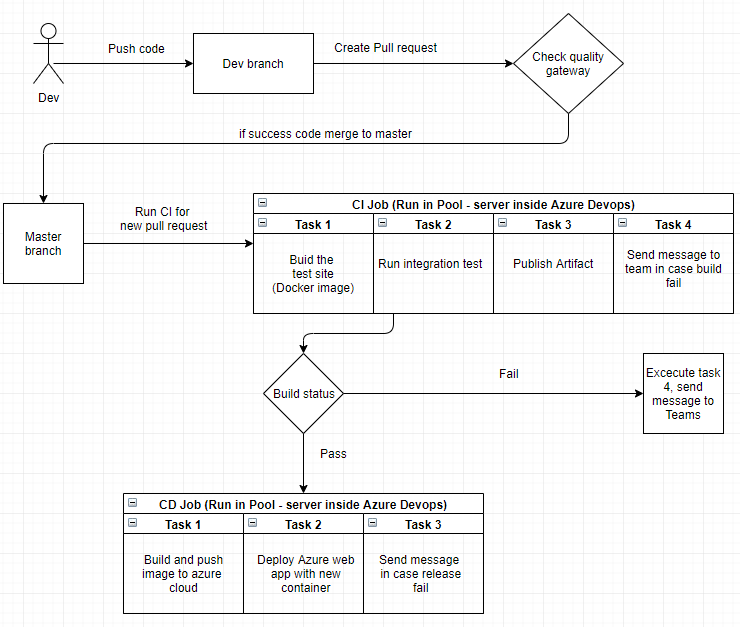
**CI/CD for auto deploy VinHR test site**

This document contains the CI/CD design for VinHR test site.



The flow includes:

1. Dev push code to dev branch
2. Dev create new pull request
3. Code check-in to master if pull request pass the quality gateway
4. CI trigger event pull request approved and run CI job (if fail, send message to Teams)
5. CD trigger event CI job success and run CD job (if fail, send message to Teams)

In this document, we focus on the process of CI/CD. The test site repo deploys base on Docker, I created a Dockerfile, and it will not change in the near future.

1. CI

CI will trigger the master branch when a pull request was merged to master (set up in Trigger and Run Job Agent). Output of CI is the artifact for source code in case build success or message in Microsoft Teams in case build fail.

CI contain four tasks:

* Build and Run image from Docker file
* Run integration test base on the Docker image that was built (for now this one just is simple test that it passes in case status return 200 and fail in another )
* Public source code to Artifact
* Send message to teams in case there was a fail job

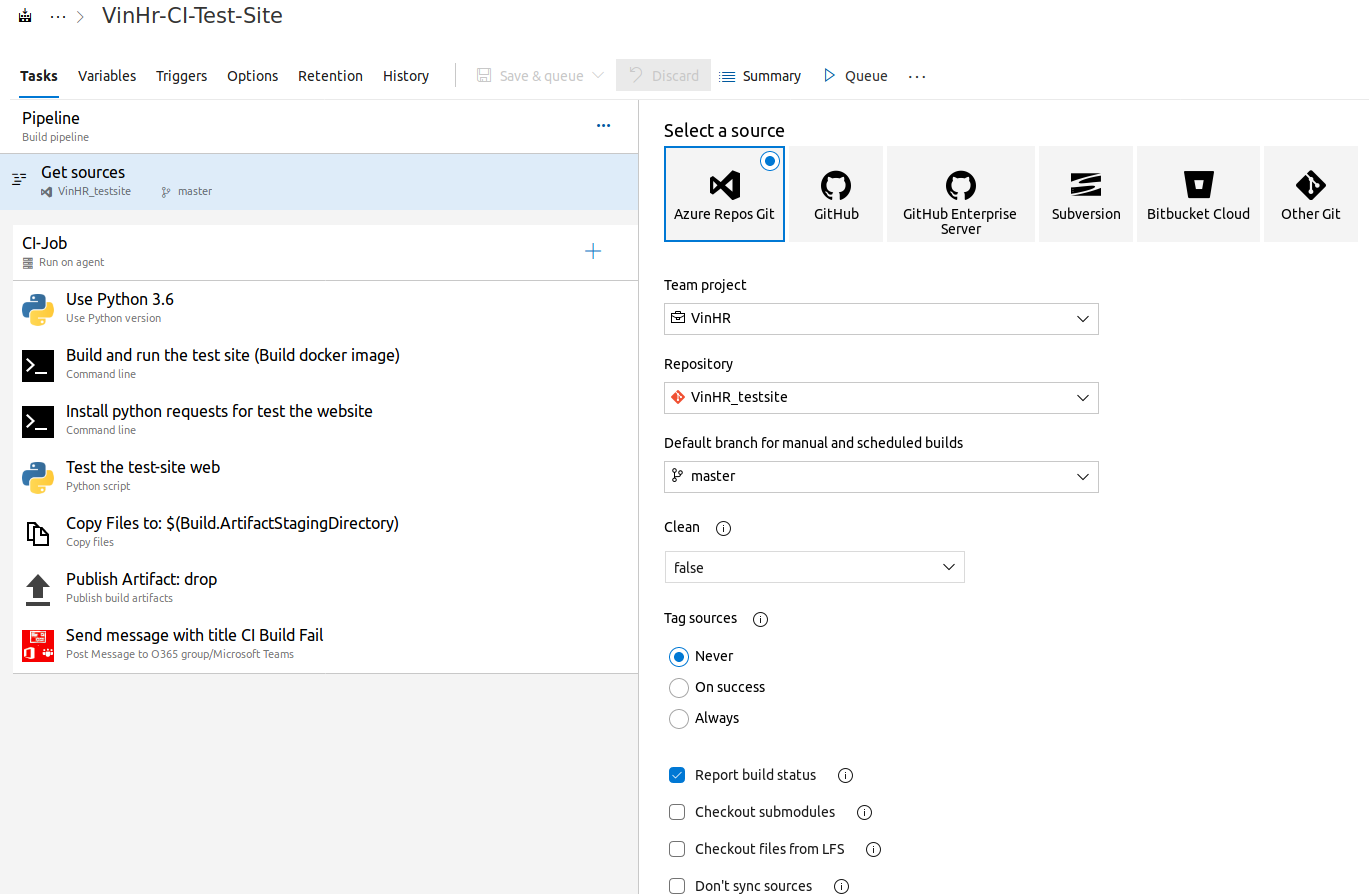
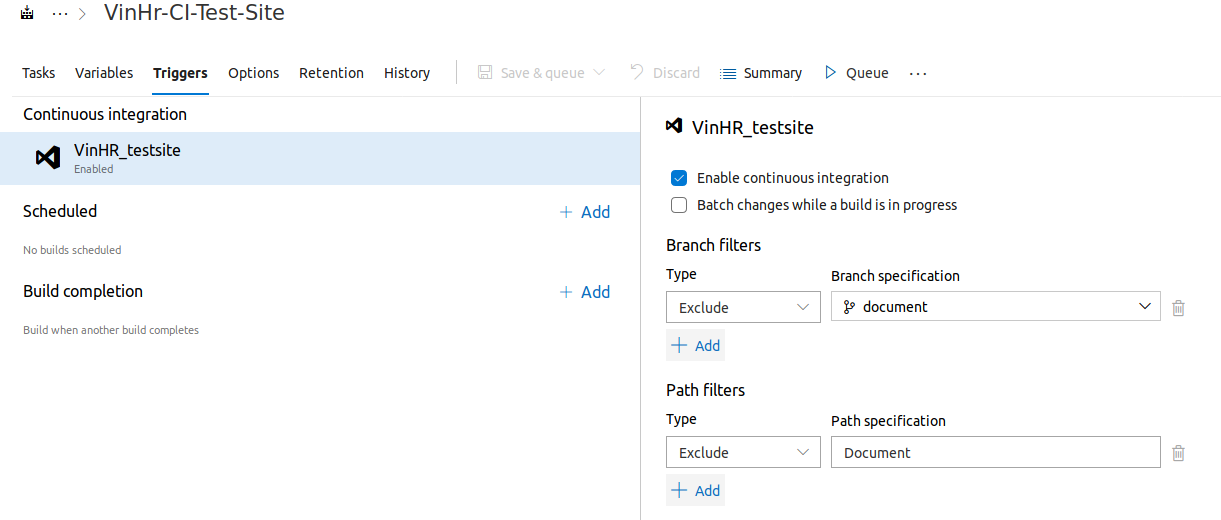


Figure 1: All task in CI test site

In CI, I set up the trigger is exclude the folder Document when new pull request is approved to master. It means that if a developer commit a Document to master, the CI/CD pipeline will not execute.



1. CD

CD will trigger an event when an artifact creates (CI build success). Output of CD is web-site (link is [here](https://test-site-docker.azurewebsites.net/))

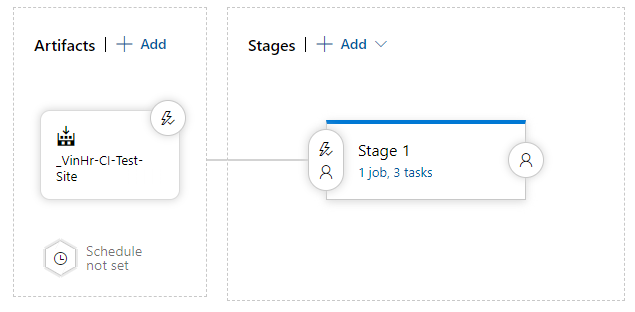


Figure 2: CD architecture

CD contains the artifact trigger and state that includes three tasks. The three tasks are:

* Build and push image from artifact to Azure cloud
* Deploy image to Web App (a service in Azure)
* Send message to Microsoft Teams in case release fail

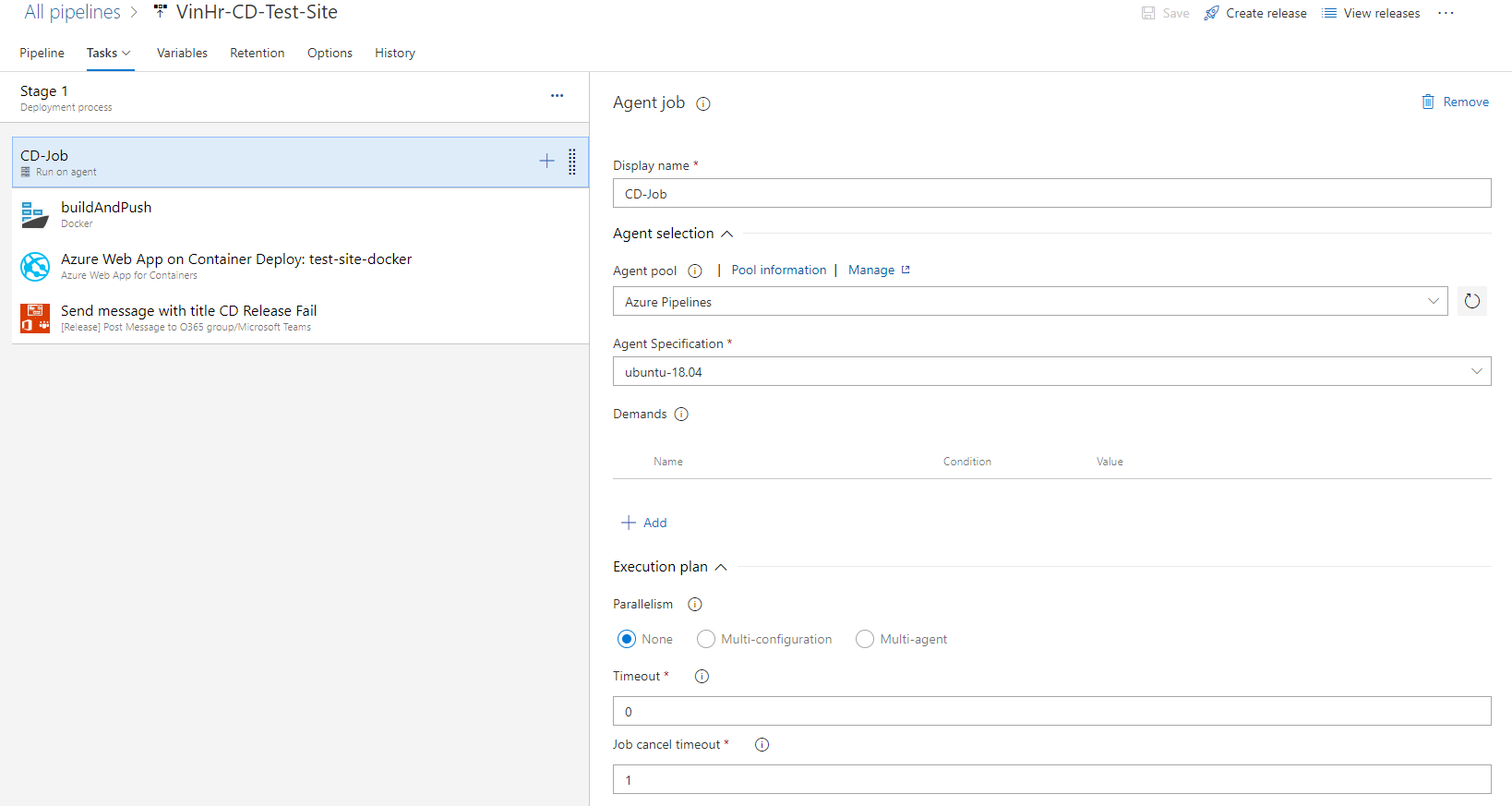
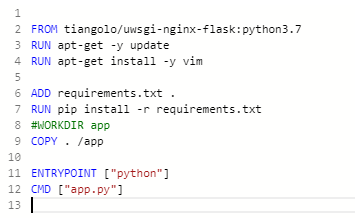


Figure 3: CD state jobs

1. Some note about writing Dockerfile for the test site

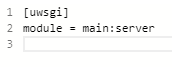
The DevOps must know about the command in Dockerfile.For example

M and D had a task that they have to deploy a web to cloud using Docker technique. M and D did not know how to configure a web server (Nginx for instance) and Docker, so they decided to make a copy Dockerfile from google and here was the Dockerfile they had get



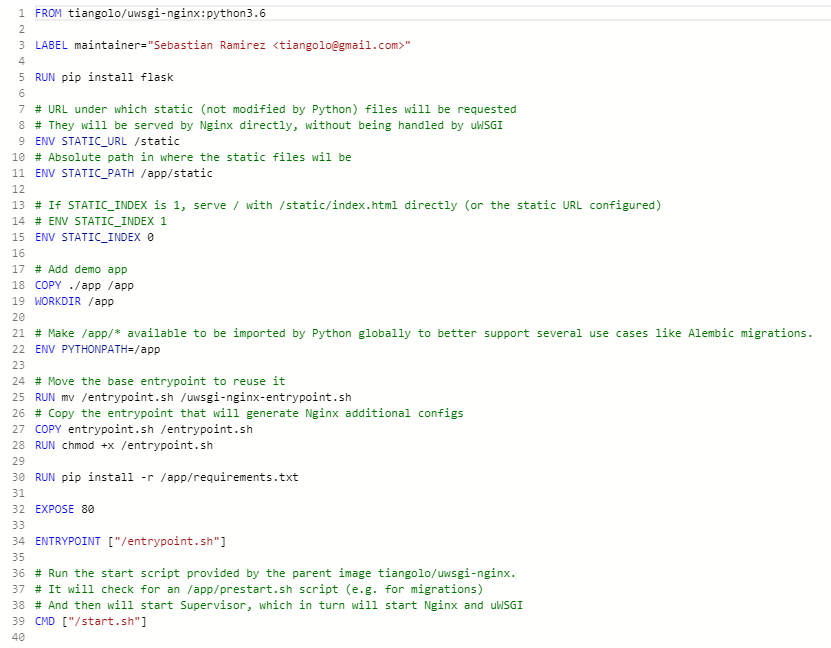
* When they tried to deploy to Azure Cloud, the web was working perfect (with one or two requests, that was OK), and they were very happy about that. However, they did not know that they made a mistake that the “CMD” and “ENTRYPOINT” started the web with the debug webserver (werkzeg). With 20 requests, I’m sure that the web site will down. Lucky that H found that and fix the Dockerfile for them. H was clone an existing Dockerfile that include Nginx and uwsgi from [here](https://github.com/tiangolo/uwsgi-nginx-flask-docker), chose version of the language (python 3.6) and modify two file: Dockerfile and uwsgi.ini

uwsgi.ini



main is file name want to execute (main.py), server is an object of framework flask (in main.py must define the variable name server that mount to flask server object)

Dockerfile



The note here is “DevOps should know how something works before deploy them.”.