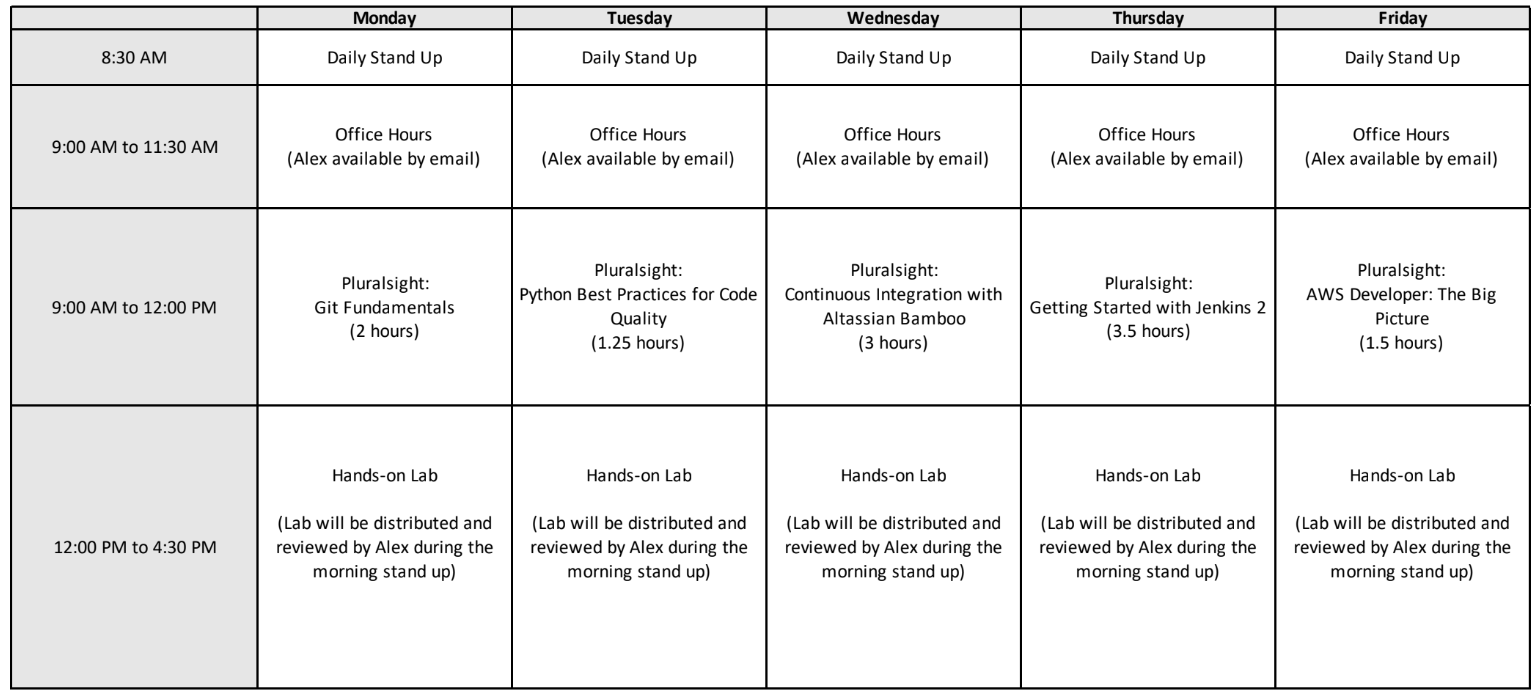
Use the class cheat sheet and materials for reference: <https://bitbucket.org/lmcohort2/materials/src/master/cheatsheet.md>



## Jenkins Pipelines

Jenkins is a build automation tool that allows you to run complex tasks with the push of a button.

Helpful links:

<https://jenkins.io/doc/book/pipeline/jenkinsfile/>

<https://jenkins.io/doc/book/pipeline/syntax/#input>

<https://jenkins.io/doc/pipeline/tour/deployment/>

<https://jenkins.io/doc/pipeline/steps/>

### Create a pipeline in Jenkins for simple flask app

Before attempting to have jenkins control our app running on the workspace, we need to clean up things created by bamboo.

|  |
| --- |
| # Stop the app  sudo kill $(ps -ef |grep -v grep | grep -w "python.\*hello.py" |awk '{print $2}')  # Remove log file  sudo rm /tmp/log.txt |

1. Install Jenkins

<https://medium.com/@itsmattburgess/installing-jenkins-on-amazon-linux-16aaa02c369c>

Our installation of jenkins creates a "jenkins" user that cannot log in and does not have a home directory. To install packages for the all users, do the following:

|  |
| --- |
| sudo python3 -m pip install --user flake8 pytest flask |

This should result in the following two command being available:

*/usr/local/bin/flake8* and */usr/local/bin/py.test*

2. Create a Declarative Pipeline

We want to create a new Job in Jenkins of type Pipeline. This should be a declarative pipeline of the structure below. You may need to make your source code repo public unless you want to configure your pipeline with ssh authentication (bonus points if you do).

|  |
| --- |
| pipeline {  agent any  stages {  stage('Checkout Code') {  steps {  // TODO: Checkout source code, https://jenkins.io/doc/pipeline/steps/git/  }  }  stage('Test Code') {  steps {  echo 'Testing..'  }  }  stage('Deploy to Dev') {  steps {  echo 'Deploying to Dev Environment....'  }  }  }  } |

3. Add deployment stages to existing pipeline requiring a manual button push for approval.

You can do this with ssh commands using the "[SSH Pipeline Steps](https://plugins.jenkins.io/ssh-steps)" plugin. To install, visit the Jenkins UI, Manage Jenkins -> Manage Plugins -> Available. Then search for ssh pipeline and install into jenkins without restarting. \*\*NOTE\*\* you must wrap your variables and sshCommand steps in a 'script' directive as below:

|  |
| --- |
| steps {  script {  def test\_remote = [:]  test\_remote.name = 'test'  test\_remote.host = '104.196.198.17'  test\_remote.user = 'ubuntu'  test\_remote.password = 'secret'  test\_remote.allowAnyHosts = true    sshCommand remote: test\_remote, command: 'kill $(ps -ef |grep -v grep | grep -w "python.\*hello.py" |awk \'{print $2}\') || true'  sshPut remote: test\_remote, from: 'hello.py', into: '.'  //TODO etc  }  } |

<https://jenkins.io/doc/pipeline/steps/ssh-steps/>

OR

You can use the plugin "Ansible" installed the same way, if you have a playbook for deployment

<https://jenkins.io/doc/pipeline/steps/ansible/>

|  |
| --- |
| stage('Deploy to Test?') {  steps {  // Ask user if they are ready to deploy to Test  }  }  stage('Deploy to Test') {  steps {  echo 'Deploying to Test Environment....'  }  }  stage('Deploy to Production?') {  steps {  // Ask user if they are ready to deploy to Production  }  }  stage('Deploy to Production) {  steps {  echo 'Deploying to Test Environment....'  }  } |

4. \*\*BONUS\*\* store your pipeline in a jenkinsfile along side the code in bitbucket