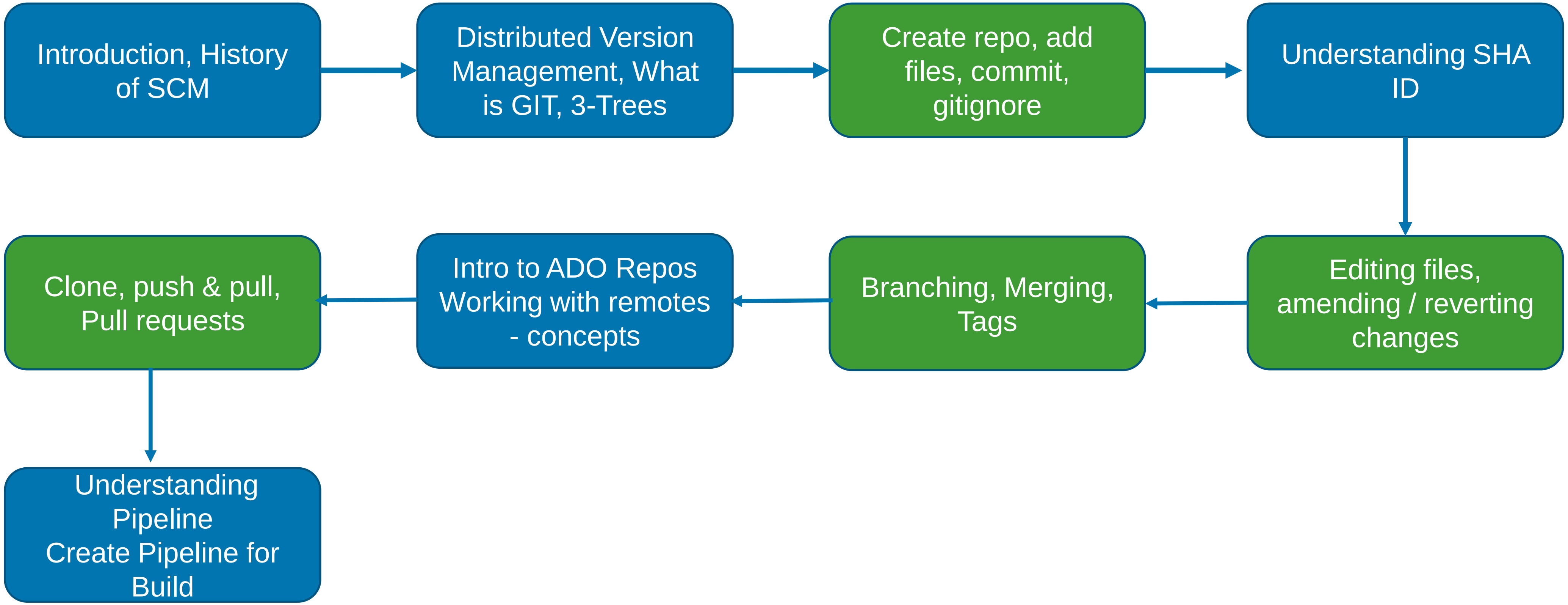
axess academy

Git and ADO

standard chartered

10/07/2025

Module Design



Introduction, History of SCM

Distributed Version Management, What is GIT, 3-Trees

Create repo, add files, commit, gitignore

Understanding SHA ID

Editing files, amending / reverting changes

Branching, Merging, Tags

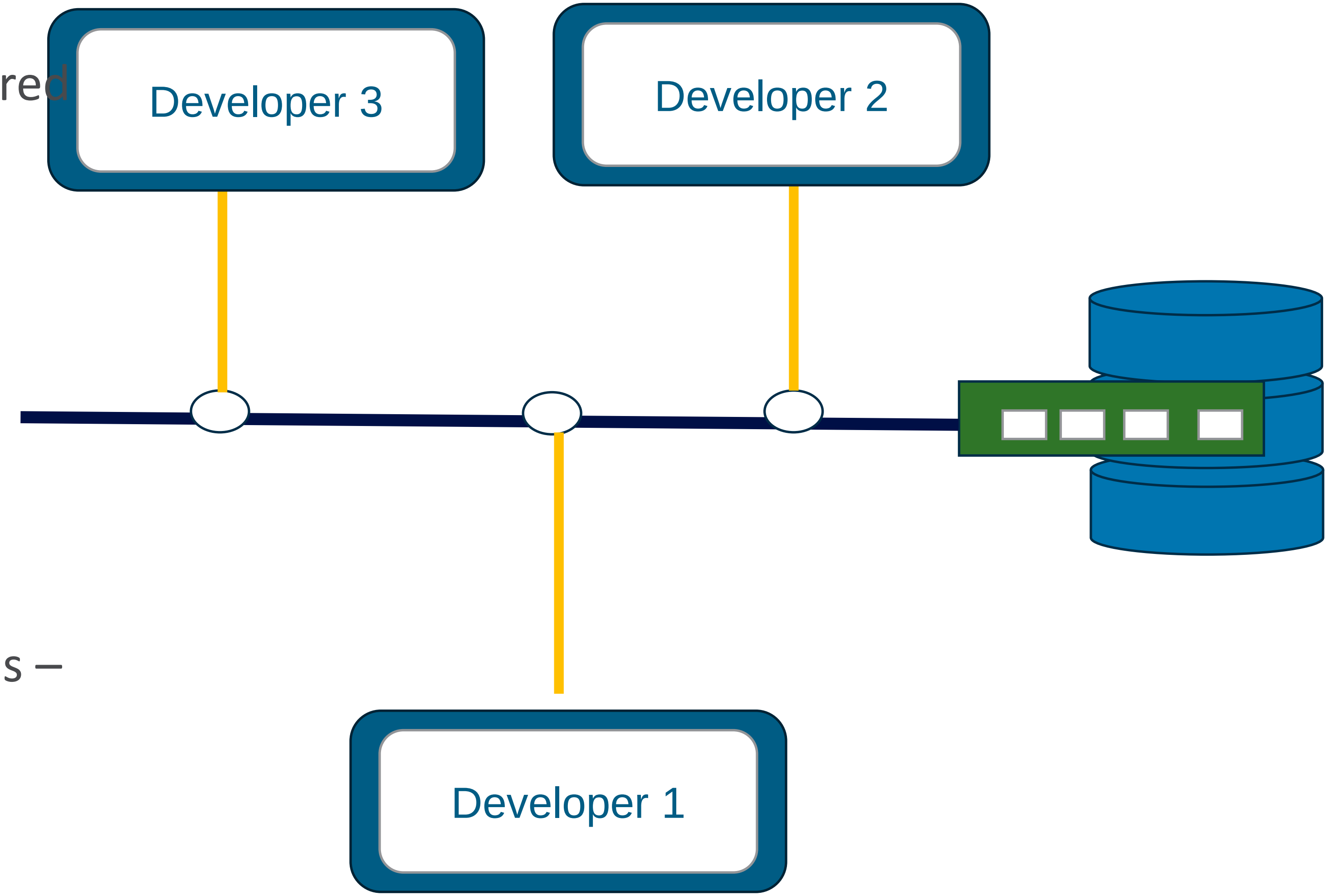
Intro to ADO Repos Working with remotes - concepts

Clone, push & pull, Pull requests

Understanding Pipeline
Create Pipeline for Build

Need for Source Code Management

- Allow multiple developers to work on shared code in parallel
- Avoid overwrites and edits of each other code
- Tracks the History of changes, allowing reverts to previous versions
- Allows Collaboration
- Parallel development on multiple activities – New feature, Bug fixes etc.

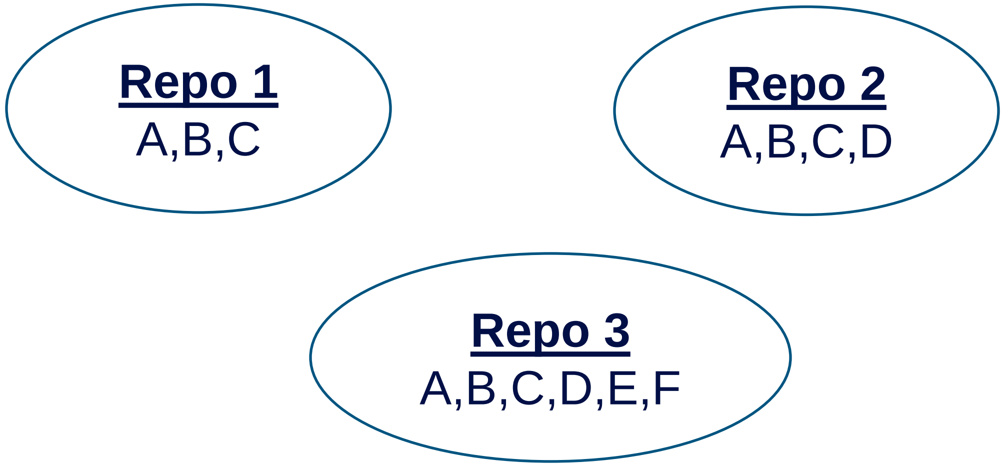


History of version management

- Source Code Control System (SCCS)
  - 1972, Closed source, free with UNIX
- Revision Control System (RCS)
  - 1982 , Open Source, cross platform, faster, more features
- Concurrent versions system (CVS)
  - 1986-1990, open source, multiple files, multiple users
- Apache Subversion (SVN)
  - 2000, open source, support for directory tracking, support for non-text
- BitKeeper SCM
  - 2000, closed source, proprietary
  - distributed version control
  - “community version” was free initially ; from 2005 no longer free
  - used for source code of Linux kernel from 2002 – 2005
- GIT
  - Born April 2005, created by Linus Torvalds
  - Replacement for BitKeeper to manage Linux kernel

Understanding Distributed Version Control

- Different users maintain their own repositories instead of working from a central repository
- Changes are stored as “Change sets” or “patches”
  - Tracks changes not versions
  - Change sets can be exchanged between repositories
- Example , suppose a file has following sets of changes – A,B,C,D,E,F
- Different repositories may have different combinations of the above change sets



Repo 1
A,B,C

Repo 2
A,B,C,D

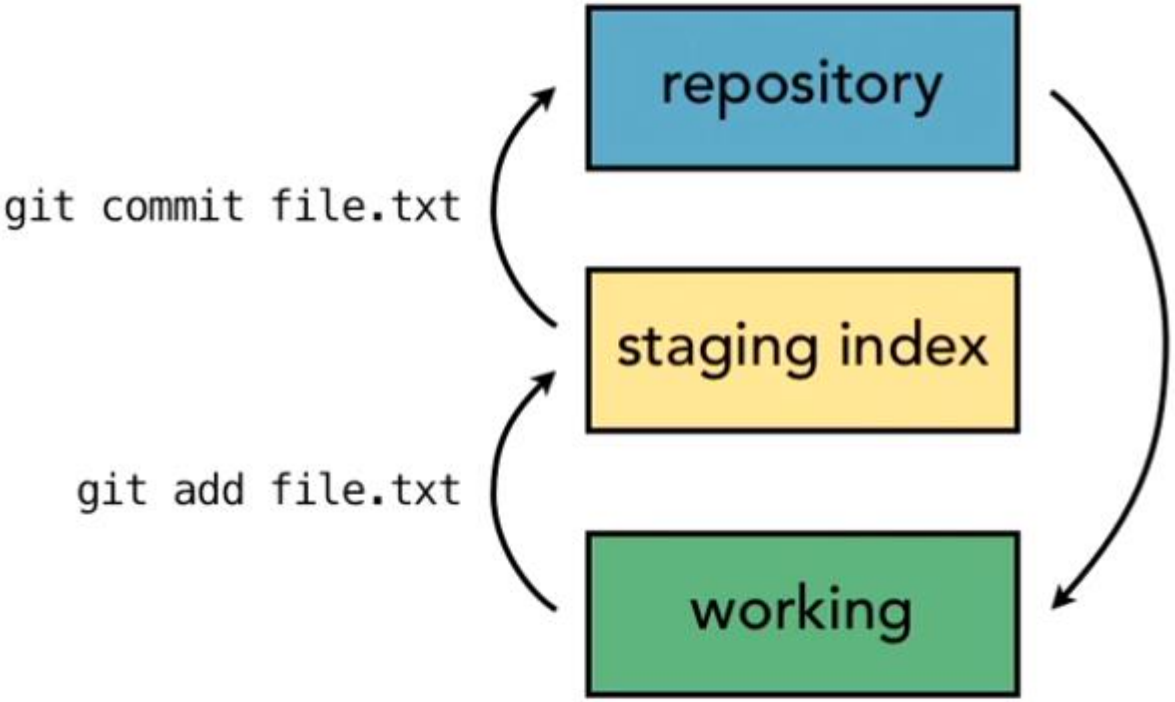
Repo 3
A,B,C,D,E,F

Why GIT

- Distributed version control system
  - No need to communicate with a single server
    - Faster
    - No network access required
    - No single failure point
  - Encourages collaborative development (“forking”)

- Open Source and free
- Compatible with UNIX like systems and Windows
- Faster than other SCMs

GIT Architecture of 3 trees



GIT - Getting started

- GIT Configuration
  - System / User / Project level
  - Listing configurations

```
git config --system
git config --global
git config
git config --list
```

- Initializing a new repository
  - Create a new repository on local
  - Browse files added by GIT

```
git init
```

- Adding new files
  - Check git status
  - Add file to staging area
  - Commit to Repo
  - View log

```
git status
git add
git commit
git log
```

- Ignoring files using .gitignore

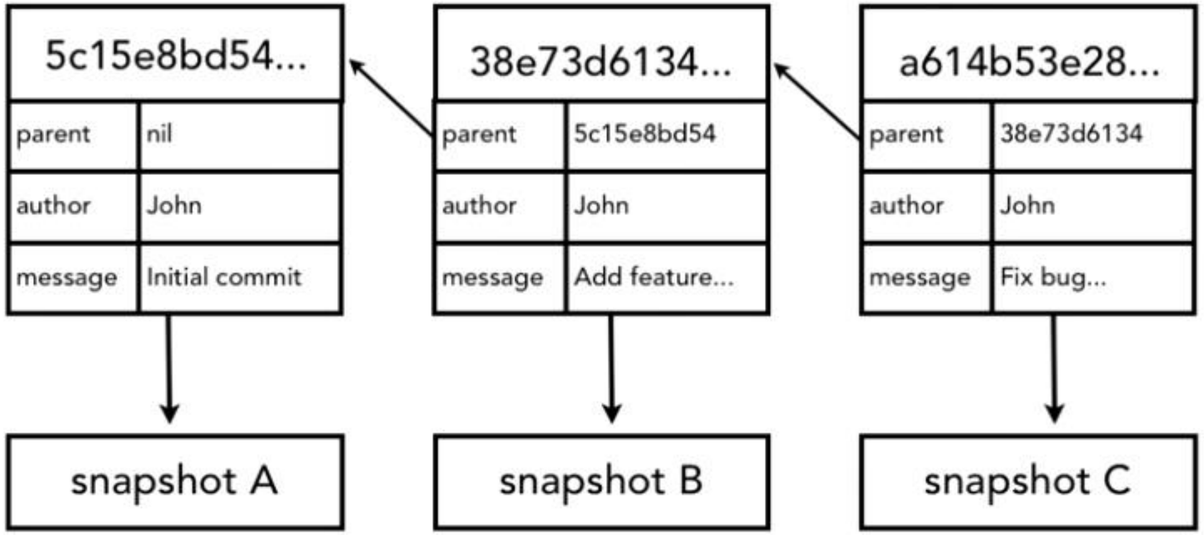
Labs Exercises

- Exploring GIT Configuration
- Create a new GIT repository on local machine
- Adding files to the local GIT repository
- Adding .gitignore file to the repository

How GIT manages and stores commits

Every change set that is committed is converted to a checksum using SHA-1 hash algorithm

SHA id is 40 character hexadecimal string



5c15e8bd54...

parent	nil
author	John
message	Initial commit

snapshot A

38e73d6134...

parent	5c15e8bd54
author	John
message	Add feature...

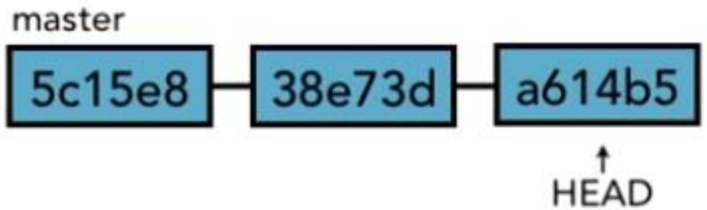
snapshot B

a614b53e28...

parent	38e73d6134
author	John
message	Fix bug...

snapshot C

HEAD Pointer points to the **tip** of the **current** branch on **repository**



Editing / Deleting / Renaming / Undoing changes

- Editing files
  - Viewing git status
  - Committing changes directly to repo
  - Viewing diff between repo and working
  - Viewing diff between repo and staging

```
git commit -a
git diff
git diff --staged
```

- Deleting / renaming
  - Delete file using git rm
  - Rename file using git mv

```
git rm
git mv
```

- Undo changes
  - To working directory
  - To staging index
  - Amend last commit
  - Retrieve older version
  - Revert commit

```
git checkout
git reset HEAD
git commit  --amend
git checkout  <SHA> -- <filename>
git revert <SHA>
```

Branching and merging

- Branching
  - Create branch
  - Switch branch
  - Show all branches
  - View HEAD pointers of all branches
  - Compare tips of two branches
  - Delete branch

```
git branch
git branch <name>
git checkout –b <branch>
git diff master..branch
git branch --merged
```

- Merge
  - Fast forward and True Merge
  - Aborting a merge
  - Resolving conflict manually

```
git merge
git merge --no-ff
git merge --ff-only
git merge --abort
```

- Tagging
  - Lightweight and annotated tags

```
git tag –a
git tag
git checkout <tag>
git tag -d
```

Lab Exercises

- Branching Merging

- Resolving Conflicts

- Resetting Changes
  - Soft Reset
  - Hard Reset

- Tagging

Referencing commits, navigating tree, using commit log

- Referencing commits “tree-ish”
  - Full / Short SHA
  - HEAD pointer
  - Branch / tag reference
  - ancestory

```
HEAD^, master^, 6s5f789^, HEAD~2
```

- Navigating the tree

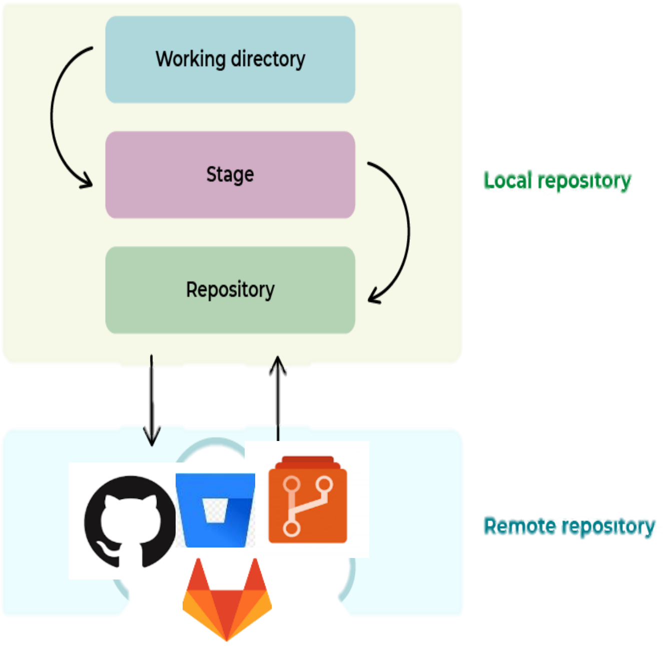
```
git ls-tree HEAD
git ls-tree master^
```

- Commit log
  - Viewing log
  - Showing details of a commit
  - Comparing commits

```
git log --oneline --graph --all –decorate
git show
git diff
```

Working with Remote Repositories

- Remote Repositories are used in GIT for sharing and exchanging code between Contributors
- Local Repo in git reside on individual laptop, while remote repositories are hosted on servers
- Contents from the local repo are pushed to remote and contents from remote repo are pulled to local repo
- Contents are then merged to a specific directory using Pull Request
- Rules can be set to ensure right content are pushed.

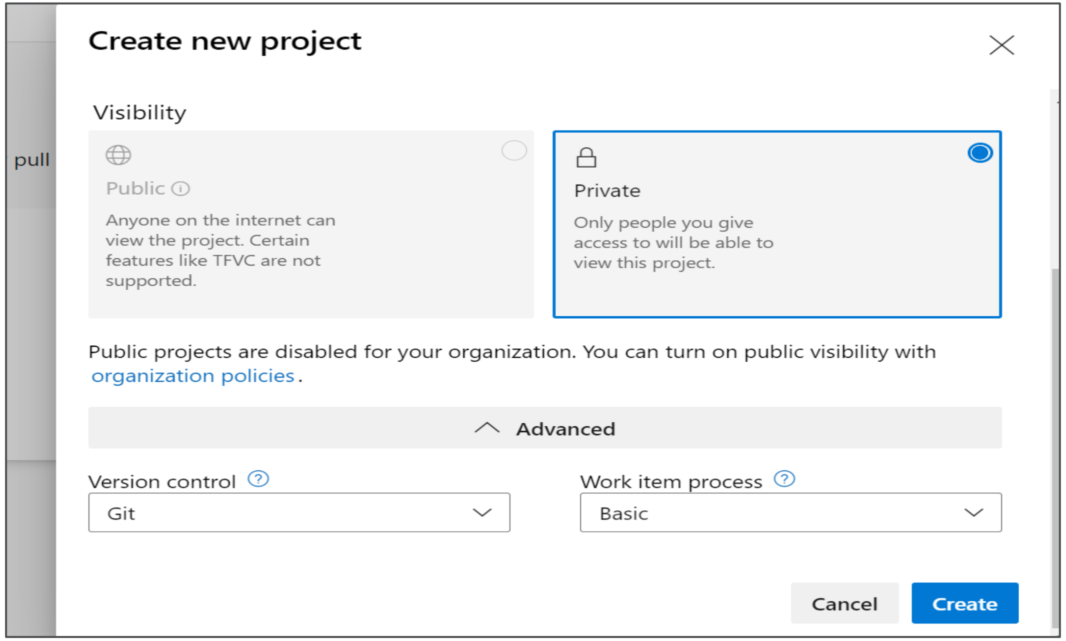


Azure Repos

- To manage source code
- Manages the code by providing
  - Branch management
  - Tracking the changes
  - Pull Request and pushes
- Azure Repos supports two types of version control
  - Distributed - Git
  - Centralized - Team Foundation Version Control (TFVC)
- Azure Git provide all the standard functionalities of standard Git
  - Integrate with IDE --- Eclipse, VS Code etc
  - Repo management including forking and cloning
  - Pull Request cycle
  - Branch management including Chery pick , Squash, rebase
  - Branch policies and permission
- GitHub Repos can also be imported into Azure Repos

Steps to Create Repos in Azure Project

- Repos can be created
  - At the time of Project Creation
  - From Project Setting ->Repositories
- Once repo is created the repo URL can be used to clone in git environment /IDE
- Changes made in the local git can be pushed to Repos and merged with the existing content



Create new project

Visibility

○ Public
Anyone on the internet can view the project. Certain features like TFVC are not supported.

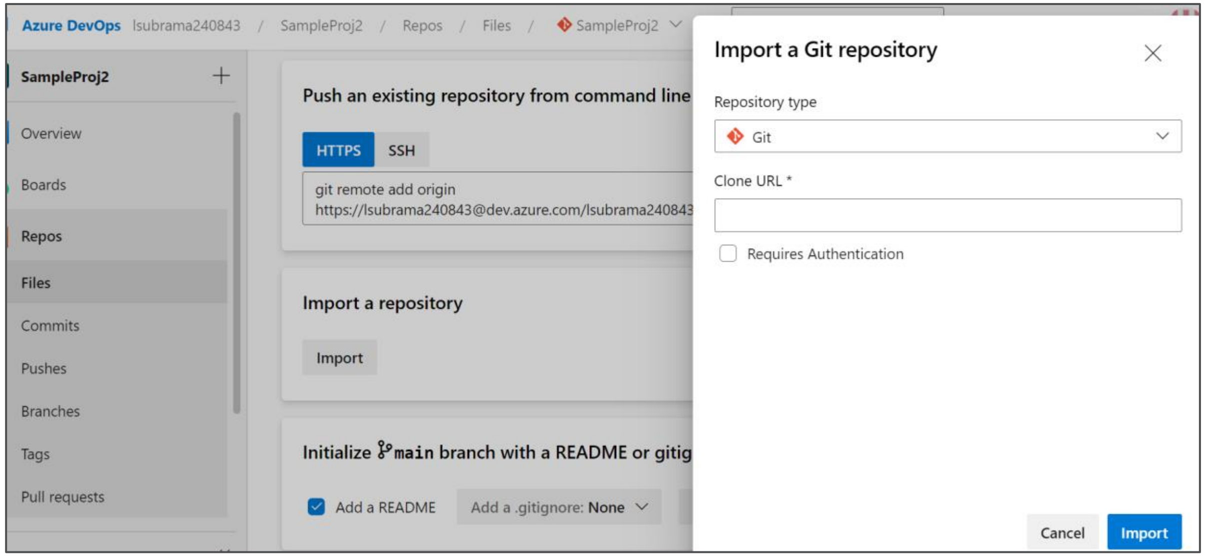
● Private
Only people you give access to will be able to view this project.

Public projects are disabled for your organization. You can turn on public visibility with organization policies.

Advanced

Version control: Git

Work item process: Basic



Azure DevOps / lsubrama240843 / SampleProj2 / Repos / Files / SampleProj2
SampleProj2
Overview
Boards
Repos
Files
Commits
Pushes
Branches
Tags
Pull requests

Push an existing repository from command line

HTTPS SSH

```
git remote add origin
https://lsubrama240843@dev.azure.com/lsubrama240843
```

Import a repository

Import

Initialize main branch with a README or gitig

☑ Add a README   Add a .gitignore: None

Import a Git repository

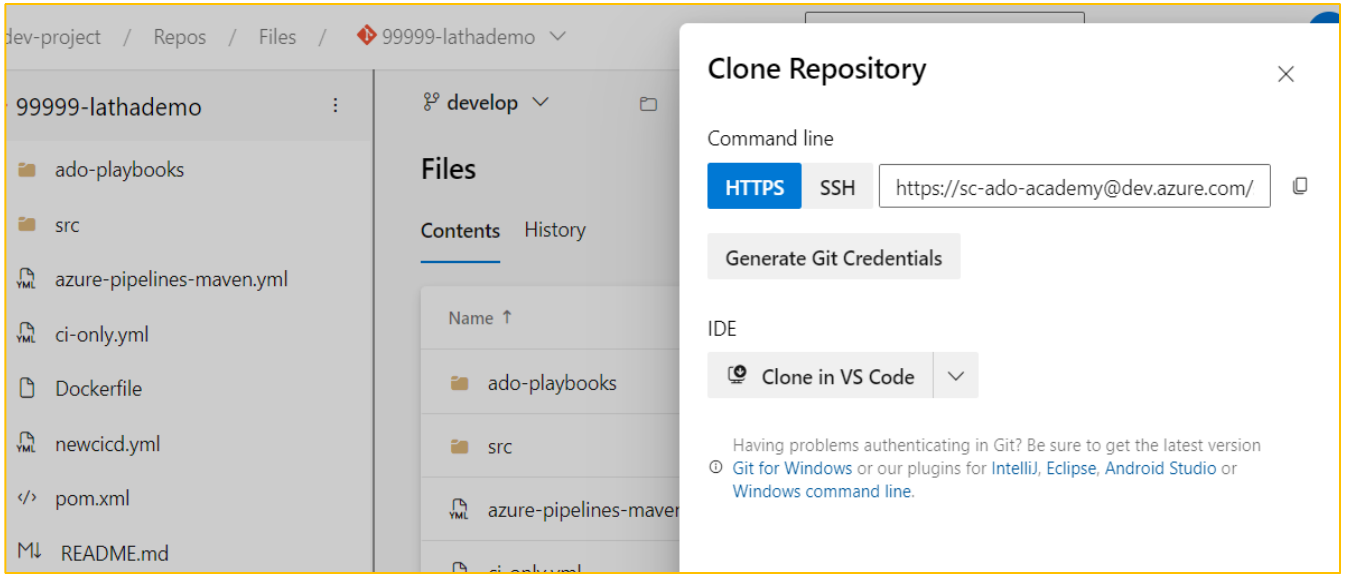
Repository type: Git

Clone URL *

□ Requires Authentication

Remotes – Cloning Repo in local git

- Get the URL of the repo https://sc-ado-academy@dev.azure.com/sc-ado-academy/dev-project/_git/99999-lathademo
- Clone the repository on your local git using git clone



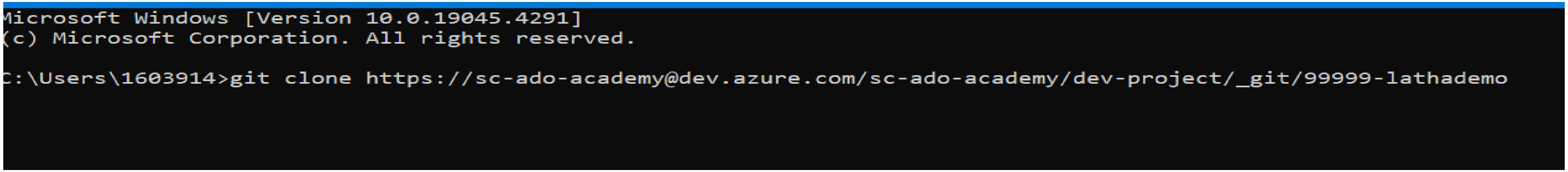
Clone Repository

Command line

https://sc-ado-academy@dev.azure.com/

IDE

Having problems authenticating in Git? Be sure to get the latest version Git for Windows or our plugins for IntelliJ, Eclipse, Android Studio or Windows command line.



```
Microsoft Windows [Version 10.0.19045.4291]
(c) Microsoft Corporation. All rights reserved.

C:\Users\1603914>git clone https://sc-ado-academy@dev.azure.com/sc-ado-academy/dev-project/_git/99999-lathademo
```

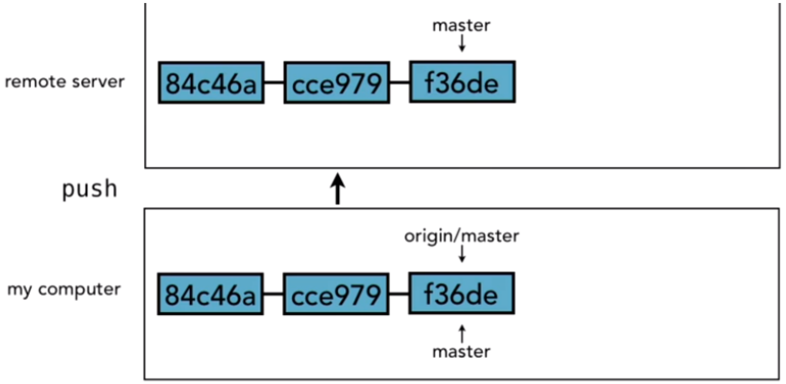
Lab Exercises

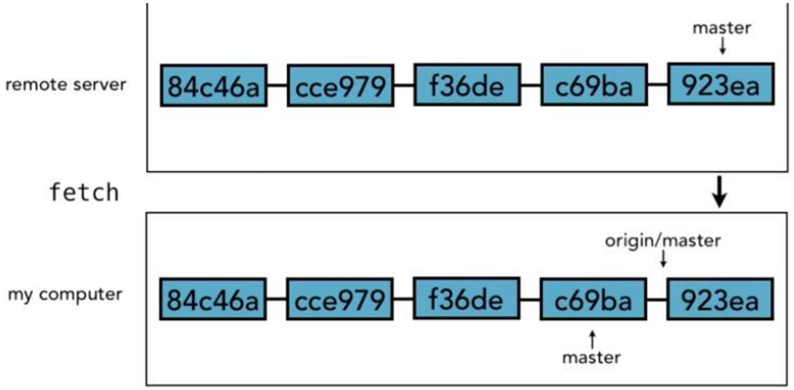
- Creating PAT
- Logging to Azure Portal
- Cloning the Repo

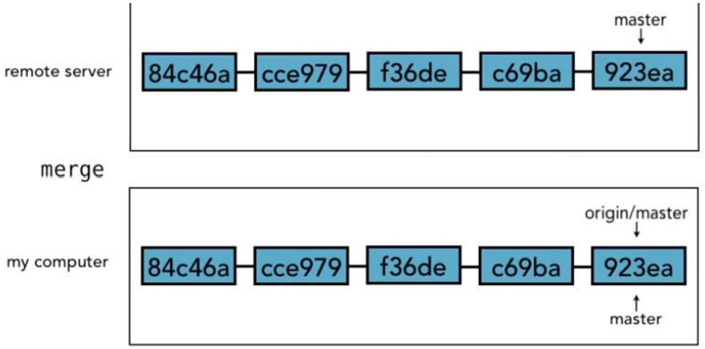
Working with remotes

- Understanding “origin”
- Listing remote branches
- Tracking and Non tracking branches

```
git push
git branch -r
```







Lab Exercises

- Creating WorkItem for the developmental activities
- Creating a local branch and committing work
- Pushing to Remote Repo

Pushing / Fetching changes to/from Remote

- Pushing changes to remote
- Viewing tracking branch
- Viewing changes on Remote
- Push a new local branch to remote

```
git push
git push -u
```

- Fetching changes from remote
- Viewing changes on tracking branches

```
git fetch
```

```
git merge
```

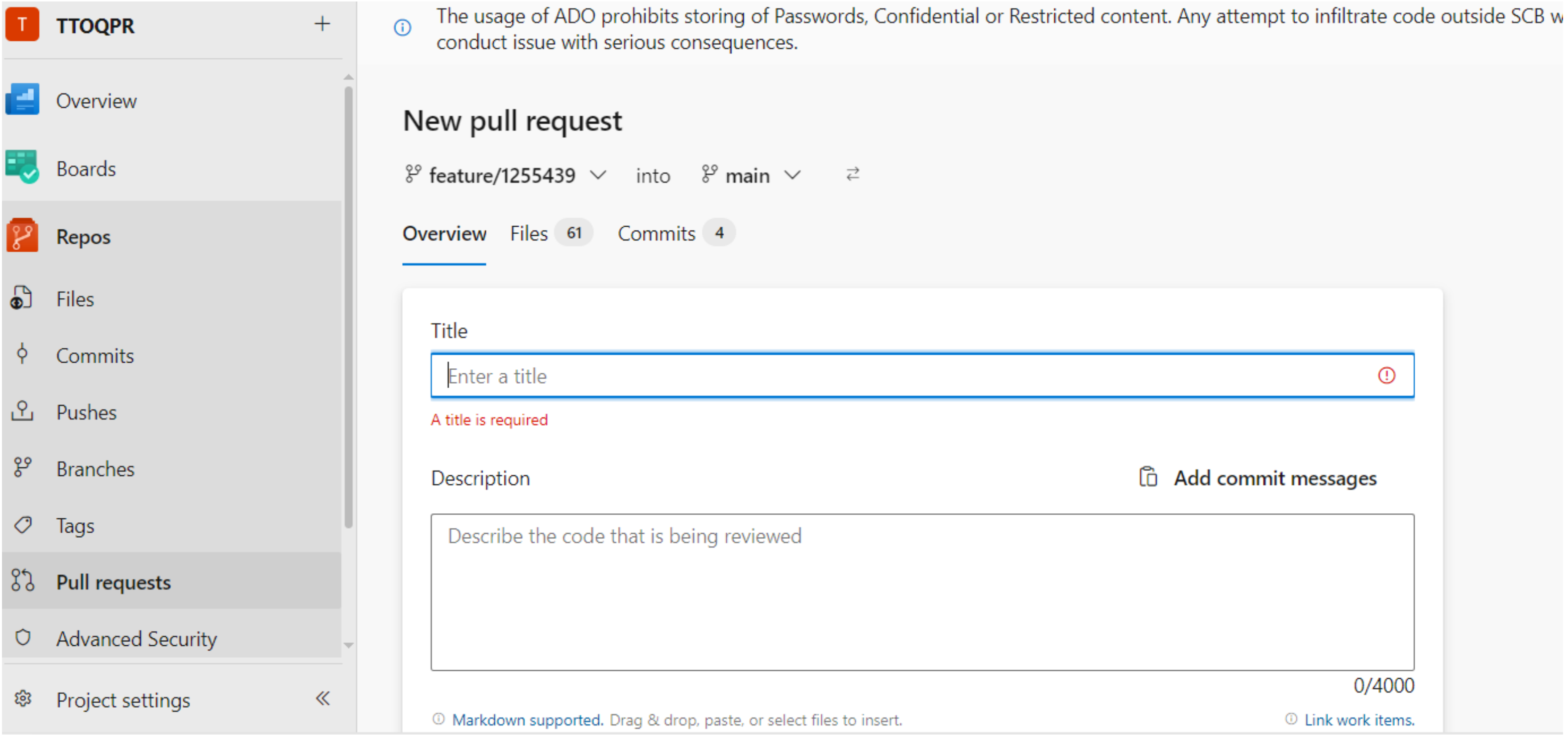
- Merging changes from remote

Notes:

- Git push will fail unless it is a fast forward merge on the origin
  - Then developer has to pull the latest and merge and re-push
- Avoid doing a push with force option as it will overwrite everything on remote with your changes

Collaborating using pull request

- Concept to let the concerned know of the changes made in a branch
- Once a Pull Request is created, discussion and review can be initiated
- Post Approval the changes are merged into a specific branch



The usage of ADO prohibits storing of Passwords, Confidential or Restricted content. Any attempt to infiltrate code outside SCB w
conduct issue with serious consequences.

New pull request

feature/1255439 into main

Title

Enter a title

A title is required

Description

Describe the code that is being reviewed

0/4000

Markdown supported. Drag & drop, paste, or select files to insert.

Link work items.

Lab Exercises

- Initiating Pull Request and merging changes
- Resolving Merge Conflicts

Branching strategies

- **Dev branch** / enhancement branch – for the actual dev work
- **Feature branch** for new feature development (local to developer’s repo ; merged to develop and pushed to repo)
- **Release branch** – used only for new releases – branches off from develop and merged back to develop and master
- **Hot fix branches** for production fixes – branches off from master and merged back to develop and master
- **Tagging done on main**

GIT Branching

axess academy

Introduction to ADO Pipeline

Campus Curriculum

standard chartered

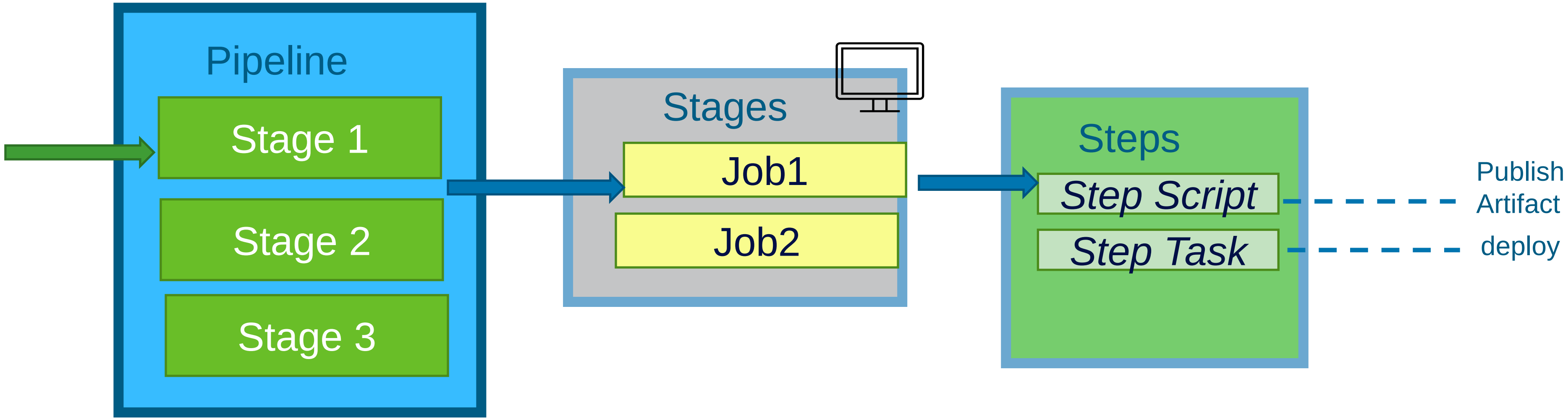
10/07/2025

Azure Pipeline

- Azure Pipelines combine CI and CD to build, test and deploy to any environment
- Supports many languages & project types
- One of the key features of Azure DevOps
- Azure Pipeline supports the following
  - Continuous Integration (CI) :
    - Used by Development Teams to automates merge, test, and build code
  - Continuous Delivery:
    - Code is built ,tested and deployed to multiple environments – production /Non-production environments, including infrastructure & apps
  - Continuous Testing
    - Supports preferred test type and test framework.
    - Rich analytics & Report
  - Package Formats
    - Supports publishing NuGet (.NET) , npm, Maven packages in Azure Package manager repository or any other repository (docker hub /antifactory for example)

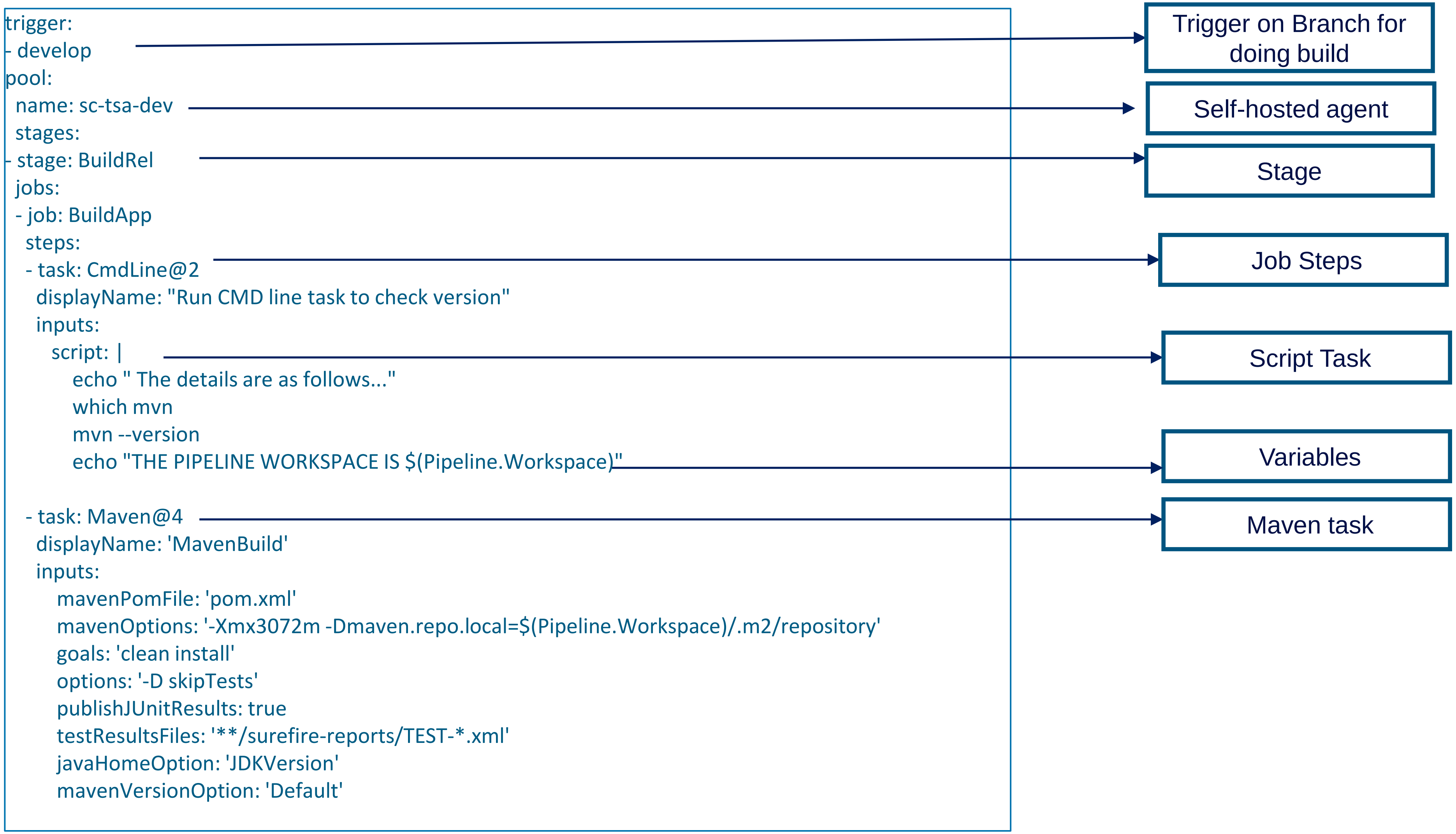
Azure Pipelines Details

- Azure Pipeline is defined “declaratively” as a YAML file, which can be created using pipeline editors
- The Pipelines can be executed by Trigger(s) – Code Merge/Pull Request approval/ Manual etc.
- A Pipeline can have multiple stages, each stage have multiple jobs
- A job has multiple tasks , each task runs in an “agent”
- Agent is a computing resource with s/w to run build jobs (Pods or VMs)



- Agent can be self-hosted or Microsoft-hosted

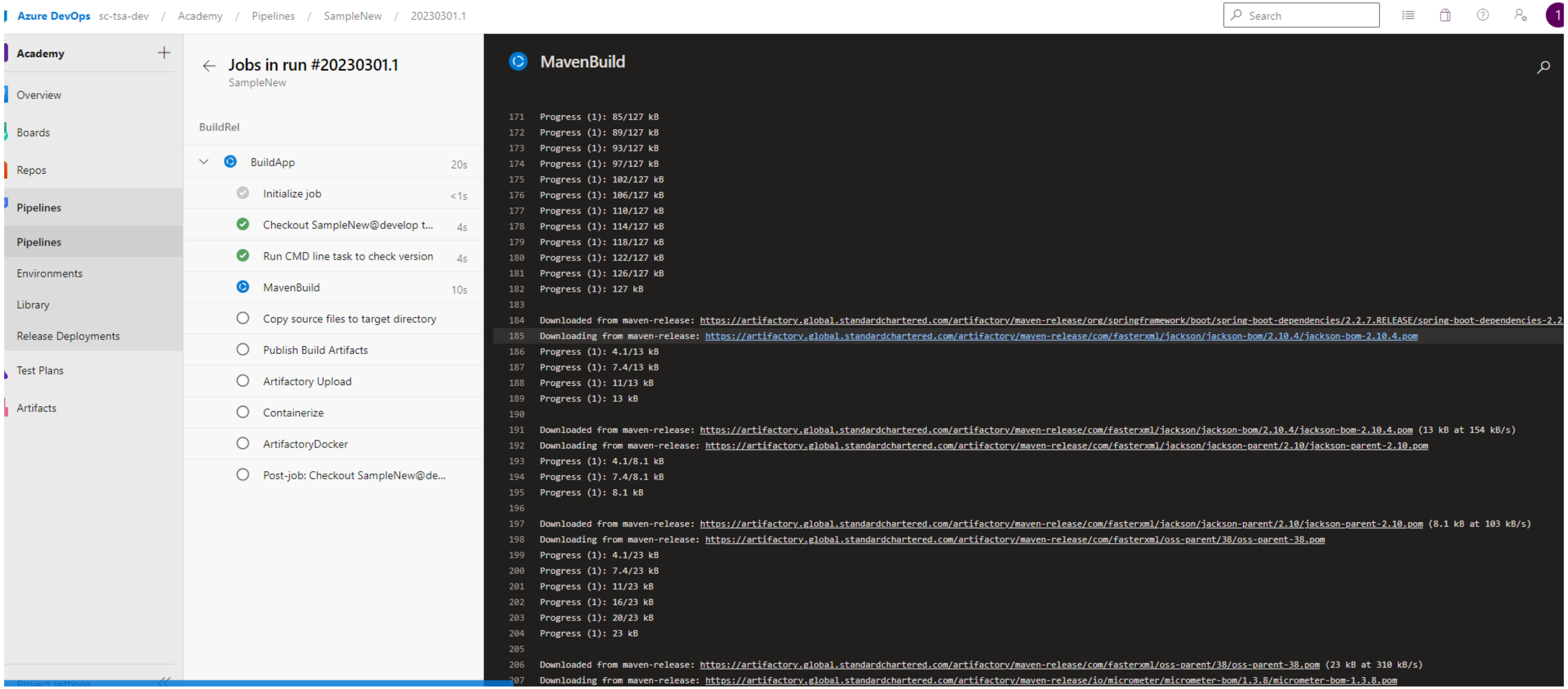
Explaining Azure Pipeline build YAML



```
trigger:
- develop
pool:
 name: sc-tsa-dev
 stages:
- stage: BuildRel
 jobs:
 - job: BuildApp
  steps:
  - task: CmdLine@2
   displayName: "Run CMD line task to check version"
   inputs:
     script: |
       echo " The details are as follows..."
       which mvn
       mvn --version
       echo "THE PIPELINE WORKSPACE IS $(Pipeline.Workspace)"

  - task: Maven@4
   displayName: 'MavenBuild'
   inputs:
     mavenPomFile: 'pom.xml'
     mavenOptions: '-Xmx3072m -Dmaven.repo.local=$(Pipeline.Workspace)/.m2/repository'
     goals: 'clean install'
     options: '-D skipTests'
     publishJUnitResults: true
     testResultsFiles: '**/surefire-reports/TEST-*.xml'
     javaHomeOption: 'JDKVersion'
     mavenVersionOption: 'Default'
```

Azure Pipeline Run



Jobs in run #20230301.1

SampleNew

BuildRel

BuildApp 20s

Initialize job <1s

Checkout SampleNew@develop t... 4s

Run CMD line task to check version 4s

MavenBuild 10s

Copy source files to target directory

Publish Build Artifacts

Artifactory Upload

Containerize

ArtifactoryDocker

Post-job: Checkout SampleNew@de...

MavenBuild

```
Progress (1): 85/127 kB
Progress (1): 89/127 kB
Progress (1): 93/127 kB
Progress (1): 97/127 kB
Progress (1): 102/127 kB
Progress (1): 106/127 kB
Progress (1): 110/127 kB
Progress (1): 114/127 kB
Progress (1): 118/127 kB
Progress (1): 122/127 kB
Progress (1): 126/127 kB
Progress (1): 127 kB

Downloaded from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/org/springframework/boot/spring-boot-dependencies/2.2.7.RELEASE/spring-boot-dependencies-2.2
Downloading from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/com/fasterxml/jackson/jackson-bom/2.10.4/jackson-bom-2.10.4.pom
Progress (1): 4.1/13 kB
Progress (1): 7.4/13 kB
Progress (1): 11/13 kB
Progress (1): 13 kB

Downloaded from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/com/fasterxml/jackson/jackson-bom/2.10.4/jackson-bom-2.10.4.pom (13 kB at 154 kB/s)
Downloading from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/com/fasterxml/jackson/jackson-parent/2.10/jackson-parent-2.10.pom
Progress (1): 4.1/8.1 kB
Progress (1): 7.4/8.1 kB
Progress (1): 8.1 kB

Downloaded from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/com/fasterxml/jackson/jackson-parent/2.10/jackson-parent-2.10.pom (8.1 kB at 103 kB/s)
Downloading from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/com/fasterxml/oss-parent/38/oss-parent-38.pom
Progress (1): 4.1/23 kB
Progress (1): 7.4/23 kB
Progress (1): 11/23 kB
Progress (1): 16/23 kB
Progress (1): 20/23 kB
Progress (1): 23 kB

Downloaded from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/com/fasterxml/oss-parent/38/oss-parent-38.pom (23 kB at 310 kB/s)
Downloading from maven-release: https://artifactory.global.standardchartered.com/artifactory/maven-release/io/micrometer/micrometer-bom/1.3.8/micrometer-bom-1.3.8.pom
```

]]>

Lab Exercises

- Creating a Simple Pipeline
- Creating a Maven Build Pipeline
- Using the Organization Standard Maven Build Pipeline