

AWS
re:Invent

D O P 2 1 0

Leadership session: Developer tools on AWS

Ken Exner

General Manager, AWS Developer Tools
Amazon Web Services

By way of introduction...

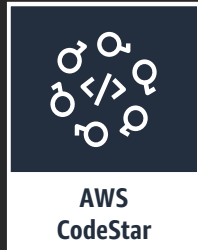
AWS
CodeBuild

AWS
CodeCommit

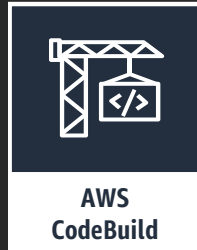
AWS
CodeDeploy

AWS
CodePipeline

CI/CD Tools



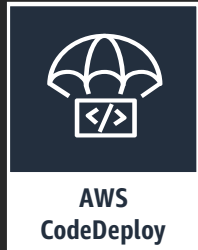
AWS
CodeStar



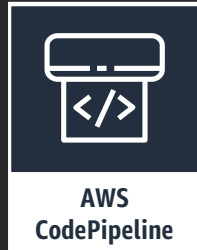
AWS
CodeBuild



AWS
CodeCommit

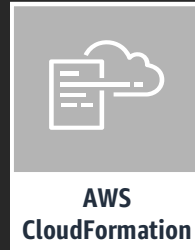


AWS
CodeDeploy

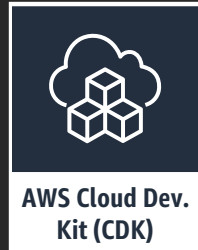


AWS
CodePipeline

Infrastructure as Code

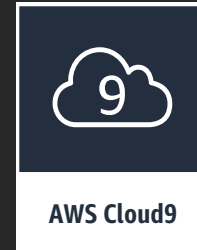


AWS
CloudFormation



AWS Cloud Dev.
Kit (CDK)

IDE

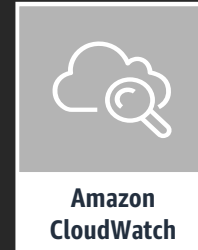


AWS Cloud9

Monitoring & Tracing

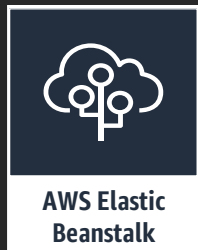


AWS X-Ray



Amazon
CloudWatch

Web Apps



AWS Elastic
Beanstalk

IDE and DevOps Toolkits



Visual Studio
Code



IntelliJ



PyCharm



Visual Studio



Eclipse



VSTS

CLI and Scripting Tools

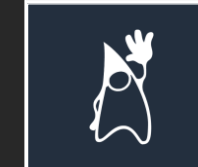


AWS CLI



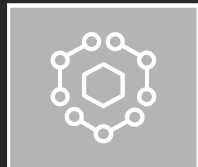
Tools for
PowerShell

Languages



Amazon
Corretto

Mobile



AWS Amplify

SDKs



JavaScript



Python



PHP



.NET



Ruby



Java



Go



Node.js



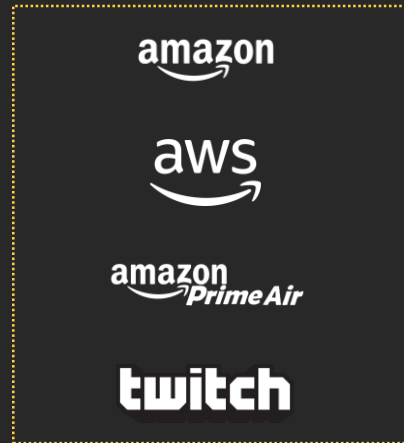
C++

By way of introduction...

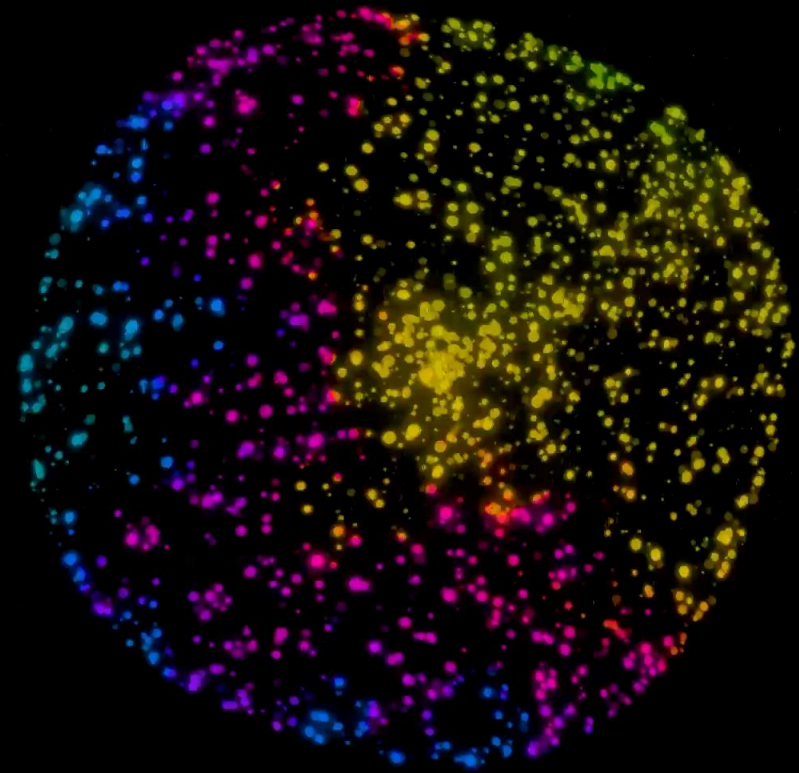


Builder.Tools

Internal and external customers across industries



How does Amazon...



build resilient, multi-region, multi-availability zone systems?

The Amazon Builders' Library

How Amazon builds and operates software



Architecture, software delivery, and operations

By Amazon's senior technical executives and engineers

Real-world practices with detailed explanations

Content available for free on the website

The Amazon Builders' Library

ARCHITECTURE

LEVEL 300



Leader election in distributed systems

Author: Marc Brooker


Improving efficiency, reducing coordination, and simplifying architectures by using leader election.

^



SOFTWARE DELIVERY AND OPERATIONS

LEVEL 300




Going faster with continuous delivery

Author: Mark Mansour


Automating the software testing and deployment process for speed and reliability

^



SOFTWARE DELIVERY AND OPERATIONS

LEVEL 400




Implementing health checks

Author: David Yanacek

Automatically detecting and mitigating server failures without unintended consequences from fleet-wide false positives.

^



ARCHITECTURE

LEVEL 400



Workload isolation using shuffle-sharding

Author: Colm MacCarthaigh

Shuffle Sharding is one of our core techniques for drastically limiting the scope of impact of operational issues

^



Many more at aws.amazon.com/builders-library

Agenda

- Dev/test and code review
- Continuous Integration and Continuous Delivery (CI/CD)
- Modern applications
- A look ahead

Dev/test and code review

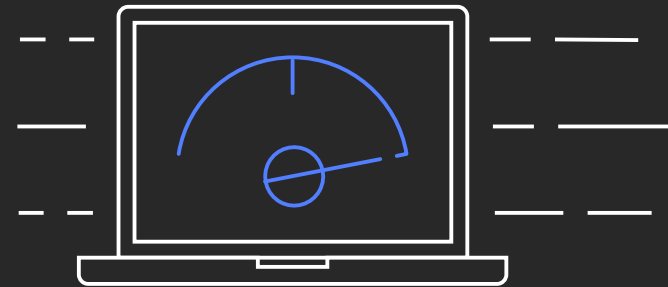
Software delivery at Amazon



Dev/Test



Code review

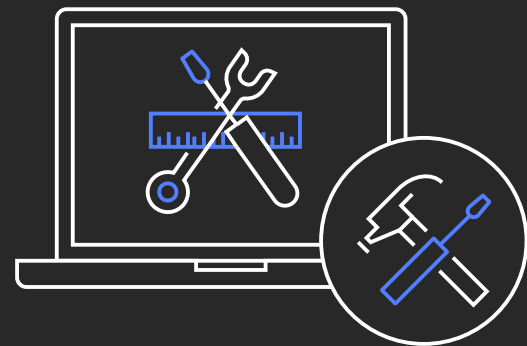


Pre-production
testing



Production

Software delivery at Amazon



Dev/Test

Code review

Pre-production
testing

Production

Software delivery at Amazon

Dev/Test



Code review

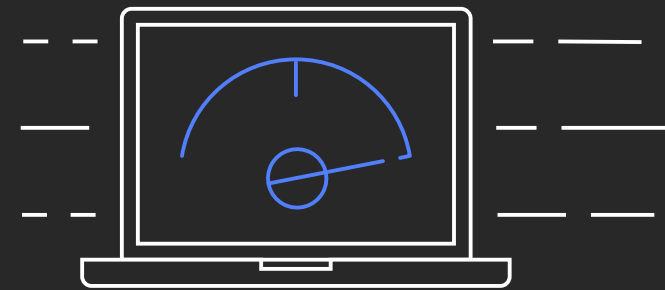
Pre-production testing

Production

Software delivery at Amazon

Dev/Test

Code review



Production

Pre-production
testing

Software delivery at Amazon

Dev/Test

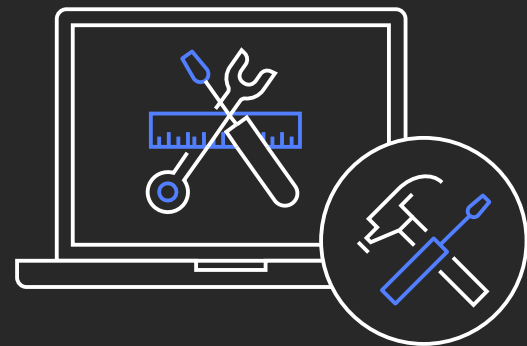
Code review

Pre-production
testing



Production

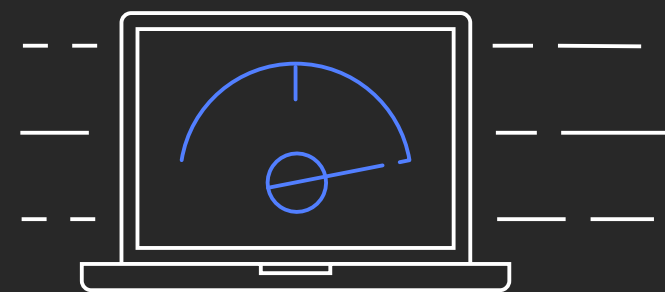
Software delivery at Amazon



Dev/Test



Code review

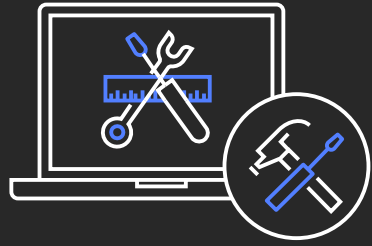


Pre-production
testing



Production

We use cloud desktops at Amazon

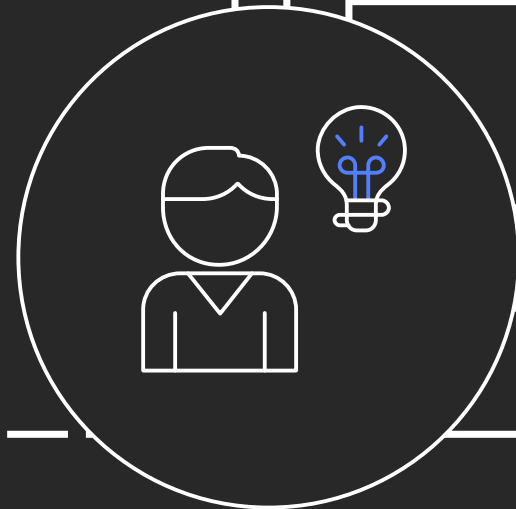


DEV/TEST

CODE REVIEW

PRE-PRODUCTION
TESTING

PRODUCTION



We also use AWS Cloud9 internally



AWS Cloud9

A cloud IDE for writing, running, and debugging code

- Code with just a browser
- Start new projects quickly
- Code together in real time
- Build serverless applications with ease
- Direct terminal access

Support for local development



AWS Toolkit
for PyCharm
Python



AWS Toolkit
for IntelliJ
Java, Python



AWS Toolkit for
Visual Studio Code
.NET, Node



AWS Toolkit for
Visual Studio
.NET

Support for local development



AWS Toolkit
for PyCharm
Python



AWS Toolkit
for IntelliJ
Java, Python



AWS Toolkit for
Visual Studio Code
.NET, Node



AWS Toolkit for
Visual Studio
.NET

New



AWS Toolkit
for Webstorm
Node.js



AWS Toolkit
for Rider
.NET

The AWS IDE Toolkits now support Cloud Debugging (beta)

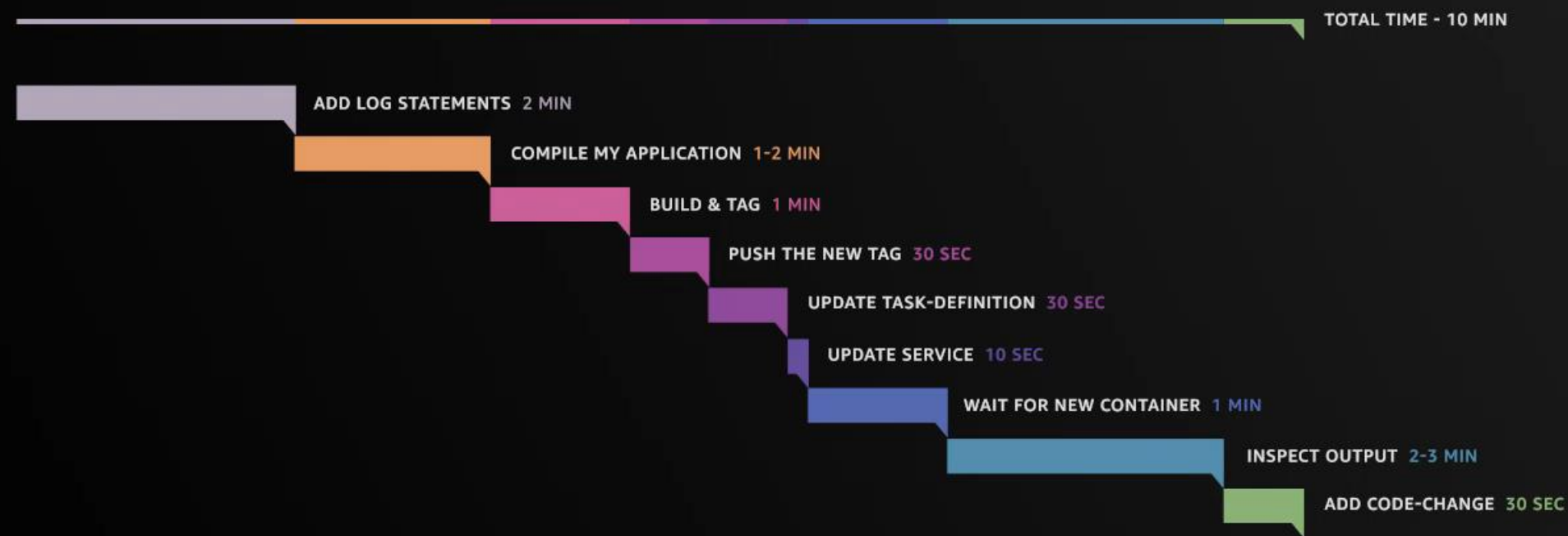


Bringing your cloud environment to your code editor



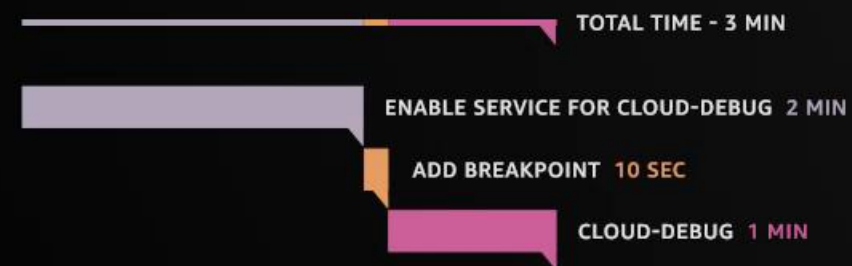
Traditional Debugging Timeline

Traditional Debugging Timeline

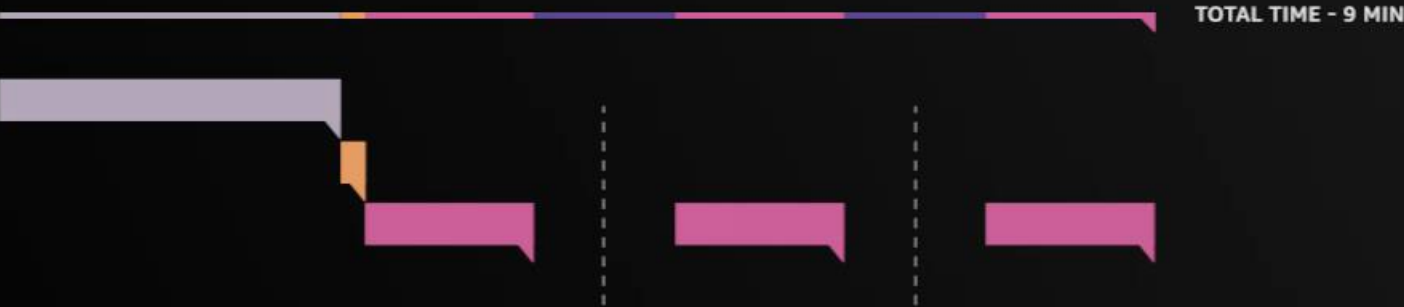


Cloud Debugging

Cloud Debugging



Cloud Debugging



Kyle Thomson

Senior Software Engineer,
AWS Developer Tools



The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project structure with packages like `com.example` and classes like `MythicalMysfitsService`.
- Code Editor:** Displays the `MythicalMysfitsService.java` file with the following code:


```

package com.example;

import ...

@Service
public class MythicalMysfitsService {

    private static final org.slf4j.Logger log = org.slf4j.LoggerFactory.getLogger(MythicalMysfitsService.class);

    private final AmazonDynamoDB client = AmazonDynamoDBClientBuilder.defaultClient();
    private DynamoDBMapper mapper = new DynamoDBMapper(client);

    public Mysfits getAllMysfits() {
        List<Mysfit> mysfits = mapper.scan(Mysfit.class, new DynamoDBScanExpression());

        Mysfits allMysfits = new Mysfits(mysfits);

        return allMysfits;
    }

    public Mysfits queryMysfitItems(String filter, String value) {
        HashMap<String, AttributeValue> attribValue = new HashMap<String, AttributeValue>();
        attribValue.put(":" + value, new AttributeValue().withS(value));

        DynamoDBQueryExpression<Mysfit> queryExpression = new DynamoDBQueryExpression<Mysfit>()
            .withIndexName(filter + "Index")
            .withKeyConditionExpression(filter + " = " + value)
            .withExpressionAttributeValues(attribValue)
            .withConsistentRead(false);

        List<Mysfit> mysfits = mapper.query(Mysfit.class, queryExpression);

        Mysfits allMysfits = new Mysfits(mysfits);

        return allMysfits;
    }
}

```
- Maven Explorer:** Shows lifecycle tasks like clean, validate, compile, test, package, verify, install, site, and deploy.
- Browser Window:** Displays a JSON response:


```

{
  "Species": "Cyclone",
  "Age": 2,
  "Good/Evil": "Neutral",
  "Lawful/Chaotic": "Neutral"
}

```



The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project structure for 'mythical-mysfits' with packages like 'com.example', 'Mysfit', 'Mysfits', 'MysfitsConfig', 'MythicalMysfitsApp', 'MythicalMysfitsContr', and 'MythicalMysfitsServic'.
- Code Editor:** Displays the `MythicalMysfitsController.java` file with the following code:


```

return allMysfits;
}

public Mysfits queryMysfitItems(String filter, String value) {
    HashMap<String, AttributeValue> attribValue = new HashMap<String, AttributeValue>();
    attribValue.put(":" + value, new AttributeValue().withS(value));

    DynamoDBQueryExpression<Mysfit> queryExpression = new DynamoDBQueryExpression<Mysfit>()
        .withIndexName(filter + "Index")
        .withKeyConditionExpression(filter + " = " + value)
        .withExpressionAttributeValues(attribValue)
        .withConsistentRead(false);

    List<Mysfit> mysfits = mapper.query(Mysfit.class, queryExpression);

    Mysfits allMysfits = new Mysfits(mysfits);

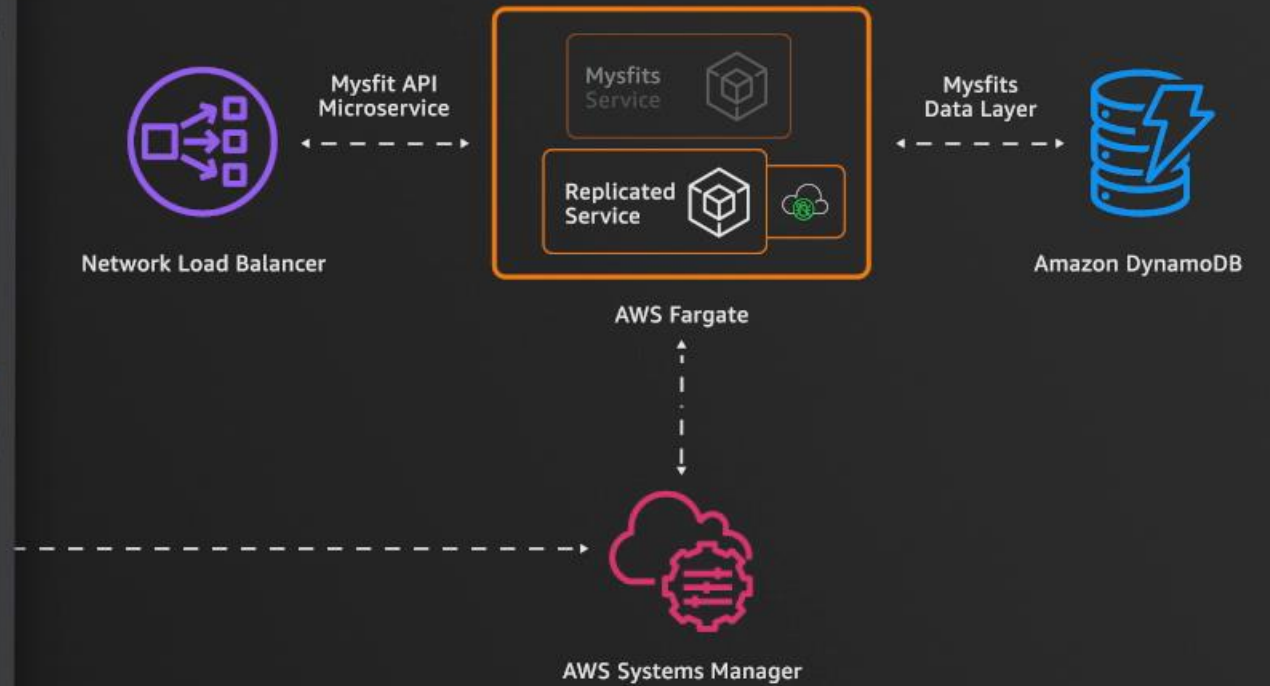
    return allMysfits;
}

```
- Terminal:** Shows the output of the AWS Cloud Debugging configuration process:


```

memory: "1024",
NetworkMode: "awsipc",
PlacementConstraints: [],
RequiresCompatibilities: ["FARGATE"],
TaskRoleArn: "arn:aws:iam:784233260762:role/MythicalMysfitsCoreStack-ECSTaskRole-MEFNT050ESH",
Volumes: [
  {
    DockerVolumeConfiguration: {
      Scope: "task"
    },
    Name: "cloud-debug-volume"
  }
]
}
Storing instrumentation info
Cloning service with name 'MythicalMysfits-Service'
Waiting for service cloud-debug-MythicalMysfits-Service to become stable
Service cloud-debug-MythicalMysfits-Service is stable
Connecting to the instrumented service cloud-debug-MythicalMysfits-Service from cluster MythicalMysfits-Cluster
Attempting to create an SSH port session with the task ID 7db7d074-234b-46d3-8c2e-865da69ac456
Dispatcher: creating session for target '7db7d074-234b-46d3-8c2e-865da69ac456'.
Dispatcher: http response status code was '200'
Dispatcher: session operation for 7db7d074-234b-46d3-8c2e-865da69ac456 succeeded: New session created for target 7db7d074-
Successfully instrumented ECS service name='MythicalMysfits-Service', cluster='MythicalMysfits-Cluster' with iamRole='arn:
target':'7db7d074-234b-46d3-8c2e-865da69ac456'

```



The screenshot shows an IDE with the following components:

- Code Editor:** Displays the `MythicalMysfitsService.java` file with the following code:


```

package com.example;

import ...

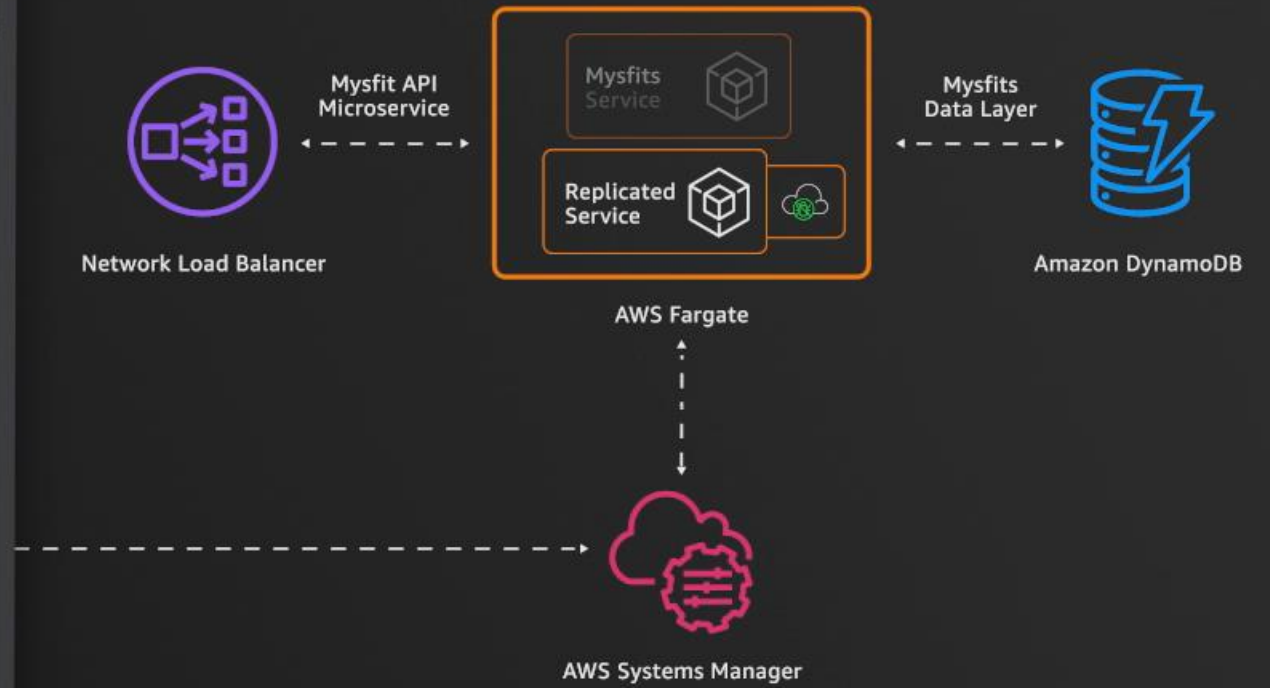
@Service
public class MythicalMysfitsService {

    private static final org.slf4j.Logger log = org.slf4j.LoggerFactory.getLogger(MythicalMysfitsService.class);

    private final AmazonDynamoDB client = AmazonDynamoDBClientBuilder.defaultClient();
    private DynamoDBMapper mapper = new DynamoDBMapper(client);

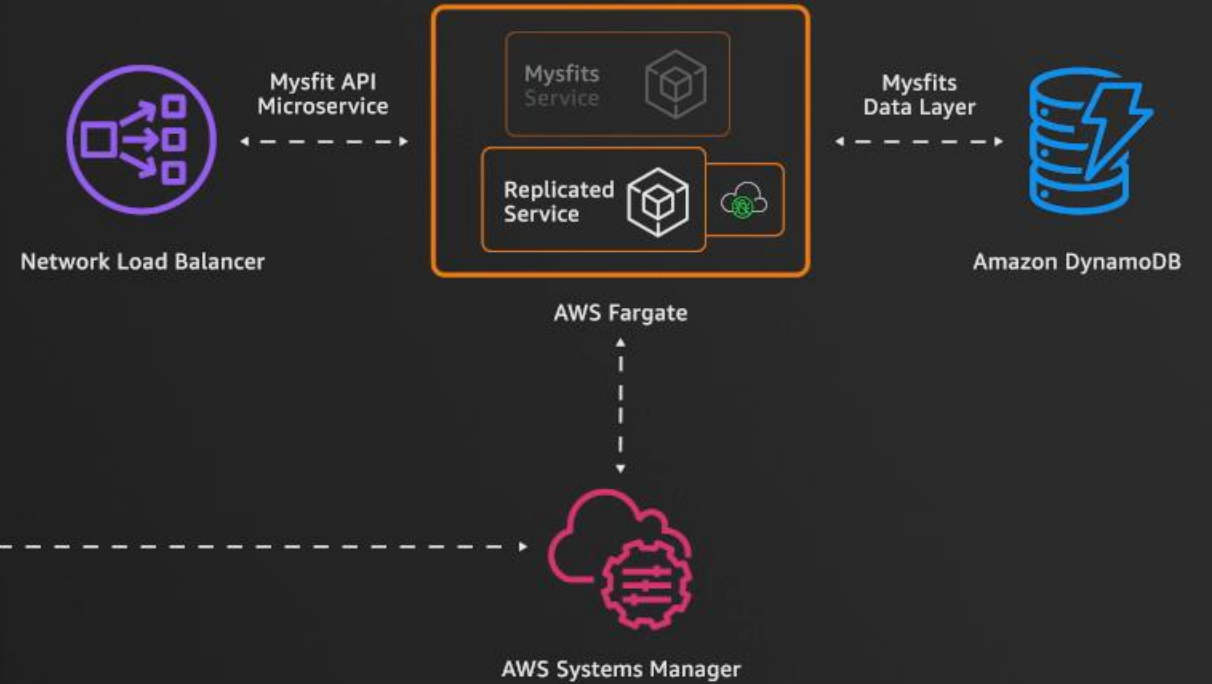
    public Mysfits getAllMysfits() {

```
- Configuration Dialog:** Shows the configuration for the `cloud-debug-MythicalMysfits-Service` container.
 - Name: `cloud-debug-MythicalMysfits-Service (beta)`
 - Cluster: `MythicalMysfits-Cluster`
 - Service: `cloud-debug-MythicalMysfits-Service`
 - Platform: `JVM`
 - Remote Debug Port: `20020`
 - Start Command: `java -Djava.security.egd=file:/dev/./urandom -jar .`
 - Port Mappings: Local Port `8881` to Remote Port `8080`.
 - Before launch: `Run Maven Goal 'mythical-mysfits: install'`
- IDE Interface:** Includes a project explorer on the left showing the package structure, a terminal at the bottom, and a status bar indicating the AWS profile and region.



The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project named 'Mythical Mysfits' with a package structure including 'com.example', 'MyMyfits', 'MyMyfitsConfig', 'MythicalMyMyfitsApplic', 'MythicalMyMyfitsContr', and 'MythicalMyMyfitsServic'.
- Code Editor:** Displays the source code for `MythicalMyMyfitsController.java`. The code includes a `ListMyMyfits` method that uses `DynamoDBScanExpression` and a `queryMyMyfitsItems` method that uses `DynamoDBQueryExpression` to filter items based on a filter and value.
- Debugger Console:** Shows logs from the application, including startup messages for Spring Framework and the application itself, and a log entry for the `getMyMyfits` method call with filter and value parameters.



Code review at Amazon: Focus on automation

DEV/TEST



CODE REVIEW

PRE-PRODUCTION TESTING

PRODUCTION

GoodCop	Status	Message
Dry Run Build	Fail	https://build.amazon.com/2007146280
InfoSecScan	Pass	Complete: 0 issues identified

53 + `$sceProvider.enabled(false);`

DiscoveryFrameworkCRUX | 2018-09-22 00:42:09 UTC Rate this comment Resolve comment

Found: AngularJS Contextual Escaping Disabled

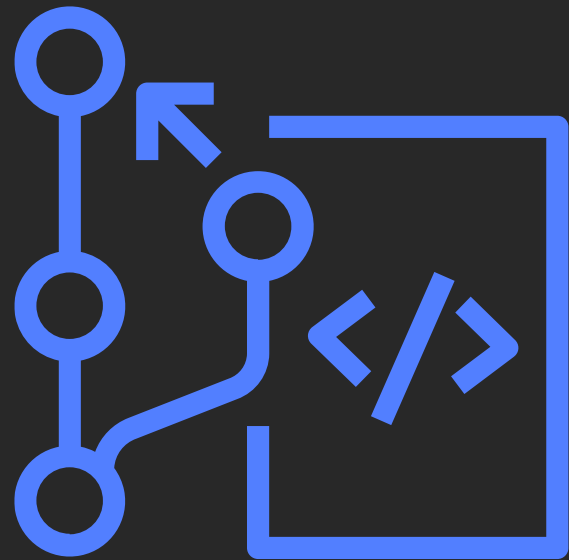
If Strict Contextual Escaping is disabled, no output values are automatically sanitized by AngularJS.

Strict contextual escaping is enabled by default from AngularJS version 1.2 and above, any previous version should be upgraded to AngularJS version 1.2. If upgrading is not an option there are steps to mitigate this vulnerability defined in the SKB article. For more information, see the [Security Knowledge Base](#)

To provide feedback on this comment, see this [survey](#).

Reply

AWS CodeCommit supports Approval Rules



AWS CodeCommit now supports **Approval Rules** that must be met before a pull request can be merged

Example Robotic Approver: SonarQube

Developer Tools ×

CodeCommit

- Source • CodeCommit
 - Getting started
 - Repositories
 - Code
 - Pull requests**
 - Commits
 - Branches
 - Git tags
 - Settings
- Build • CodeBuild
- Deploy • CodeDeploy
- Pipeline • CodePipeline

Go to resource

Feedback

Developer Tools > CodeCommit > Repositories > addition > Pull requests > 6601

6601: Remove extra debugging statements and add CodeCommit commenting on failure

Close pull request Merge

Open Approved No merge conflicts Destination master ◀ Source new-pr Author: Approvals: 1

Details | Activity | Changes | Commits | **Approvals**

Approvals

Override approval

Approver	Status
AWSCodeBuild-	✔ Approved

Approval rules

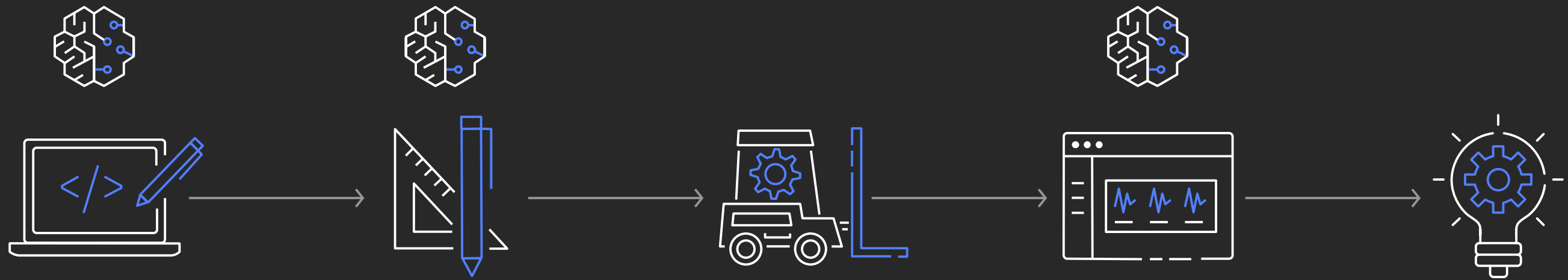
Delete Edit Create approval rule

Approval rule	Status
No results There are no results to display.	

Applied approval rule templates

Approval rule	Origin template	Status
require SonarQube analysis	require SonarQube analysis	✔ Rule satisfied

Amazon CodeGuru: Using ML to build and run high-performing applications



Built-in **code reviews** with intelligent recommendations

Detect and **optimize** the expensive lines of code pre-prod

Easily identify **application inefficiencies** in production environment

Amazon CodeGuru Reviewer

How it works

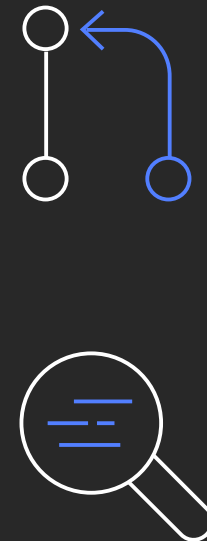
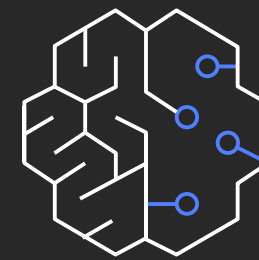
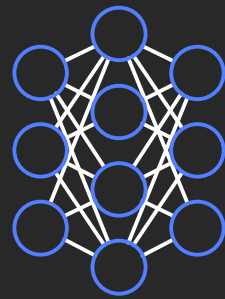
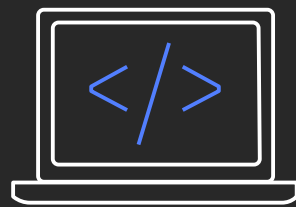


Customer provides source code as input

Extract semantic features/patterns

ML algorithms identify similar code for comparison

Customers see recommendations as pull request feedback



Input: source code

Feature extraction

Machine learning

Output: Recommendations

Amazon CodeGuru Profiler

How it works

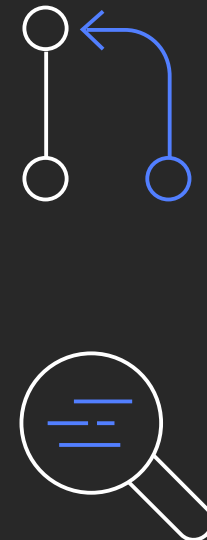
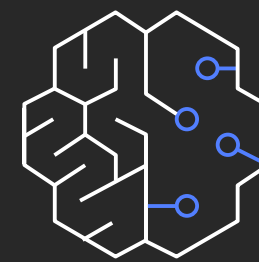
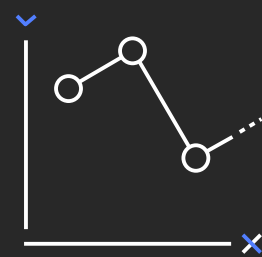


Customer's application runs in production

Profiler continuously captures application runtime information

Profiler detects inefficiencies in the live application

Customers see recommendations in their automated efficiency reports and visualizations



Input: Live application stack trace

Application profile sampling

Pattern matching

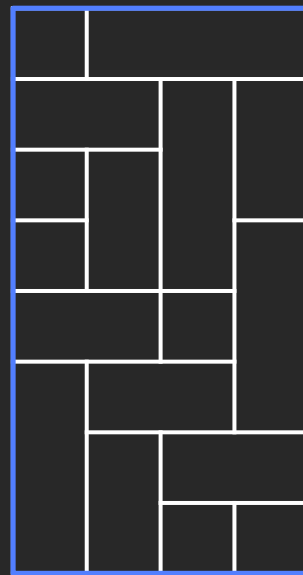
Output: Method names, recommendations, and searchable visualizations

Continuous integration and continuous delivery

Development transformation at Amazon: 2001–2002

Lesson learned: **Decompose for agility**

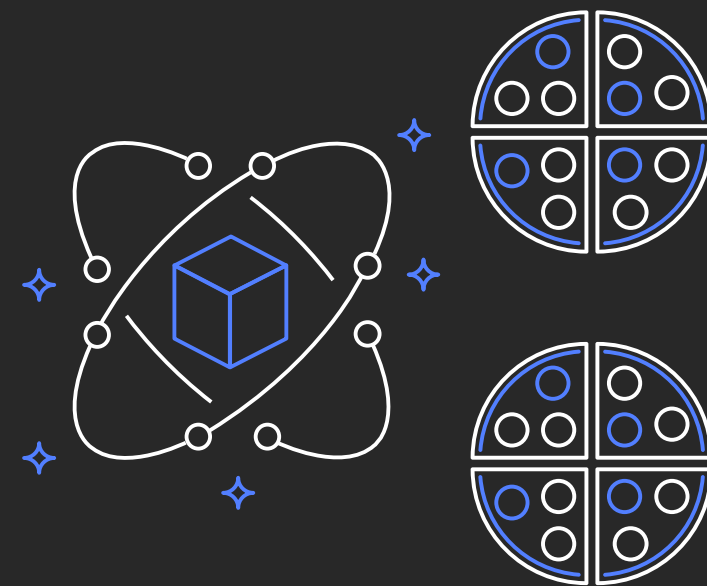
2001



Monolithic
application + teams

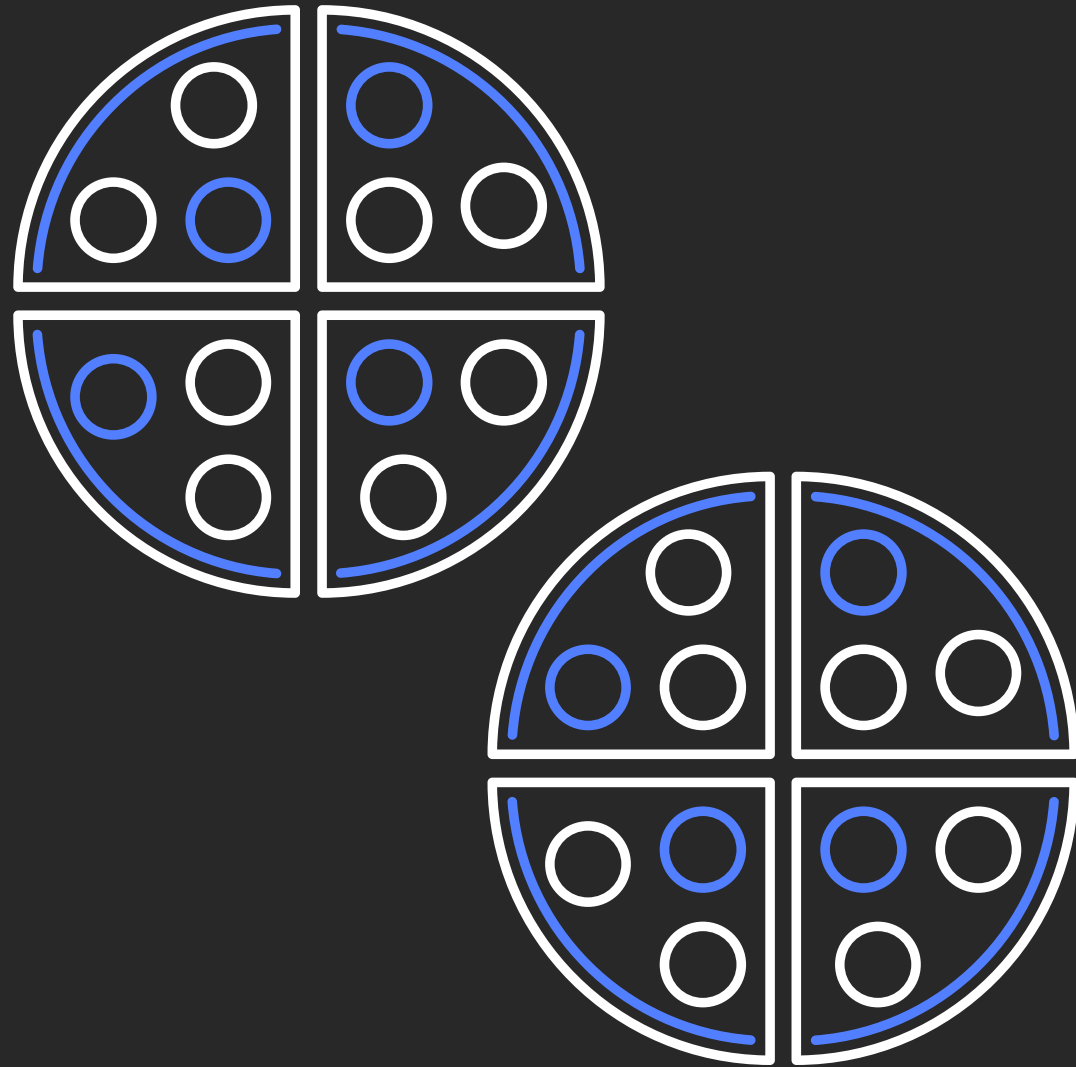


2002



Microservices
+ 2-pizza teams

Two-pizza teams



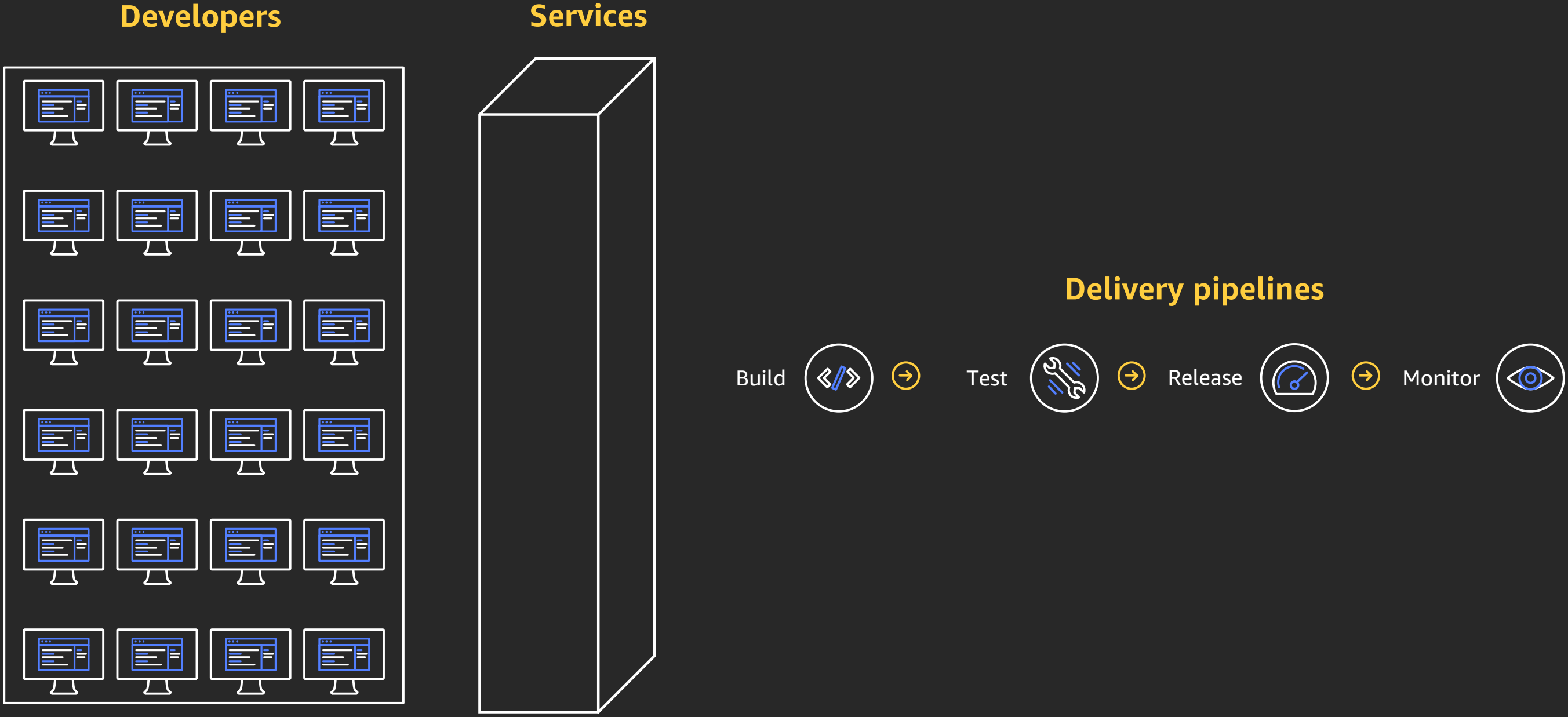
Full ownership

Full accountability

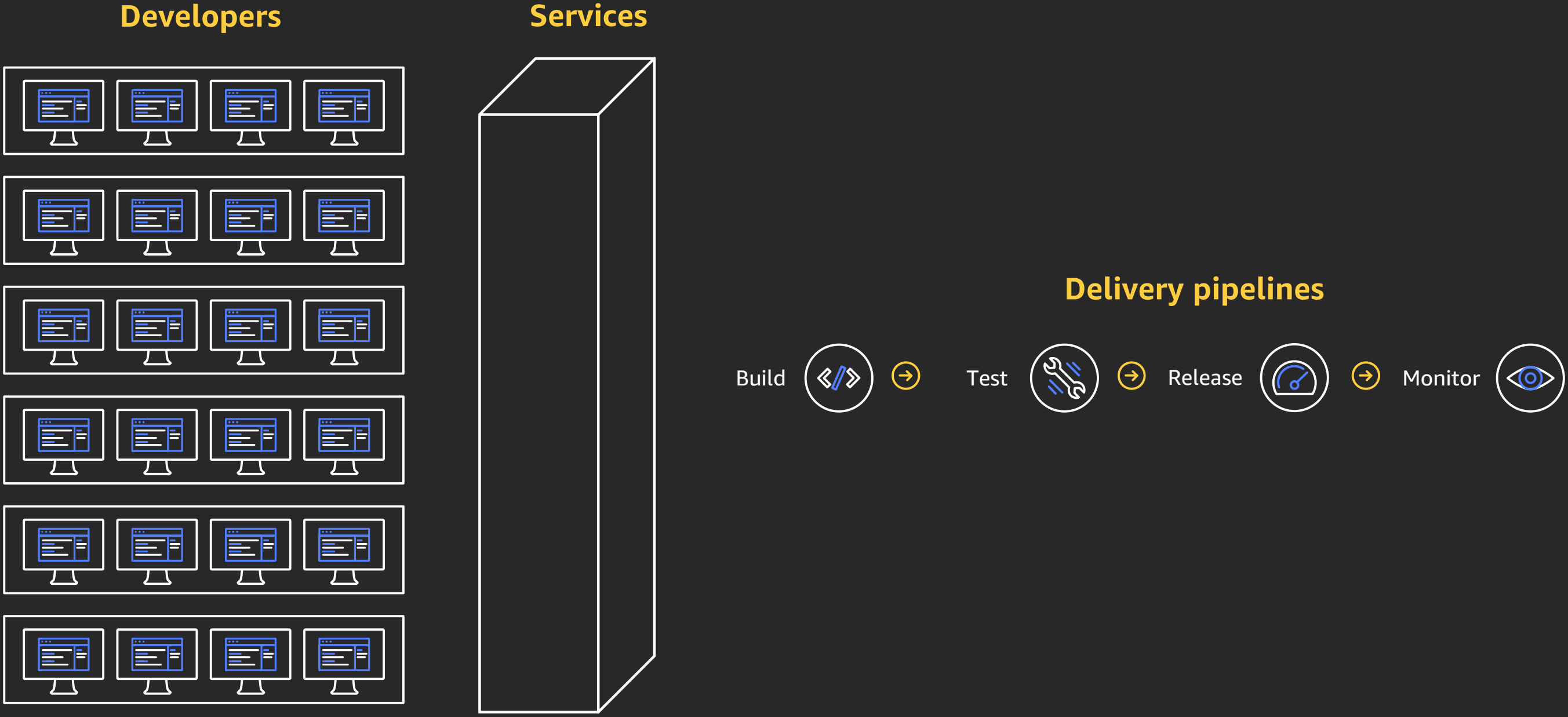
“DevOps”

Focused innovation

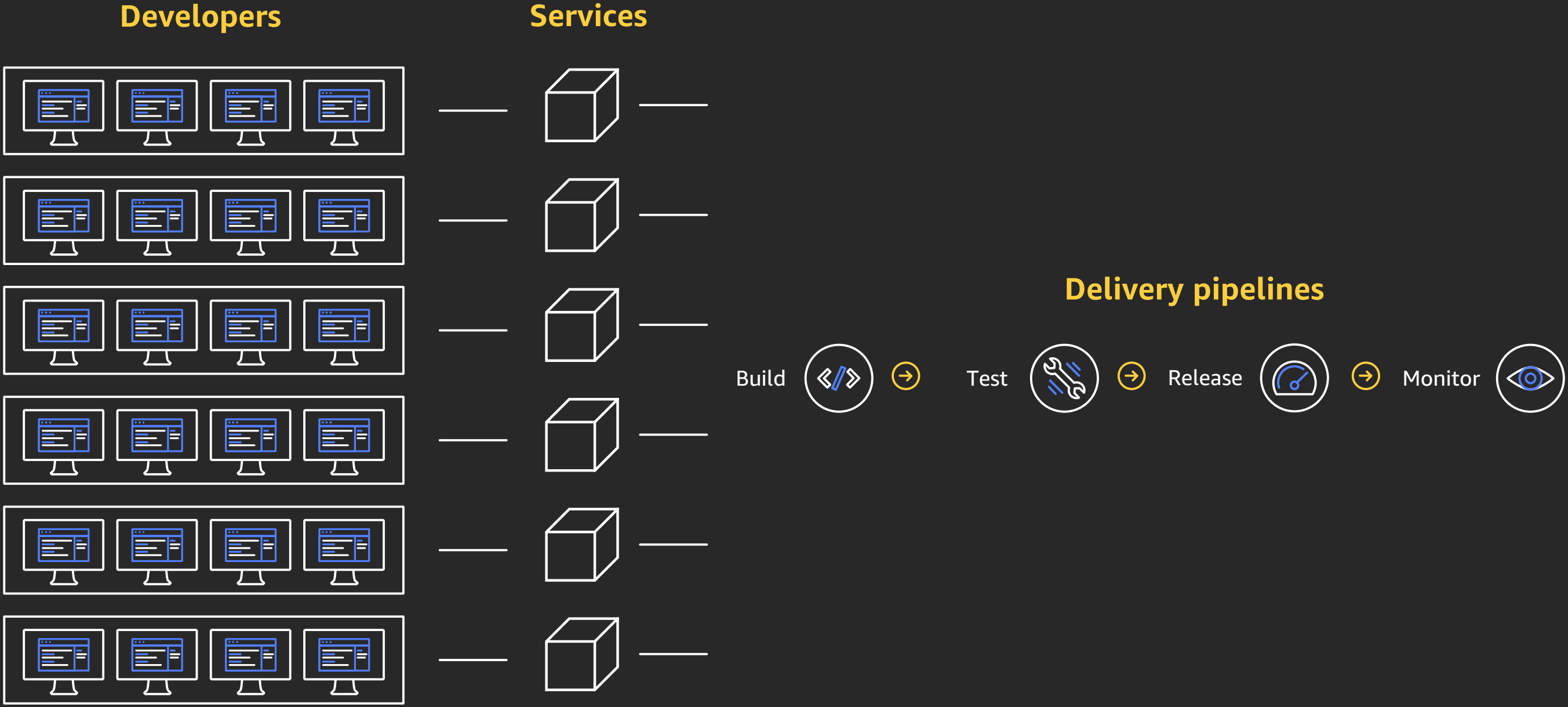
Monolith development lifecycle



Monolith development lifecycle



Microservice development lifecycle



Microservice development lifecycle

Developers



Services



Continuous Delivery at Amazon

Dev/Test

Code review



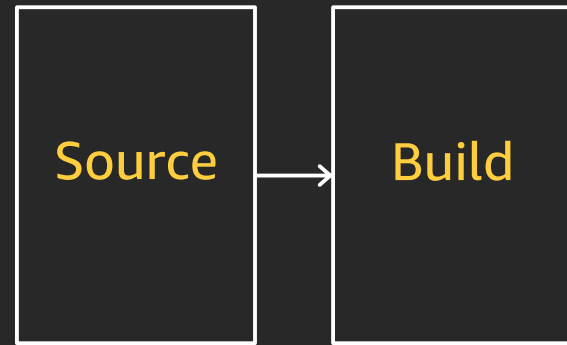
Pre-production
testing

Production

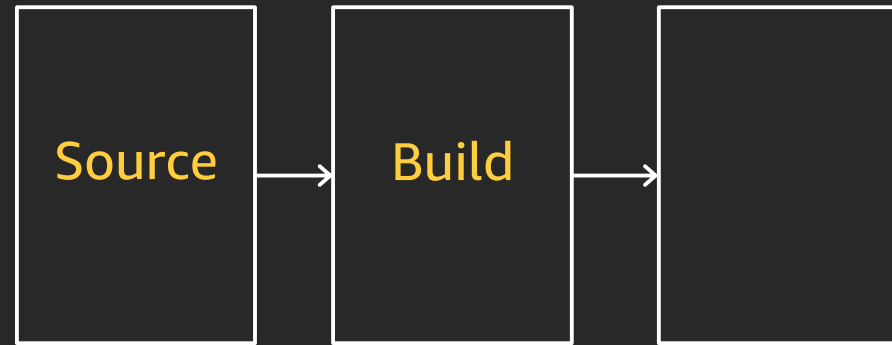
Continuous Delivery at Amazon: Deep Dive

Source

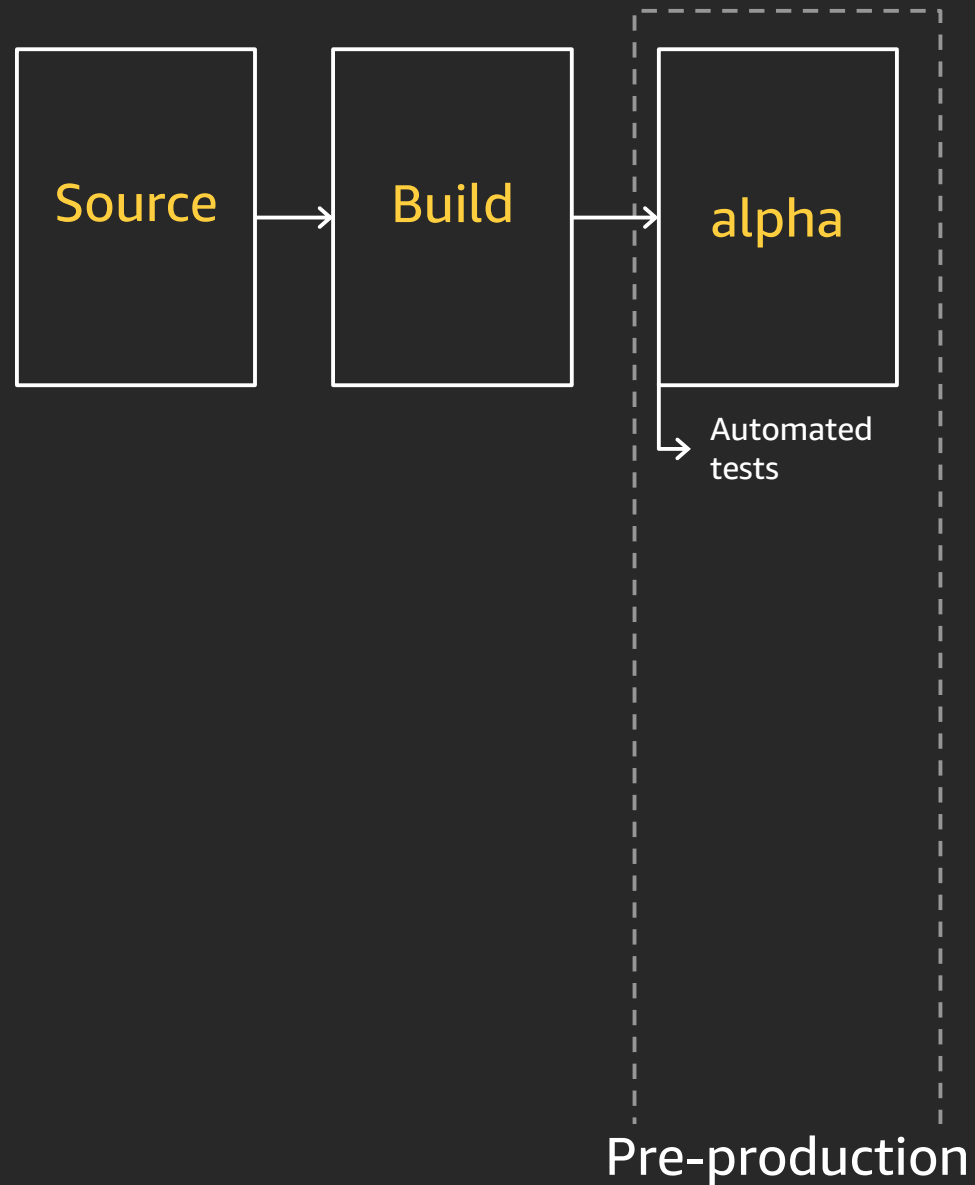
Continuous Delivery at Amazon: Deep Dive cont.



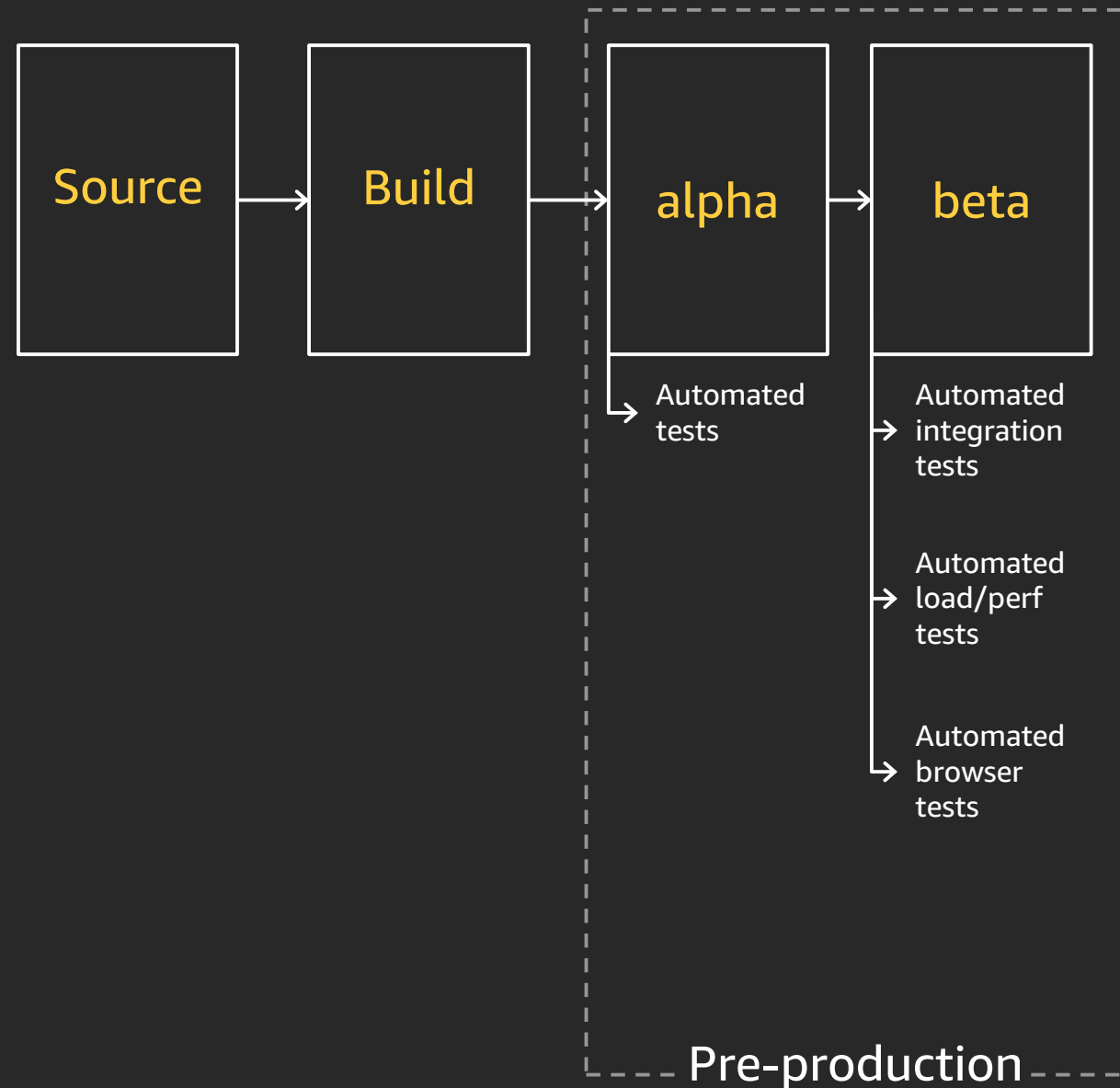
Continuous Delivery at Amazon: Deep Dive cont.



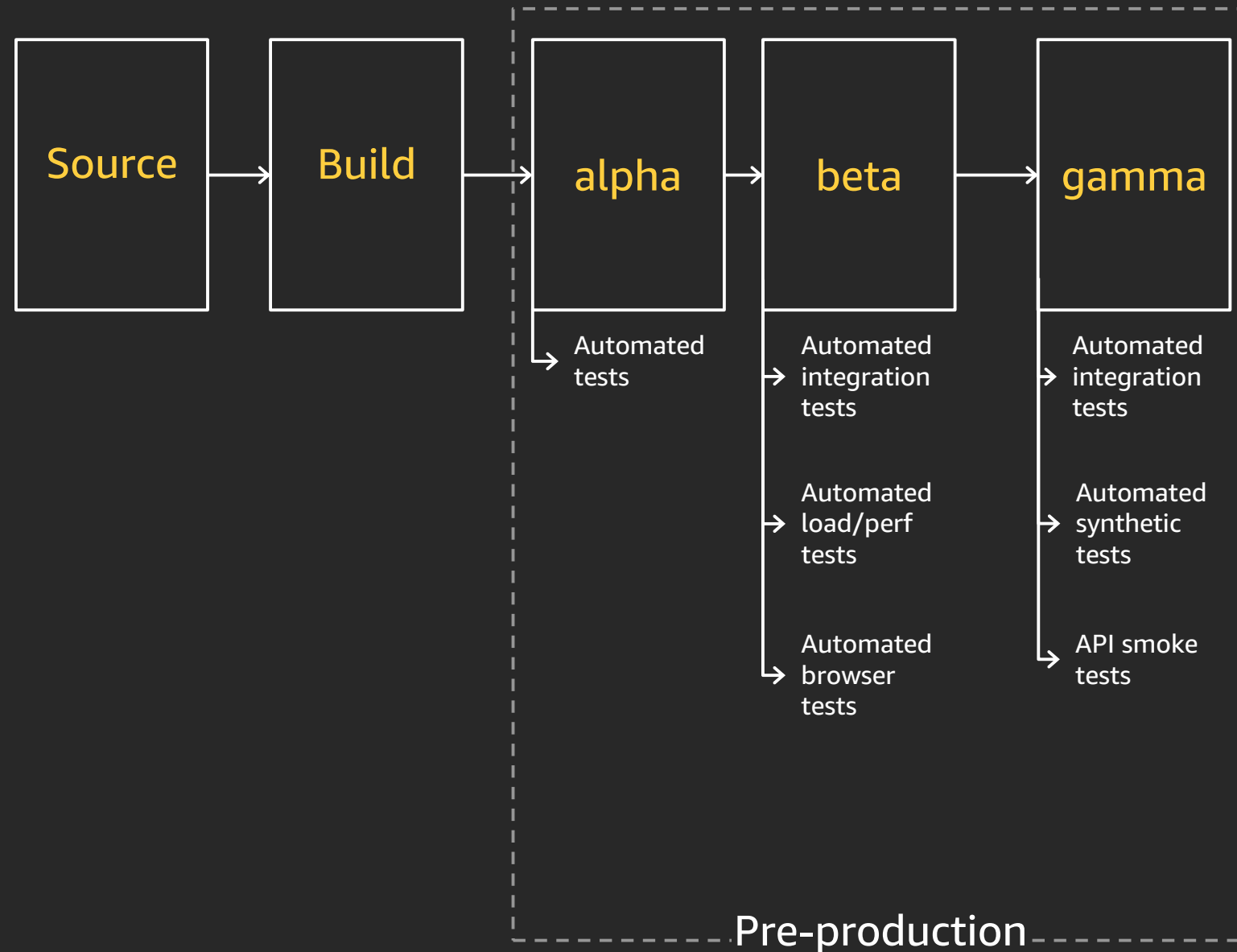
Continuous Delivery at Amazon: Deep Dive cont.



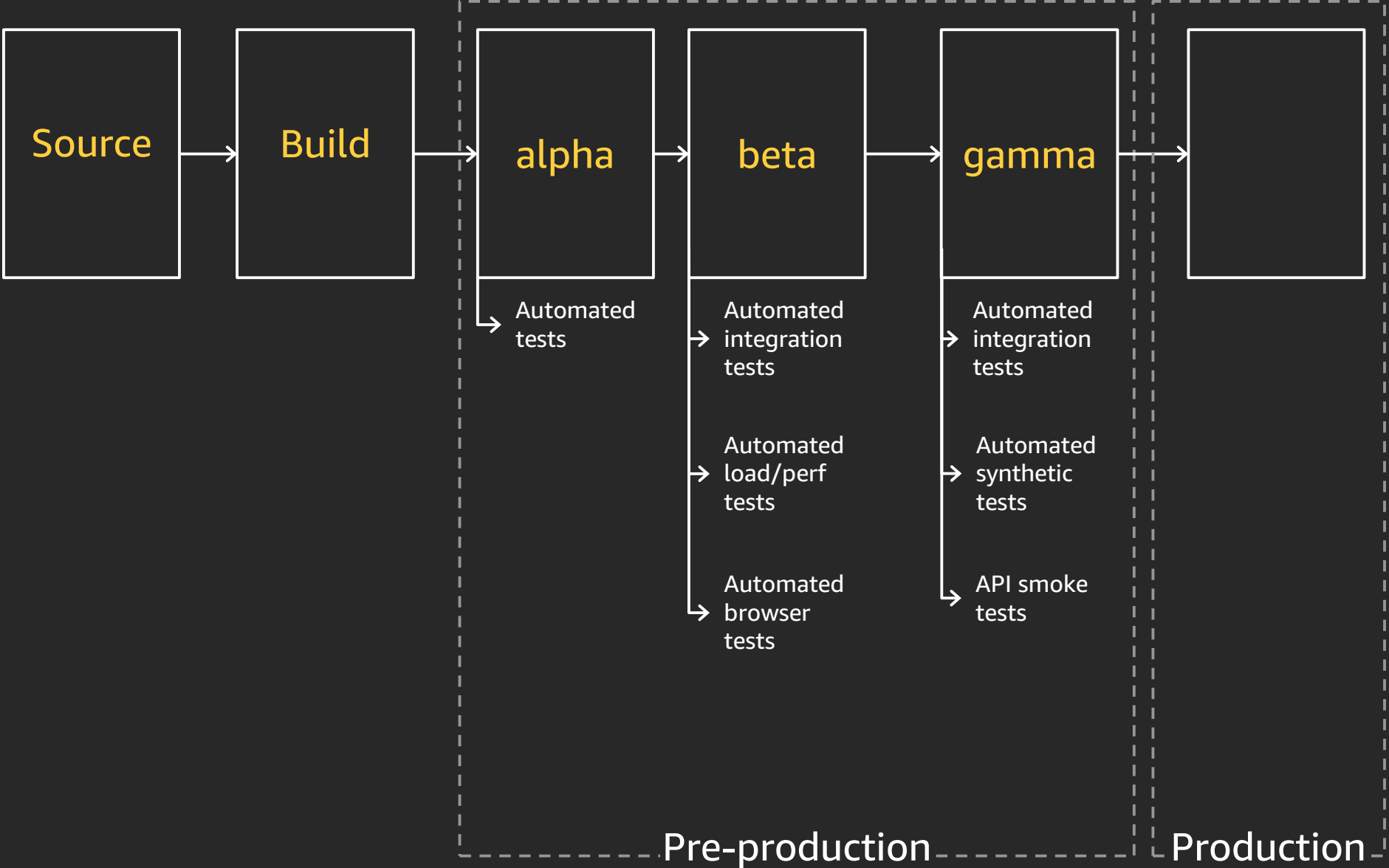
Continuous Delivery at Amazon: Deep Dive cont.



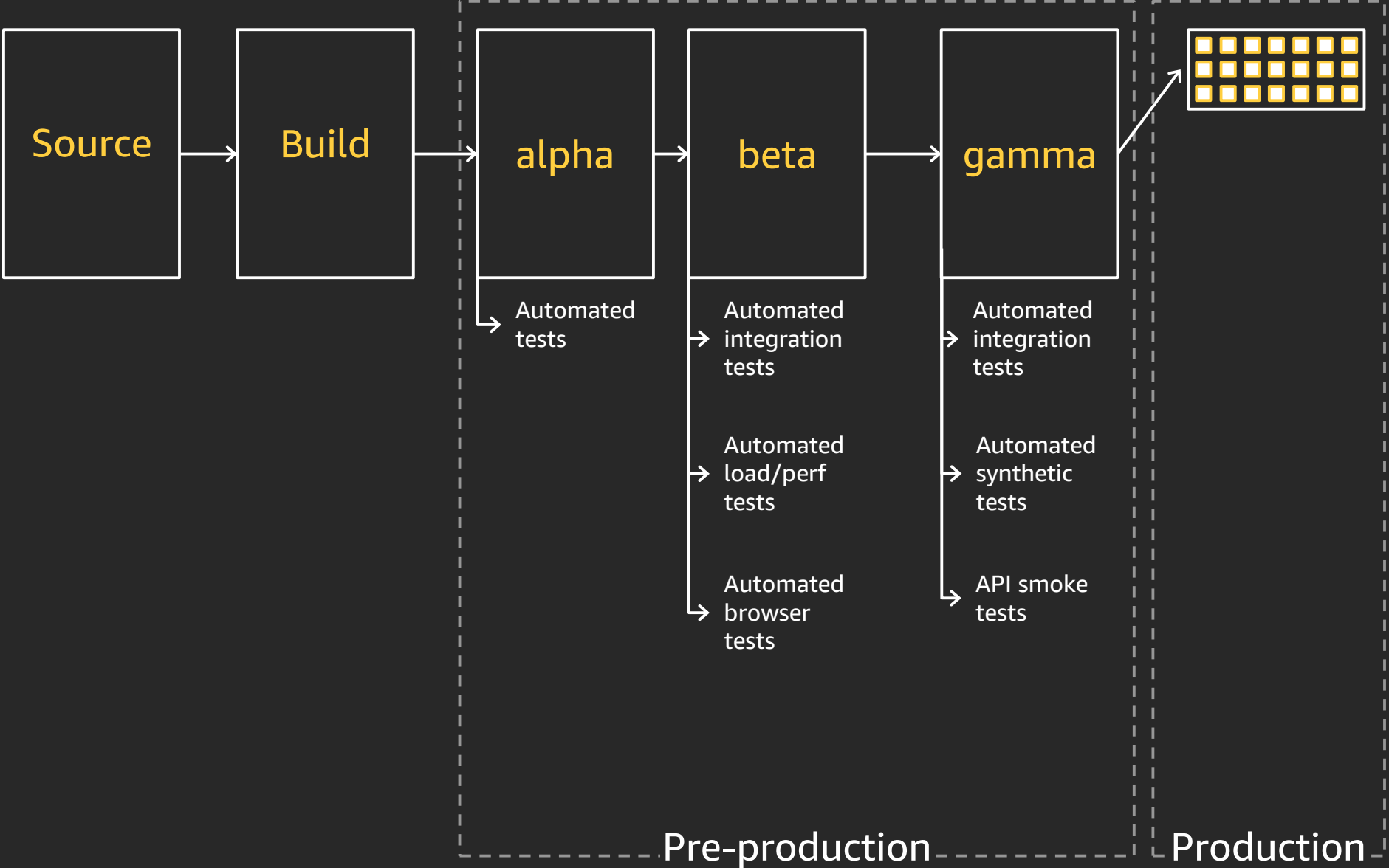
Continuous Delivery at Amazon: Deep Dive cont.



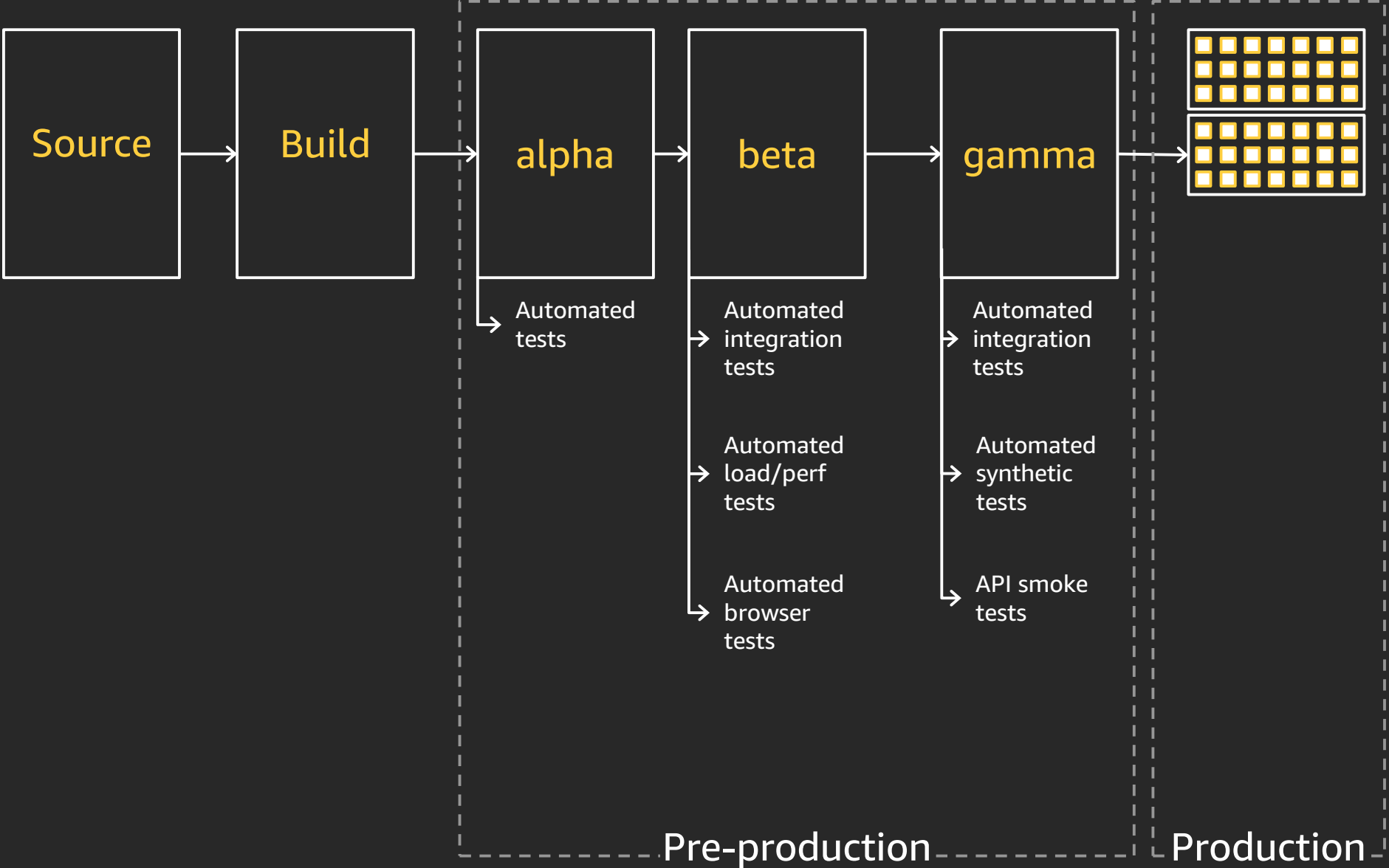
Continuous Delivery at Amazon: Deep Dive cont.



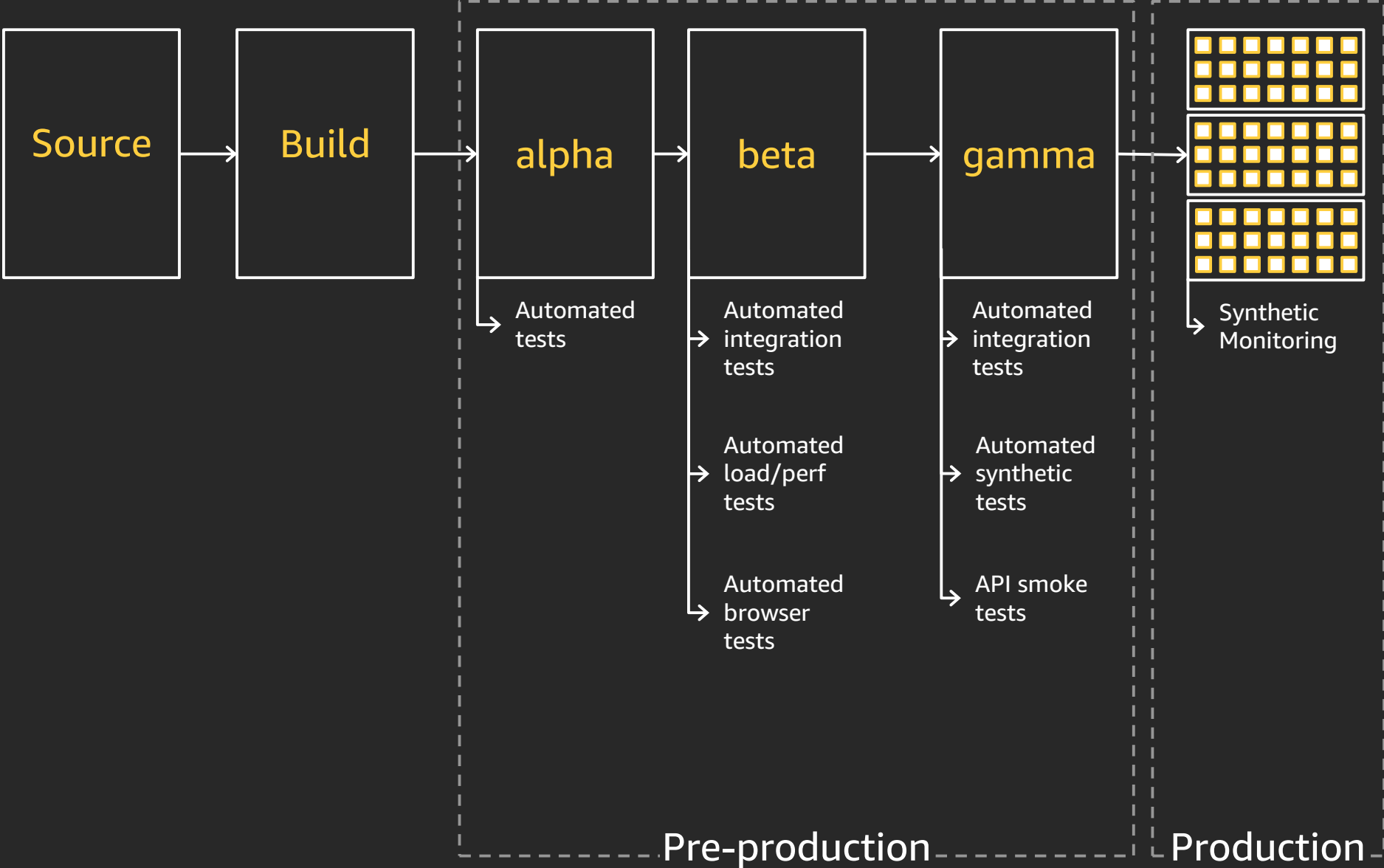
Continuous Delivery at Amazon: Deep Dive cont.



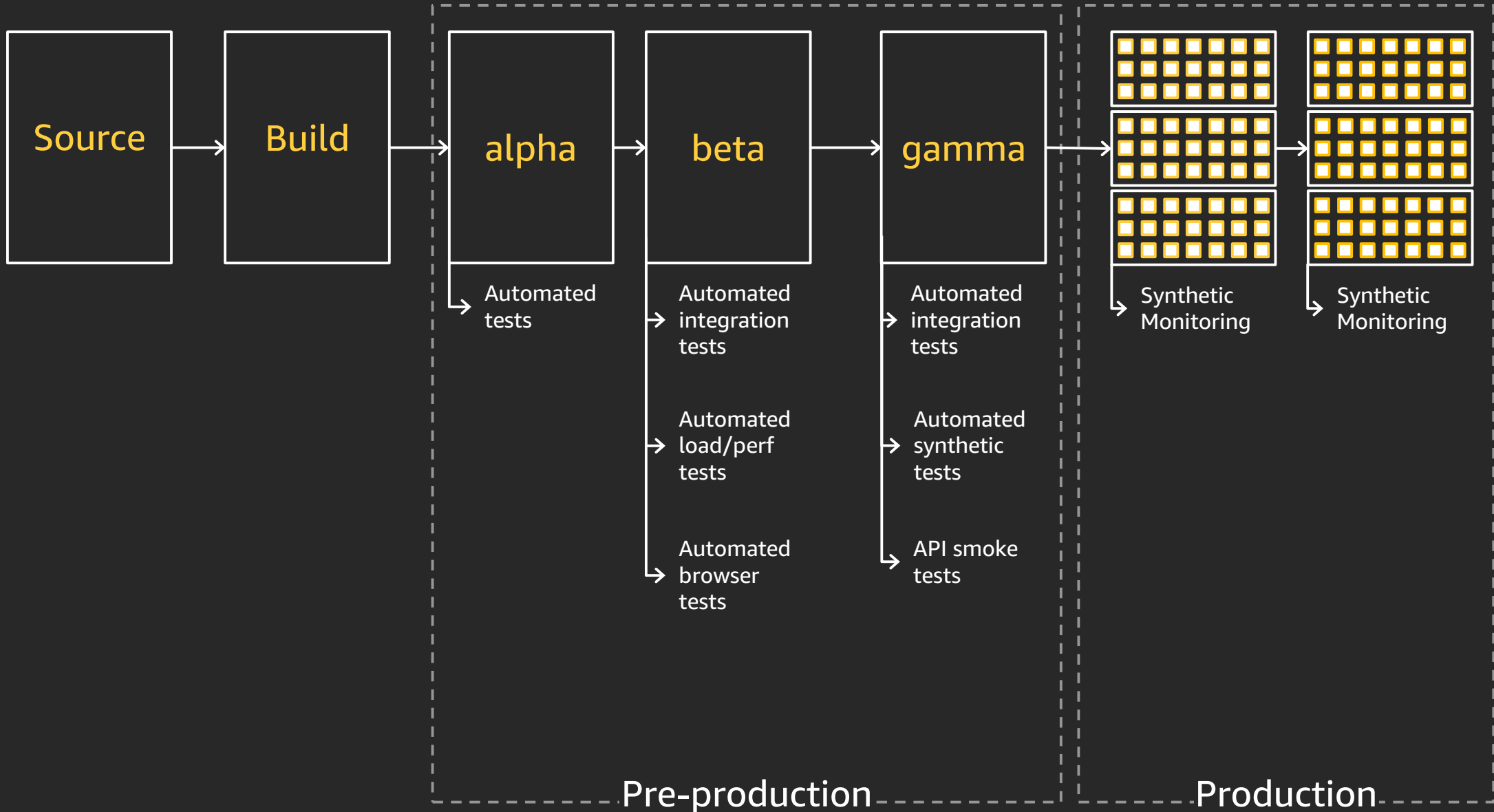
Continuous Delivery at Amazon: Deep Dive cont.



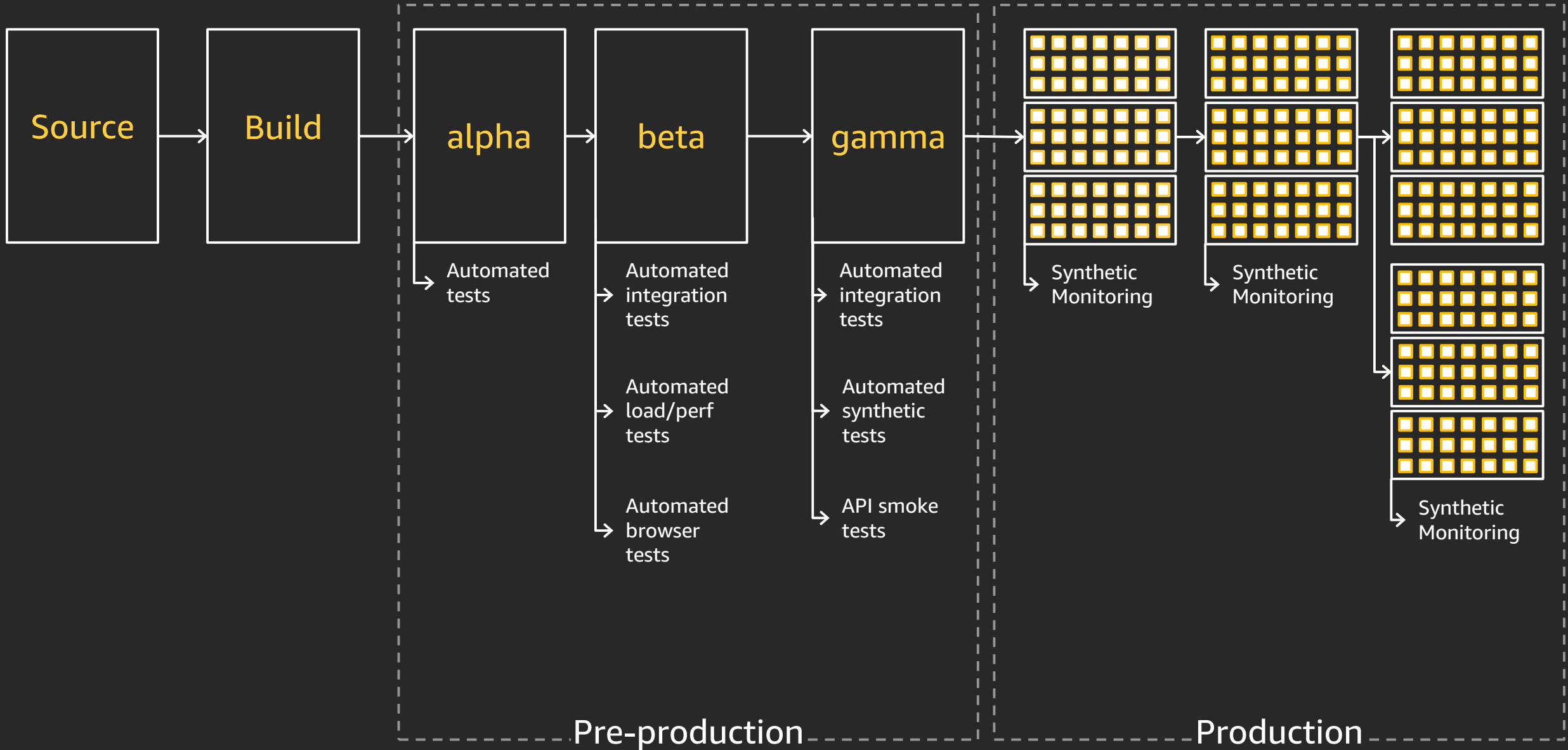
Continuous Delivery at Amazon: Deep Dive cont.



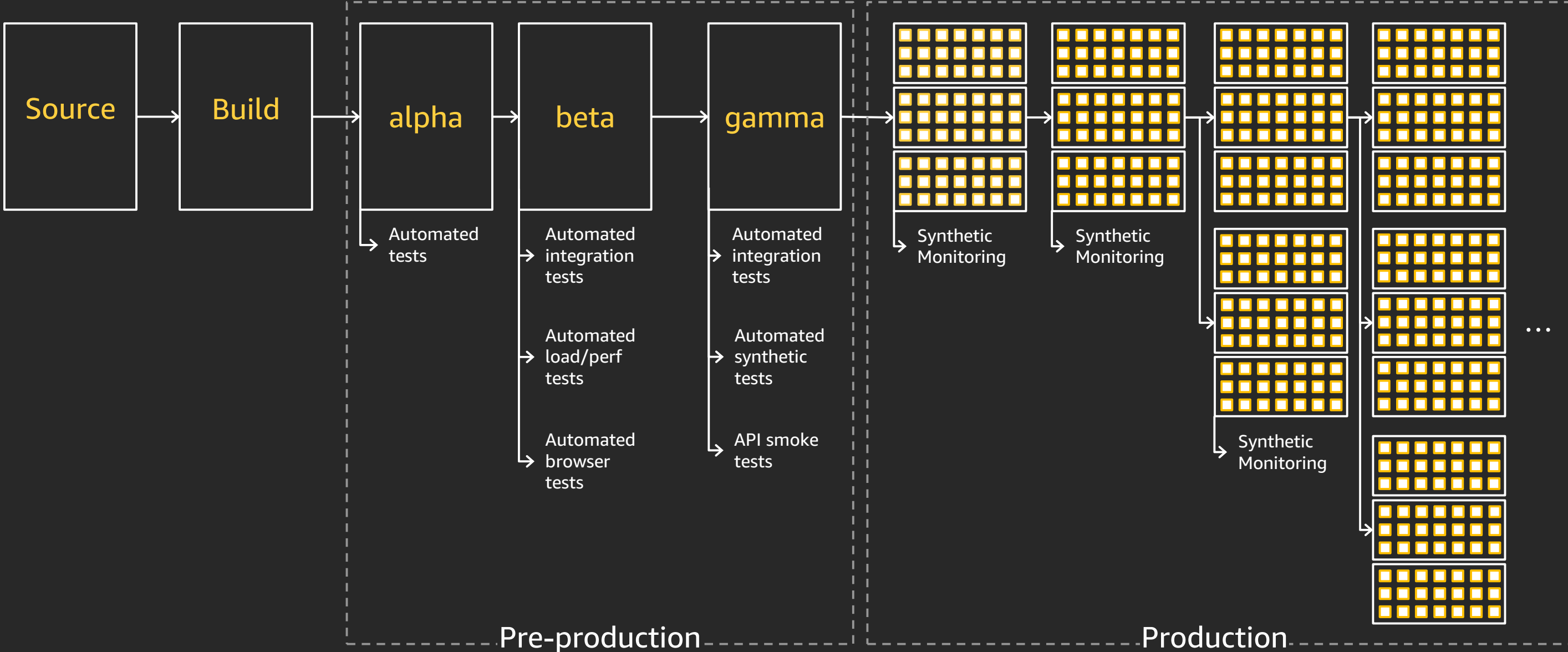
Continuous Delivery at Amazon: Deep Dive cont.



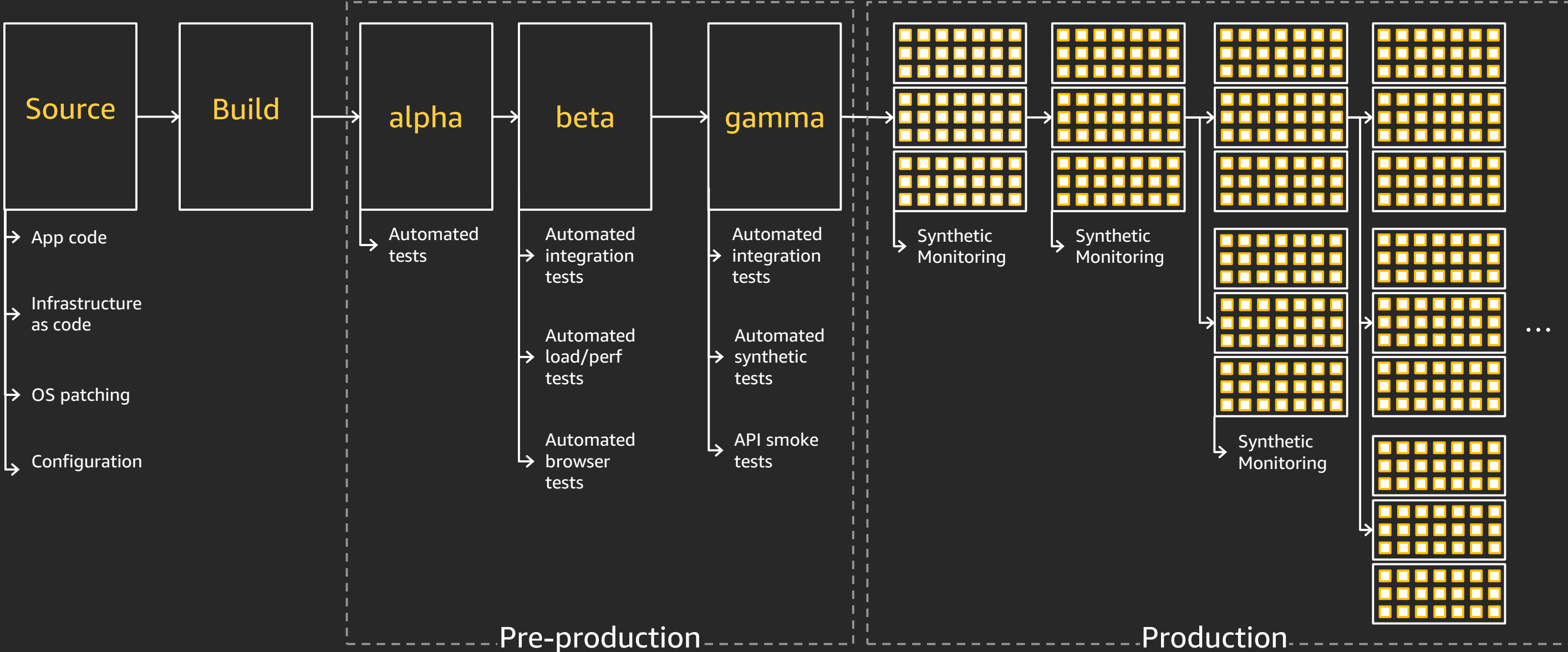
Continuous Delivery at Amazon: Deep Dive cont.



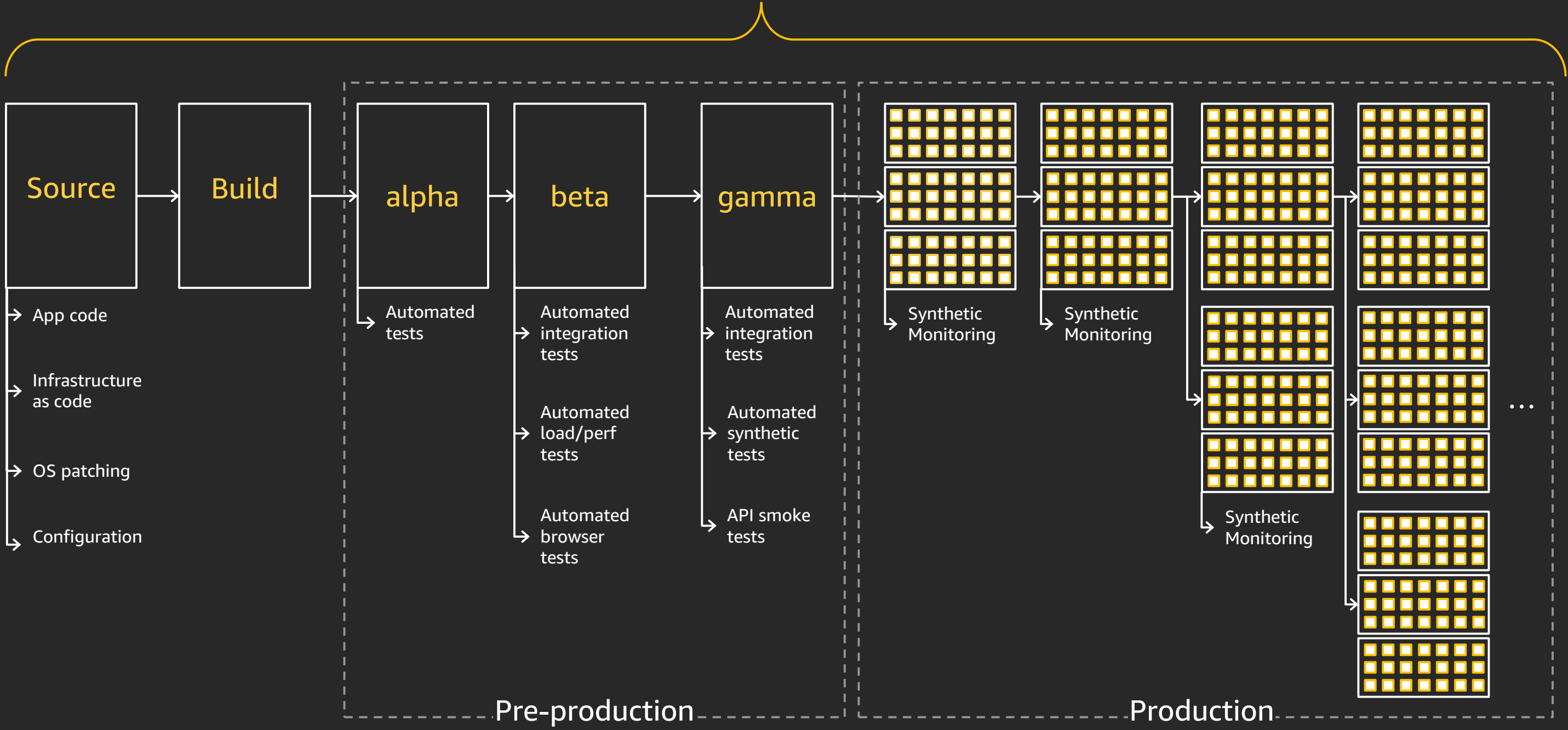
Continuous Delivery at Amazon: Deep Dive cont.



Continuous Delivery at Amazon: Deep Dive cont.



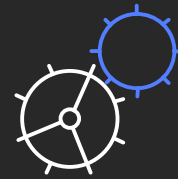
Pipeline blockers: Time windows, pipeline policies, code coverage, code review, security scans, dependency updates, etc.



Lessons learned at Amazon for developing and delivering software



Decompose for agility
(microservices, 2-pizza teams)



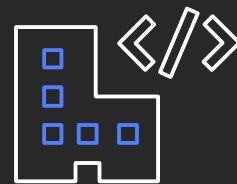
Automate everything



Standardized tools

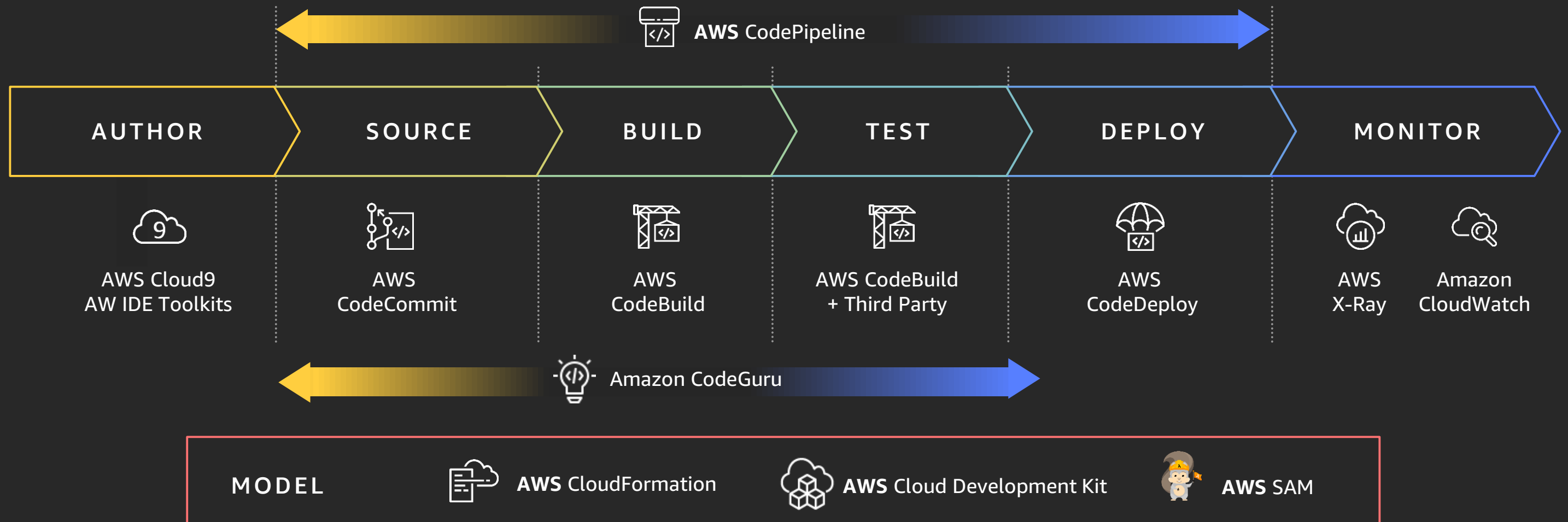


Belts and suspenders
(governance, templates)



Infrastructure as code

AWS Developer Tools for modern software delivery

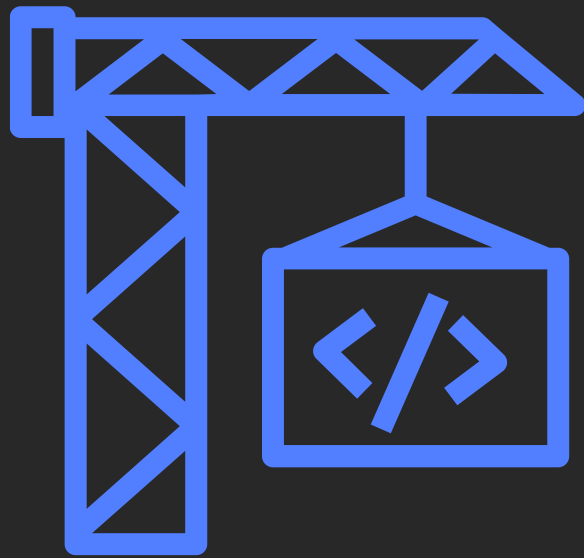


AWS CodeBuild launches



AWS CodeBuild

- Larger instance types
- ARM-based instances
- AWS Secrets Manager integration
- Export environment variables
- Test reporting (beta)



Developer Tools

CodeBuild

Source • CodeCommit

Build • CodeBuild

Getting started

Build projects

Build history

Reports

Report

Report run history

Account metrics

Deploy • CodeDeploy

Pipeline • CodePipeline

Go to resource

Feedback

Developer Tools > CodeBuild > Reports > ReportA > reporta160

testreport160

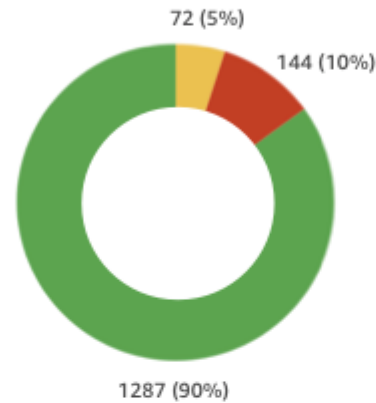
View report

View build run

View artifacts

Delete

Summary



Pass rate

74.6%

Report duration

35.62 sec

Created

1 minute ago

Passed Failed Other

Details

To see the full list of 1440 test cases for this report, [view the artifacts](#).

Test cases (1000)

View details

Search test case

Any Status

Any Prefix

< 1 2 > ⚙

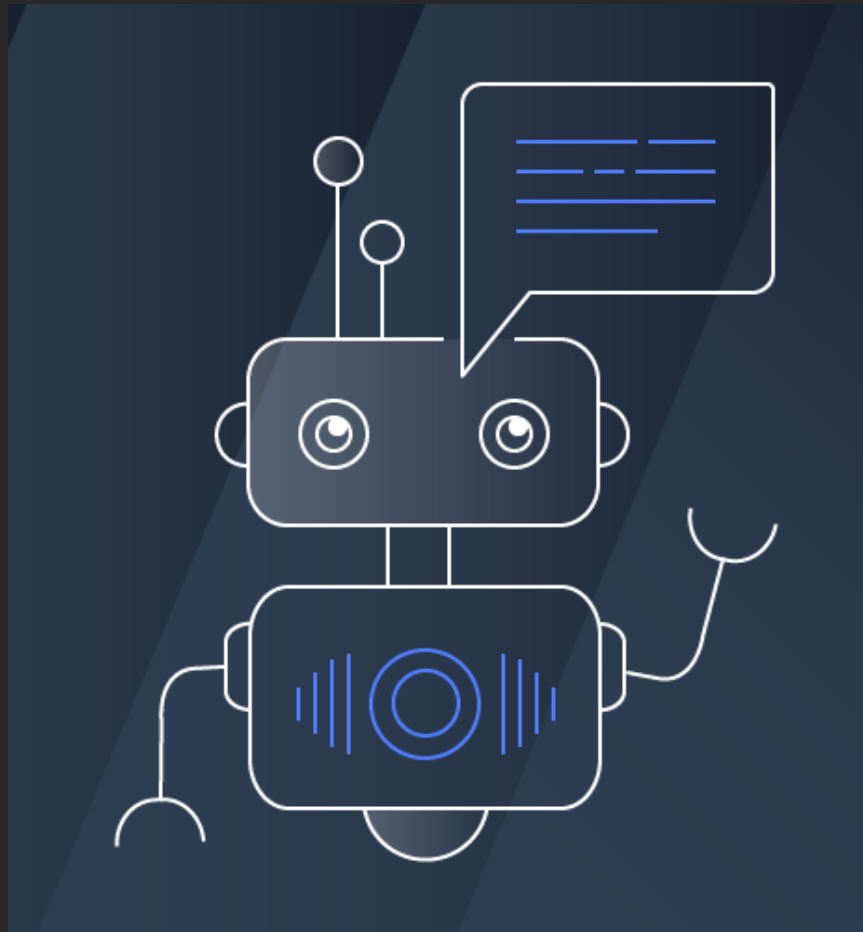
Test case	Status	Prefix	Message	Duration
TEST1	Failed	Unit Tests A	Exception in thread "main" java.lang.NullPointerException...	10 seconds
TEST2	Failed	Unit Tests A	Exception in thread "main" java.lang.NullPointerException...	2 seconds
TEST3	Failed	Unit Tests A	Exception in thread "main" java.lang.NullPointerException...	2 seconds

AWS CodePipeline dynamic variables



AWS CodePipeline now enables passing **dynamic values between pipeline stages**

AWS Chatbot (beta) can now run commands



AWS Chatbot

Interactive agent for ChatOps on AWS

- Receive notifications
- **Run commands for diagnostic information**
- Pre-defined IAM policy templates
- Support for Slack and Chime

NEW



Send notifications from an AWS Code* Service

NEW

Subscribe Code* services to Amazon SNS topics for receive notifications. Integrated with AWS Chatbot.

 **AWS CodeCommit Notification | us-west-2 | Account:** 

Comment published on pull request:

-lsengard : This looks good.



Repository

Hello-Dublin

Pull Request ID

12

Time

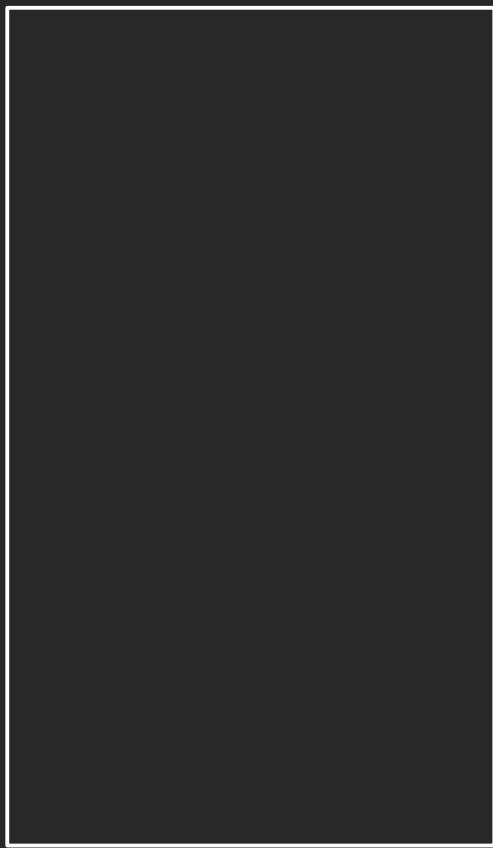
2019-11-21T19:17:47Z

Resource ARN

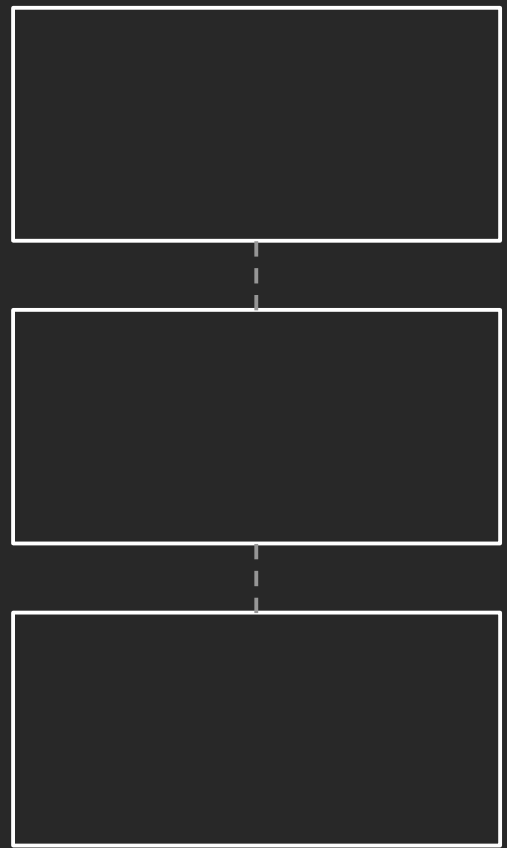
arn:aws:codecommit: :Hello-Dublin

Modern architectures

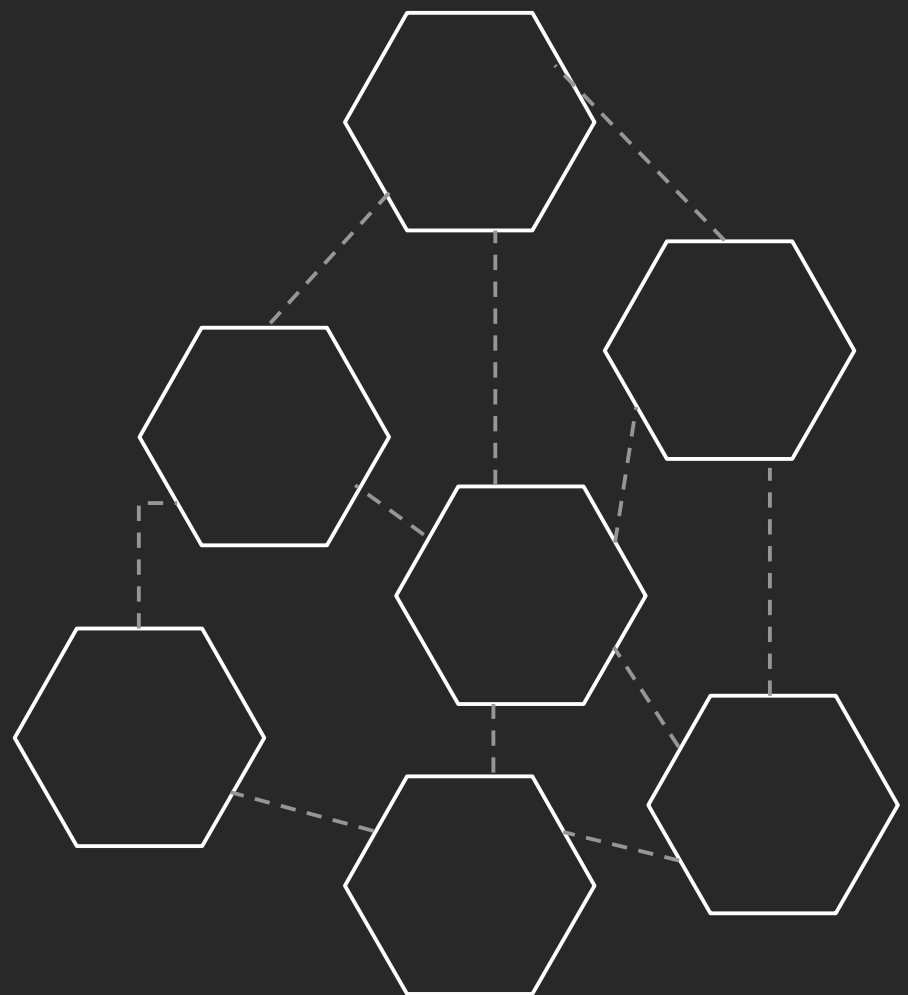
Better performance, more complexity



Monolithic



n-tier



**Microservices
& serverless**

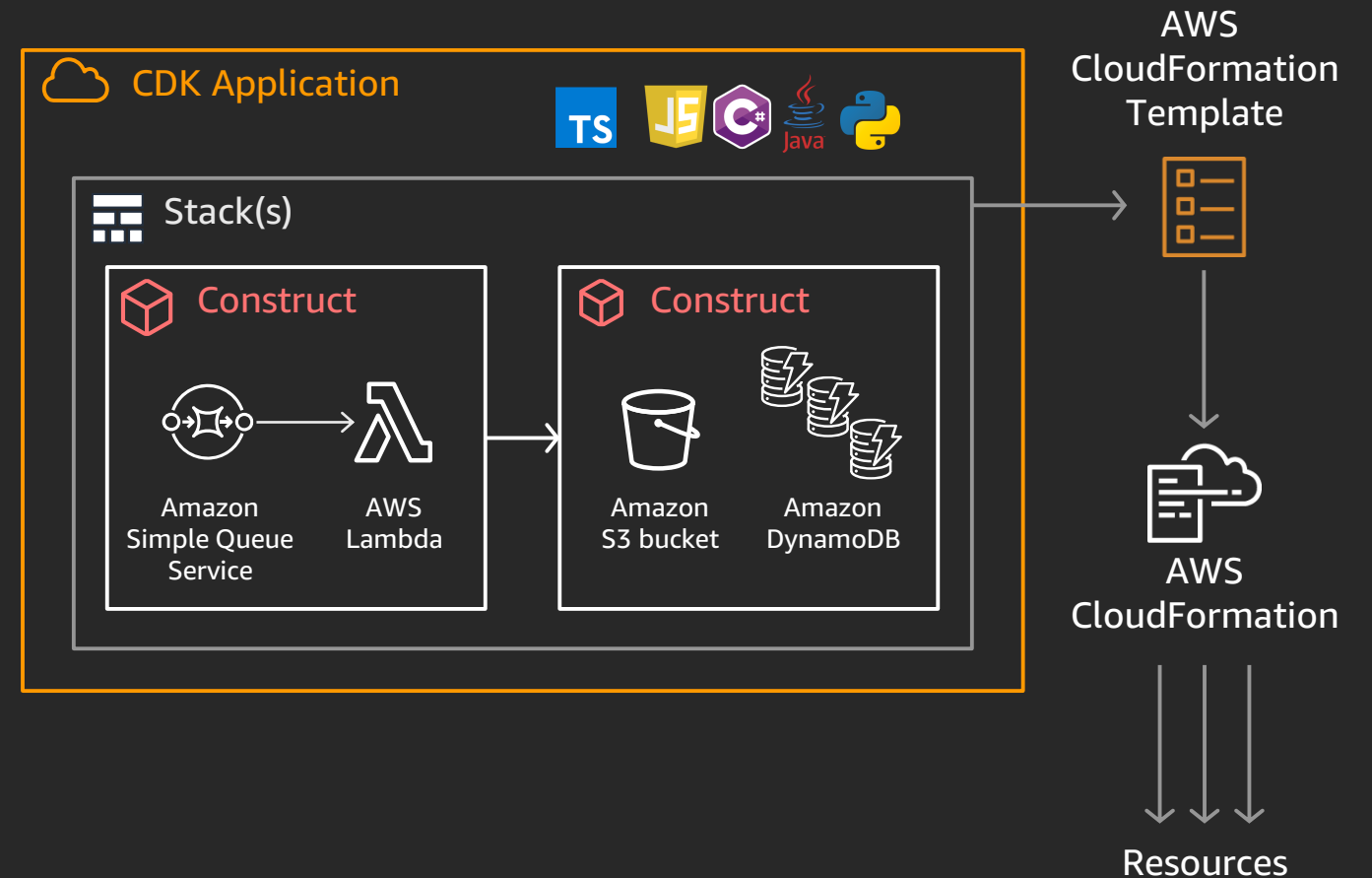
AWS Cloud Development Kit (AWS CDK)

Define cloud infrastructure using familiar programming languages

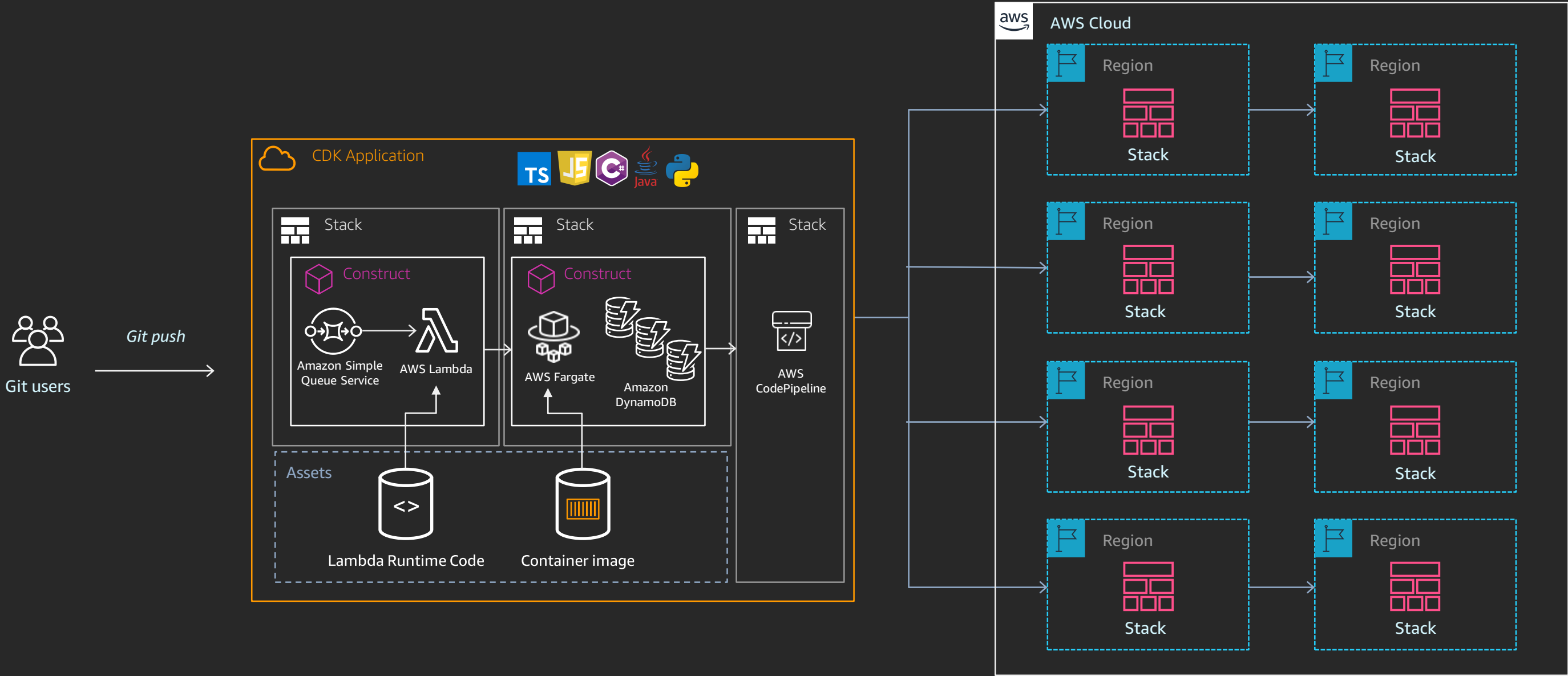


NEW

Higher-level components to preconfigure cloud resources



CI/CD with the AWS CDK



Under comment at CDK GitHub repo!

AWS CDK Explorer (preview) for visualizing stacks and resources defined in CDK constructs

NEW

The screenshot displays the AWS CDK Explorer interface. On the left, the 'EXPLORER' pane shows a project structure under 'MY-CDK-PROJECTS' with folders 'bin', 'cdk.out', and 'lib'. The 'lib' folder contains 'hello-world-stack.d.ts', 'hello-world-stack.js', and 'hello-world-stack.ts'. Below this is the 'AWS CDK EXPLORER (PREVIEW)' pane, which shows a tree view of resources for 'my-cdk-projects/hello-world', including 'HelloWorldStack', 'HelloWorldQueue', 'HelloWorldTopic', 'HelloVpc', and 'PublicSubnet1'. The 'HelloVpc' resource is expanded to show properties like 'cidrBlock: 10.0.0.0/16', 'enableDnsHostnames: true', 'enableDnsSupport: true', and 'instanceTenancy: default'. The main editor pane shows the TypeScript code for 'hello-world-stack.ts', which defines a 'HelloWorldStack' class extending 'cdk.Stack'. The code includes imports for 'sns', 'subs', 'sqs', 'cdk', and 'ec2', and defines a constructor that creates an 'SqsQueue', an 'SnsTopic', and an 'Ec2Vpc'. The terminal at the bottom shows the command 'cdk synth' being executed.

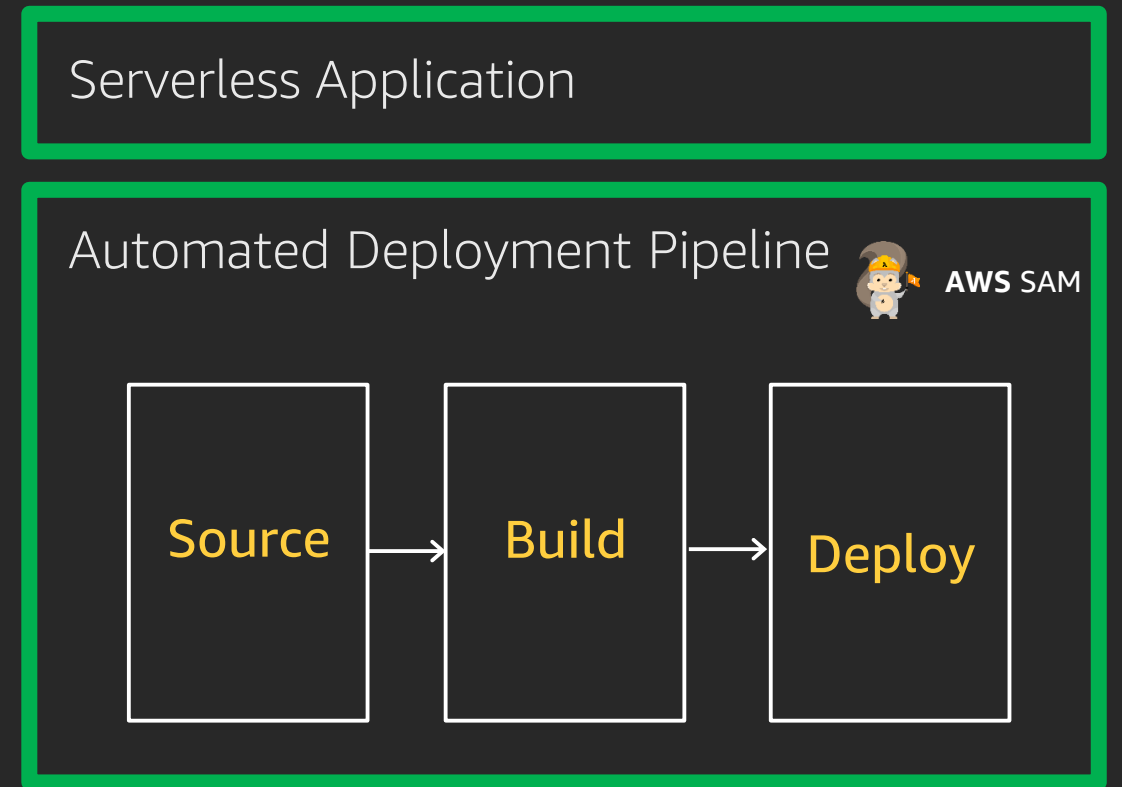
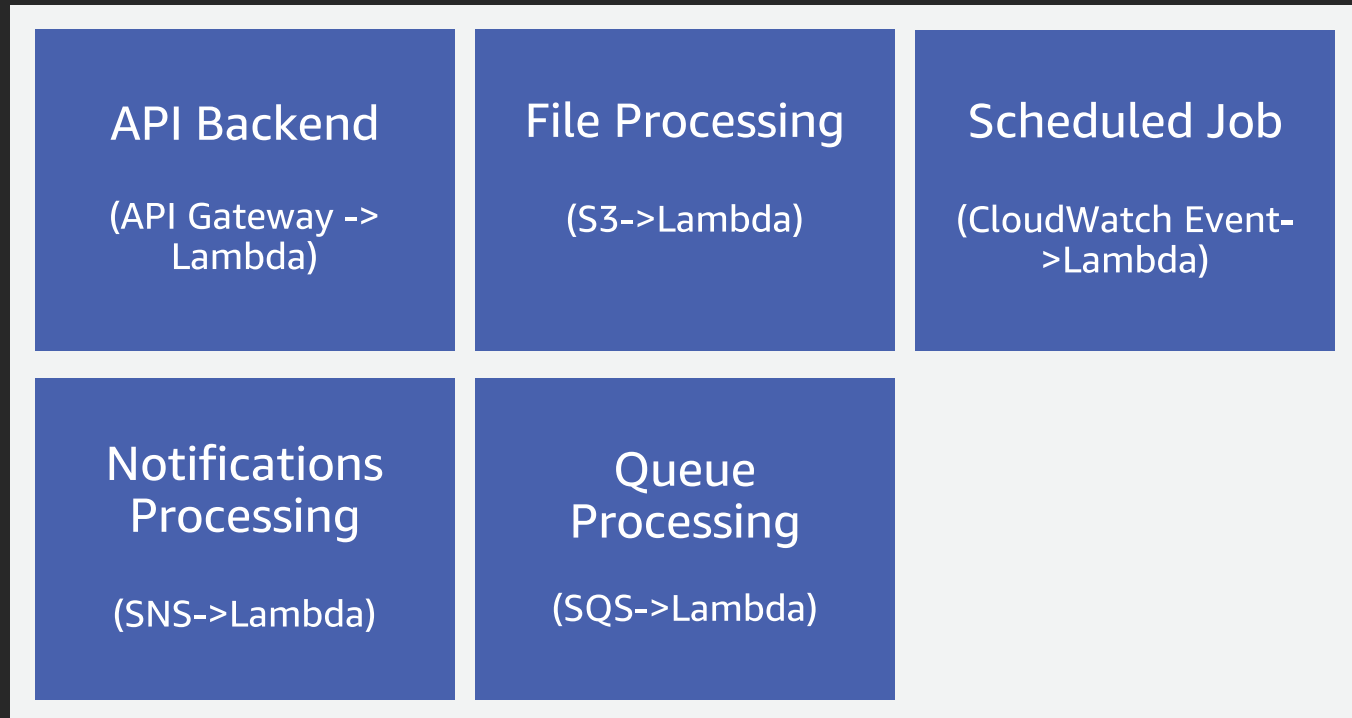
```
hello-world > lib > TS hello-world-stack.ts > ...  
You, a few seconds ago | 1 author (You)  
1 import sns = require('@aws-cdk/aws-sns');  
2 import subs = require('@aws-cdk/aws-sns-subscriptions');  
3 import sqs = require('@aws-cdk/aws-sqs');  
4 import cdk = require('@aws-cdk/core');  
5 import ec2 = require('@aws-cdk/aws-ec2');  
6  
You, a few seconds ago | 1 author (You)  
7 export class HelloWorldStack extends cdk.Stack {  
8   constructor(scope: cdk.App, id: string, props?: cdk.StackProps) {  
9     super(scope, id, props);  
10  
11     const queue = new sqs.Queue(this, 'HelloWorldQueue', {  
12       visibilityTimeout: cdk.Duration.seconds(300)  
13     });  
14  
15     const topic = new sns.Topic(this, 'HelloWorldTopic');  
16  
17     topic.addSubscription(new subs.SqsSubscription(queue));  
18  
19     const vpc = new ec2.Vpc(this, 'HelloVpc');  
20   }  
21 }  
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 3: bash
bash-3.2\$ cdk synth
Resources:

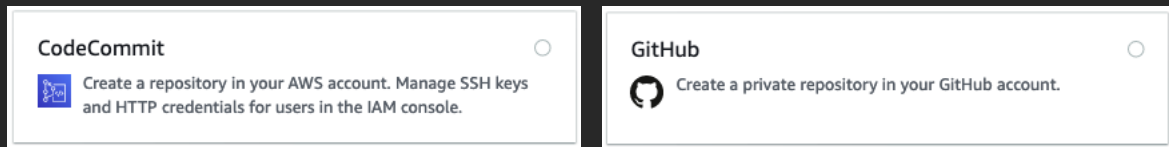
Built-in CI/CD for serverless applications



1 Pick a quick start from AWS Lambda Console



2 Configure your application's repository

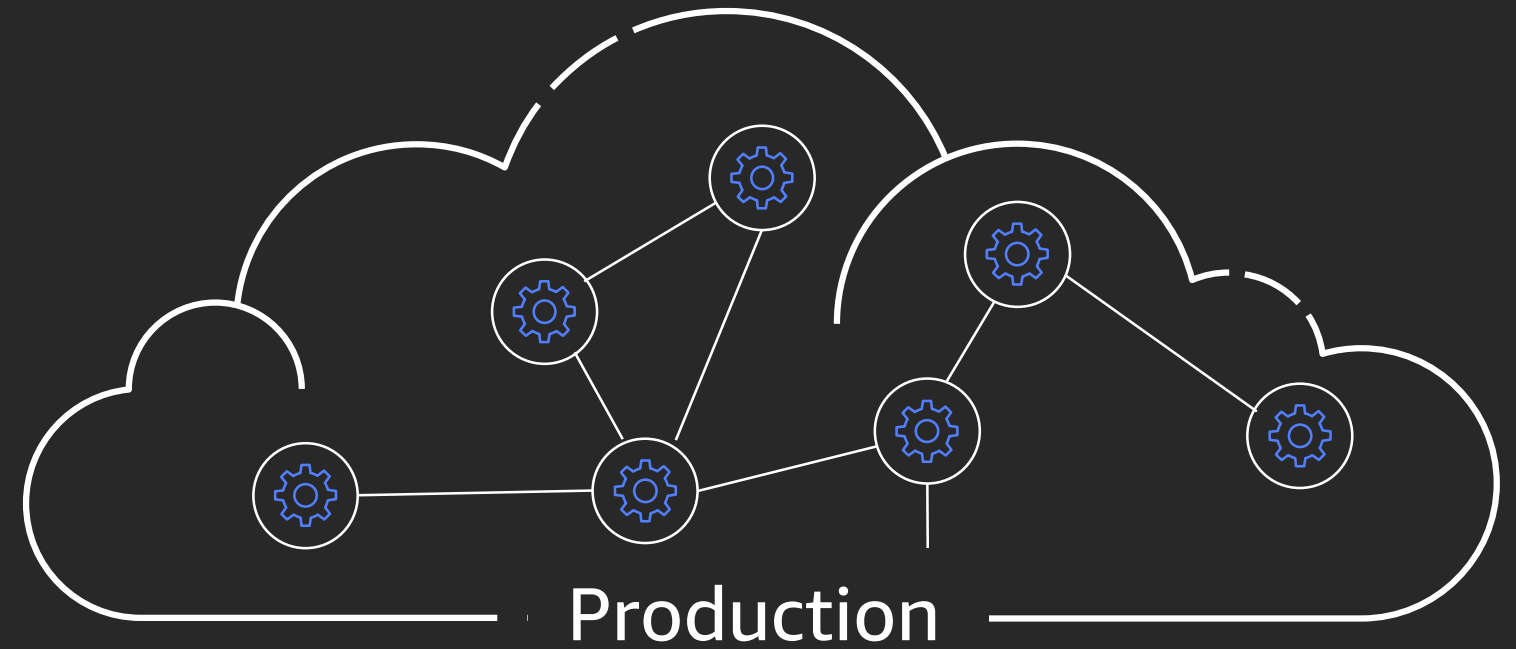
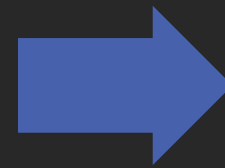


A look ahead

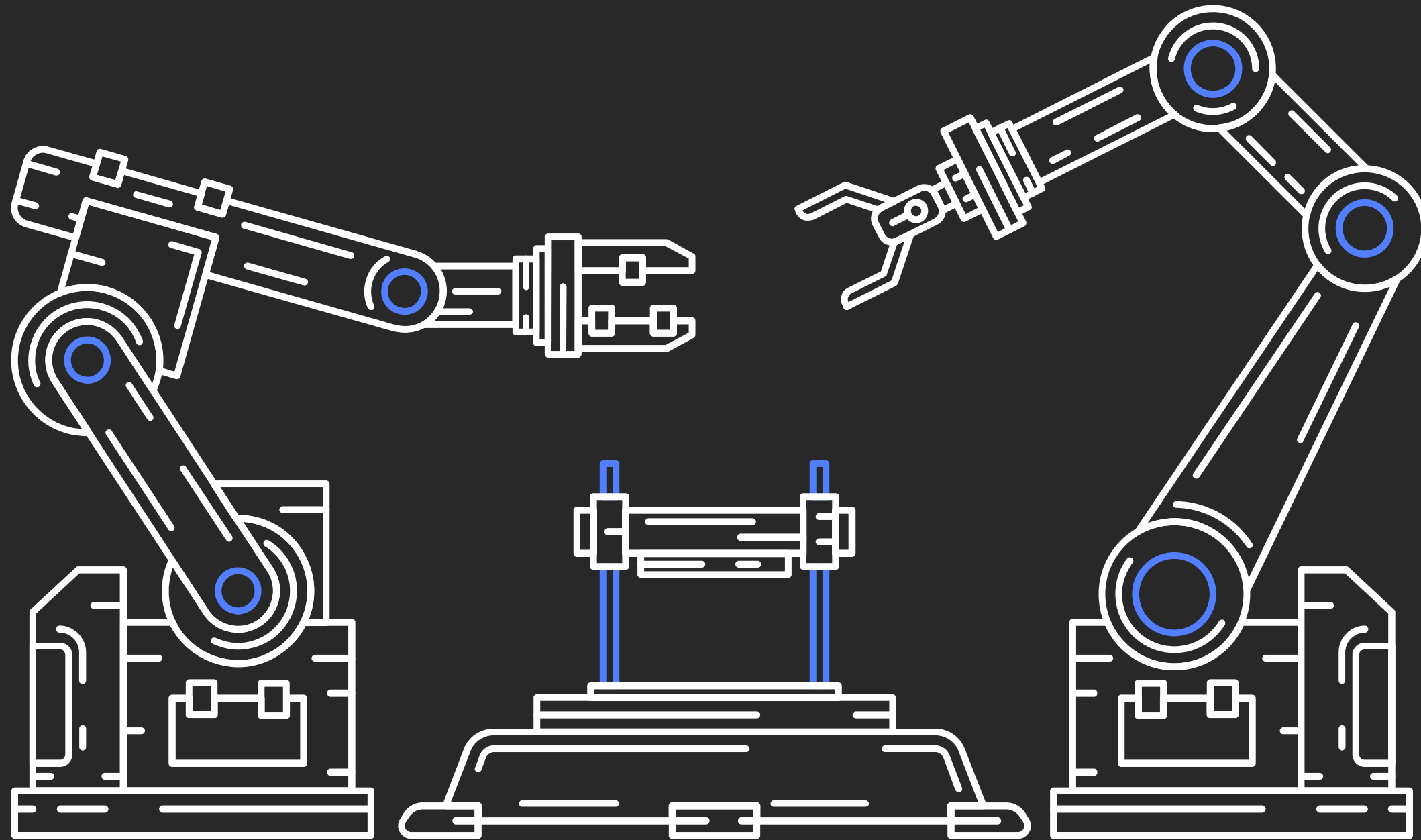
Security and safety (while moving quickly)



Where we're headed: Modern applications



Automate away the muck



Thank you!



Please complete the session survey in the mobile app.