Overview

This lab is intended to familiarize you with the Bamboo UI and terminology by creating a Build Plan for testing our astronomy app.

As a review, Bamboo relies on a few different concepts of the following structure to organize and run CI/CD pipelines.
Projects
  -> Build Plans
    -> Stages
      -> Jobs
        -> Tasks

A **project** is like a folder that contains everything related to an application. An application may have multiple build plans associated with it, perhaps one build plan runs all tests while another generates documentation for the project. It is possible to have the execution of one build plan to trigger another build plan or to do everything in a single build plan. One advantage of breaking something into its own build plan is that it can be run independently. Generating application documentation is a good example of a Build Plan you may have for an application that is separate from a Build Plan representing a CI/CD pipeline for the application itself, but may live in the same project.

Stages are run sequentially but all jobs in a stage are run in parallel. Tasks within a job are also run sequentially. Therefore, if you want to run two things at once, they should be different jobs in the same stage.

1. Create a Project

→ Explore the Bamboo UI for how you might create a project. Then create a project named "Astronomy".

2. Create a Build Plan

→ Create a new Build Plan named "test" in your project.

3. Create a Stage

→ Rename or replace the "Default Stage" created for you and create a stage called "static analysis"

4. Add Jobs

→ Add 2 jobs "style check" and "pytest"

5. Configure tasks

We have installed Bamboo in a way that Jobs are run directly in the AWS Workspace. This is referred to as a Local Agent. Therefore, we can use any commands or tools installed on the Workspace itself from within our Tasks. An alternative would be to install a Bamboo agent on a remote machine and our Bamboo server could then run Jobs on said machine.

→ Add appropriate Tasks to run flake8 and pytest in their respective Jobs. Explore what types of tasks are available and consider what might make sense for running our flake8 and py.test commands.

6. Add a trigger

→ Under "Plan Configuration" ensure there is a trigger to run the pipeline when new code is pushed into the repository located in Bitbucket.