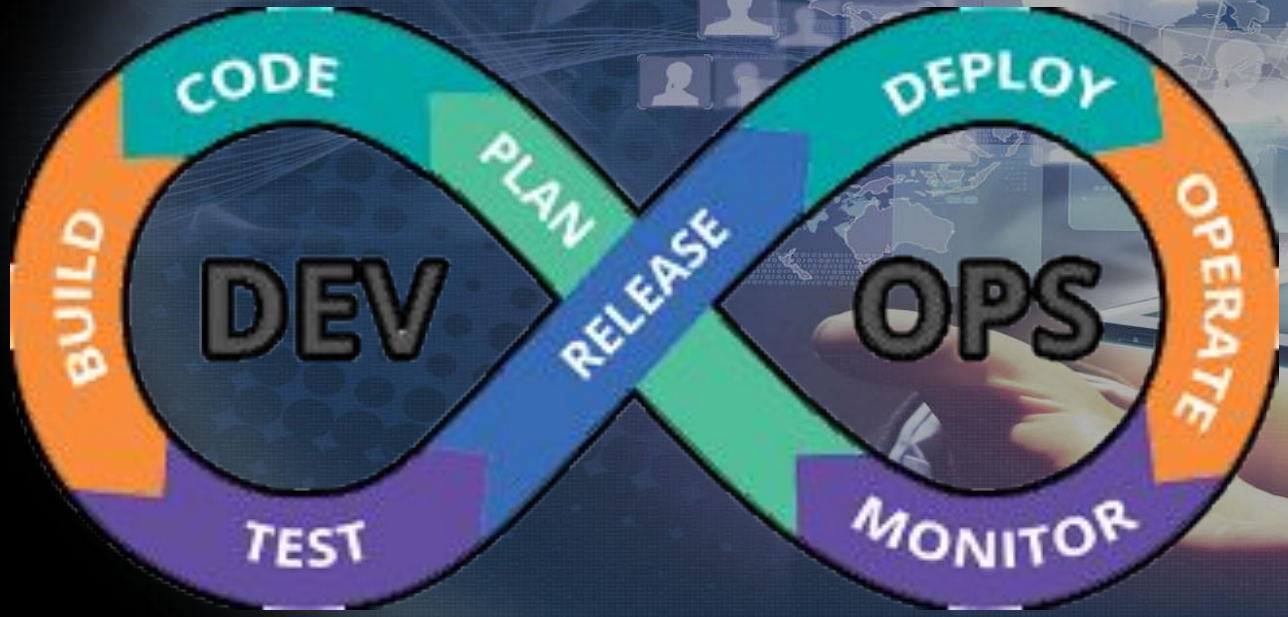
REVATURE PROJECT 3
DEV-OPS PIPELINEFORCE

April 2020 Revature Batch
Trainer: August Duet

TEACH ME DEV OPS

What Is DevOps?



DEV OPS FOUNDER
PATRICK DUBOIS

Presenter:
Paityn Maynard

WHY DO WE USE DEV OPS?

- Deployment to Clientele
- Reduce Deployment Failures
  - Continuous Integration and Deployment
- Load Testing
- Application Monitoring

Presenter:
Paityn Maynard

Ansible + Initializing the Pipeline

Objective

To automatically initialize the pipeline with as little manual interaction as possible.

Approach

- Write a bash script that asks for pipeline parameters
- Create a parameterized Jenkins job template
- Utilize Ansible to automatically initialize jobs and webhooks
- Configure the Jenkins job to automatically send email notifications

What is Ansible?

- A simple and flexible IT automation engine that can automate
  - cloud provisioning
  - configuration management
  - application deployment

Presenter:
Reginald Jefferson

Why Use Ansible For This Project?

- Need to automate multiple tasks with a simple command
- Communication with different hosts for job completion
  - Communicate with Jenkins to create jobs
  - Communicate with GitHub to create webhooks
- Easy to parameterize

Presenter:
Reginald Jefferson

Job Creation Workflow

```
#! /bin/bash

read -p "Please enter your Github username: " gitUser
read -sp "Please enter your Github API Token: " gitToken
echo
read -p "Please enter your full repository url (including .git): " gitUrl
read -p "Please enter the name of the branch this pipeline will use: " gi
tBranch

[[ $gitUrl =~ .*github\.com/(.*)\.git ]] && gitRepoFull=${BASH_REMATCH[1]
} #grab the fullname of the github repo from the url
[[ $gitRepoFull =~ .*/(.+) ]] && gitRepoName=${BASH_REMATCH[1]} #grab onl
y the repo name from the github repo fullname

jenkinsUrl="http://jenkins-ci.revaturelabs.com/" #url for Jenkins server
jenkinsUsername="svc_revature" #Username for Jenkins authentication
jenkinsPassword="password" #Password for Jenkins authentication
causeString="Pushed to branch" #Reason for Jenkins Job build
uniqueVal="${gitUser}_${gitRepoName}_${gitBranch}" #unique value used for
 token and job name

echo $gitRepoFull
echo $gitRepoName

ansible-playbook test_job_create.yml -e "cause_string=$causeString token_
val=$uniqueVal repo_url=$gitUrl git_branch=$gitBranch jenkins_url=$jenkin
sUrl jenkins_job_name=$uniqueVal jenkins_user=$jenkinsUsername jenkins_pa
ssword=$jenkinsPassword git_repo=$gitRepoFull payload_url=${jenkinsUrl}ge
neric-webhook-trigger/invoke?token=${uniqueVal} github_user=$gitUser gith
ub_token=$gitToken"
```

Triggers Playbook

test_job_create.yml

Creates Job from Template

Job: username_repositoryName_branchName

Presenter:
Joshua Brewer

Parameterizing the Jenkins Job

```
    <triggers>
      <org.jenkinsci.plugins.gwt.GenericTrigger plugin="generic-webhook-trigger@1.67">
        <spec></spec>
        <regexpFilterText></regexpFilterText>
        <regexpFilterExpression></regexpFilterExpression>
        <printPostContent>false</printPostContent>
        <printContributedVariables>false</printContributedVariables>
        <causeString>{{cause_string}}</causeString>
        <token>{{token_val}}</token>
        <silentResponse>false</silentResponse>
        <overrideQuietPeriod>false</overrideQuietPeriod>
      </org.jenkinsci.plugins.gwt.GenericTrigger>
    </triggers>
  </org.jenkinsci.plugins.workflow.job.properties.PipelineTriggersJobProperty>
</properties>
<definition class="org.jenkinsci.plugins.workflow.cps.CpsScmFlowDefinition" plugin="workflow-cps@2.80">
  <scm class="hudson.plugins.git.GitSCM" plugin="git@4.2.2">
    <configVersion>2</configVersion>
    <userRemoteConfigs>
      <hudson.plugins.git.UserRemoteConfig>
        <url>{{repo_url}}</url>
      </hudson.plugins.git.UserRemoteConfig>
    </userRemoteConfigs>
    <branches>
      <hudson.plugins.git.BranchSpec>
        <name>*/{{git_branch}}</name>
      </hudson.plugins.git.BranchSpec>
    </branches>
```

Presenter:
Joshua Brewer

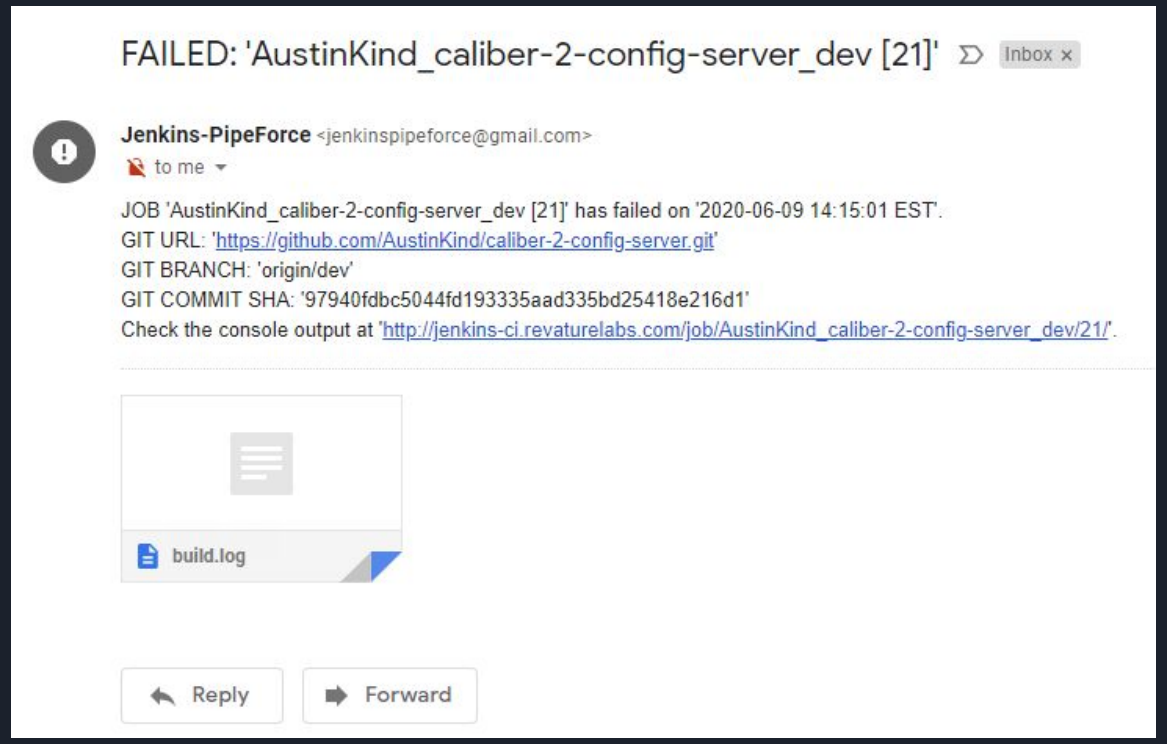
Writing the Ansible Playbook

```yaml
---
- hosts: localhost
  tasks:
        - name: create a jenkins job
          jenkins_job:
                config: "{{ lookup('template', 'templates/jenkins_job.j2') }}"
                name: "{{ jenkins_job_name }}"
                url: "{{ jenkins_url }}"
                user: "{{ jenkins_user }}"
                password: "{{ jenkins_password }}"
        - name: create webhook on github
          github_webhook:
                repository: "{{ git_repo }}"
                url: "{{ payload_url }}"
                events:
                  - push
                user: "{{ github_user }}"
                token: "{{ github_token }}"
```

Presenter:
Christopher Hill

Adding Email Notifications

```
post {
    failure {
        emailext (
            subject: "FAILED: '${env.JOB_NAME} [${env.BUILD_NUMBER}]'",
            body: "JOB '${env.JOB_NAME} [${env.BUILD_NUMBER}]' has failed on '${env.BUILD_TIMESTAM
            to: "centerofexcellence@revature.com",
            attachLog: true
            )
    }
}
```



Presenter:
Christopher Hill

Hurdles & Solutions

Hurdles:

1. Long wait to gain access to Revature's AWS network
2. Large scope (Jack of all trades)

Our Approach:

1. Created our own Test servers for Jenkins and Ansible
2. Collaborated with other teams to determine Ansible role
   a. Some roles are more suited towards other tools like Spinnaker or Jenkins

Presenter:
Dylan Tran

Next Phase

- Automate the ECR repo creation
- Create a “Clean-up Playbook”
  - Delete unused Jenkins Jobs
  - Remove images created from builds

Presenter:
Dylan Tran

Jenkins

Presenters: Danarrius Broadway,
William Chung,
Damier Raymond

- What is Jenkins?
- How does it fit in DevOps?
- What are benefits of Jenkins
- How we implement it to our Project?

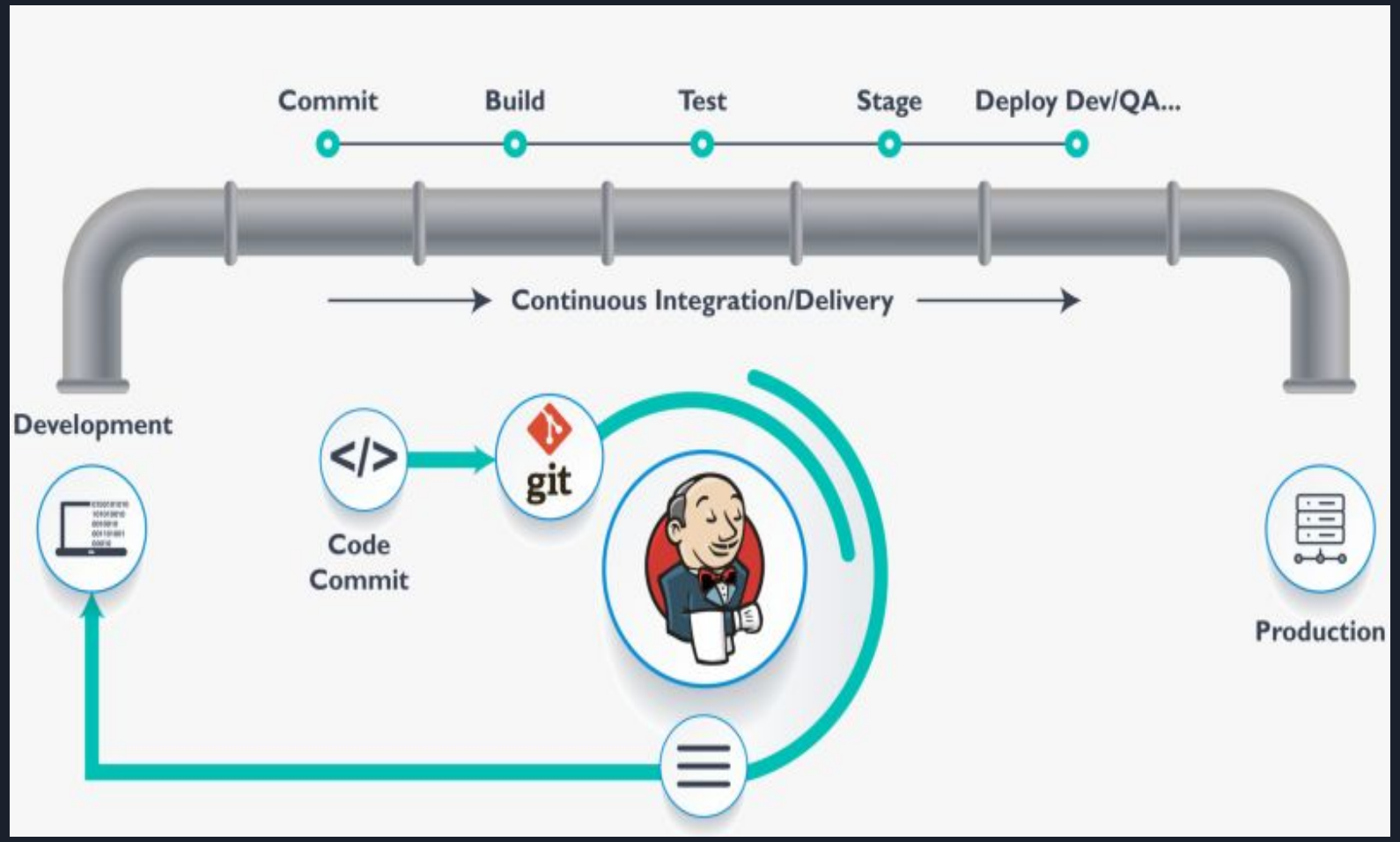
Presenter:
Danarrius Broadway

What is Jenkins ?

Jenkins is a open-source tool to automate Continuous Integration tasks

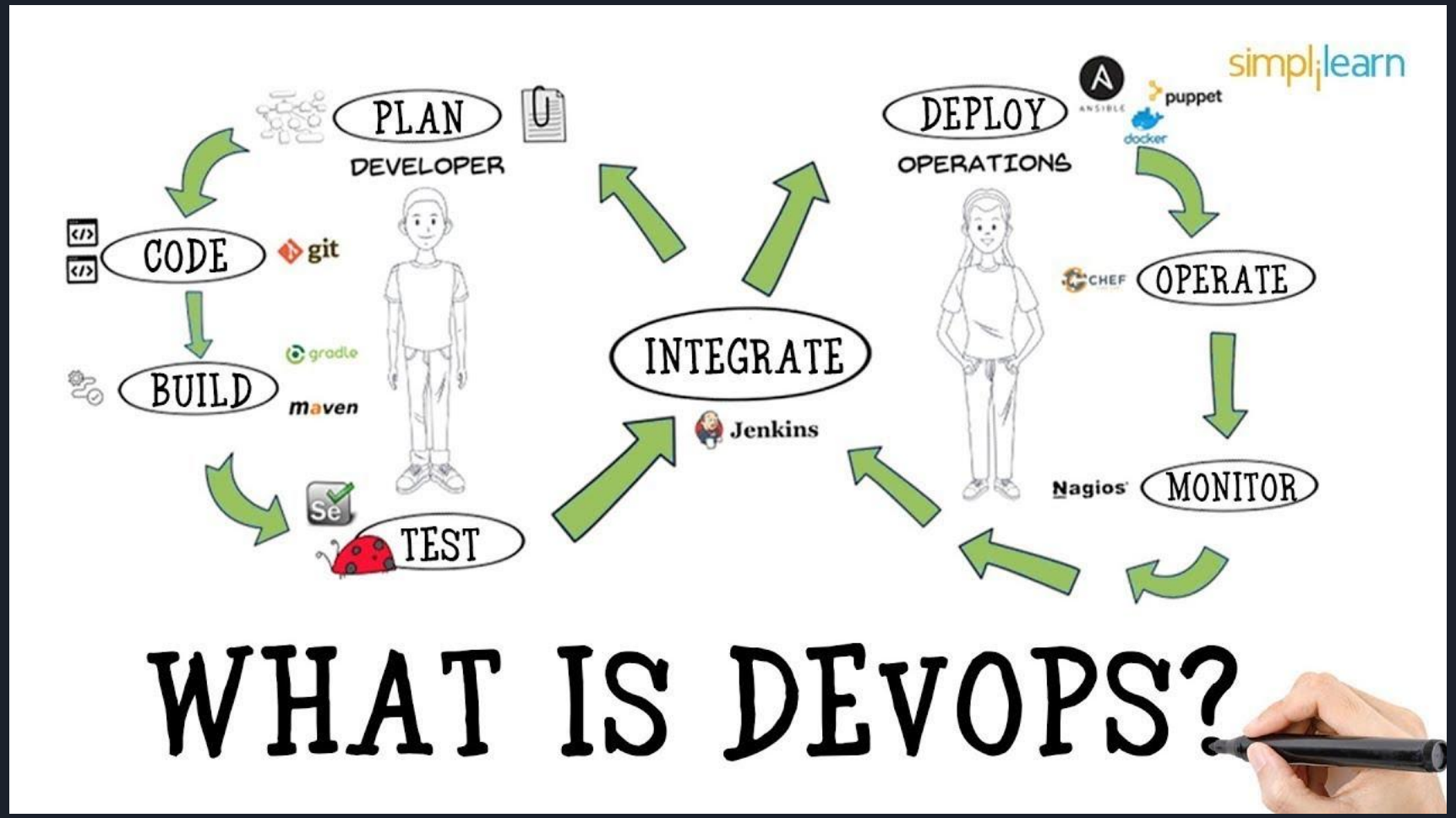
We can use Jenkins to:

- Test code
- Build application
- Package project
- Push package or image to registry



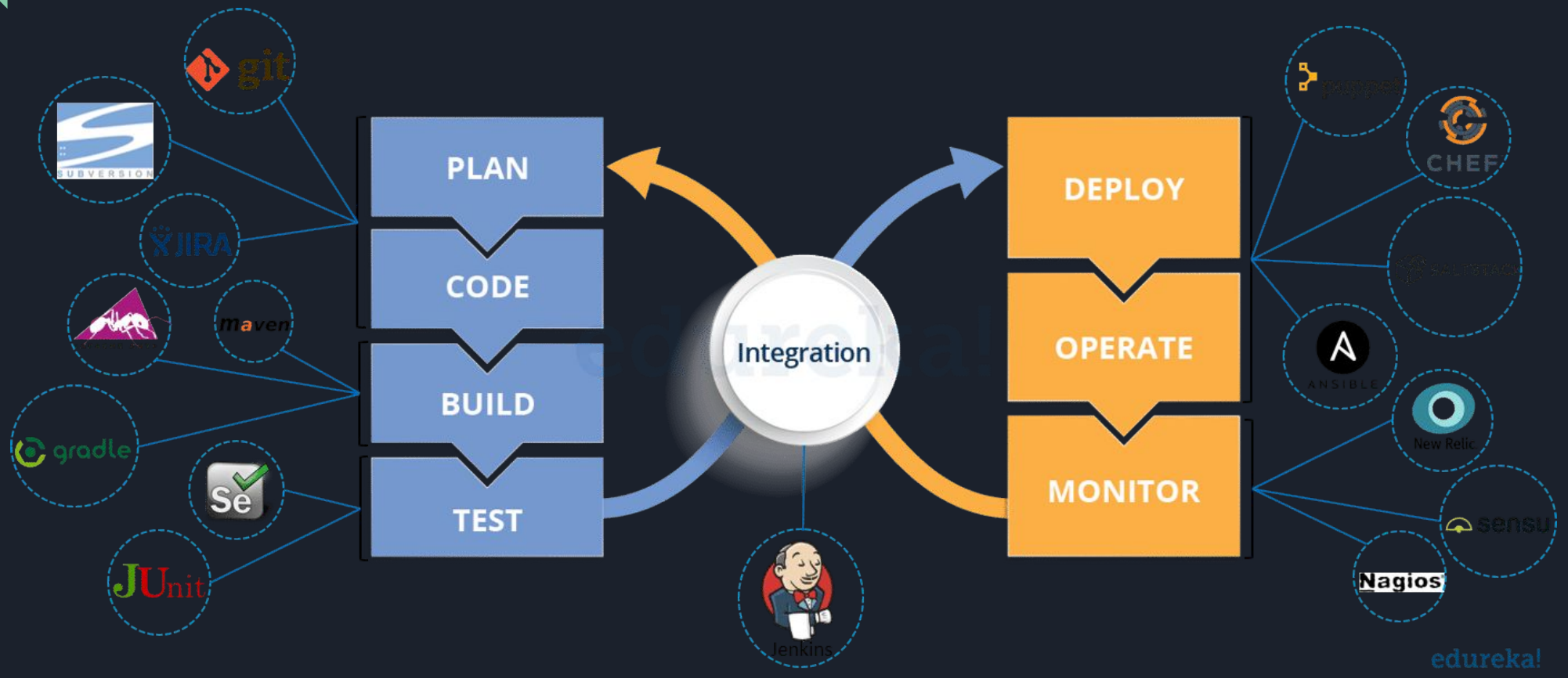
Presenter:
Danarrius Broadway

Jenkins' role in the DevOps Process



Presenter:
Danarrius Broadway

What are benefits of Jenkins ?
(History & Advantages)



Presenter:
Damier Raymond

Implementation of Jenkins

1) Installed a Jenkins server onto AWS EC2
2) SSH into the Jenkins instance to install required tools and dependencies (Git, Maven, etc)
3) Configured Source Code Management to specify Repository URL (GitHub)
4) Used web gui to configure Jenkins and manage its plugins
   - Maven, angular, Sonar Scanner, email notification
   - Create the webhook for Ansible to trigger the Job

5) JenkinsFile (Declarative pipeline)

Once triggered by Ansible, Jenkins will run a Build with a Declarative pipeline specified in a Jenkinsfile

Presenter:
William Chung

Jenkinsfile - Pipeline as Code

Configured with a Declarative pipeline using a **Jenkinsfile**

- Jenkinsfile is commonly written in **Groovy** language.

A Build happens in **"Stages"**, such as 'Testing' or 'Building' or 'Pushing Image'.

Stages happen in **"Steps"**, where shell commands and functions can be executed.

```
stages{
    stage ("initialize") {
        steps {
            sh '''
            echo "PATH = ${PATH}"
            echo "M2_HOME = ${M2_HOME}"
            '''
        }
    }

    stage('install'){
        steps{
            //fixes JDBC driver dependancies prioer to packaging .jar
            sh 'mvn install:install-file -Dfile="./src/main/resources/ojdbc7.jar" -Dgrou
            sh "mvn clean package -DskipTests=true"
        }
    }

    stage('Build the image'){
        steps{
            script{
                sh 'docker build -t ${Register}:latest --build-arg JAR_FILE=${Service}-
            }
        }
    }

    stage ('Deploy image to ECR'){
        steps{
            script{
                docker.withRegistry("${ECRRepo}", "${Region}:${RegisterCredential}") {
                    sh 'docker push ${Register}:latest'
                }
            }
        }
    }
```

Presenter:
William Chung

Challenges

- Configuring Plugins
  - Waiting for Credentials
  - ssh the Jenkins instance
  - have the correct plugins for the pipeline to be running

we have 2 separates file configuration

- Jenkinsfile (Java and Angular)

Way we Overcome all of that:

Presenter:
Damier Raymond

Future works

The current Pipeline is compatible with Java/Spring and Angular projects.

We hope to increase this flexibility to support any type of project.

Presenter:
Damier Raymond

SPINNAKER

Spinnaker

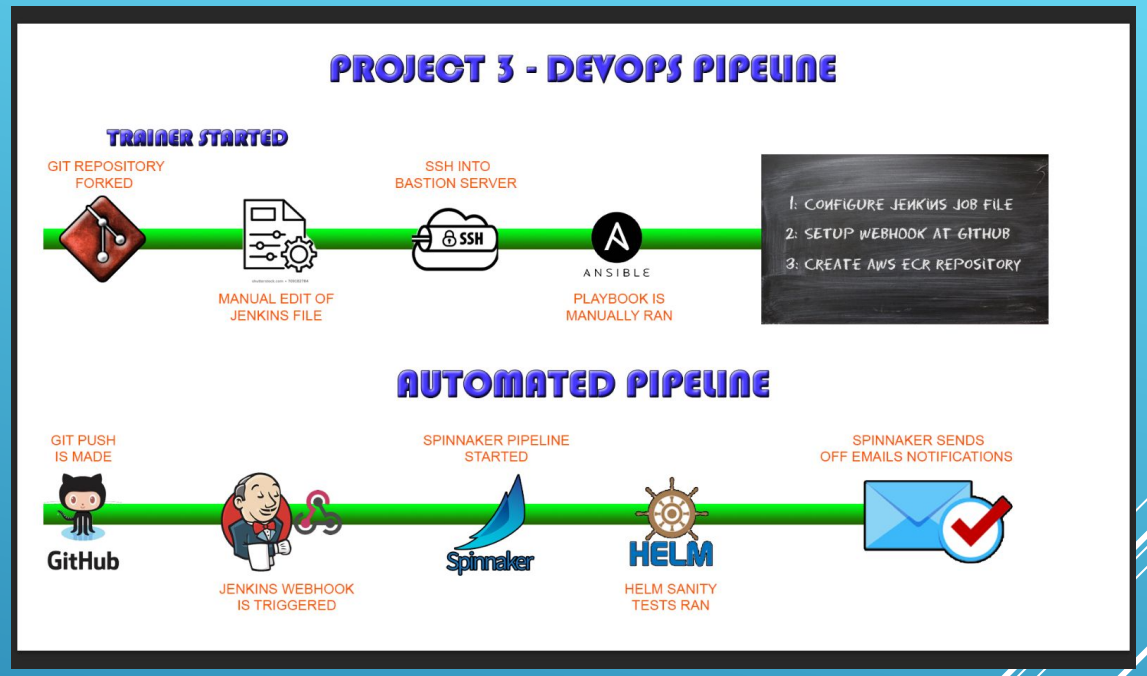
Spinnaker has two core features:

1. Application management
2. Application Deployment

Presenter:
Paityn Maynard

What Were The Goals?

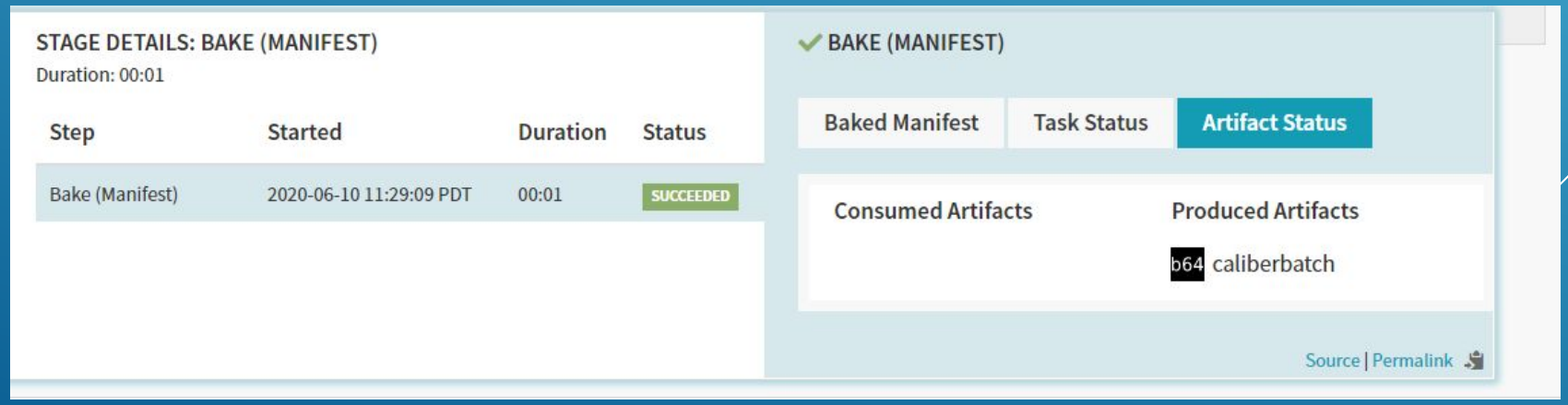
- Pick up pipeline after Jenkins
- To Bake Helm Charts
- To Deploy Artifacts to Kubernetes Cluster



Presenter:
Johnny Sheerin

Bake

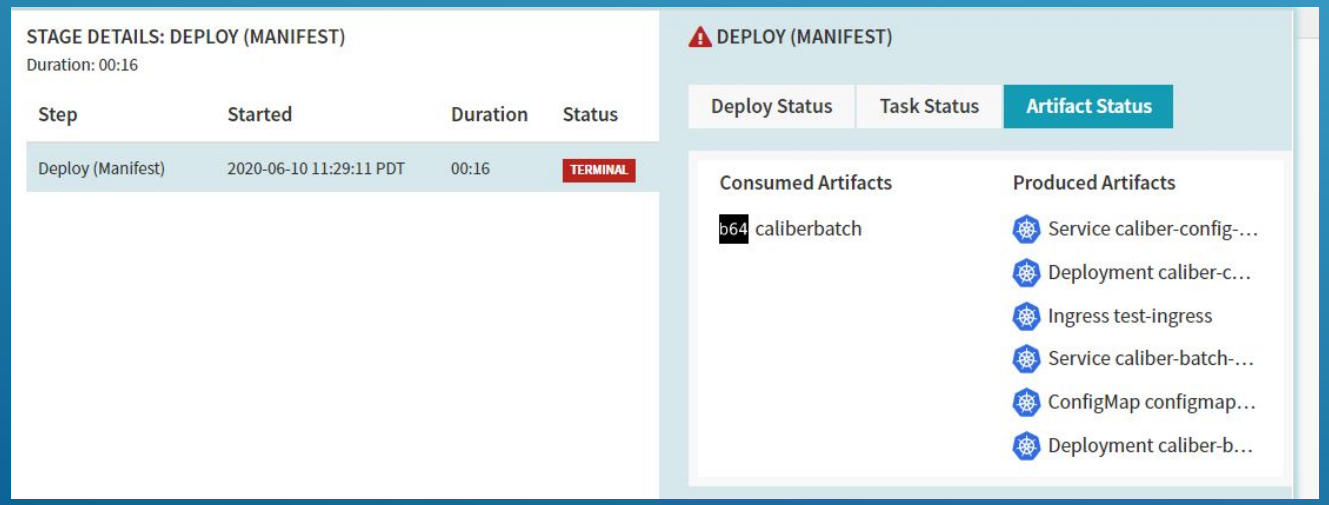
- Pull Helm Charts from S3
- Builds Artifact using Charts
  - JSON object matching specification, not actual contents
  - Reference to resource, not actual resource
- Stage takes in “expected Artifact” (pointers to S3 charts)
- Creates Artifact that encapsulates resource and metadata



Presenter:
Bart Martinon

Deploy

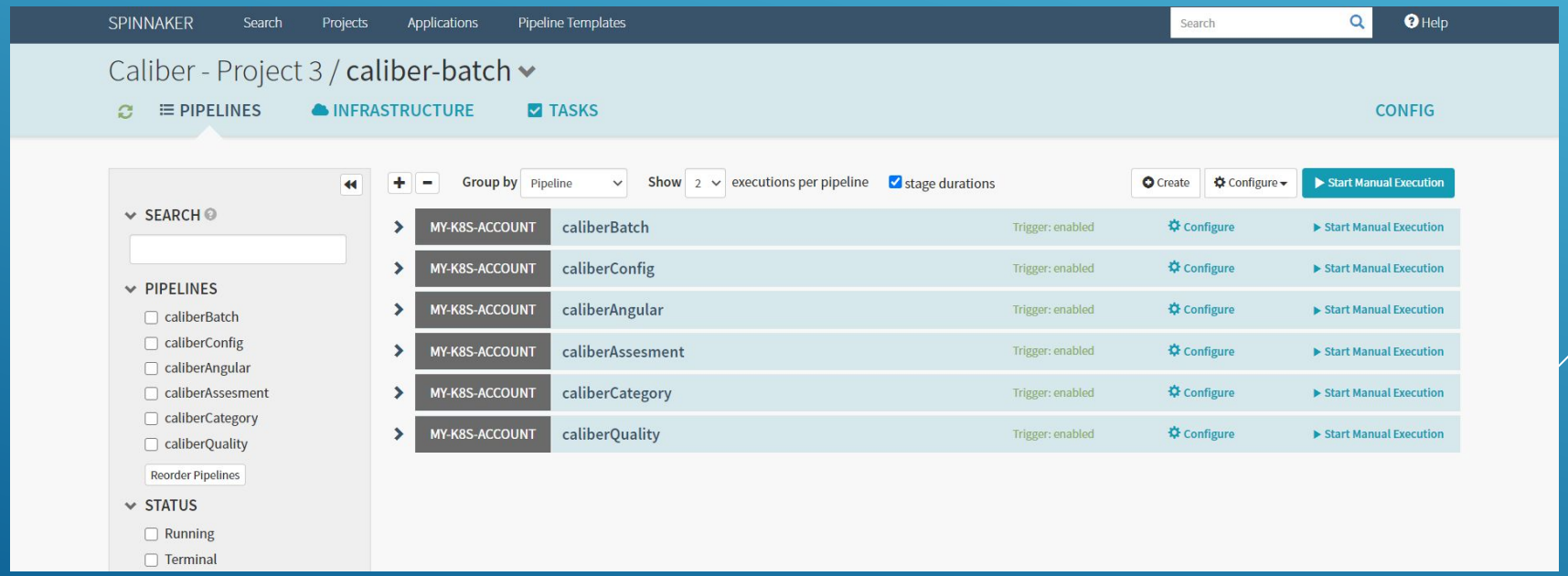
- Takes the the artifacts produced by the bake stage and deploys it to the cluster using helm.
  - Pulls the Helm chart from the AWS S3.
  - Pulls the docker image from the AWS ECR.
  - Deploys the container using the helm chart.



Presenter:
Bart Martinon

Accomplishments

- Spinnaker is installed in the Kubernetes Cluster
- Pipelines are templated and in Spinnaker Application



Presenter:
Jordan Severance

Hurdles

- Waited for user accounts and authentifications
  - Extended R&D
  - Compressed hands on time
- Spinnaker Documentation Not Thorough
- Communication issues between UI (spin-deck) and API (spin-gate)
  - Communication is blocked
    - CORS issue
  - We have been working with assisting cloud specialist Uday and our trainer August on this issue
- Deploy Stage did not act as expected or documented

```
Access to XMLHttpRequest at 'http://spin-gate.revatur (index):1
elabs.com/credentials?expand=true' from origin 'http://spin-dec
k.revaturelabs.com' has been blocked by CORS policy: Response to
preflight request doesn't pass access control check: The value
of the 'Access-Control-Allow-Origin' header in the response must
not be the wildcard '*' when the request's credentials mode is
'include'. The credentials mode of requests initiated by the
XMLHttpRequest is controlled by the withCredentials attribute.
▸GET http://spin-gate.revaturelabs.com/creden  angular.js:12994
tials?expand=true net::ERR_FAILED
Access to XMLHttpRequest at 'http://spin-gate.revatur (index):1
elabs.com/jobs/preconfigured' from origin 'http://spin-deck.reva
turelabs.com' has been blocked by CORS policy: Response to
preflight request doesn't pass access control check: The value
of the 'Access-Control-Allow-Origin' header in the response must
not be the wildcard '*' when the request's credentials mode is
'include'. The credentials mode of requests initiated by the
XMLHttpRequest is controlled by the withCredentials attribute.
▸GET http://spin-gate.revaturelabs.com/jobs/p  angular.js:12994
reconfigured net::ERR_FAILED
```

Presenter:
Jordan Severance

Solution Attempts

- Load Balancers for UI and API
  - Reachable, Console shows error stream, timeout
- Added expected CORS pattern by editing config file
  - Through Halyard
  - Manual addition
  - No change in results
- Made DNS addresses instead of ELB addresses
  - Timeout issue resolves
  - UI reachable but not talking to API

Presenter:
Dylan Chhin

Temporary Working Solution

- Use default of directing UI to localhost:9000 and API to localhost:8084 on pod
- Use kubectl port forwarding to route traffic to ports 9000 and 8084 on the pod from those ports on the local machine
- Not a permanent solution
  - unstable
  - not extensible

Presenter:
Perry Lee

Future Work

- Get the UI exposed more effectively
  - Add authentication to mitigate CORS problems?
- Incorporate Spinnaker throughout the pipeline
  - Trigger on github
  - Run jenkins jobs
- Automate process of creating Spinnaker pipeline
  - Dinghyfile
  - Spin cli
- Configure notifications to email address

Presenter:
Shawyn Kane

KUBERNETES & HELM

Presenter:
Rayan Vakil

Kubernetes

- Orchestrate docker container services
  - Containers consist of all the components a program or process needs to run.
  - A pod is a structure containing one or more containers
- Replicates the pods to allow load balancing and build failure resistance.
- Spinnaker also resides in the cluster
- Features:
  - Automating manual processes
  - Self-healing
  - Scaling
  - Storage orchestration
  - Automated rollout and rollback
- Container tech to achieve end-to-end automation, ensuring CD.

Presenter:
Rayan Vakil

Helm

- Package, configure, and deploy applications and services onto Kubernetes clusters.
  - Install software.
  - Automatically install software dependencies.
  - Upgrade software.
  - Configure software deployments.
  - Fetch software packages from repositories.
- Helm provides this functionality through the following components:
  - Helm
  - Tiller
  - Charts
  - Helm hub

Presenter:
Rayan Vakil

Implementation

- Created Docker Images
- Wrote deployment and service YAML files
- Wrote an Ingress YAML file
- Created Helm chart and stored appropriate YAML files
- Tested Helm chart

Presenter:
Joshua Ragasa

Challenges with Caliber

Caliber

- Lack of documentation
- Application tests were often unrunnable

Adapting to Spring

- Minimal exposure to Spring
- Extra time spent on understanding and researching

Presenter:
Joshua Ragasa

Challenges with Helm and Kubernetes

- Learning basic GO
- Add dynamic functionality within the YAML files
- Navigating the Helm and Kubernetes documentation for setting up a deployment
- Port mapping configurations
- Setting up environment variables

Presenter:
Joshua Ragasa

Helm Hurdles Overcome

- Implementation of Google's Go within helm charts
  - Enables dynamic creation of charts through the use of variables
- Building templates and charts
  - Aids in the successful deployment of the kubernetes cluster
- Manipulate the YAML to match external configurations
  - Files needed to match the configuration of Caliber

Presenter:
Gregory Cox

What can be done with more time?

- The creation of a bash to be run with Ansible
  - Deployments
  - Ingress
  - Configmaps
- Some additional files
  - Values
  - Secrets

Presenter:
Gregory Cox

LETS SEE A DEMONSTRATION!

Presenter:
Jean Aldoph II