ENTERPRISE SYSTEMS INTEGRATION

ACIT4850 – WINTER 2024

AGENDA – LESSON 5

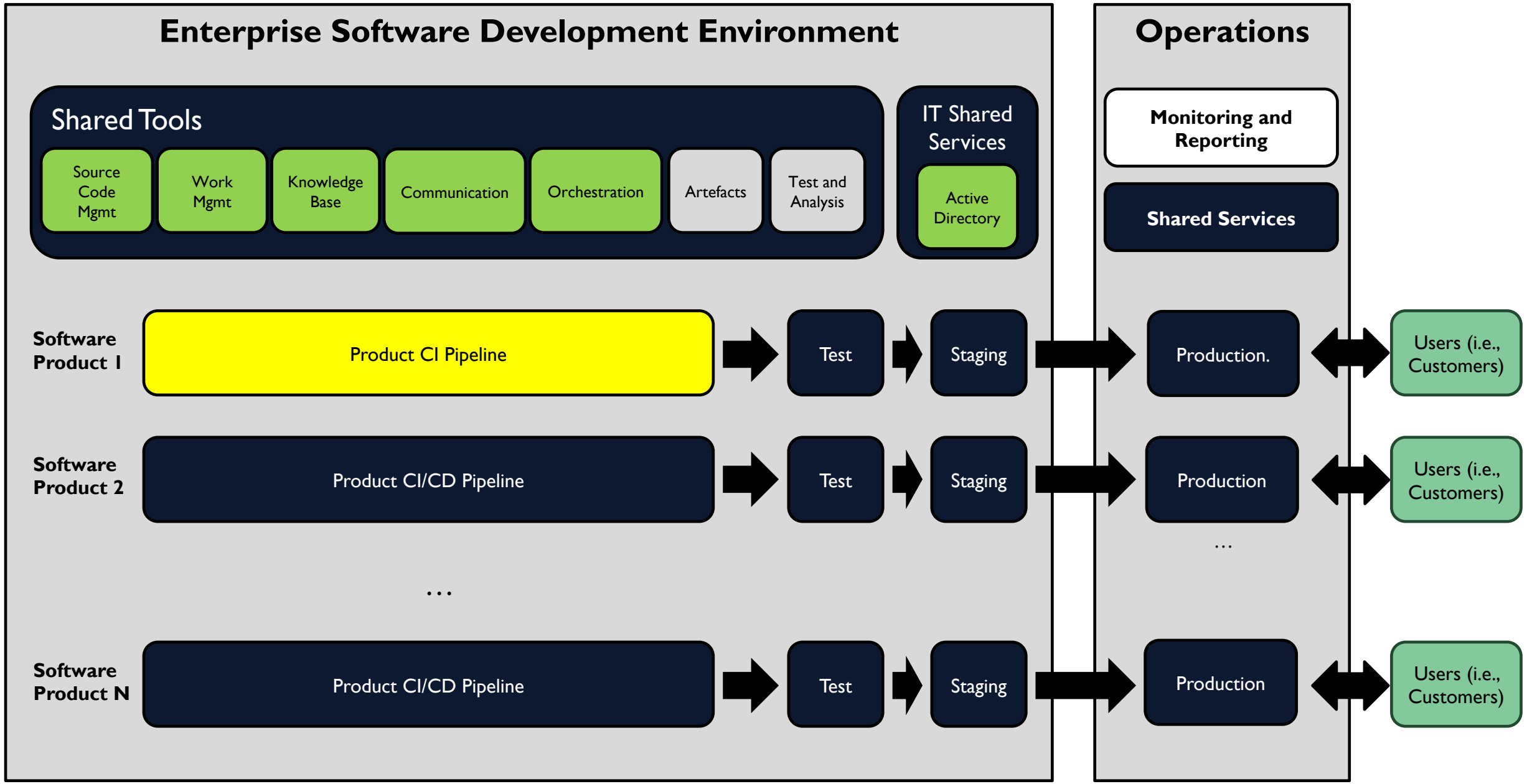
- Quick Review
- Quiz 4 on D2L
- Topics
  - Jenkins Pipeline
- Lab Requirements
- Lab
  - Demo of Lab 4
  - Start on Lab 5

REVIEW QUESTIONS

- Where do we define a Jenkins Pipeline (i.e., which file)?
- Where do we store our Jenkins Pipeline definition? Why – what are the advantages?
- What are the main keywords used in a Jenkins Pipeline definition?
- When should we trigger a Jenkins Pipeline?

QUIZ 4

- On Jenkins Pipelines
- On the Learning Hub (aka D2L), Open Book
- Your have 15 minutes to complete it


Enterprise Software Development Environment
Shared Tools
Source Code Mgmt
Work Mgmt
Knowledge Base
Communication
Orchestration
Artefacts
Test and Analysis
IT Shared Services
Active Directory
Operations
Monitoring and Reporting
Shared Services
Software Product 1
Product CI Pipeline
Test
Staging
Production.
Users (i.e., Customers)
Software Product 2
Product CI/CD Pipeline
Test
Staging
Production
Users (i.e., Customers)
...
...
Software Product N
Product CI/CD Pipeline
Test
Staging
Production
Users (i.e., Customers)

THE ROADMAP (AKA COURSE SCHEDULE)

Week	Topics	Notes
1	• Components of an Enterprise Development Environment • Software Source Code Management	Lab 1
2	• Work Management and Knowledge Base Tools	Lab 2, Quiz 1
3	• Tool Selection – Requirements • Integration and Security	Lab 3, Quiz 2
4	• Tool Selection – Stakeholders/Process • Continuous Integration (CI) Tool • CI Tool Setup	Lab 4, Quiz 3
5	• CI Pipelines – Python	Lab 5, Quiz 4
6	• CI Pipelines – Shared Libraries	Lab 6, Quiz 5, Assignment 1 Due
7	• CI Pipelines – Java and Static Code Analysis *Note: At home lab for Monday set*	Lab 7, Quiz 6 (Sets A and B)
8	• Midterm	Midterm Review Quiz
9	• CI Pipelines – Alternate Tools	Lab 8, Quiz 6 (Set C), Quiz 7
10	• Spring Break	
11	• CI Pipelines – Artifact Management (Java)	Lab 9, Quiz 8, Assignment 2 Due
12	• Continuous Delivery (CD) • CD Pipelines - Containerization	Lab 10, Quiz 9
13	• CD Pipelines – Deployment • Developer Workflows *Note: At home lab for Monday Set*	Lab 11, Quiz 10 (Sets A and B)
14	• Microservices Pipelines • Final Exam Preview	Quiz 10 (Set C), Assignment 3 Due
15	**Final Exam**	

REMINDERS

- Assignment 1 – Due Feb. 16th
- JIRA – If you've demoed Labs 2 and 3 and received your marks you can delete JIRA
  - Delete the items in the JIRA resource group, not just the VM. Otherwise they will continue billing you for the other resources (i.e., the disk)

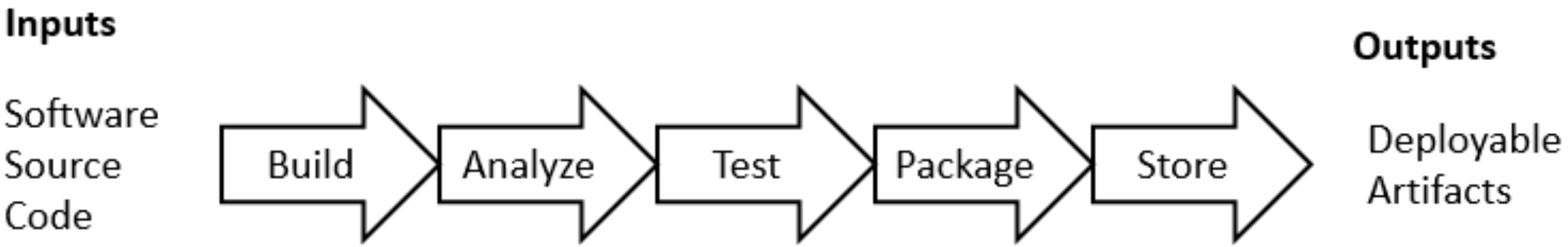
DEFINITIONS

- **Jenkins Pipeline** is the definition of a Continuous Integration/Delivery (CI/CD) pipeline in Jenkins
- **Continuous Integration Pipeline** – Automated expression of your process for getting software from source code management (i.e., from Git) through to build, test and packaging steps.
- **Continuous Delivery Pipeline** – Extension of the CI Pipeline to include automated deployments to development and test environments.

JENKINSFILE

- This is the definition for a Jenkins pipeline
- It uses a Domain Specific Language and Declarative Syntax based on the Groovy language syntax
- It should be committed to the repository of the code it builds
- We will see, in a future class, that you can create re-usable pipeline modules that can be called from your Jenkinsfile
  - This is useful if you have a lot of repositories that are built in the same manner
- Benefits
  - Creates a pipeline automated in a Jenkins job
  - You can review and collaborate on the pipeline when checked into source code management
  - Audit trail
  - Source of truth for the pipeline

TYPICAL PIPELINE STAGES



ADVANTAGES

Three Key Advantages:

- **Code**: Pipelines are implemented in code. This makes it easier to review and collaborate on the pipeline.
- **Durable**: Can survive when the Jenkins server goes down. They are "backed up" with the source code.
- **Modular**: Can reuse the same patterns in multiple pipelines.

NEW AND UPDATED LAB REQUIREMENTS

- **REQ1150** – The Enterprise Development Environment shall automatically trigger a Continuous Integration Pipeline on a Software Project (i.e., repository) in Source Code Management when the source code changes. Note: A push to a project in GitLab should trigger the corresponding build job.
- **REQ1160** – The Enterprise Development Environment shall support Continuous Integration Pipelines for Python projects. At minimum, the pipeline will include build, test, packaging and artifact storage.
- **SEC1020** – All web applications and API endpoints shall be encrypted (i.e., https endpoints). Note: We will use a signed certificate to better emulate a production-like environment.

NEW LAB REQUIREMENTS

- **REQ1150** – The Enterprise Development Environment shall automatically trigger a Continuous Integration Pipeline on a Software Project (i.e., repository) in Source Code Management when the source code changes. Note: A push to a project in GitLab should trigger the corresponding build job.

You will be using a GitLab webhook to trigger a Jenkins job on a push to your GitLab project (i.e., repository). This will require configuration in both GitLab and Jenkins.

NEW LAB REQUIREMENTS

- **REQ1160** – The Enterprise Development Environment shall support Continuous Integration Pipelines for Python projects. At minimum, the pipeline will include build, test, packaging and artifact storage.

You will be creating a simple pipeline that builds and tests your sample Python application.

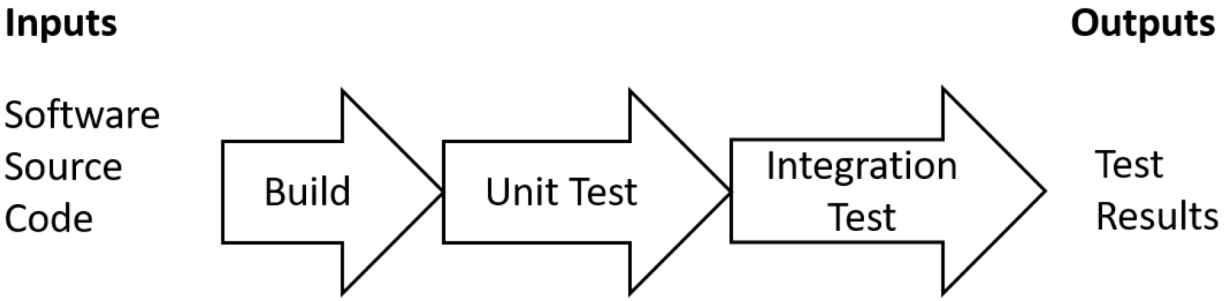
Build consist of installing the Python requirements (i.e., module dependencies – SQLAlchemy, Flask, Etc)

Build consists of unit tests (PointManager) and integration tests (PointApi).

JENKINSFILE

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                Some Steps Here
            }
        }
        stage('Unit Test') {
            steps {
                Some Steps Here
             }
        }
        stage('Integration Test') {
            steps {
                Some Steps Here
             }
        }
    }
}
```

- pipeline – overall pipeline definition
- agent – specification for running the pipeline
- stages – one or more stages in the pipeline
- stage – specific state in the pipeline
- steps – one or more actions to perform in the stage



SECURITY – *SIGNED* CERTIFICATE

SEC1020 *– All web applications and API endpoints shall be encrypted (i.e., https endpoints). Note: We will use a signed certificate to better emulate a production-like environment.*

We will now use a signed certificate in our prototype environment to better emulate a production-like environment and prevent integration issues using self-signed certificates.

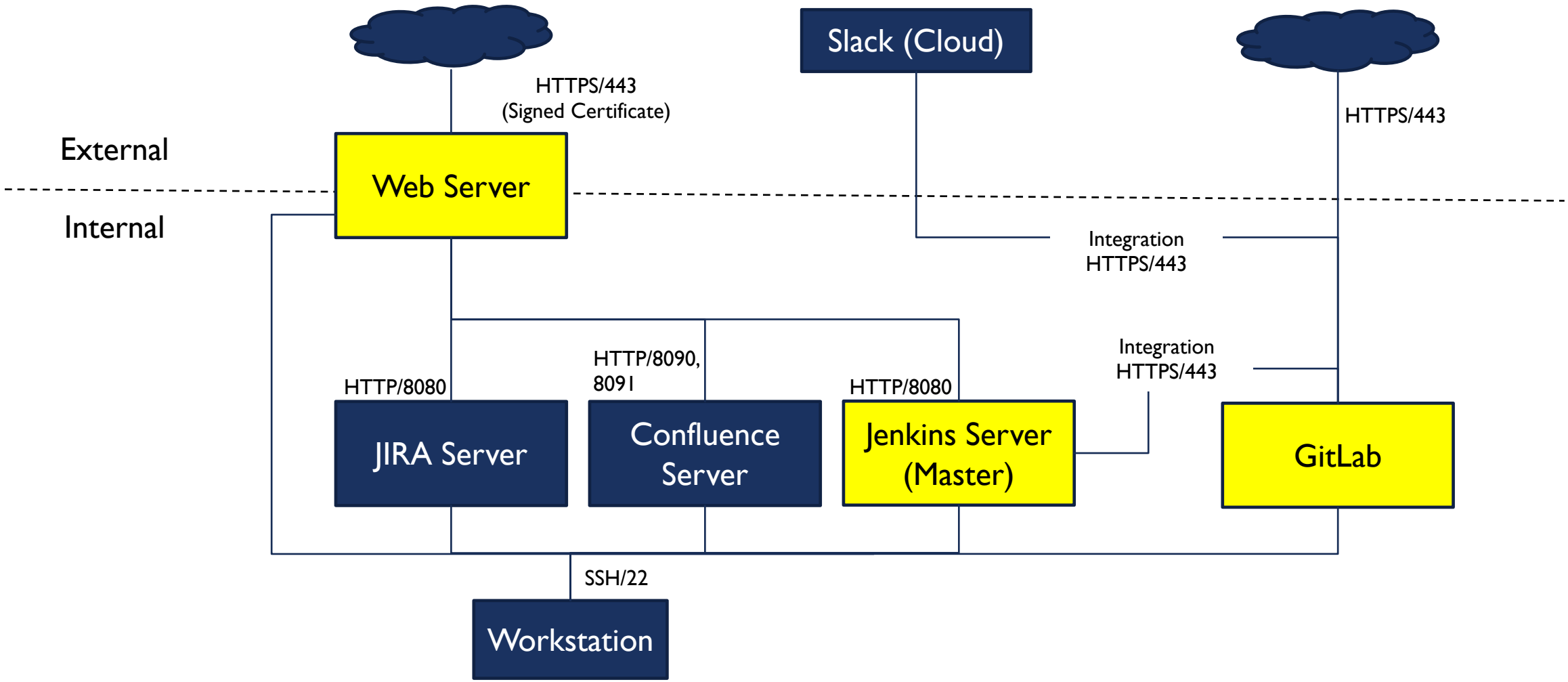
Most applications will not support integration with endpoints with self-signed certificates (others have an option to ignore these checks, but it's not ideal).

We will be using Lets Encrypt to provide us with a signed certificate, which is same provider as used by the GitLab installer.

LETS ENCRYPT

- Free, automated and open certificate authority (CA)
  - Typically used for web services, personal/startup websites
  - Not used for highly secure/trusted sites (like your bank, so be careful)
- Free, but short-term (around 3-4 months vs. 1+ years)
  - We may have to renew it before the end of the term
- Automated – you can't manually have a certificate signed, need to use an automated process
  - We will use CertBot for Apache
  - It will be based on the ServerName defined in you default-ssl.conf configuration file. It must match the DNS Name for you Apache VM.
- See https://letsencrypt.org/how-it-works for details

YOUR ENTERPRISE DEVELOPMENT ENVIRONMENT (SO FAR)



TODAY'S LAB

1. Demo Lab 4 Before the End of Class
2. Start on Lab 5
    1. You will do this together with your partner
    2. Demo is due by end of next class.
3. The next several labs we will continue on Jenkins pipeline jobs