure_node_run_list" : {  
            "command" : "knife node run_list add -z `knife  
node list -z` recipe[wordpress]",  
            "cwd" : "/var/chef/chef-repo",  
            "env" : { "HOME" : "/var/chef" }  
    }  
}
```

[ow.ly/DiNkz](http://ow.ly/DiNkz)



# AWS::CloudFormation::Init

**Supports role-based auth**

```
"YourInstance": {  
    "Metadata": {  
        "AWS::CloudFormation::Authentication": {  
            "S3AccessCreds": {  
                "type": "S3",  
                "roleName": { "Ref" : "InstanceRole"},  
                "buckets" : ["your-bucket"]  
            }  
        },  
        "AWS::CloudFormation::Init": {}  
    },  
    "ow.ly/DqkrB"
```

Choose auth type.  
IAM Role is  
recommended

Securely download



# Use AWS::CloudFormation::Init

```
"UserData": {  
  
    "# Get the latest CloudFormation helper scripts package\n",  
    "yum update -y aws-cfn-bootstrap\n",  
  
    "# Trigger CloudFormation::Init configuration\n",  
    "/opt/aws/bin/cfn-init --stack ", {"Ref": "AWS::StackId"},  
        " --resource WebServerInstance ",  
        " --region ", {"Ref": "AWS::Region"}, "\n",  
  
    "# Signal completion\n",  
    "/opt/aws/bin/cfn-signal -e $? --stack ", {"Ref": "AWS::StackId"},  
        " --resource WebServerInstance ",  
        " --region ", {"Ref": "AWS::Region"}, "\n"  
}
```



# Use CloudWatch Logs for debugging

```
"install_logs": {  
    "packages" : { ... "awslogs" ... },  
    "services" : { ... "awslogs" ... }  
    "files": {  
        "/tmp/cwlogs/cfn-logs.conf": {}  
    }
```

```
file = /var/log/cfn-init.log  
log_stream_name = {instance_id}/cfn-init.log  
  
file = /var/log/cfn-hup.log  
log_stream_name = {instance_id}/cfn-hup.log
```

# Use CloudWatch Logs for debugging

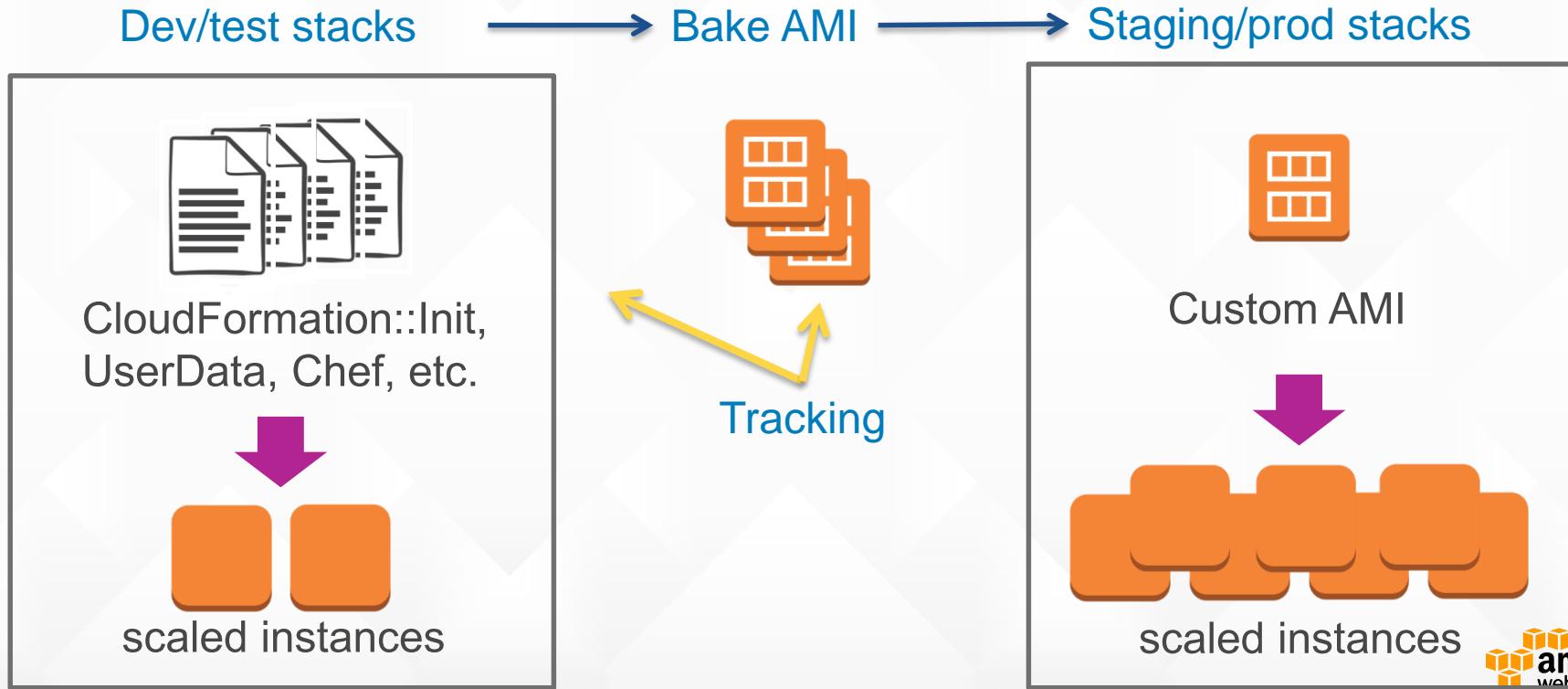
The screenshot shows the AWS CloudWatch Logs interface. The left sidebar is titled "Logs" and includes links for Dashboard, Alarms, ALARM, INSUFFICIENT, OK, Billing, Metrics, Selected Metrics, DynamoDB, EBS, EC2, RDS, SNS, and Custom Metrics... A yellow box highlights the "Logs" link. The main content area shows a breadcrumb navigation path: Log Groups > Streams for my-web-app-stack-6-CloudFor... > Events for i-e1deacd/cfn-init.log. Above the table is a "Jump To:" input field set to 2014/07/19 00:05:05 Local (GMT-07:00). The table has two columns: "Creation Time" and "Event Data". The "Event Data" column contains log entries starting with "2014-07-19 00:05:05 UTC-7" and ending with "2014-07-19 00:05:15 UTC-7".

Creation Time	Event Data
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:04:19,885 [INFO] Running configSets: default
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:04:19,886 [INFO] Running configSet default
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:04:19,963 [INFO] Running config cfn-logs-config
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:02,297 [INFO] Command install-logs-agent succeeded
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:02,342 [INFO] Running config cfn-hup-config
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:02,437 [INFO] enabled service cfn-hup
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:03,638 [INFO] Restarted cfn-hup successfully
2014-07-19 00:05:05 UTC-7	▶ 2014-07-19 07:05:03,704 [INFO] Running config application-config
2014-07-19 00:05:14 UTC-7	▶ 2014-07-19 07:05:14,092 [INFO] Yum installed [u'httpd', u'php']
2014-07-19 00:05:14 UTC-7	▶ 2014-07-19 07:05:14,243 [INFO] enabled service httpd
2014-07-19 00:05:15 UTC-7	▶ 2014-07-19 07:05:14,603 [INFO] Started httpd successfully
2014-07-19 00:05:15 UTC-7	▶ 2014-07-19 07:05:14,846 [INFO] disabled service sendmail
2014-07-19 00:05:15 UTC-7	▶ 2014-07-19 07:05:14,974 [INFO] ConfigSets completed

[ow.ly/E0zO3](http://ow.ly/E0zO3)

# Bake AMIs for faster booting

# Bake AMIs for maintaining golden images



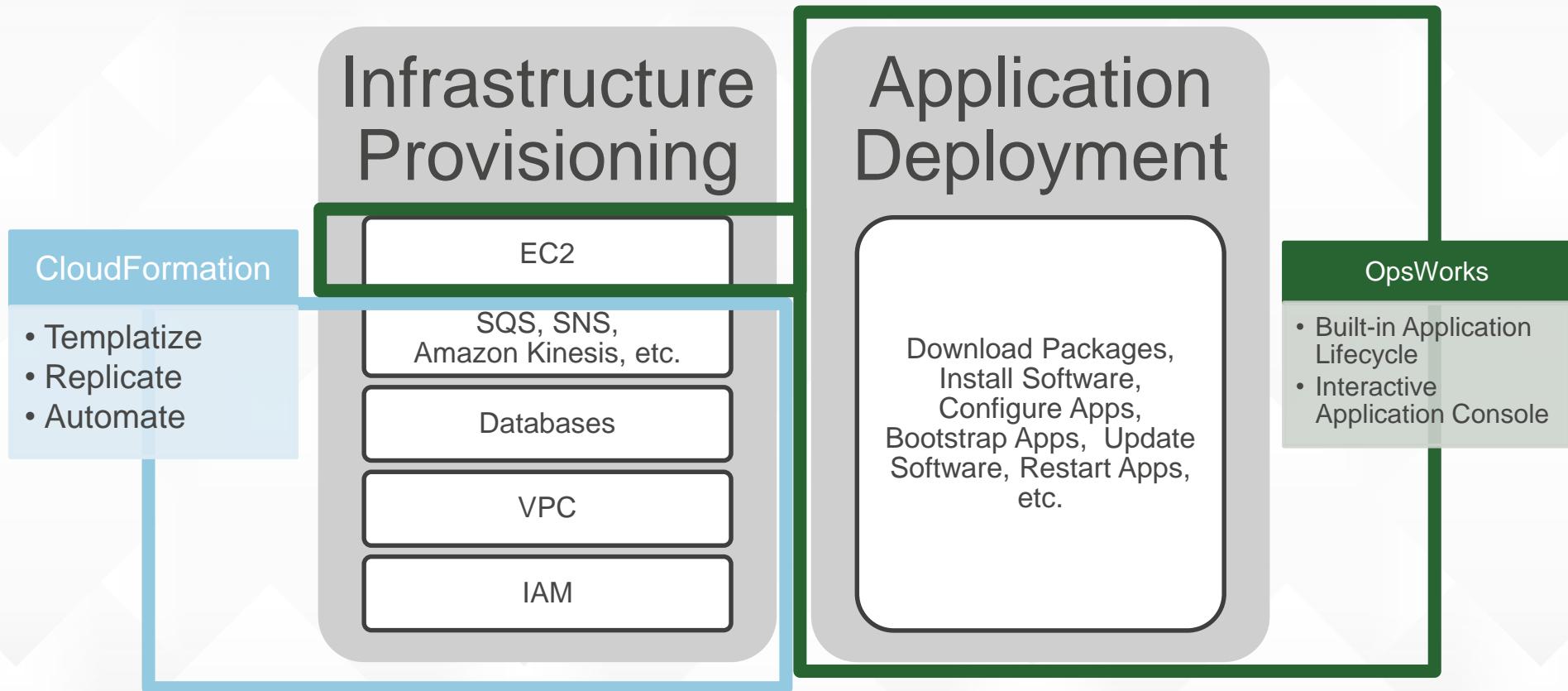


---

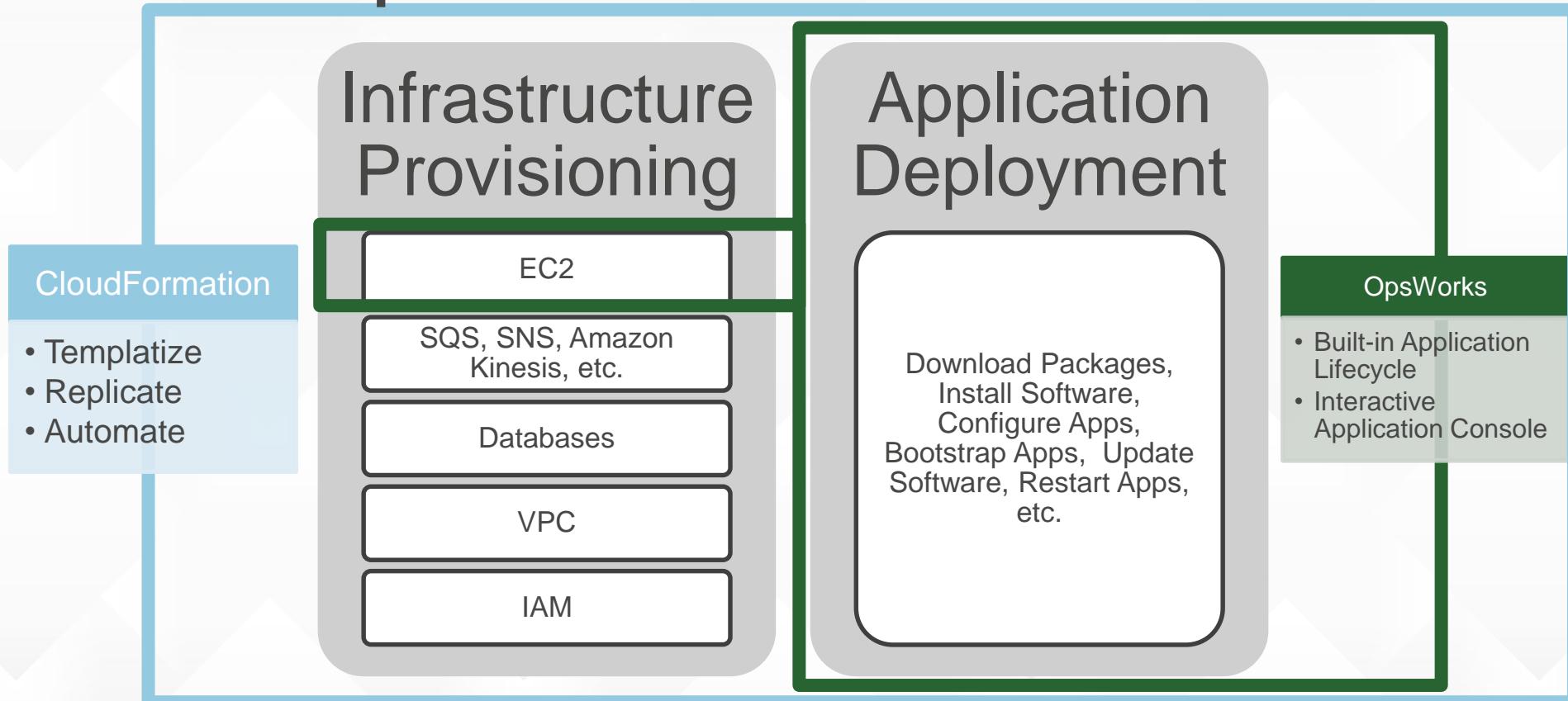
# Using CloudFormation and OpsWorks together



# OpsWorks & CloudFormation “side-by-side”



# OpsWorks “inside” CloudFormation

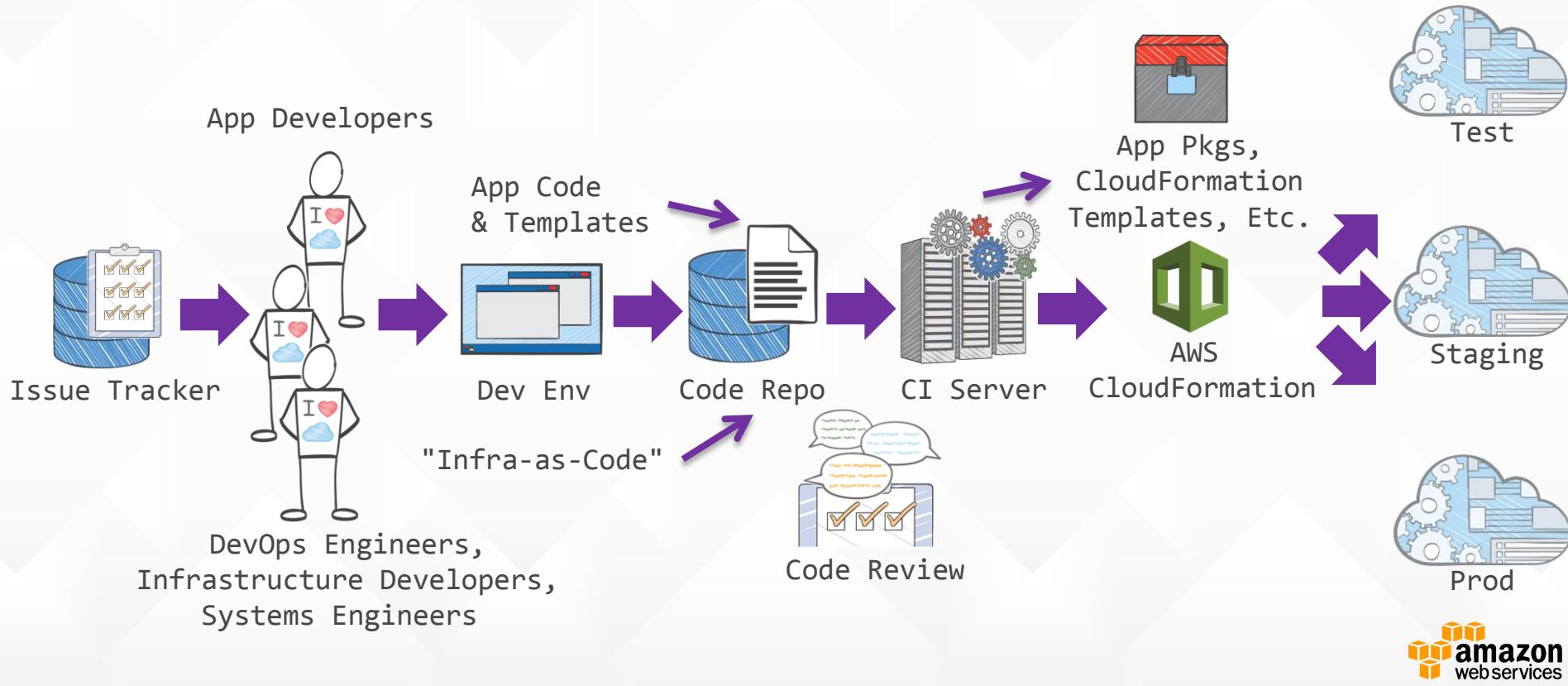




---

# Infrastructure-as-code in a CI/CD pipeline

# CloudFormation in a CI/CD pipeline

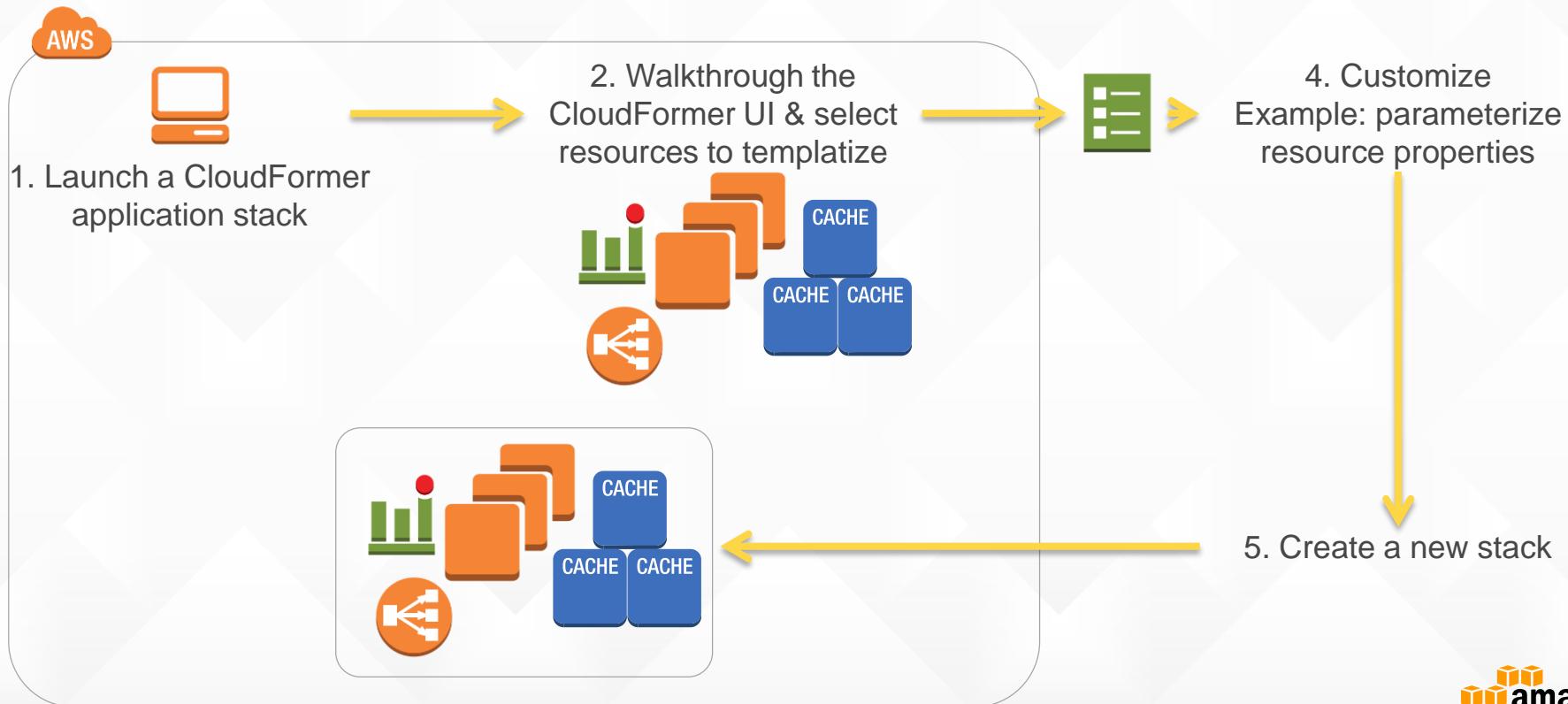




---

# Templatize existing resources

# CloudFormer: Templatize existing resources



# Practitioners of infrastructure-as-code

- **Developers/DevOps** teams value CloudFormation for its ability to treat infrastructure as code, allowing them to apply software engineering principles, such as SOA, revision control, code reviews, integration testing to infrastructure.
- **IT Admins and MSPs** value CloudFormation as a platform to enable standardization, managed consumption, and role-specialization.
- **ISVs** value CloudFormation for its ability to support scaling out of multi-tenant SaaS products by quickly replicating or updating stacks. ISVs also value CloudFormation as a way to package and deploy their software in their customer accounts on AWS.



---

Thank You  
SAN FRANCISCO





---

# AWS Summit

SAN FRANCISCO

---

