Jenkins Fundamentals

WELCOME!

Develop Intelligence

Axel Sirota
MCT and AI Consultant

Join Us in Making Learning Technology Easier

Develop Intelligence

Our mission...

Over 16 years ago, we embarked on a journey to improve the world by making learning technology easy and accessible to everyone.

...impacts everyone daily.

And it's working. Today, we're known for delivering customized tech learning programs that drive innovation and transform organizations.

In fact, when you talk on the phone, watch a movie, connect with friends on social media, drive a car, fly on a plane, shop online, and order a latte with your mobile app, you are experiencing the impact of our solutions.

Upskilling and Reskilling Offerings

Develop Intelligence

Intimately customized learning experiences just for your teams.

- **Workshop** — 2-3 day upskilling experiences
- **Fast Track** — 5-day reskilling experiences
- **Learning Spike** — 1-day technology overviews
- **Target Topics** — 90-minute instructor-led micro-learnings
- **Hack-a-thon** — Learn and build an MVP in 2-3 days

BACK END DEVELOPMENT

BIG DATA

CLOUD COMPUTING

DEVOPS

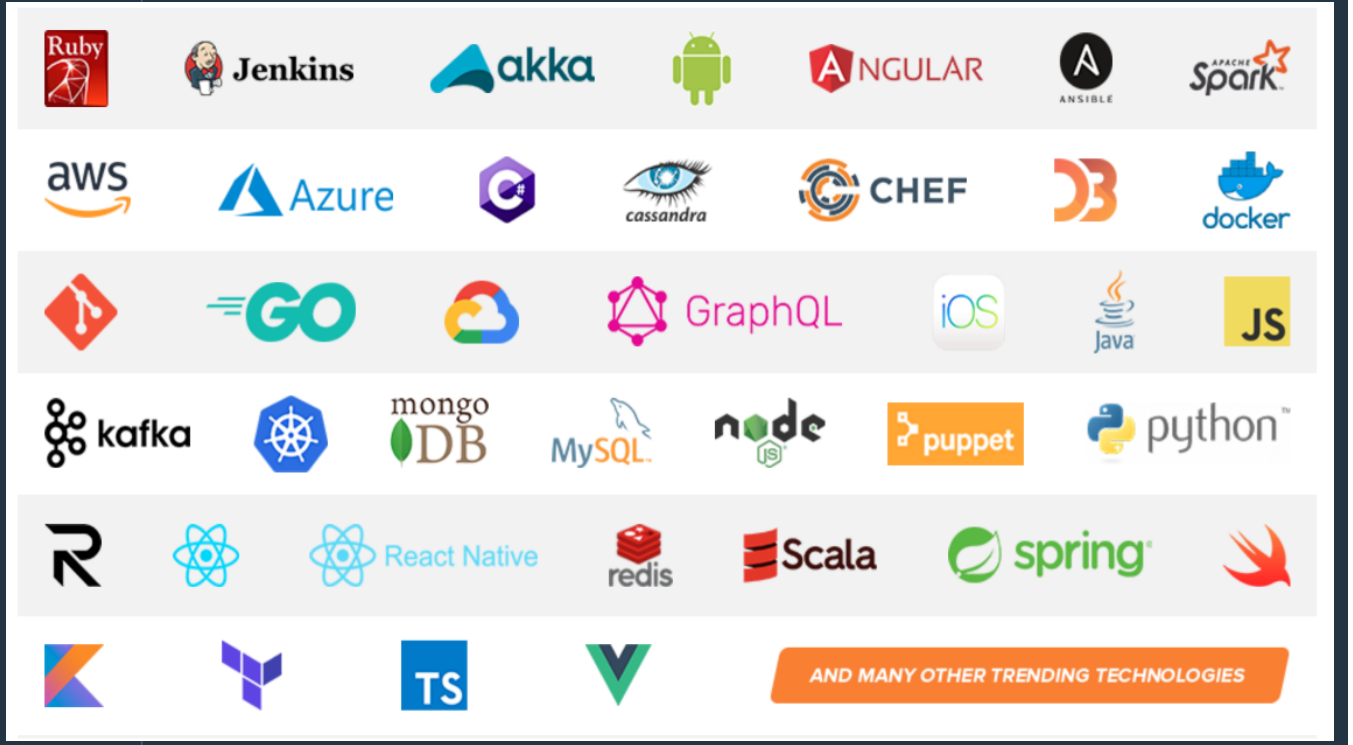
FRONT END DEVELOPMENT

MACHINE LEARNING

MOBILE APP DEVELOPMENT

SOFTWARE ENGINEERING

SYSTEM ADMINISTRATION



World Class Practitioners

Note About Virtual Trainings

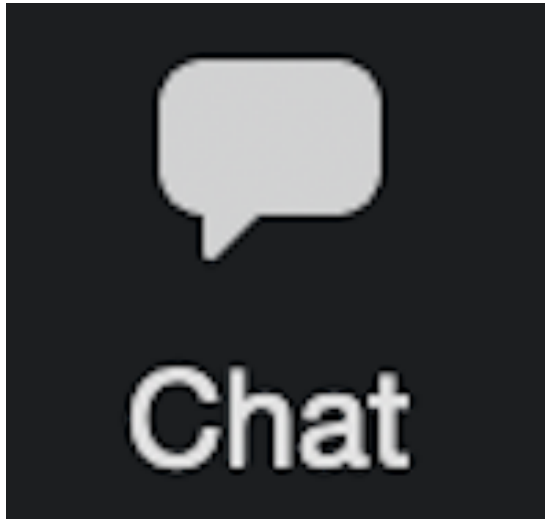
What we want

...what we've got

Virtual Training Expectations for You

Arrive on time / return on time

Mute unless speaking



Use chat or ask questions verbally

Virtual Training Expectations for Me

I pledge to:

- Make this as interesting and interactive as possible
- Ask questions in order to stimulate discussion
- Use whatever resources I have at hand to explain the material
- Try my best to manage verbal responses so that everyone who wants to speak can do so
- Use an on-screen timer for breaks so you know when to be back

Prerequisites

- Development and Testing overview knowledge
- Java (for the code examples and example project)

What this course is about

Develop Intelligence

This course **is about**:

- Learning what CI/CD **is** and **how** to devise a plan to **implement it**
- Learning to **develop** Jenkins Jobs that **implement CI/CD pipelines**
- Searching for Plugins to **extend functionality** in our pipeline

This course **is not about:**

- Advanced Multi-Node Jenkins Management
- Java Programming

Objectives

Develop Intelligence

At the end of this course you will be able to:

- Understand what CI/CD **is** and how it can help an organisation
- Devise a plan to implement CI/CD with Jenkins
- Develop Jenkins jobs to implement CI/CD Pipelines
- Measure Code quality and release gates
- Setup notifications for any pipeline

Structure of the Course / Course Takeaways

- We will be using a Github repository for labs throughout the training
- The labs will be on your own laptops
- Download the code and instructions for labs here: https://github.com/axel-sirota/jenkins-course
- The slides were sent by mail

Agenda

Develop Intelligence

Introduction

- Introductions
- Expectations
- Why CI/CD?

First Steps

- Freestyle Jobs
- Configuring a CI job

Advanced Pipelines

- Code quality
- Notifications
- Parallel

Jenkins

- Why Jenkins?
- Architecture of Jenkins
- Setting Up

Jenkins Pipelines

- Scripted vs Declarative
- Jenkinsfile
- Developing Pipelines

Agenda

Develop Intelligence

Introduction

- Introductions
- Expectations
- Why CI/CD?

First Steps

- Freestyle Jobs
- Configuring a CI job

Advanced Pipelines

- Code quality
- Notifications
- Parallel

Jenkins

- Why Jenkins?
- Architecture of Jenkins
- Setting Up

Jenkins Pipelines

- Scripted vs Declarative
- Jenkinsfile
- Developing Pipelines

Introduction

Develop Intelligence

Who are we?

Introductions

- One by one introduce yourself:
  A. Name
  B. Role
  C. Previous experience with Jenkins or CI/CD
  D. What do you want to learn from this class?
  E. Do you want to share a fun fact of yourself?

Who am I?

Develop Intelligence

– Microsoft Certified Trainer

– Author, Instructor, and Editor at Pluralsight, Develop Intelligence, and O'Reilly Media

– AI and Cloud Consultant

QR to my Pluralsight courses

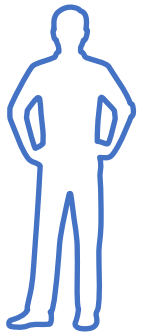
QR to my O'Reilly trainings

Introduction

Develop Intelligence

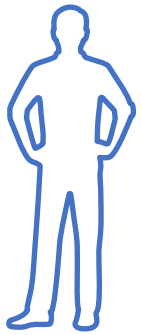
Why CI/CD?

Case Study: E-Commerce

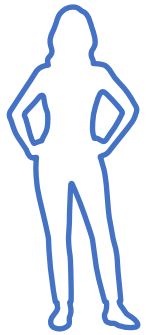


Product Managers

Case Study: E-Commerce

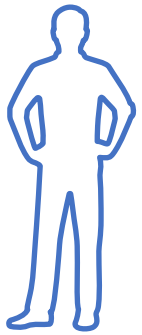


Product Managers

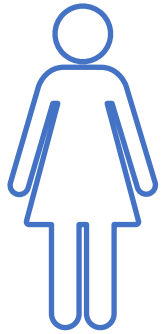


Software Development

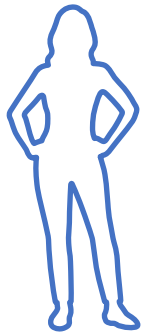
Case Study: E-Commerce



Product Managers

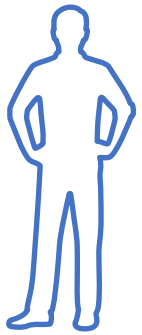


Integration & Operations

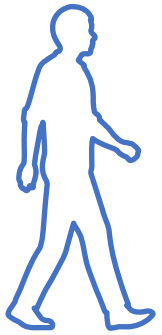


Software Development

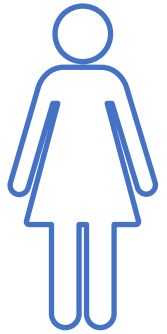
Case Study: E-Commerce



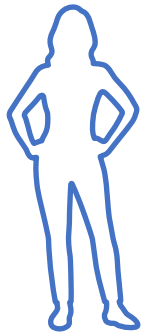
Product Managers



Quality Assurance

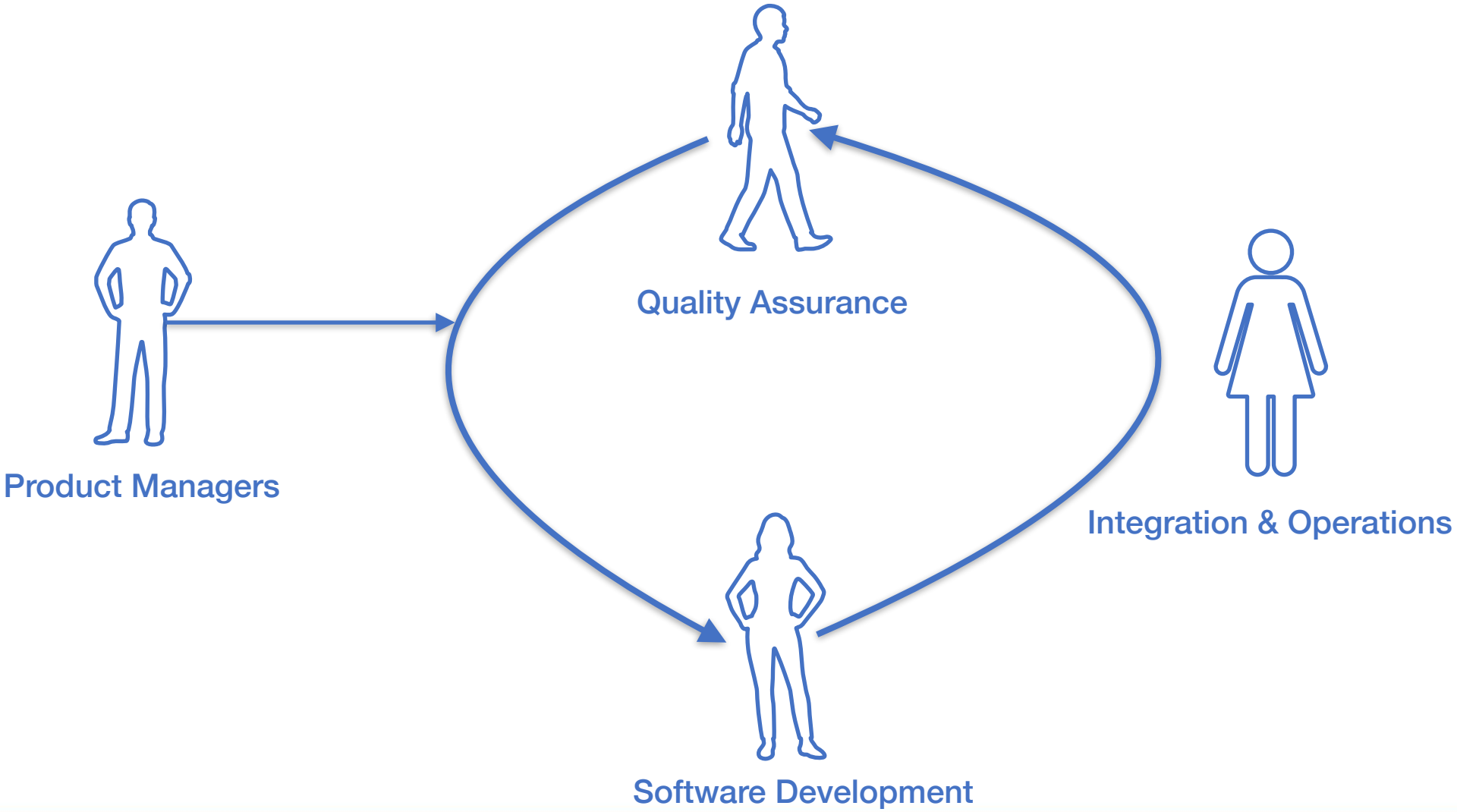


Integration & Operations

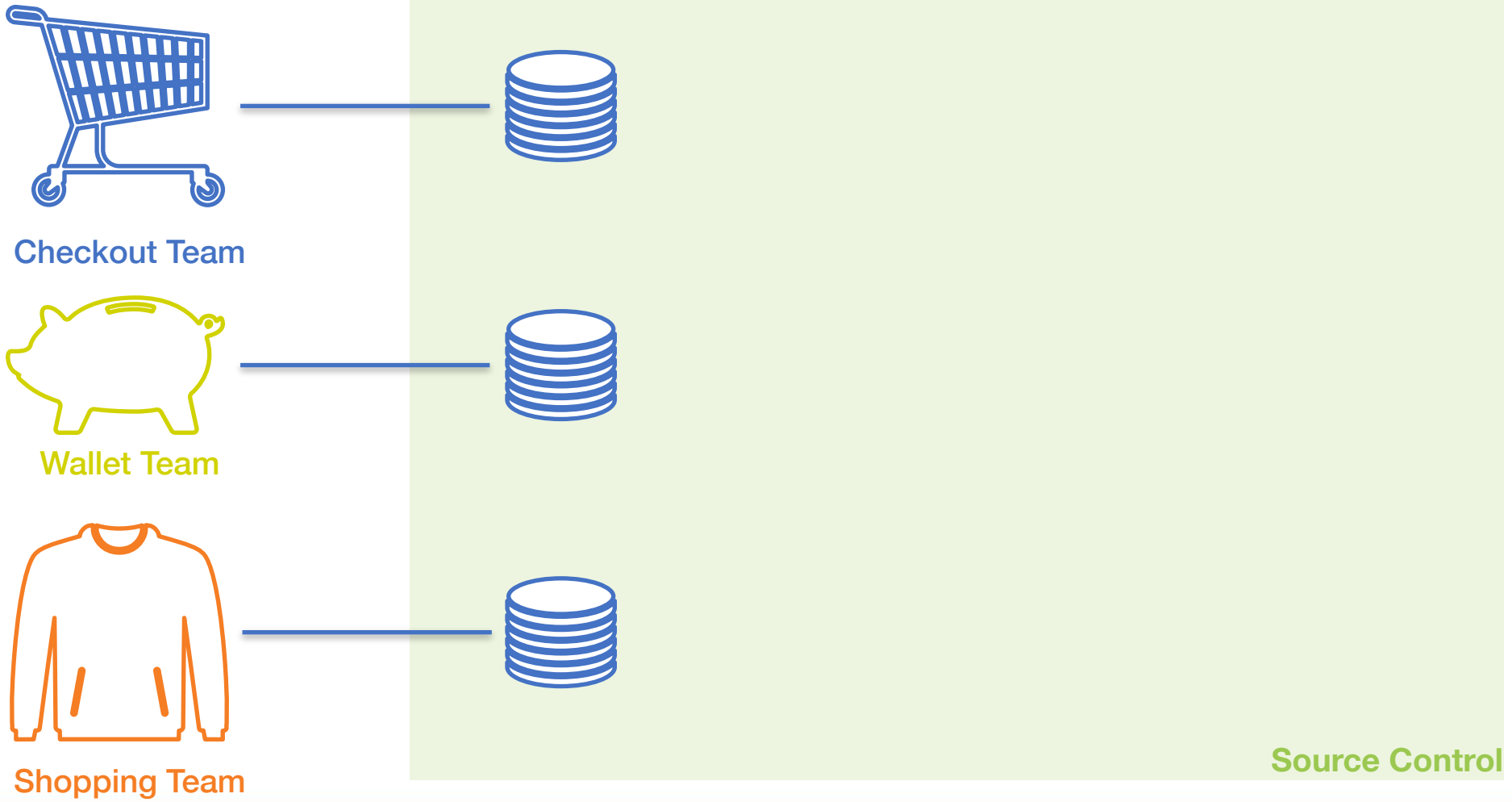


Software Development

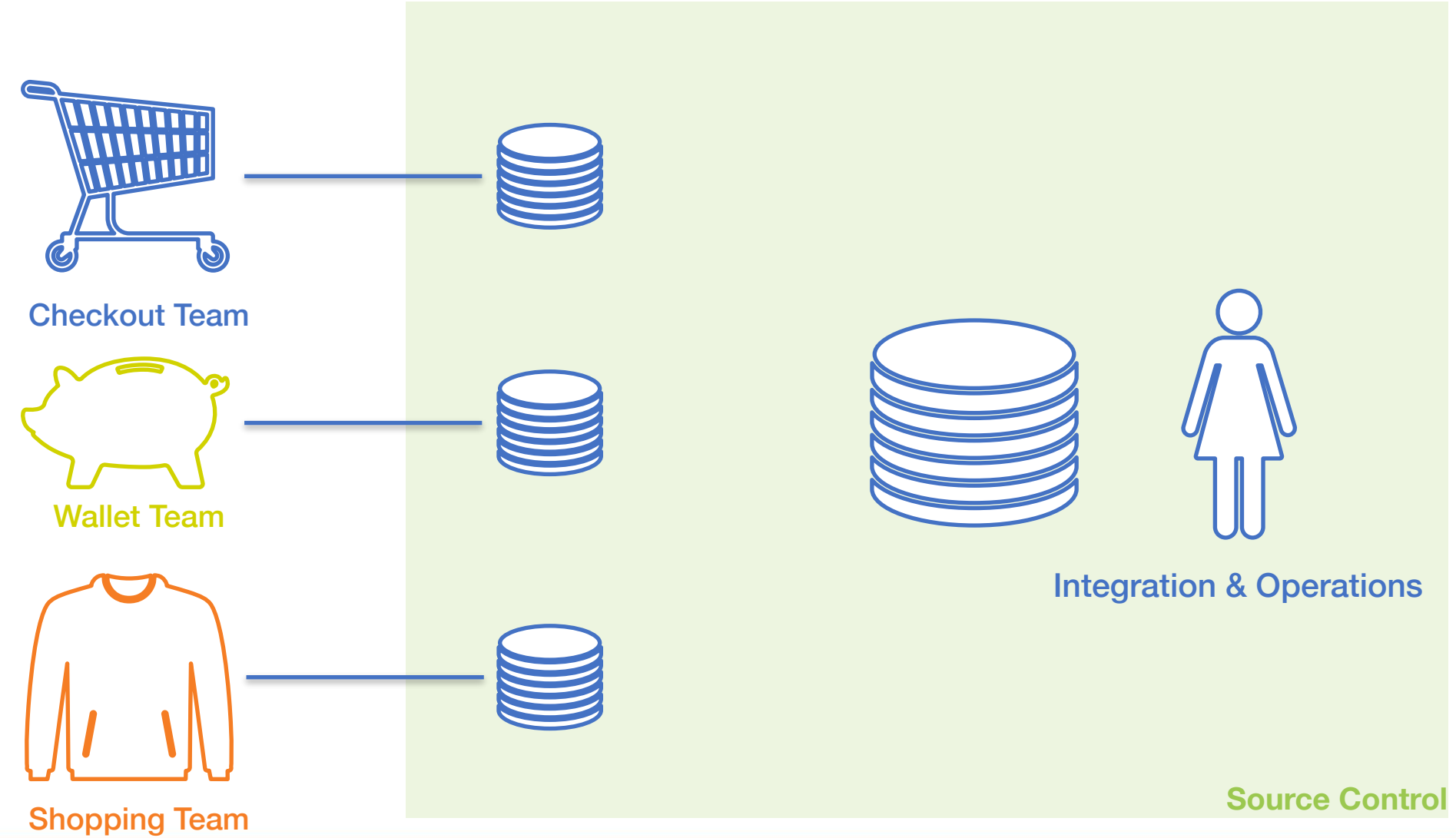
Case Study: E-Commerce



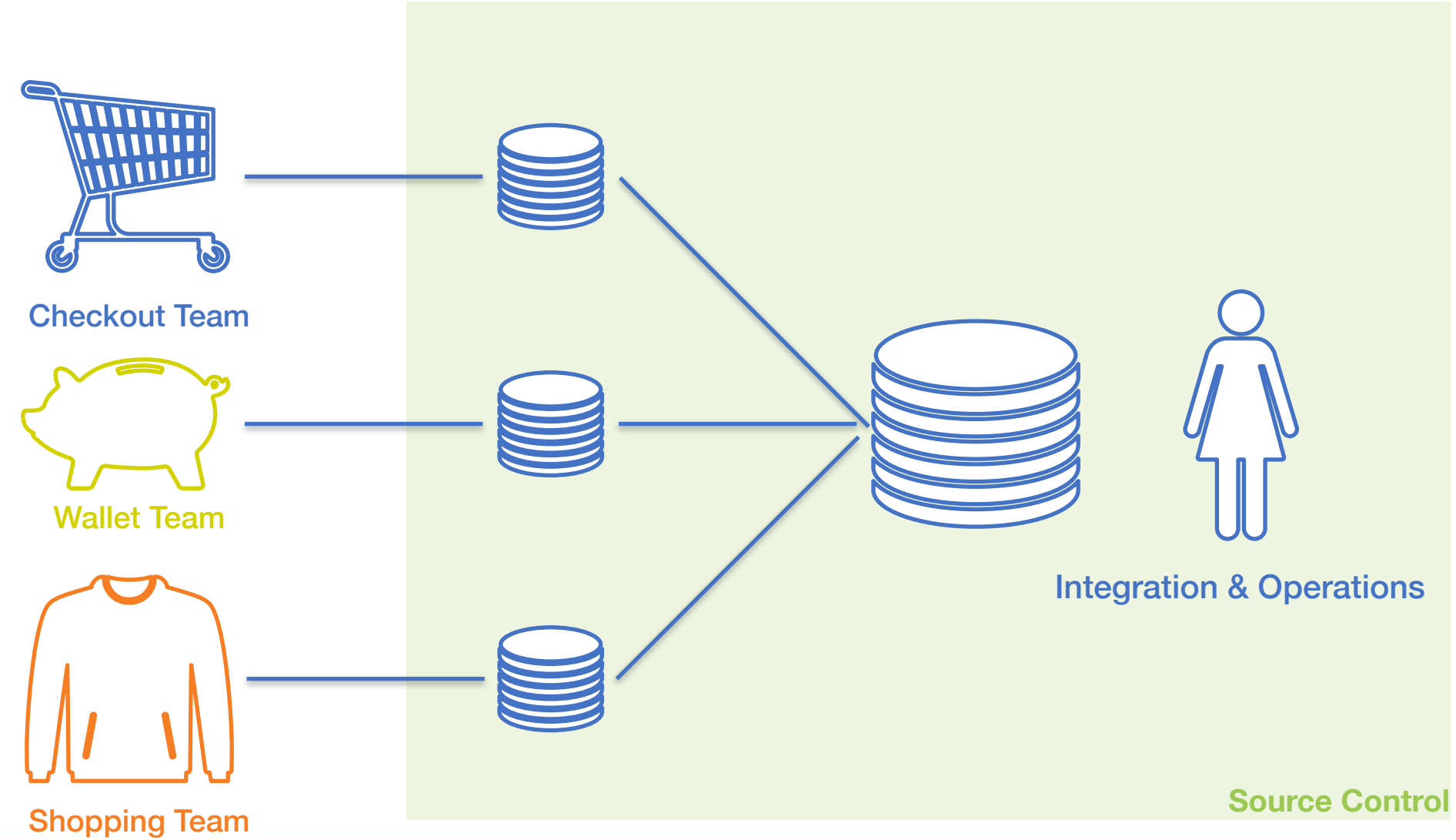
Case Study: E-Commerce



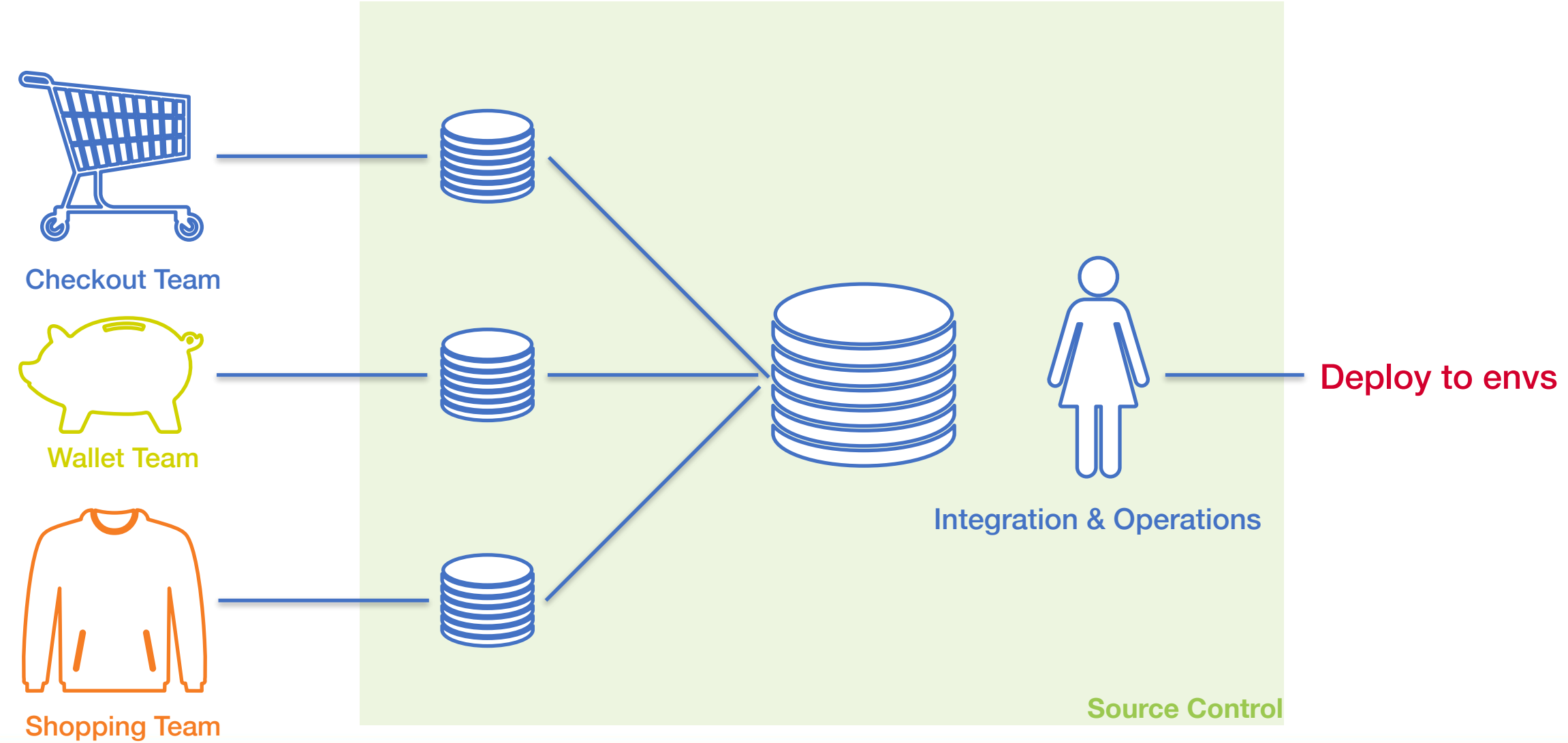
Case Study: E-Commerce



Case Study: E-Commerce



Case Study: E-Commerce



Case Study: E-Commerce

Develop Intelligence

Pain Points

Integration

Painful merge (“integration”) per cycle

Possible bugs caught late

Not reproducible compile (“build”) process to create release

Case Study: E-Commerce

Develop Intelligence

Pain Points

Integration

Painful merge (“integration”) per cycle

Possible bugs caught late

Not reproducible compile (“build”) process to create release

Deployment

Possible wrong Instructions

Not reproducible Environments

Q&A enters late after deployment, reporting bugs at end of cycle

Pain Points = Errors, Not Happy Customers and Teams

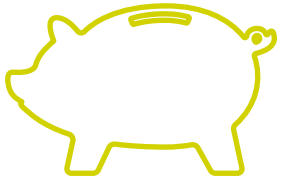
Case Study: E-Commerce

Develop Intelligence

Continuous Integration and Continuous Deployment come to solve this!

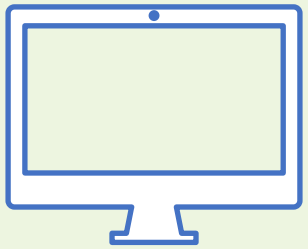
Case Study: E-Commerce

Checkout Team



Wallet Team

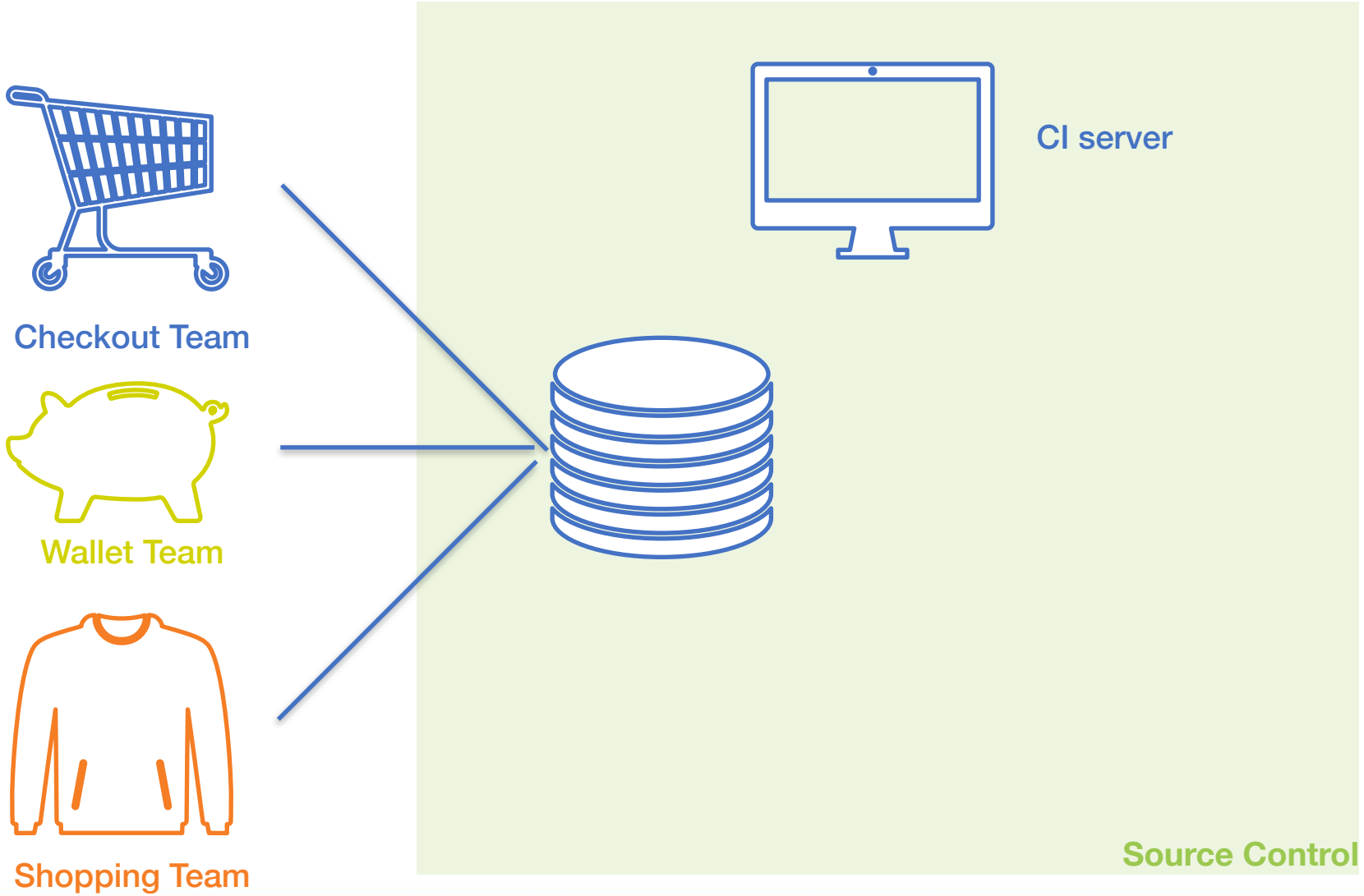
Shopping Team



CI server

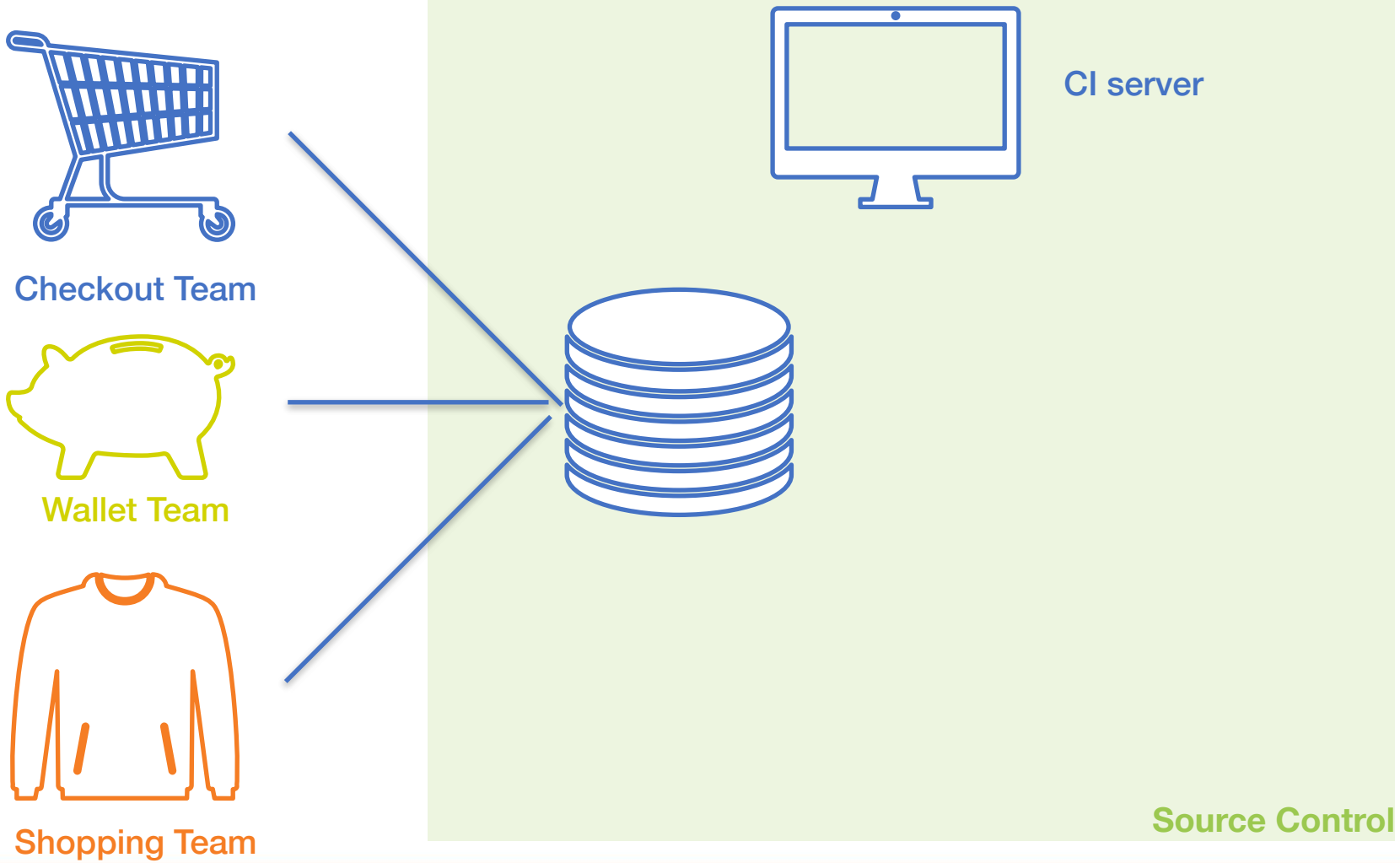
Source Control

Case Study: E-Commerce



On every change one integrates to the main line and run tests to verify work

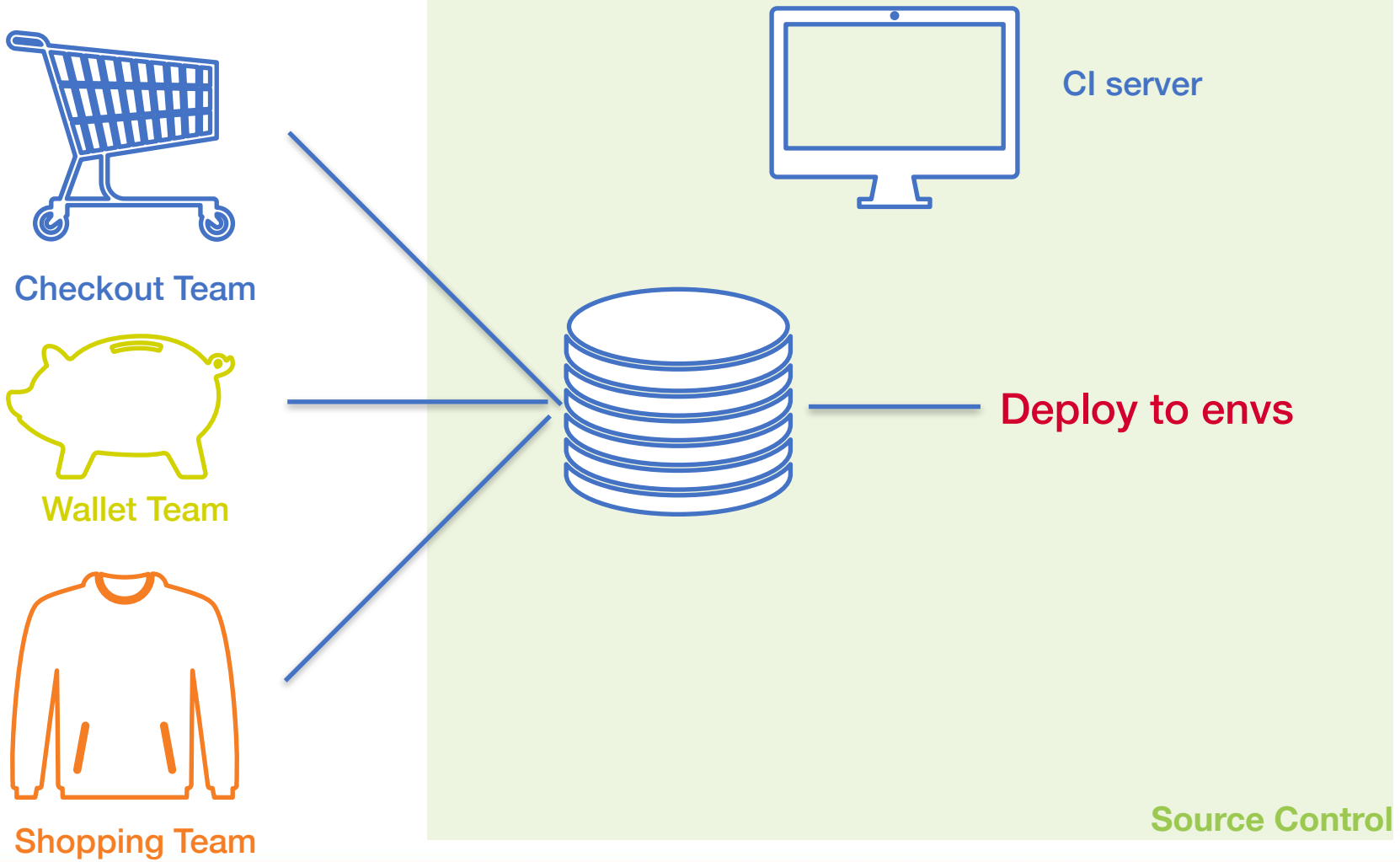
Case Study: E-Commerce



On every change one integrates to the main line and run tests to verify work

Every commit leaves the main line deployable

Case Study: E-Commerce



On every change one integrates to the main line and run tests to verify work

Every commit leaves the main line deployable

Deployment is run through automation

CI principles

- Have a single place where the code lives
- Commit everyday
- Automate the build and test process
- Maintain the build healthy
- Commit triggers Build
- Visibility

CD principles

Develop Intelligence

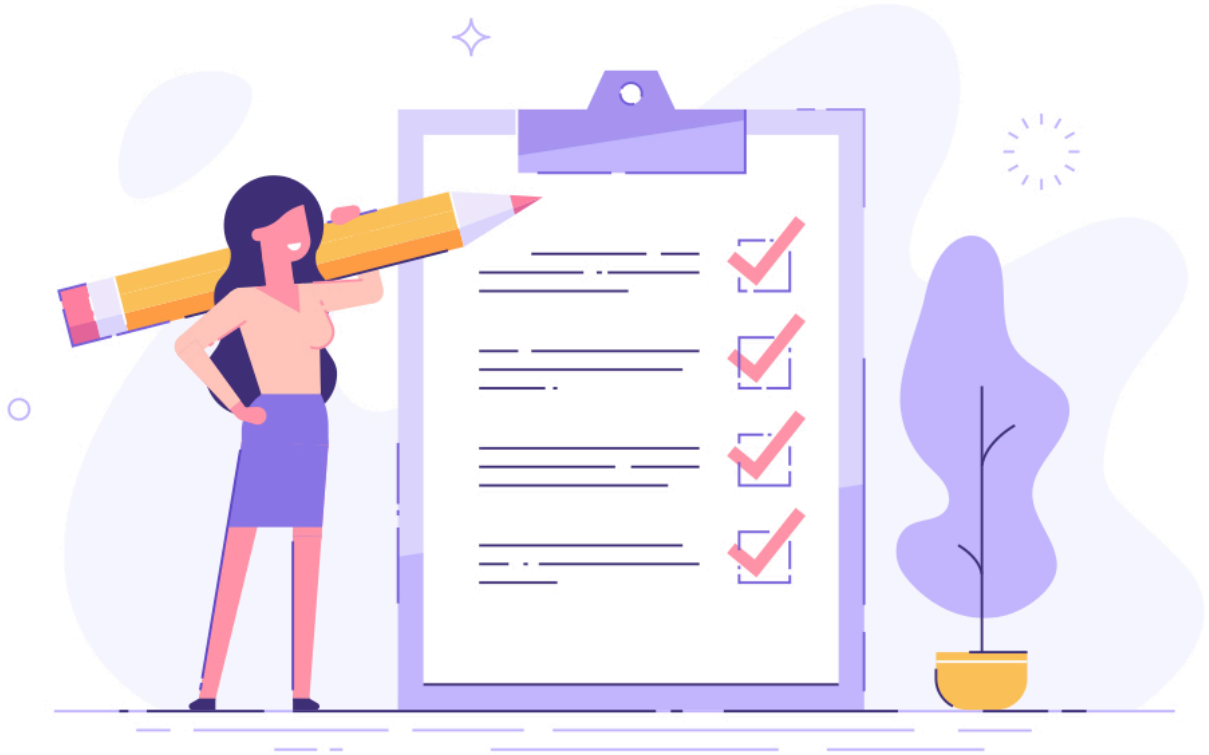
- Be able to deploy through an automated job at every commit
- Treat infrastructure as code artifacts
- Automate environment creation
- Create acceptance tests and manual/auto deployment approvals
- Release should be on-demand
- Visibility

Exercise Time

Develop Intelligence

Map in a diagram where the following live in a CI/CD process

- Build scripts
- Unit Test
- Integration Tests
- Performance / Security Tests
- Deployment Scripts
- Environment creation code



Bonus: How many environments should we need? When do they take place in this process?

Discussion Time

Develop Intelligence

Do you have CI or CD in your organisation?

What would it look like?

What does CD stand for in your case?

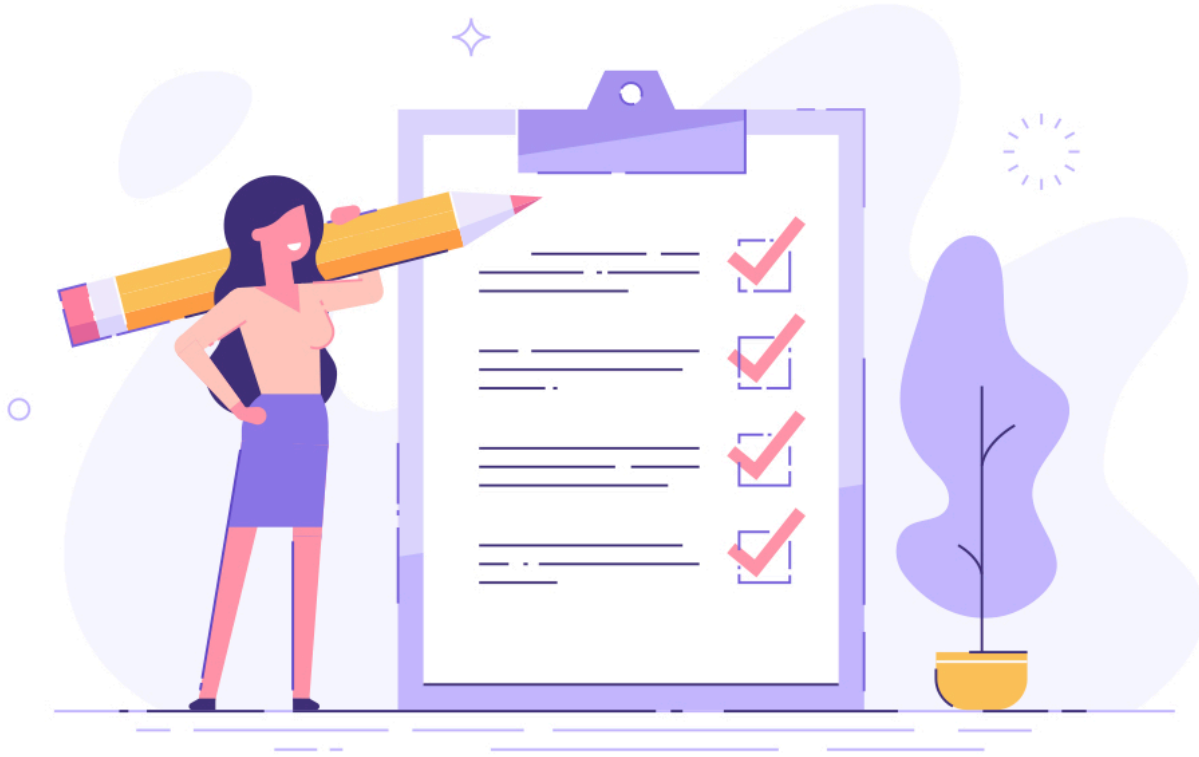
Allocated time: 15 minutes

Summary: What do we gain with CI/CD?

Develop Intelligence

- Higher quality software: Because tests are run rapidly and code integrated often
- Faster delivery: Because an automated process can run to deploy, and each integration and release is quicker and simpler
- Lower costs: Because teams spend less time integrating and deploying
- More Flexibility: To deploy whatever version and whatever number of versions on every cycle.

Pulse Check



Break: 15 minutes

Develop Intelligence

Jenkins

Develop Intelligence

Why Jenkins?

Agenda

Develop Intelligence

Introduction

- Introductions
- Expectations
- Why CI/CD?

First Steps

- Freestyle Jobs
- Configuring a CI job

Advanced Pipelines

- Code quality
- Notifications
- Parallel

Jenkins

- Why Jenkins?
- Architecture of Jenkins
- Setting Up

Jenkins Pipelines

- Scripted vs Declarative
- Jenkinsfile
- Developing Pipelines

Best Practices

- Scaling Jenkins
- Jenkins Deployments
- Security

Jenkins

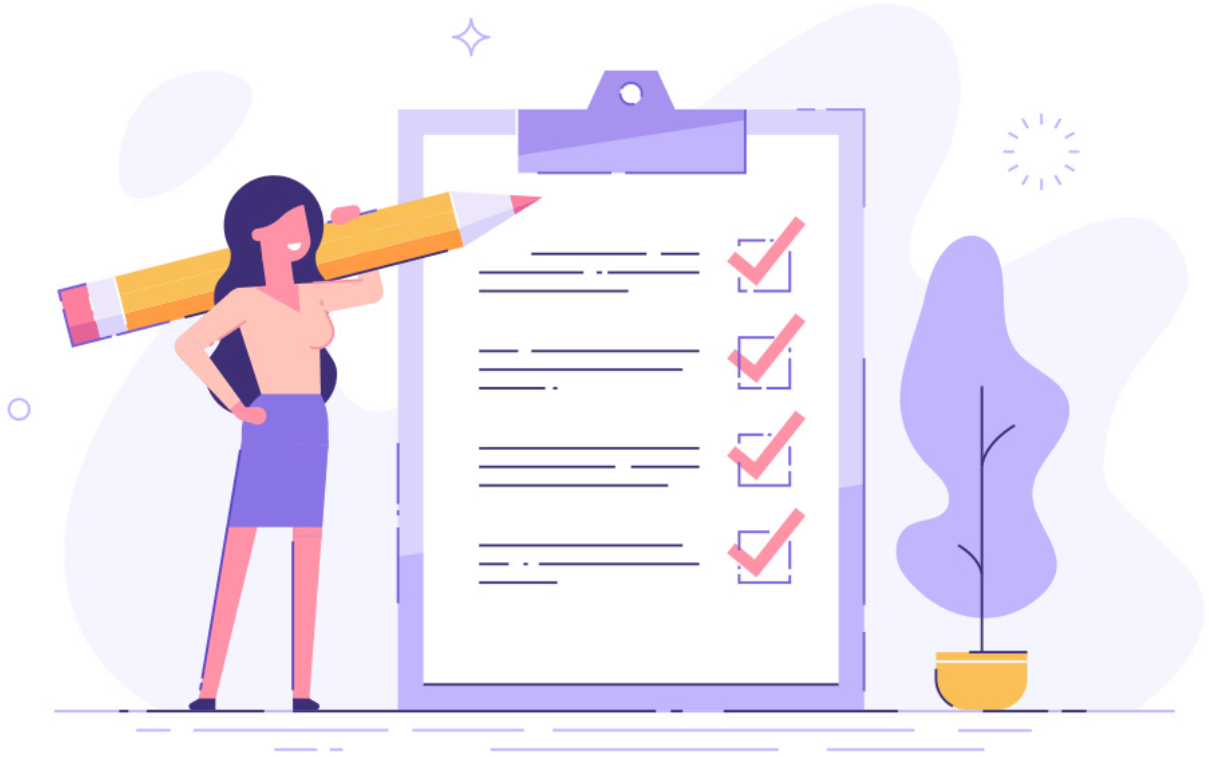
- CI server
- Extremely configurable and powerful
- Distributed and Battle tested

Jenkins Features

- Amazing community and plugins available
- Highly distributed
- Easy installation and easy to start going
- Extensible

LAB: Installing Jenkins

- Installing Jenkins both as WAR as well as Docker
- Clone the project to work on



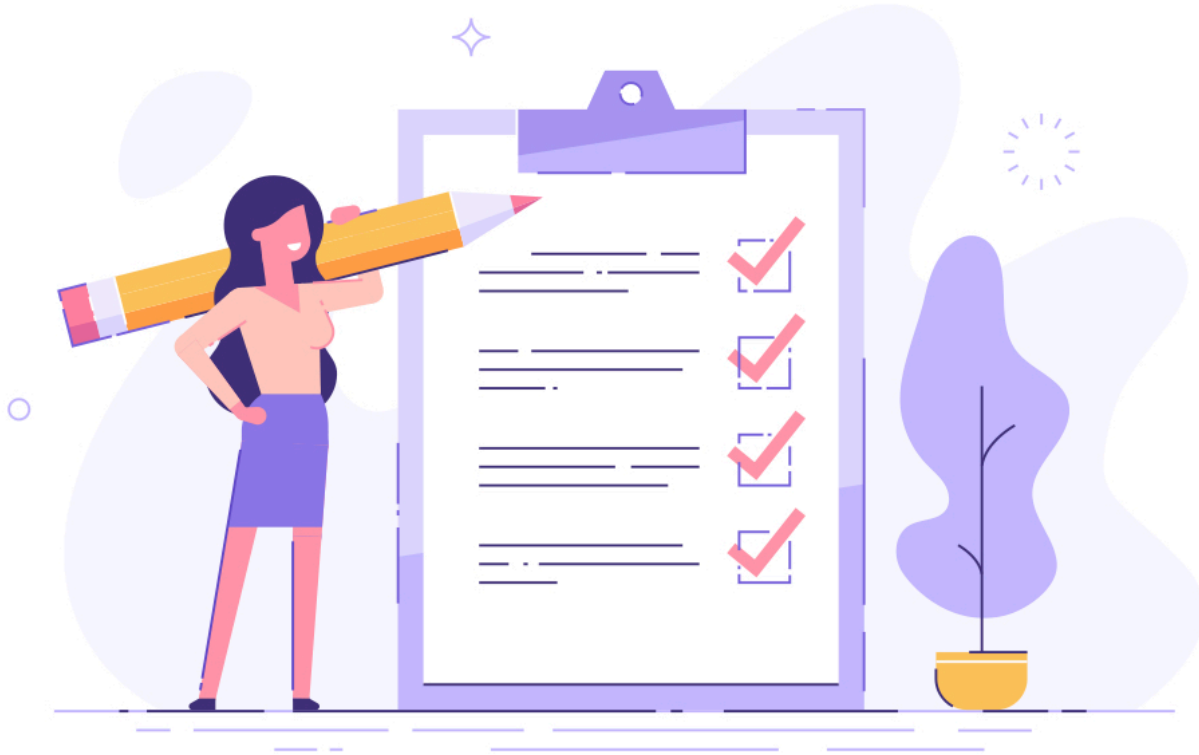
Allocated time: 20 minutes

(depending on troubleshooting!)

LAB: Creating out first Freestyle Job

- Create a Freestyle job that outputs “Hello Jenkins”

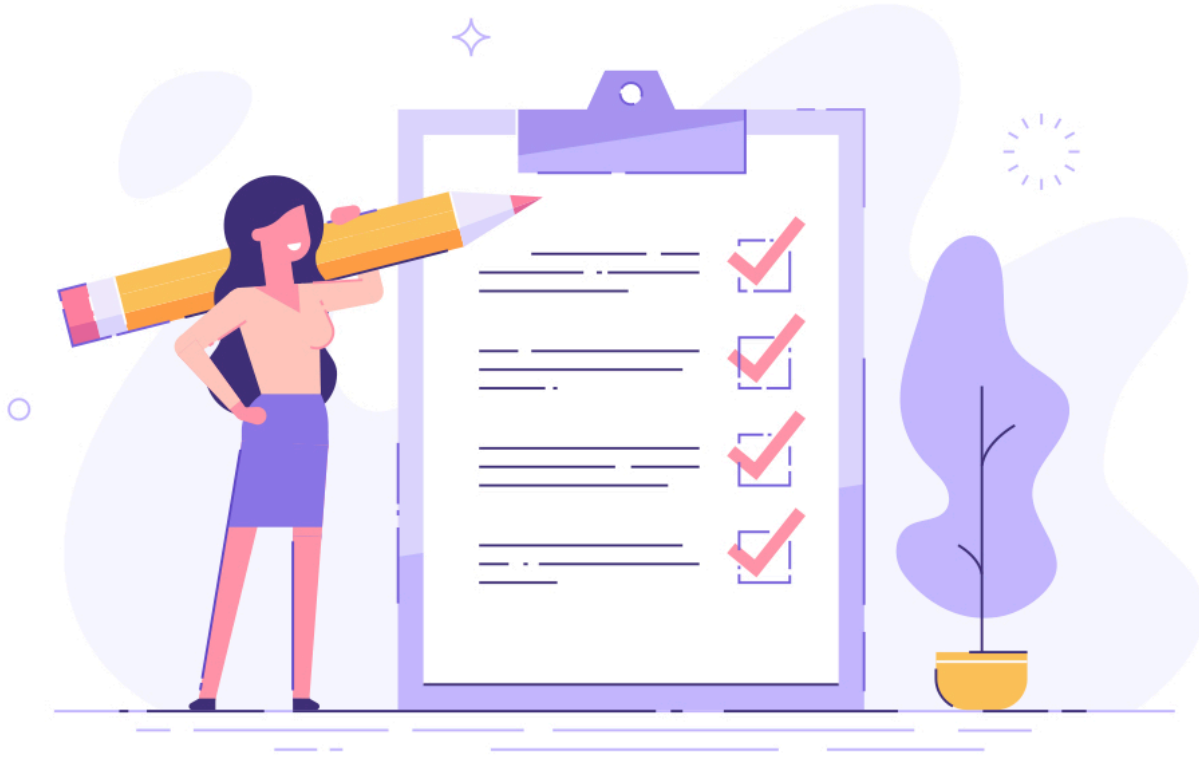
Allocated time: 10 minutes



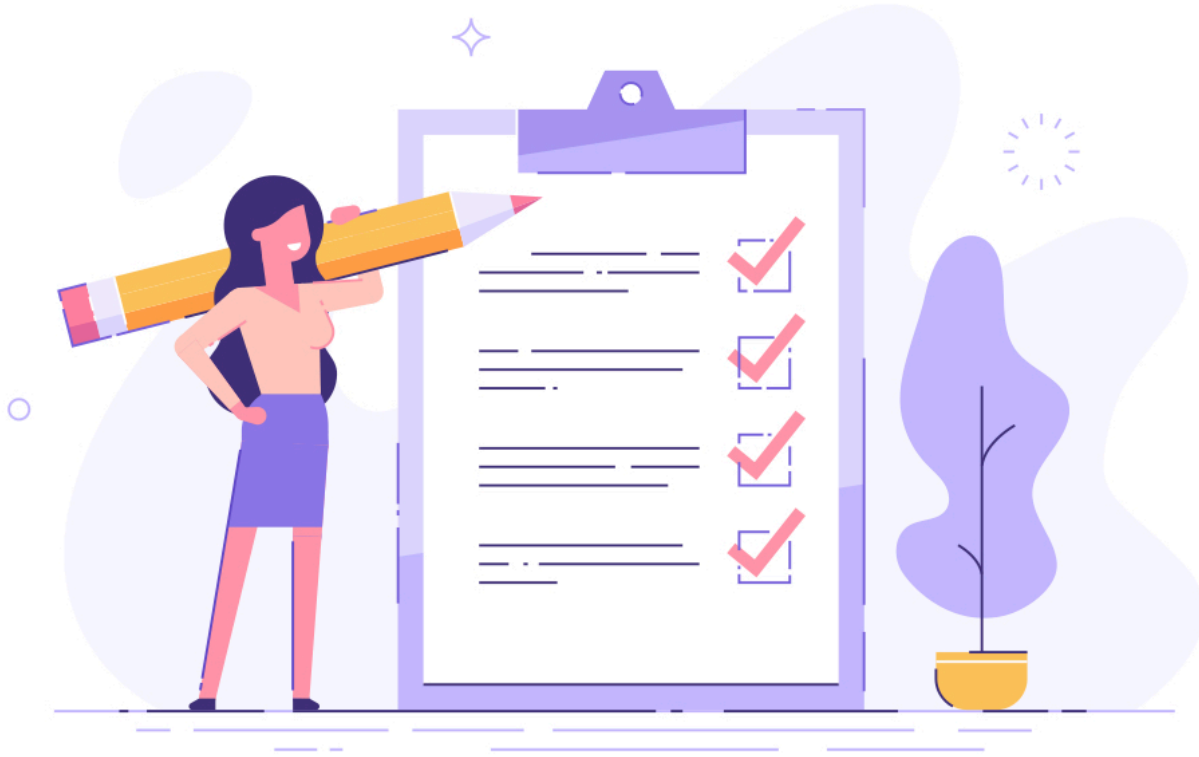
LAB: Creating our second Freestyle Job

- Compile the simple app in the repo in a new Jenkins Job

Allocated time: 15 minutes



Pulse Check

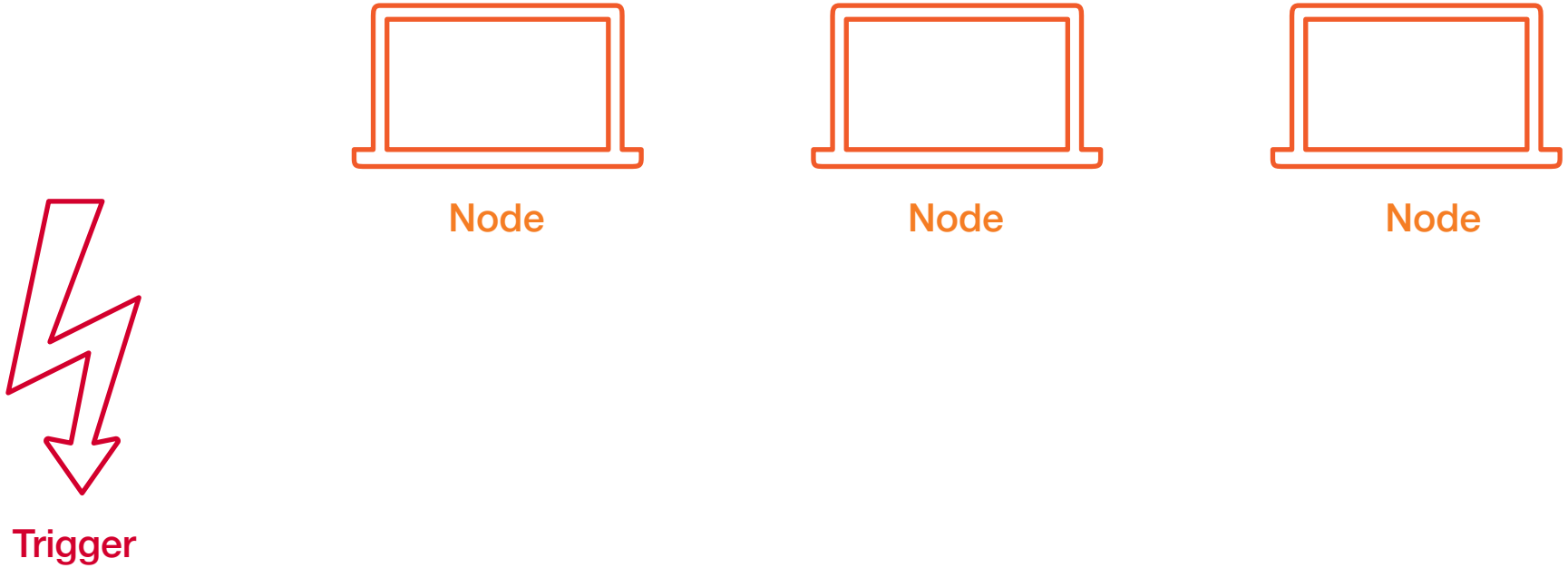


Jenkins

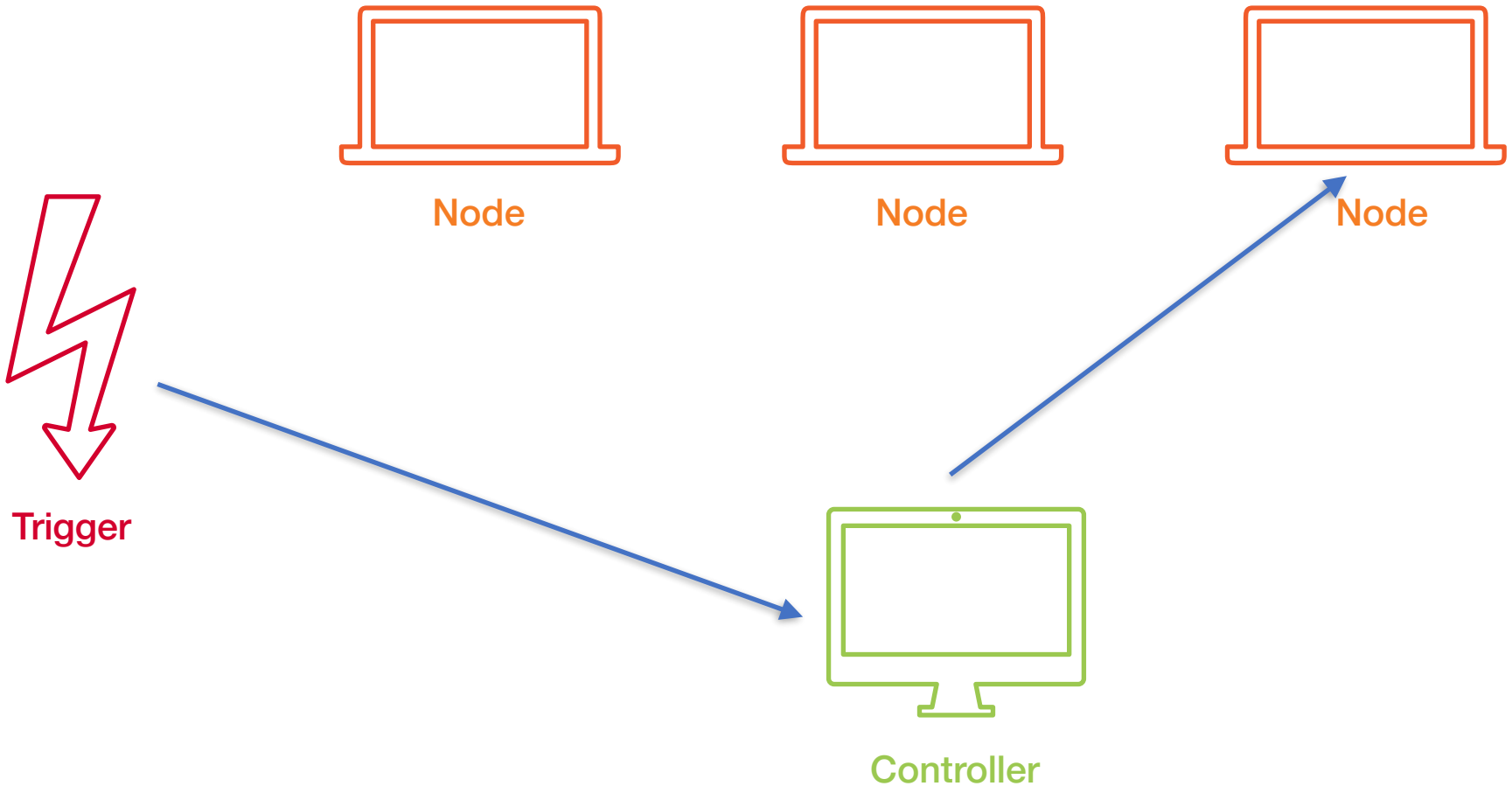
Develop Intelligence

Jenkins Architecture

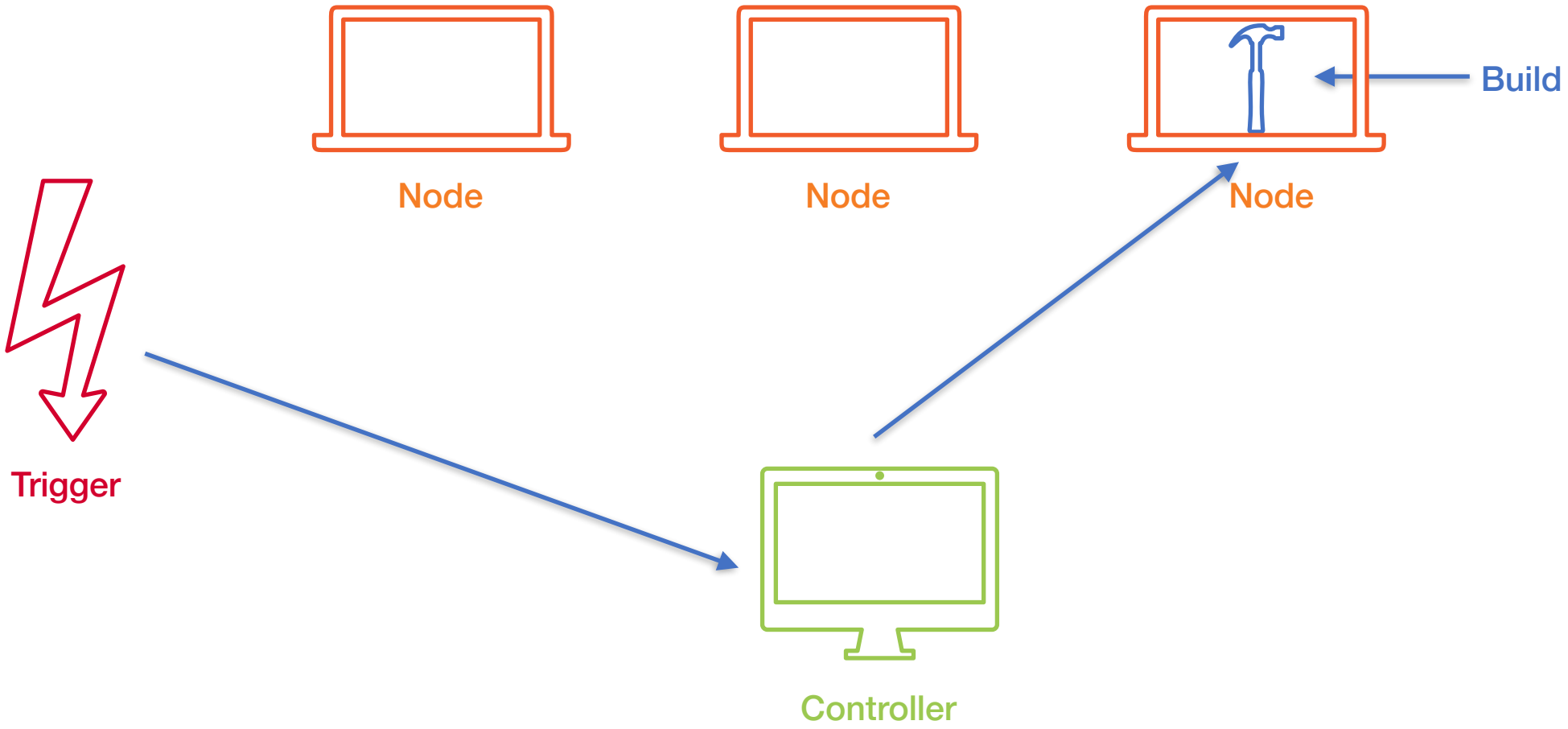
Controller-Node architecture



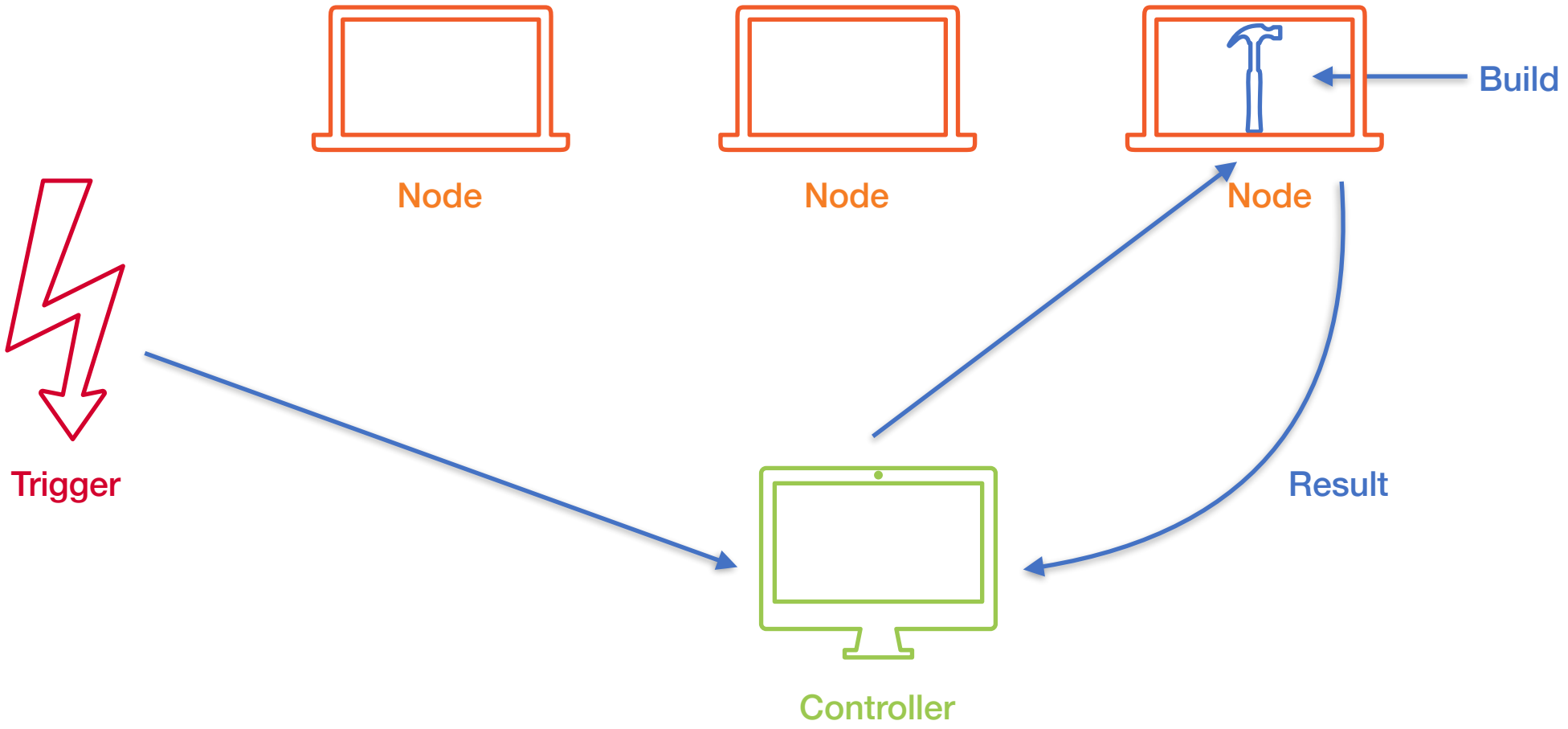
Controller-Node architecture



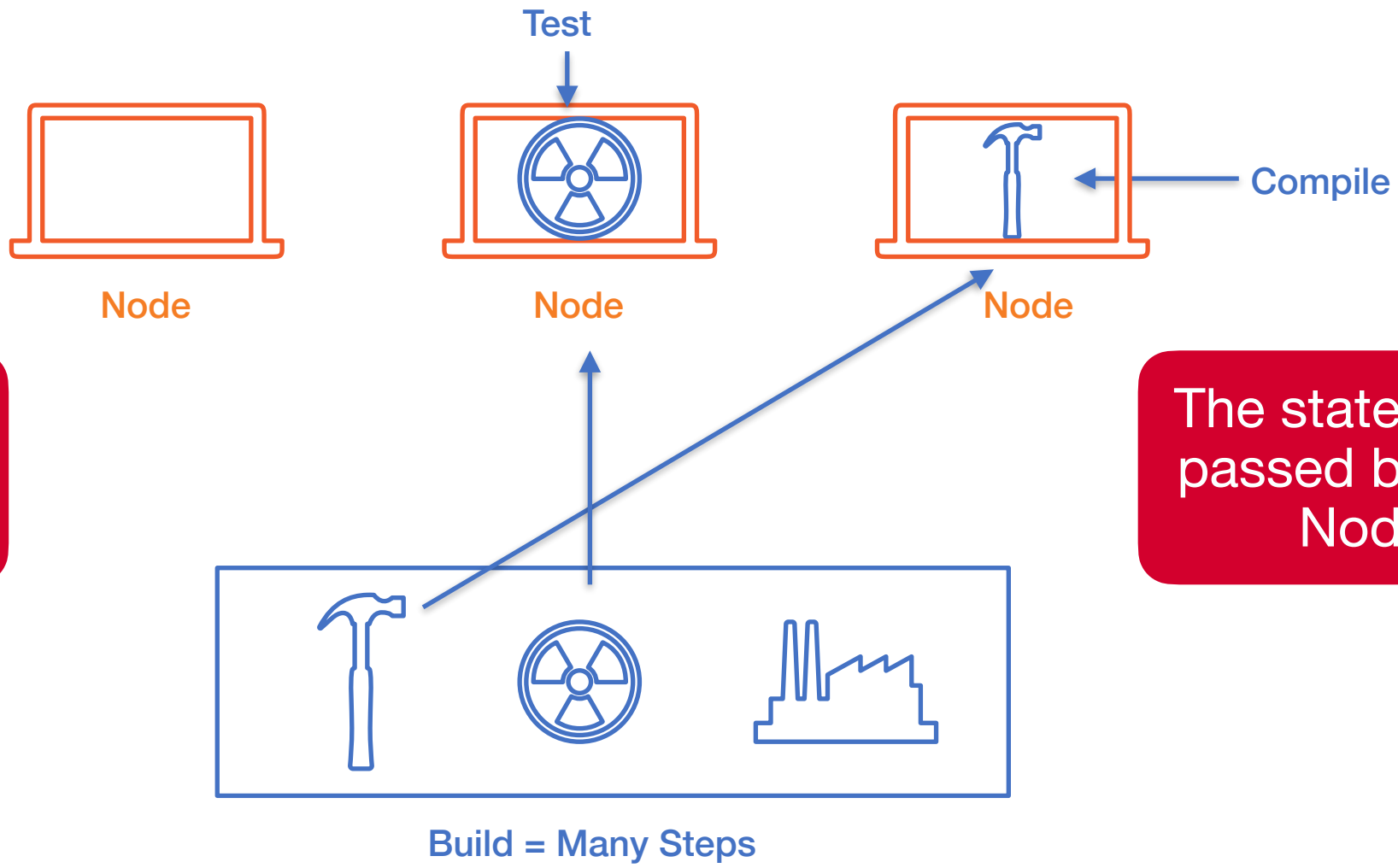
Controller-Node architecture



Controller-Node architecture



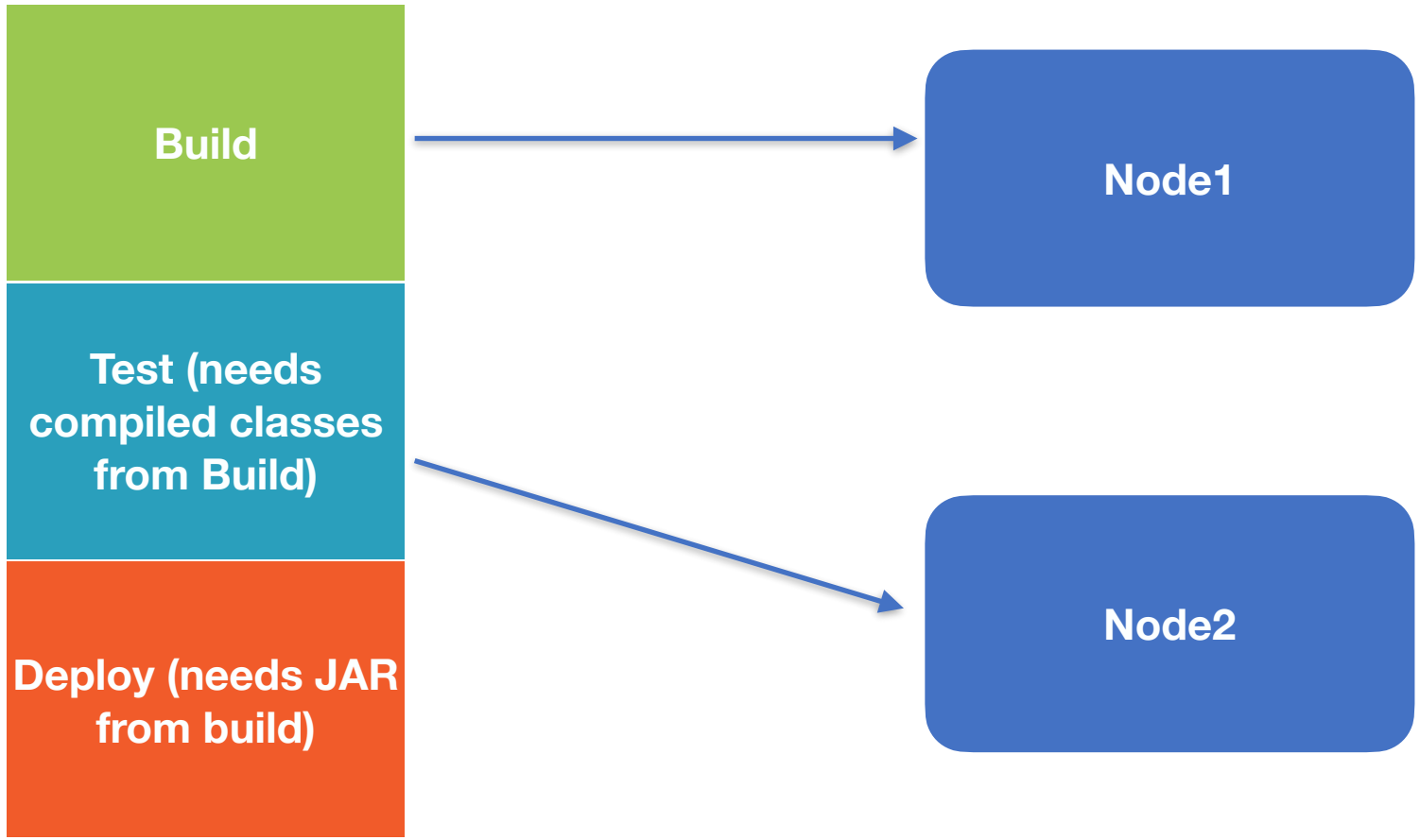
Controller-Node architecture



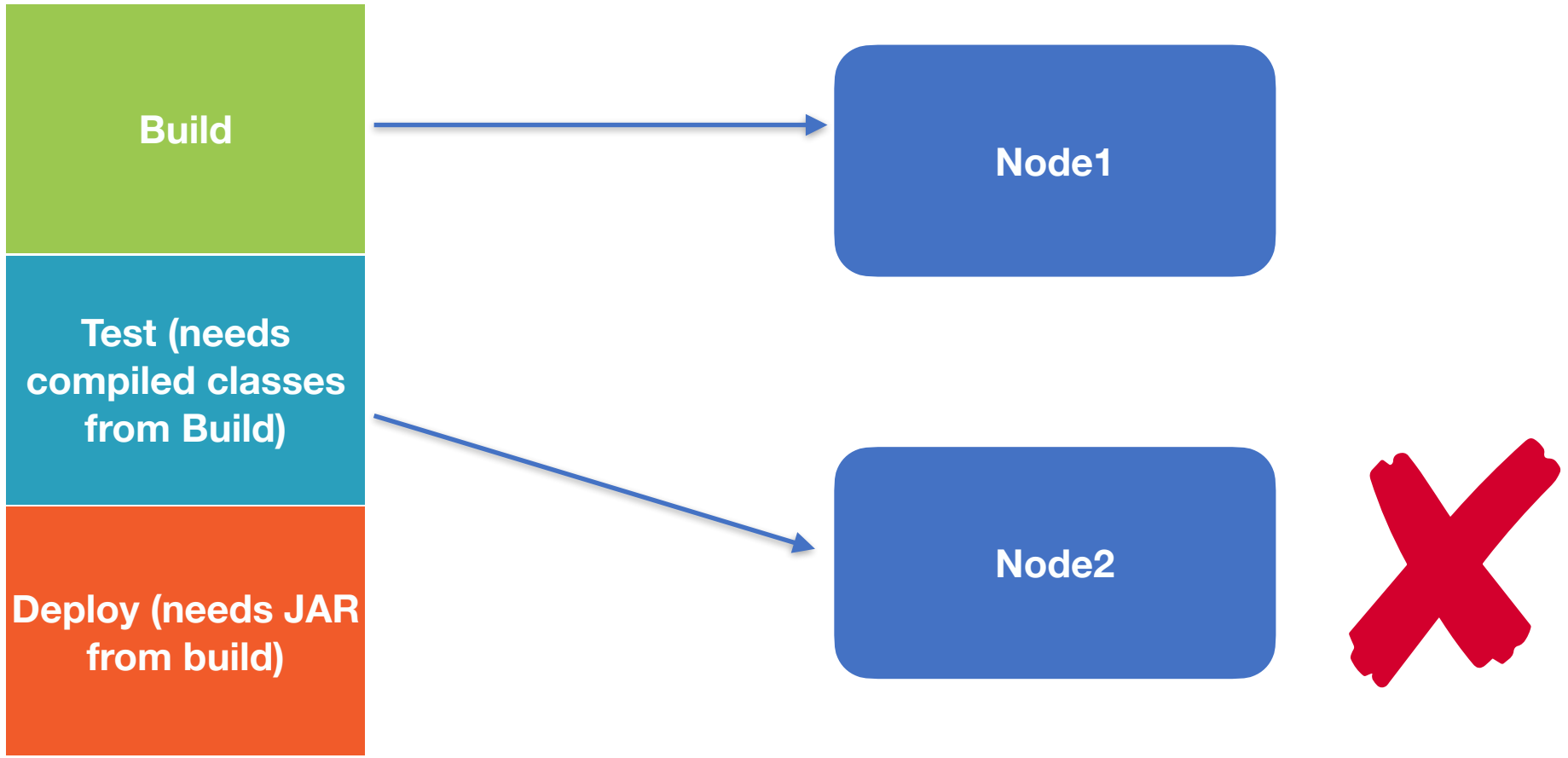
Each Step may run in a separate Node

The state is NOT passed between Nodes

Controller-Node architecture



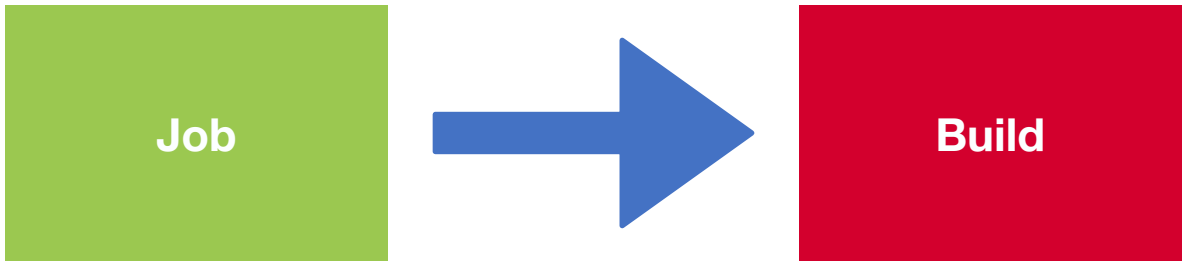
Controller-Node architecture



Job vs Build vs Step

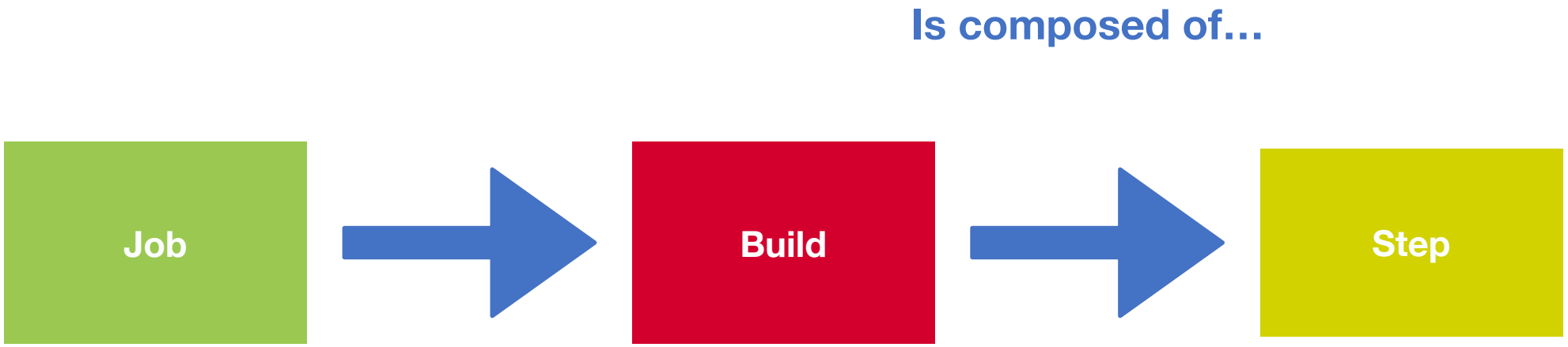
Develop Intelligence

Job vs Build vs Step



Is a template for…

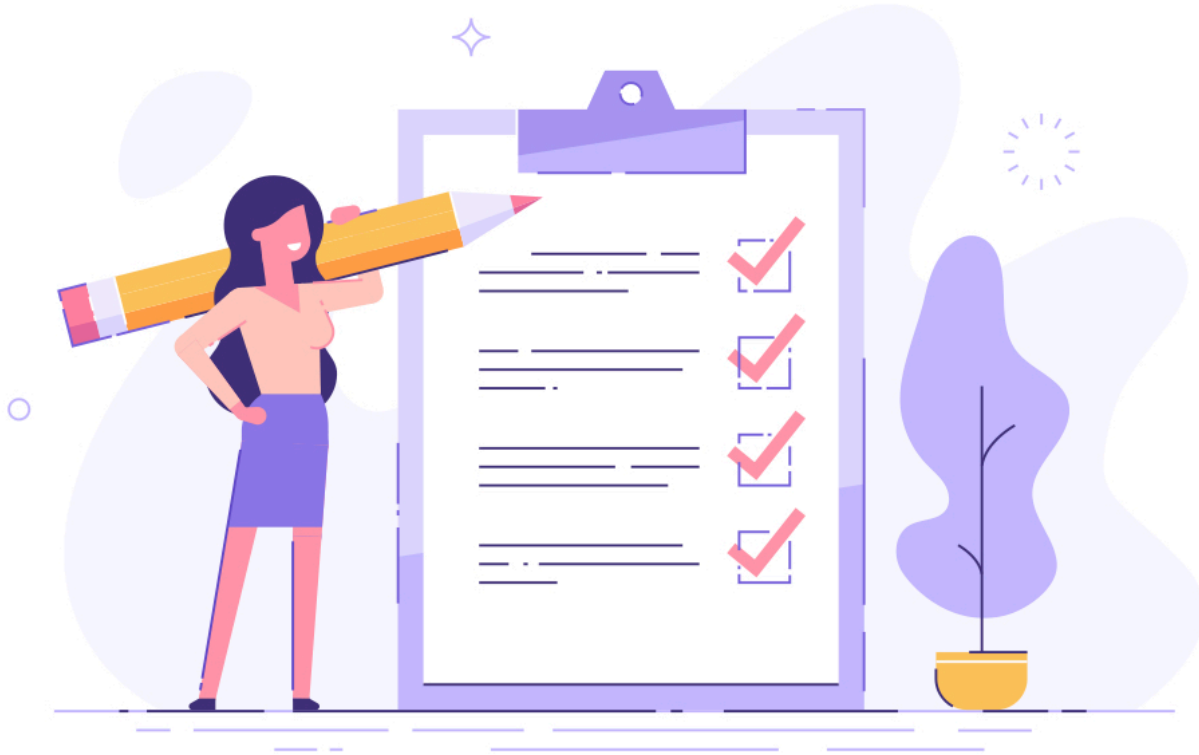
Job vs Build vs Step



LAB: Adding 2 nodes to Jenkins

- Add 2 nodes to Jenkins
- Set the executors on the controller as 0

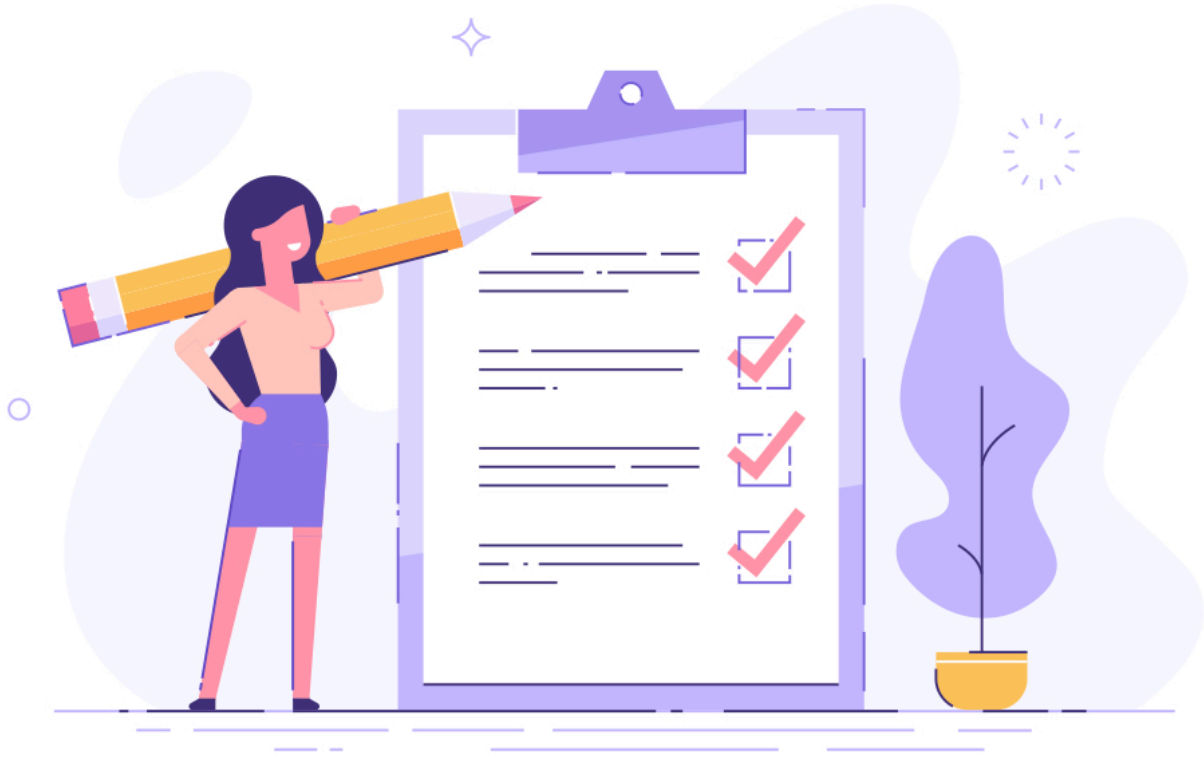
Allocated time: 15 minutes



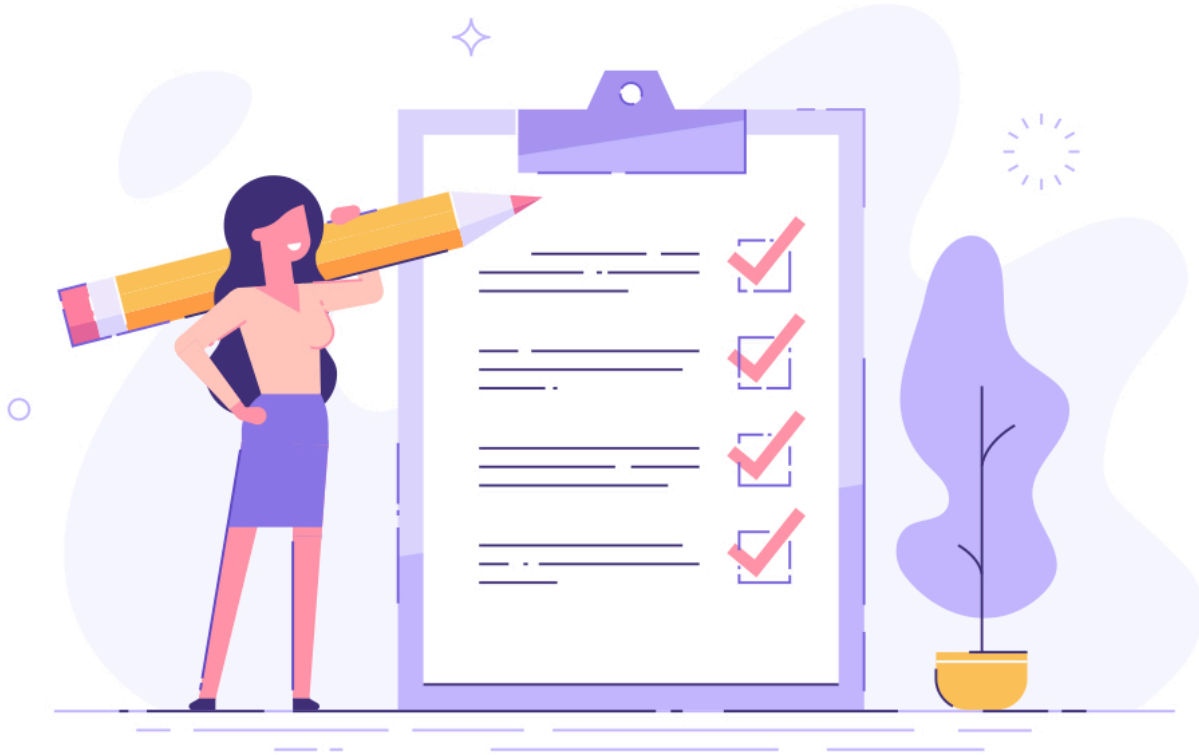
LAB: Storing artifacts

- Store the JAR file
- Execute the app in your terminal

Allocated time: 15 minutes



Pulse Check



Summary

- Jenkins has a controller - nodes architecture
- Each step of a build can run in a separate node, distributing the workload
- A Job is a template for the Builds, the oldest job is Freestyle job.

Lunch Break: 60 minutes

Develop Intelligence

First Steps

Develop Intelligence

Configuration

Agenda

Develop Intelligence

Introduction

- Introductions
- Expectations
- Why CI/CD?

First Steps

- Freestyle Jobs
- Configuring a CI job

Advanced Pipelines

- Code quality
- Notifications
- Parallel

Jenkins

- Why Jenkins?
- Architecture of Jenkins
- Setting Up

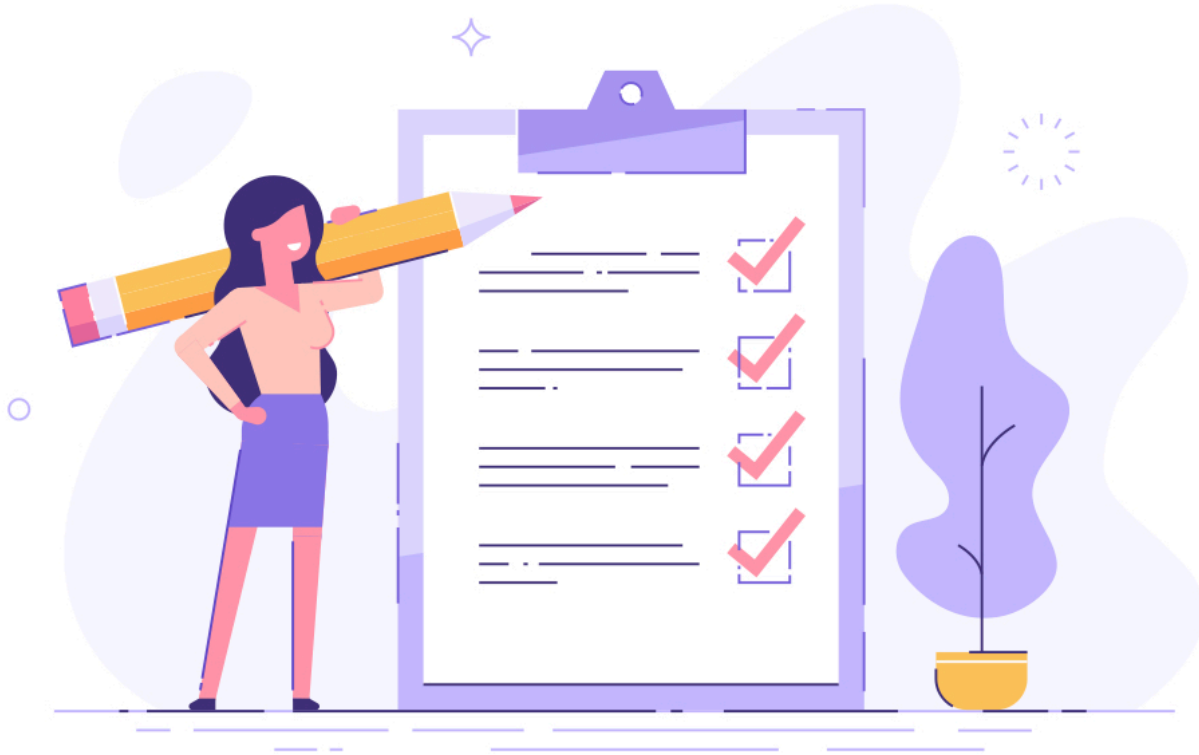
Jenkins Pipelines

- Scripted vs Declarative
- Jenkinsfile
- Developing Pipelines

LAB: Creating Users in Jenkins

- Creating users in Jenkins
- Manage authorisation and authentication

Allocated time: 20 minutes



Plugins

- Jenkins comes with many plugins to install
- Some are available at setup
- Some famous ones:

Pipeline, Blue Ocean, JUnit, Cobertura,

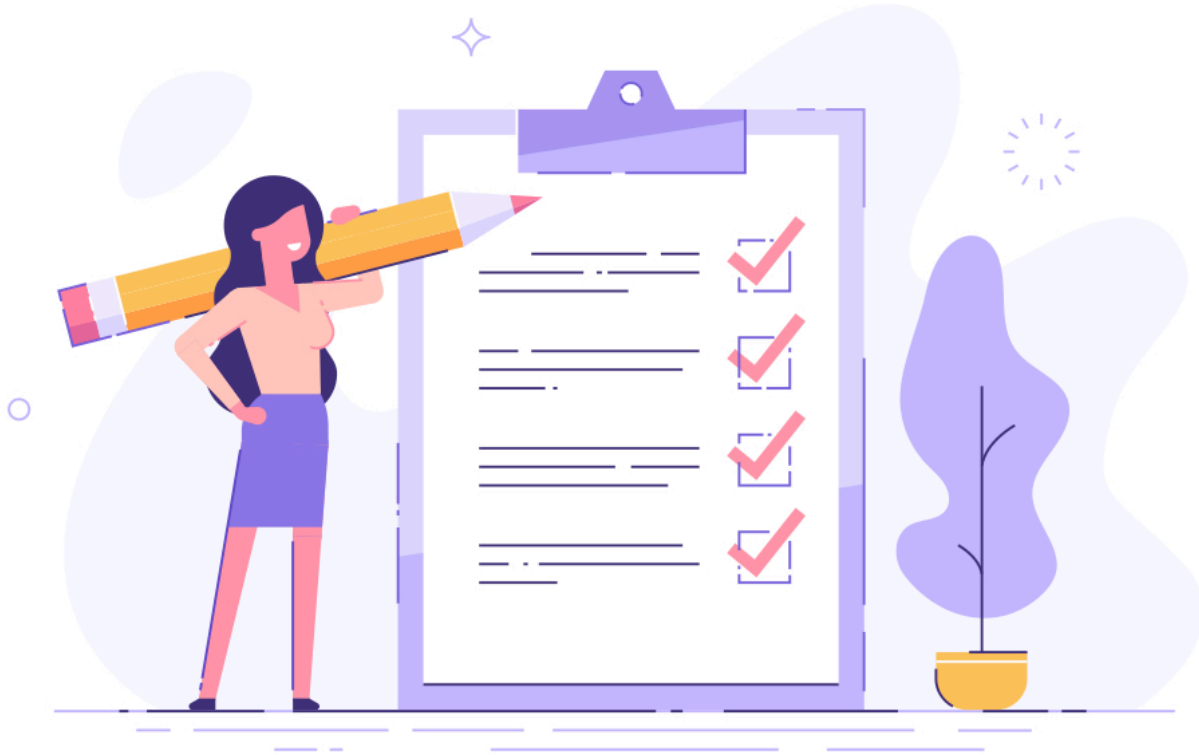
Docker, Git, Amazon Elastic EC2,

Conditional, SAML, etc…

LAB: Reporting code coverage in Jenkins

- Install Plugins
- Report Test report and Code coverage report in Jenkins

Allocated time: 30/45 minutes



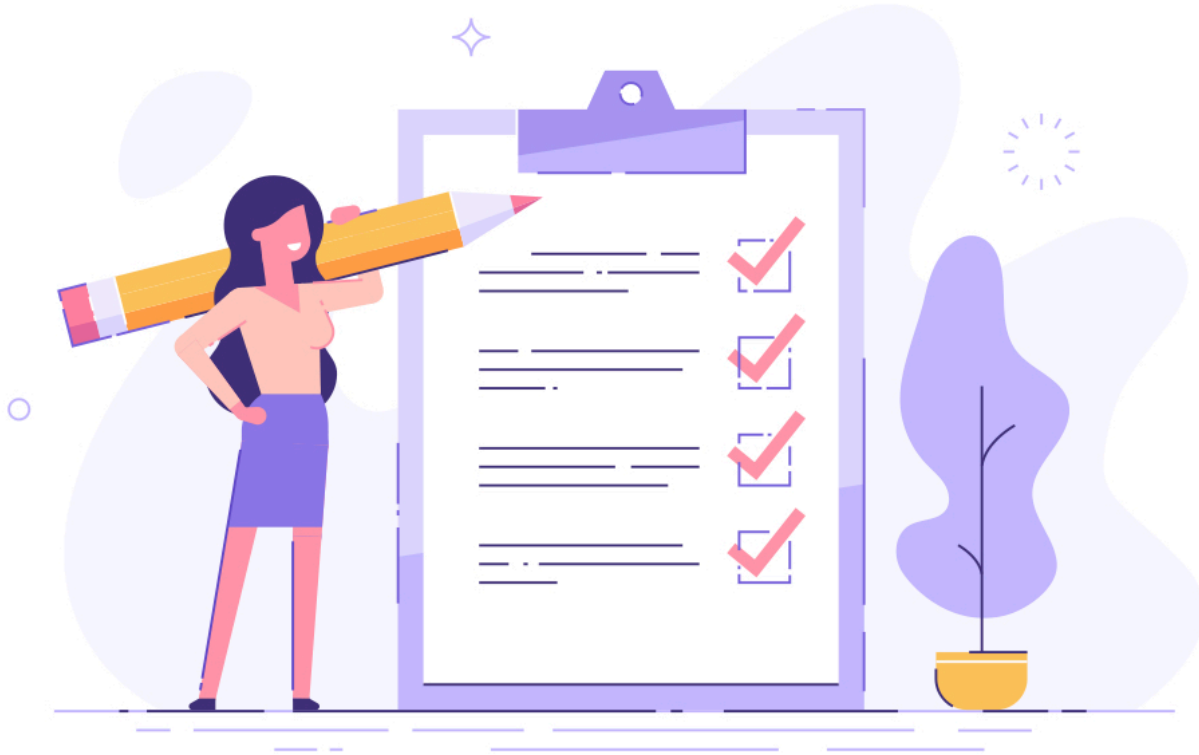
Break: 15 minutes

Develop Intelligence

LAB: Working with Jobs

- Build periodically
- Parametrise Builds
- Trigger remotely

Allocated time: 15 minutes

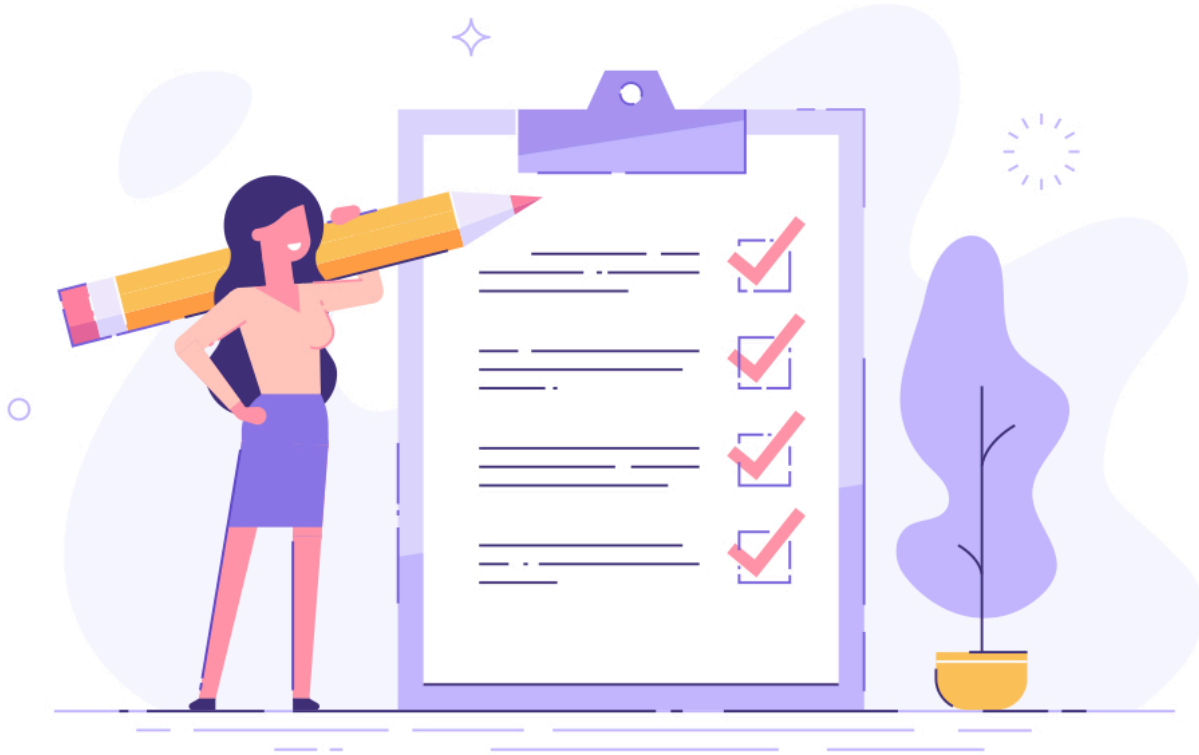


Summary

- One can define users via Jenkins database or LDAP
- From there define roles and manage permissions
- Jenkins has many plugins to extend functionality
- With them is easy to create pretty powerful pipelines

Workshop

- Feel free to use the remaining time to revisit the old core labs
- I would recommend taking one of **your projects** that need to migrate into Jenkins and create the Job for it. Probably fork it to play around safely.



End of Day 1

Develop Intelligence

Pipelines

Develop Intelligence

Declarative vs Scripted

Agenda

Develop Intelligence

Introduction

- Introductions
- Expectations
- Why CI/CD?

First Steps

- Freestyle Jobs
- Configuring a CI job

Advanced Pipelines

- Code quality
- Notifications
- Parallel

Jenkins

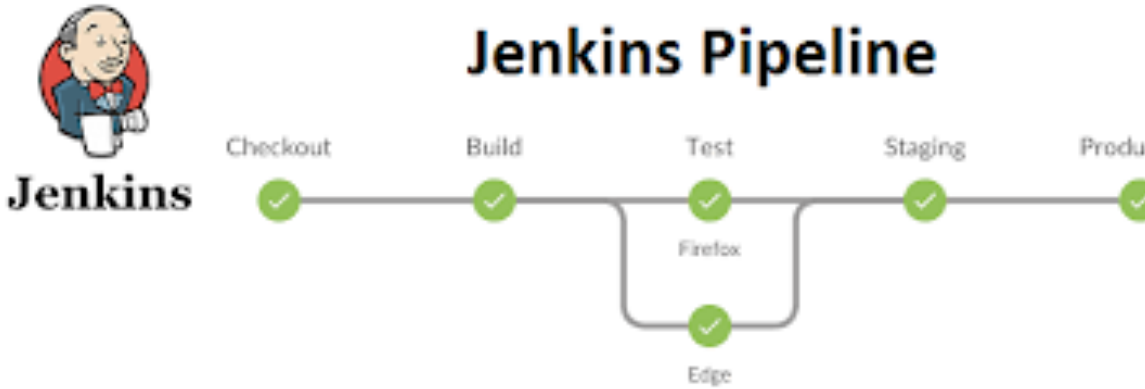
- Why Jenkins?
- Architecture of Jenkins
- Setting Up

Jenkins Pipelines

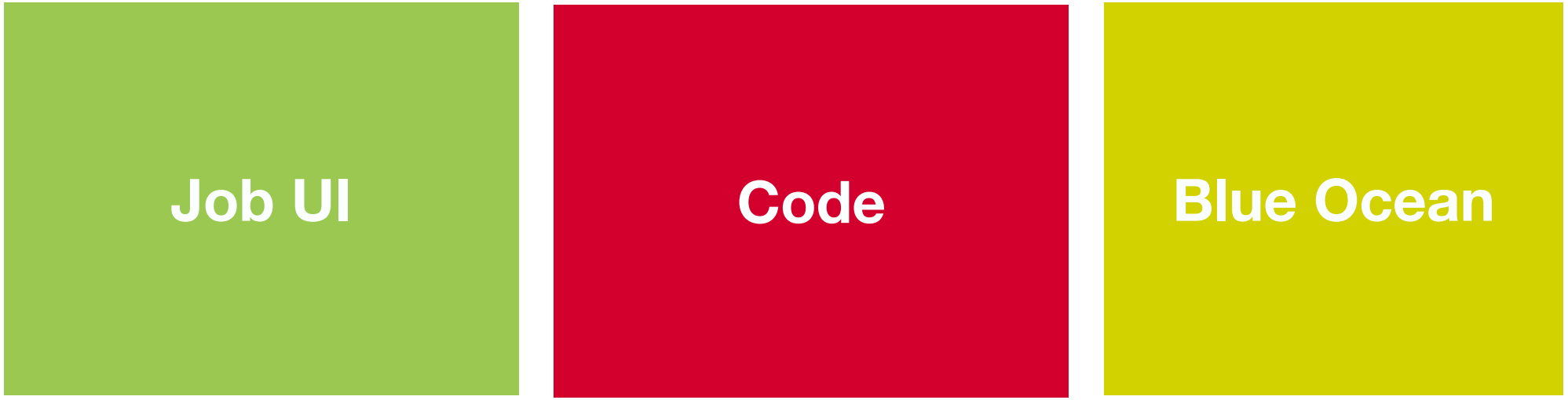
- Scripted vs Declarative
- Jenkinsfile
- Developing Pipelines

Pipelines

- Jenkins created the Pipeline plugin, which change the way we create jobs
- It approaches pipeline as code
- Makes it easier to review jobs, adds functionality, and increases reproducibility.



Ways to create a Pipeline



Ways to create a Pipeline

Develop Intelligence

Job UI

Code

Blue Ocean

Declarative Pipelines

Declares a Pipeline

Where does it run?

Aggregate steps into a Stage

Defines the steps to run in a Stage

The specific step. This is Jenkins language, but you can use Groovy!

Scripted Pipelines

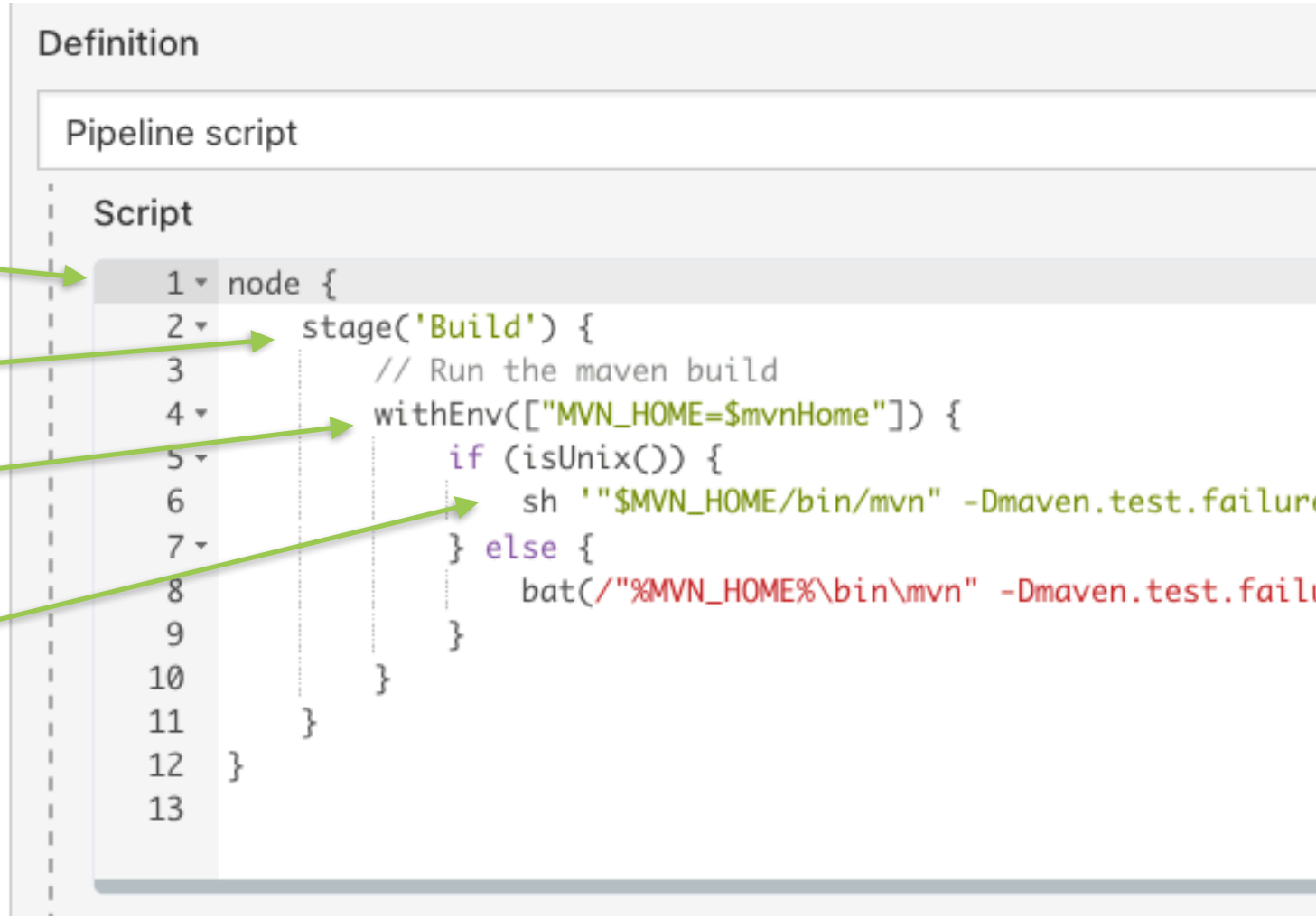
Develop Intelligence

Where does it run?

Aggregate steps into a Stage

We can use Groovy!

Shell script. This is just syntax sugar for a traditional Groovy method



Scripted vs Declarative

Develop Intelligence

- Oldest way of creating pipelines
- Closer to coding
- Can use any object loaded in the classpath
- More difficult to pass state between nodes
- Full compatibility with Freestyle jobs

- New way of creating pipelines
- More declarative (ie: not imperative). We declare what we want, not how to do it
- State and scoping is handled by Jenkins
- Only uses Objects that are Steps (inherit from…)
- Not 100% compatibility yet

Pipeline Syntax Helper

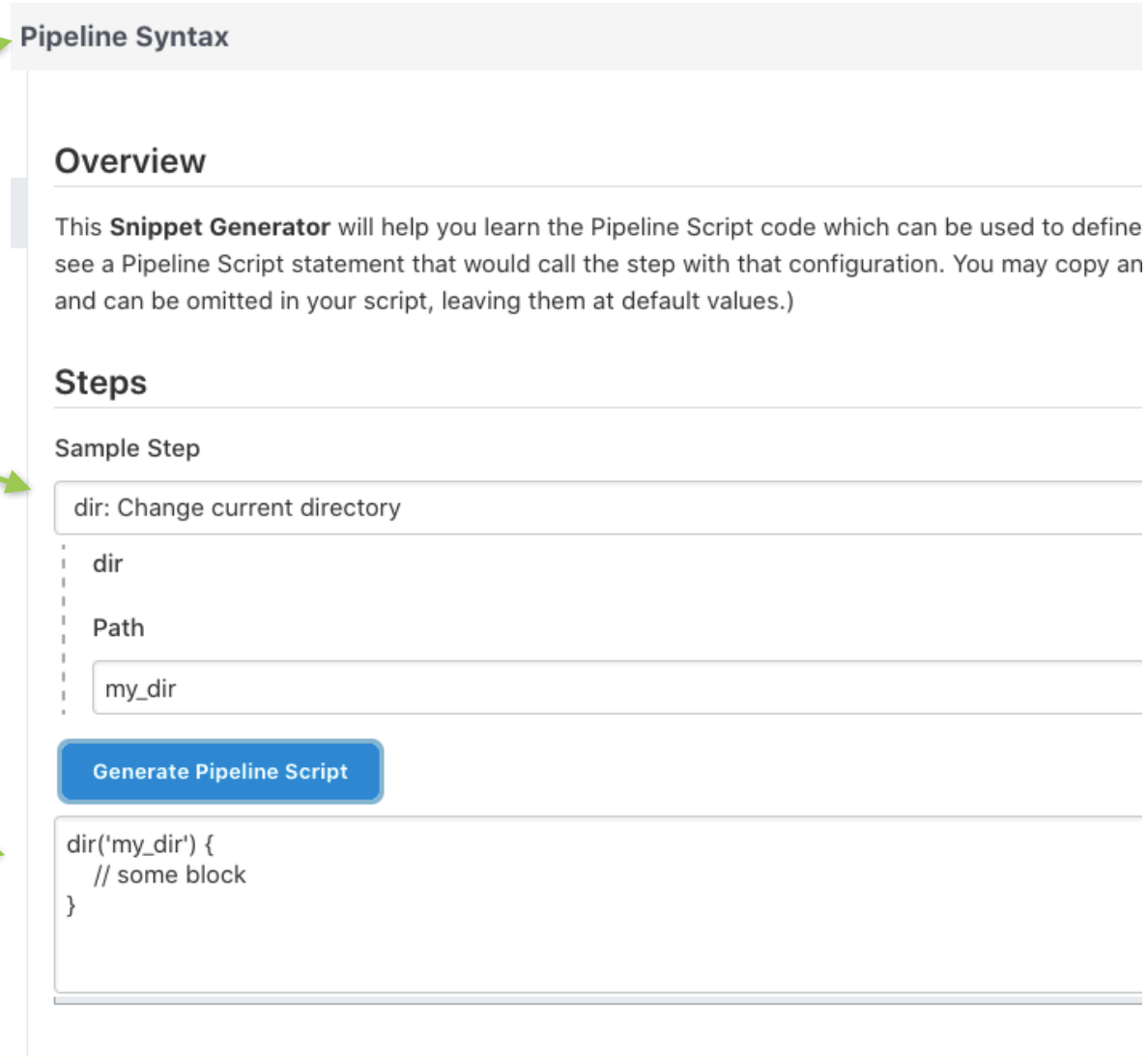
Develop Intelligence

Syntax helper

You select your command

Define the parameters easily

It generates the code to copy!

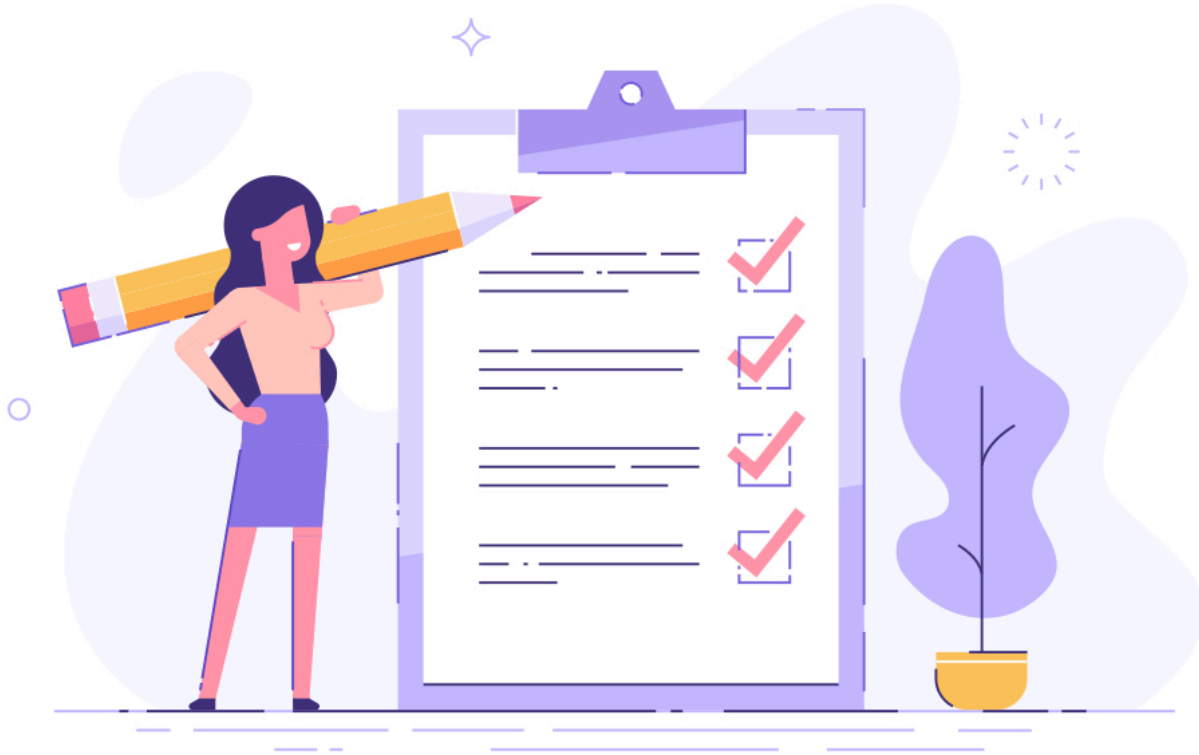


The docs are amazing! https://www.jenkins.io/doc/book/pipeline/syntax/

LAB: Working with Pipeline

- Create declarative pipeline through UI
- Use post actions

Allocated time: 30 minutes



LAB: Conditionals and Parameters

- Declare a conditional stage based on parameters

Allocated time: 20 minutes

LAB: Restart a Stage

- Restart a stage due to flakiness
- Make steps independent
- Run on separate nodes
- Use string interpolation

Allocated time: 20 minutes

Break: 15 minutes

Develop Intelligence

Ways to create a Pipeline

Develop Intelligence

Job UI

Code

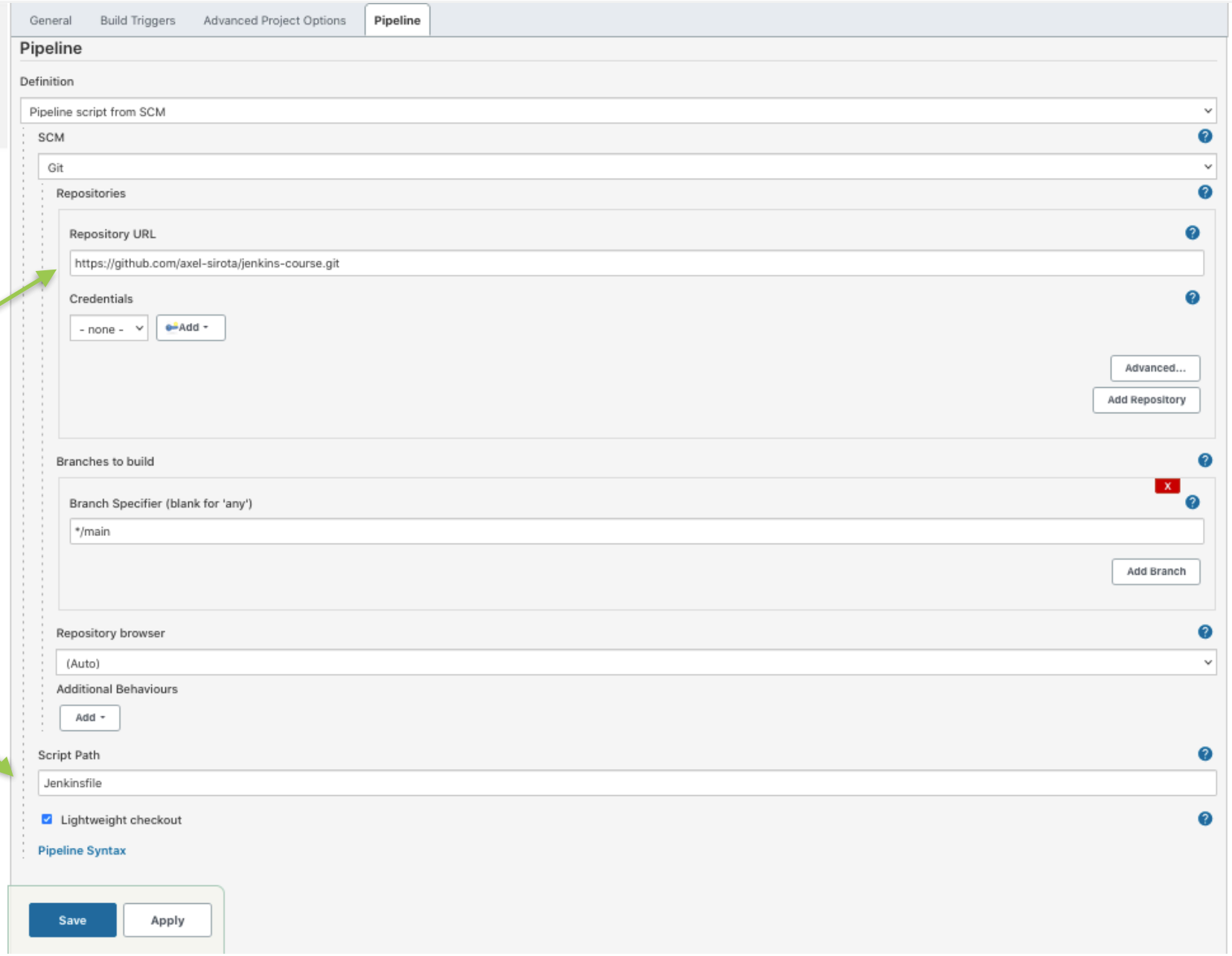
Blue Ocean

Jenkinsfile

The Code for the pipeline will be in source control!

Where does it live?
It may be a separate repo from the one we build

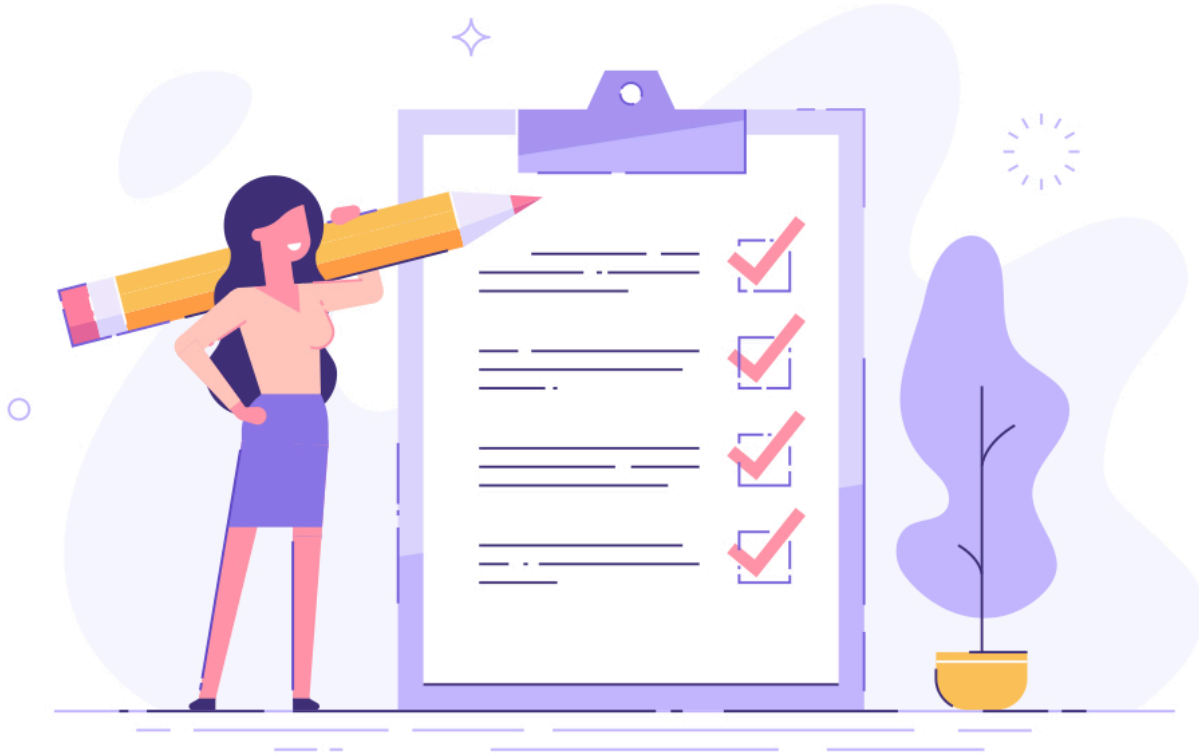
The name of the file, default is Jenkinsfile



LAB: Scripted pipeline in a Jenkinsfile

- Create a scripted pipeline
- Create a Jenkinsfile
- Publish the code coverage

Allocated time: 20 minutes



LAB: Scripted pipeline in a Jenkinsfile

- Create a scripted pipeline
- Change the value of a variable in a stage and use it in another stage

Allocated time: 15 minutes

Summary

Develop Intelligence

- Pipelines are the new way of creating Jobs
- They provide flexibility and an infra-as-code style
- One can create them through the UI, Jenkinsfile, or Blue Ocean.
- Scripted pipelines are for code only solutions, and they run on the controller
- Declarative pipelines are more restricted but easier and run on the nodes

Advanced Pipelines

Develop Intelligence

Code Quality

Agenda

Develop Intelligence

Introduction

- Introductions
- Expectations
- Why CI/CD?

First Steps

- Freestyle Jobs
- Configuring a CI job

Advanced Pipelines

- Code quality
- Notifications
- Parallel

Jenkins

- Why Jenkins?
- Architecture of Jenkins
- Setting Up

Jenkins Pipelines

- Scripted vs Declarative
- Jenkinsfile
- Developing Pipelines

Best Practices

- Scaling Jenkins
- Jenkins Deployments
- Security

Ways to create a Pipeline

Develop Intelligence

Job UI

Code

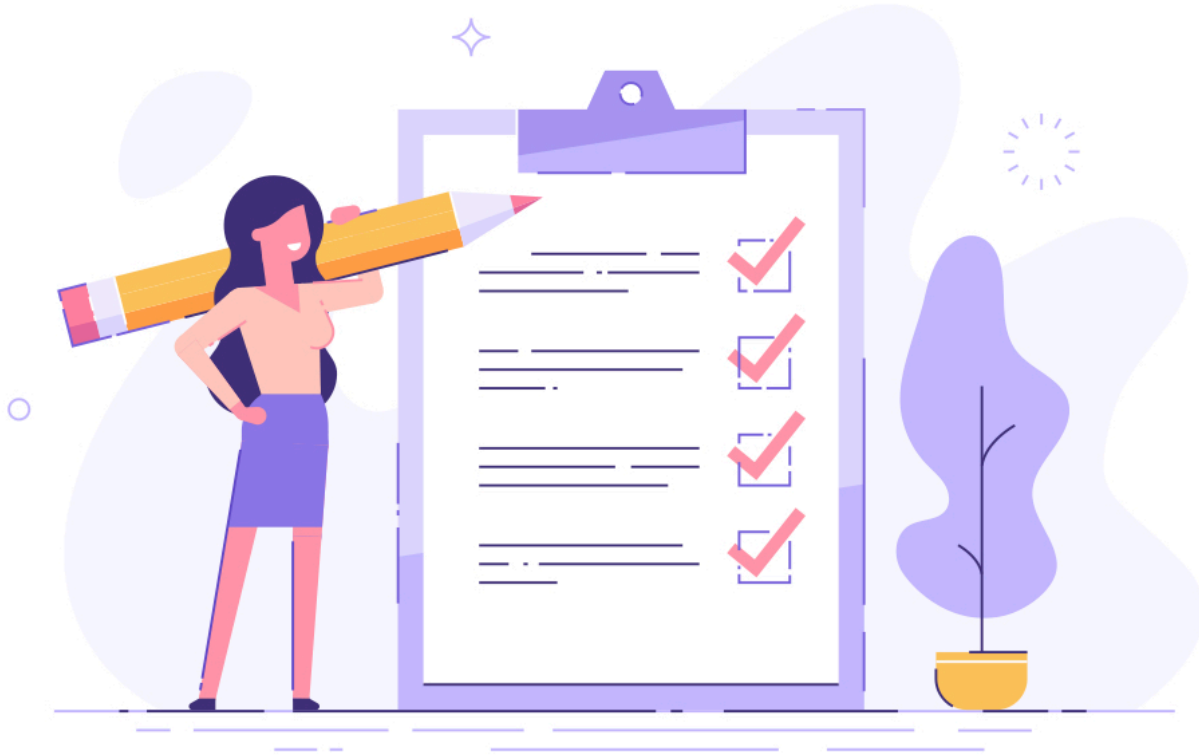
Blue Ocean

Blue Ocean Flow

LAB: Blue Ocean and Static Analysis

- Create a pipeline through Blue Ocean
- Add static code analysis using spot bugs (do you have SonarQube in your company?)

Allocated time: 30 minutes



Parallel

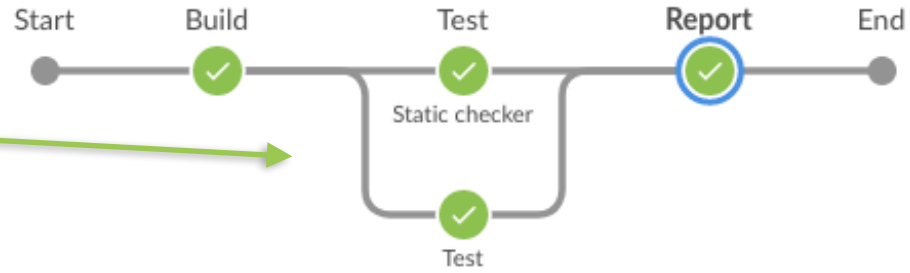
Develop Intelligence

Parallel execution

Paralel directive

It may enclose steps or stages

Both may determine an agent or be “free”




```
parallel {
  stage('Test') {
    steps {
      sh 'mvn -f maven-static-code-analysis clean test'
    }
  }

  stage('Static checker') {
    steps {
      sh 'mvn -f maven-static-code-analysis site'
    }
  }

}
}
```

Matrix

For each combination of values creates a parallel build

It goes at the stage directive

One may exclude options

```
matrix {
    axes {
        axis {
            name 'PLATFORM'
            values 'linux', 'mac', 'windows'
        }
        axis {
            name 'BROWSER'
            values 'chrome', 'edge', 'firefox', 'safari'
        }
        axis {
            name 'ARCHITECTURE'
            values '32-bit', '64-bit'
        }
    }
```

LAB: Parallelisation

- Matrix execution

Allocated time: 15 minutes

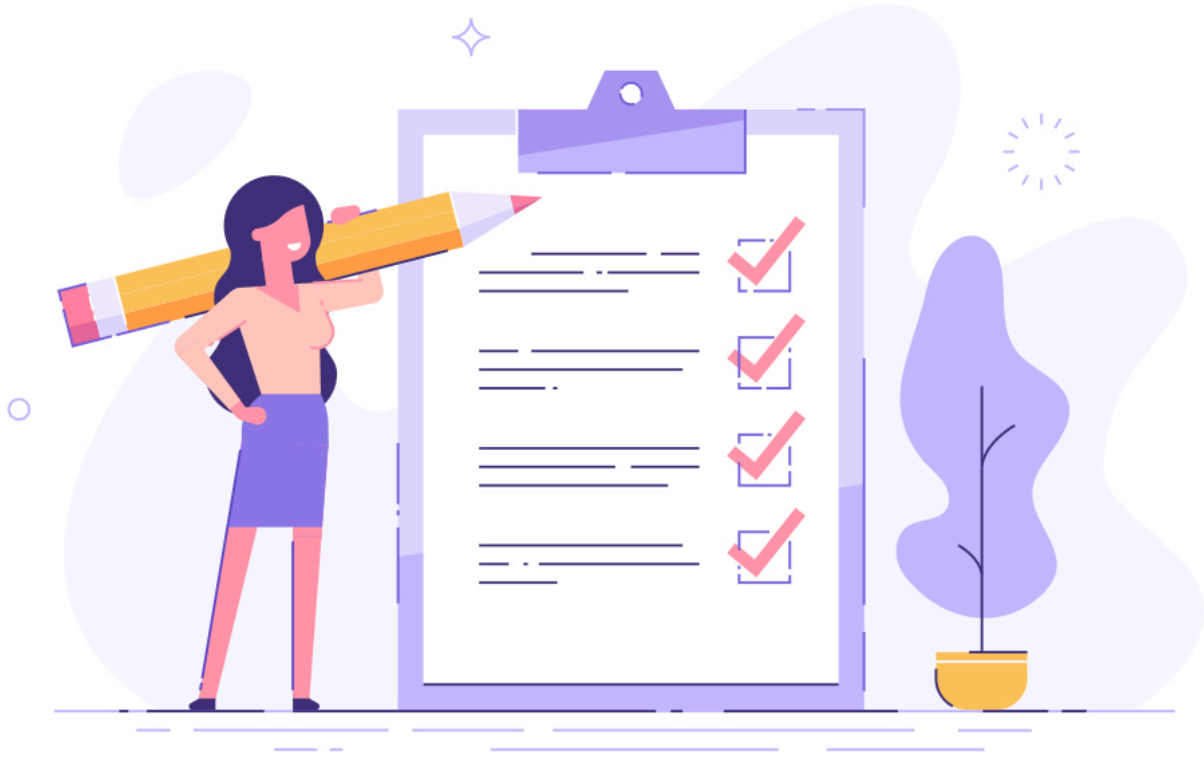
LAB: Git Flow in Jenkins

- Create a CI/CD job in Jenkins

Allocated time: 30 minutes

LAB: Slack Notifications in Jenkins

- Add Slack notifications in Jenkins



Allocated time: 15 minutes

Shared Libraries

- Sometimes we want to share some logic
- For that we have shared libraries
- It involves a Groovy script that later we load

Any *.groovy file that has a call method will be a new Step

main jenkins-shared-library-example / vars / notification.groovy

axel-sirota example-shared-library

1 contributor

32 lines (28 sloc) | 968 Bytes

```
1  #!/usr/bin/env groovy
2
3  /**
4   * Send notifications based on build status string
5   */
6  def call(String buildStatus = 'STARTED') {
7    // build status of null means successful
8    buildStatus = buildStatus ?: 'SUCCESS'
9
```

Shared Libraries

We can use any Groovy logic

You later access the step by importing the library as @Library _

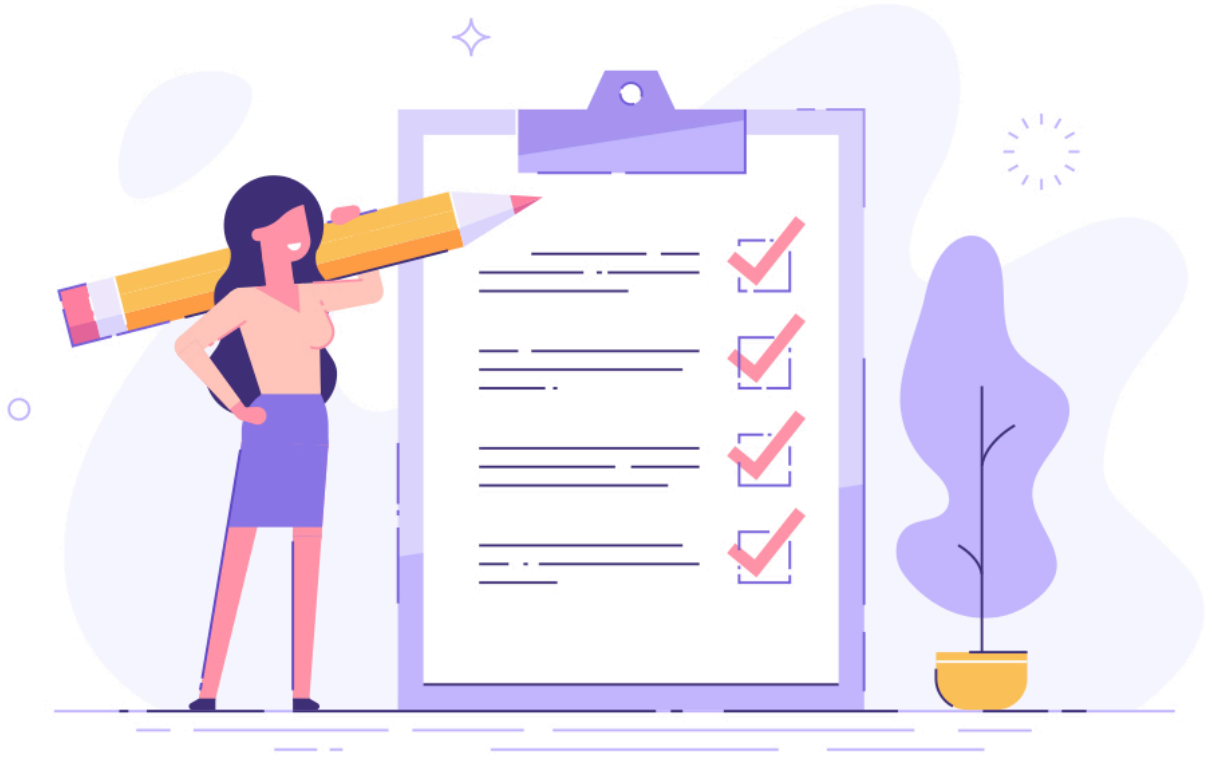
Or even access methods from Plugins!

```
// Override default values based on build status
if (buildStatus == 'STARTED') {
  color = 'YELLOW'
  colorCode = '#FFFF00'
} else if (buildStatus == 'SUCCESS') {
  color = 'GREEN'
  colorCode = '#00FF00'
} else {
  color = 'RED'
  colorCode = '#FF0000'
}

// Send notifications
slackSend (color: colorCode, message: summary)
}
```

LAB: Shared Libraries

- Create a simple shared library for Slack notifications



Allocated time: 30 minutes

Pipeline Best Practices

- Groovy code runs on controller -> avoid resource contention!
- Use shared libraries to avoid repetition
- Avoid sharing workspaces on different jobs
- Ensure variables (state) are serialisable or use @NonCPS
- Use credentials and scopes!

Summary

- Combining Pipelines and Plugins one can create a perfectly valid CI/CD pipeline
- Using parallel or matrix you can even test for multiple platforms/browsers/combinations easily
- There are simple notification systems to Slack or other platforms such as Google chat or Microsoft Teams
- Shared libraries enable us to share code between Jenkinsfiles

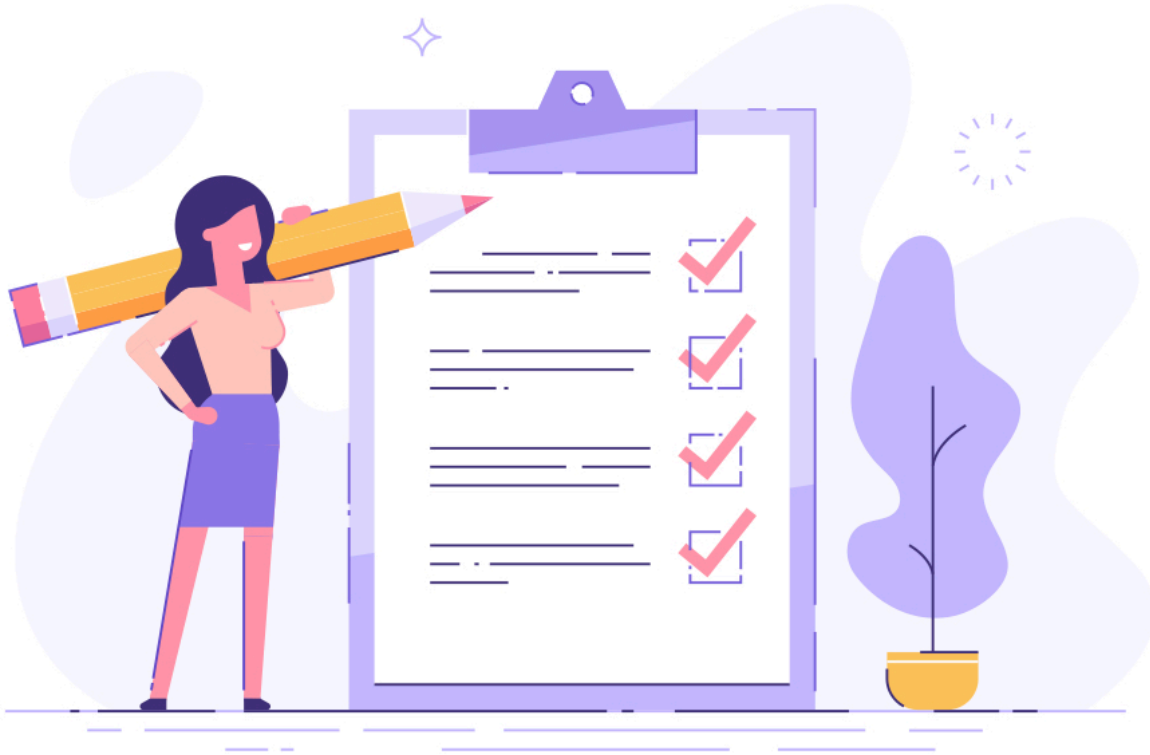
Please Fill in Survey

Lunch Break: 60 minutes

Develop Intelligence

Workshop

- Take the job that you added CI yesterday and migrate it into Pipelines. Add CD, notifications and all the quality checks we saw today!



Recording Policy

Recordings are provided to participants who have attended the training, in its entirety. Recordings are provided as a value-add to your training, and should not be utilized as a replacement to the classroom experience.

Participants can expect the following:

- Recordings will be provided the Monday after class is completed
- Recordings will be provided for 14 days
- Recordings will be accessible as “View Only” status and cannot be copied
- Sharing recordings is strictly prohibited

To request recordings, please fill out the form linked in Learn++.

Thank you for your understanding and adhering to the policy.

THANK YOU

Develop
Intelligence