

Runbook

Advanced Development Factory

Version 1.3

Development Security Operations Team

June 21, 2021

SWEN 670 9040 Software Engineering Project (2215) Summer 2021

Jeroen Soeurt,

Michelle Monfort,

Robert Wilson

Revision History

Author	Date	Reason for Change	Version
Michelle Monfort	06/21/2021	Initial draft	1.0
Rob Wilson	06/27/2021	Build Sections	1.1
Rob Wilson	06/29/2021	Update AKS section and format	1.2
Jeroen Soeurt	06/30/2021	Review, typos, small changes	1.3

Table of Contents

1	Introduction	6
1.1	Overview	6
1.1.1	System Components	6
1.2	Infrastructure	7
1.2.1	The system Infrastructure varies based on development team. The DevSecOps team has provided and general infrastructure via	7
2	GitHub	8
2.1	Master Branch: Main Branch of working components that have been previously reviewed and merged to the branch	8
2.2	Developer Branch: Branch of ideally working components to be reviewed and merged to the main branch	8
2.3	Feature Branch: Branch for developing features and creating pull request for working components to be reviewed and merged to the developer branch	8
2.4	GitHub Initial Configuration	8
2.5	Creation of Development Teams	8
2.5.1	Create Team	8
2.5.2	Add Members to the GitHub team	9
2.6	Repository Creation	10
2.7	Assign repository to development teams	10
3	GitHub Client Installation	11
3.1.1	MacOS	11
3.1.2	Windows	11
3.1.3	Linux	12
3.2	Initial Configuration	12
3.2.1	Verify Git Installation	12
3.2.2	Set Git Global Configuration	12
3.2.3	Routine Git Hub Tasks	12
3.2.4	Check out the development branch.	12
3.2.5	Create a new feature branch.	13
3.2.6	Publish To Remote	13
3.2.7	Update code, add it, commit and push changes	13
4	Submitting Code to be merged	13

4.1	Create pull request	13
4.2	Merge pull request	14
4.3	Merge a Pull Request	15
5	Microsoft Teams	15
6	Build Pipeline – GitHub Actions	16
6.1	ADF Pipeline with GitHub Actions	16
6.1.1	Add Users	17
6.2	Access the Capstone Azure DevOps Project	17
6.2.1	Creation of a continuous integration pipeline	18
7	Docker	20
7.1	Get and run Docker images	20
7.1.1	Ensure that docker images are installed	20
7.1.2	Pull docker image to local machine.....	20
7.1.3	Download and execute docker image	20
7.1.4	Get a particular version of docker	20
7.1.5	Bind Port to Host	20
7.1.6	To run docker image in detached mode	20
7.1.7	Delete an image	21
7.2	Actions with Docker containers.....	21
7.2.1	Check running docker containers	21
7.2.2	Check both running and stopped images.....	21
7.2.3	Stop docker image	21
7.2.4	Start docker image again	21
7.2.5	Delete container.....	21
7.3	Docker network.....	21
7.3.1	List networks	21
7.3.2	Create docker network for image communication	21
7.3.3	Tell container to run on a specific docker network	21
7.4	Docker Compose file (running multiple containers with different configurations)	21
7.4.1	Start images using Compose configurations	21
7.4.2	Shut down images using Compose	21
7.5	Create a Dockerfile (creating a Docker image)	22

7.5.1	Create docker image	22
8	Build Development Team Pipelines.....	22
8.1	SonarCloud	22
8.1.1	Create a Sonar Cloud account	22
8.1.2	Link Sonar Cloud to a GitHub repository	23
8.1.3	GitHub Analysis Files	24
8.1.4	From the SonarCloud set up continue	25
8.2	GitHub Actions	25
	Configure Enterprise Application.....	27
8.3	27
8.4	Setup secrets for AKS.....	29
	Deployment	Error! Bookmark not defined.
8.5	31

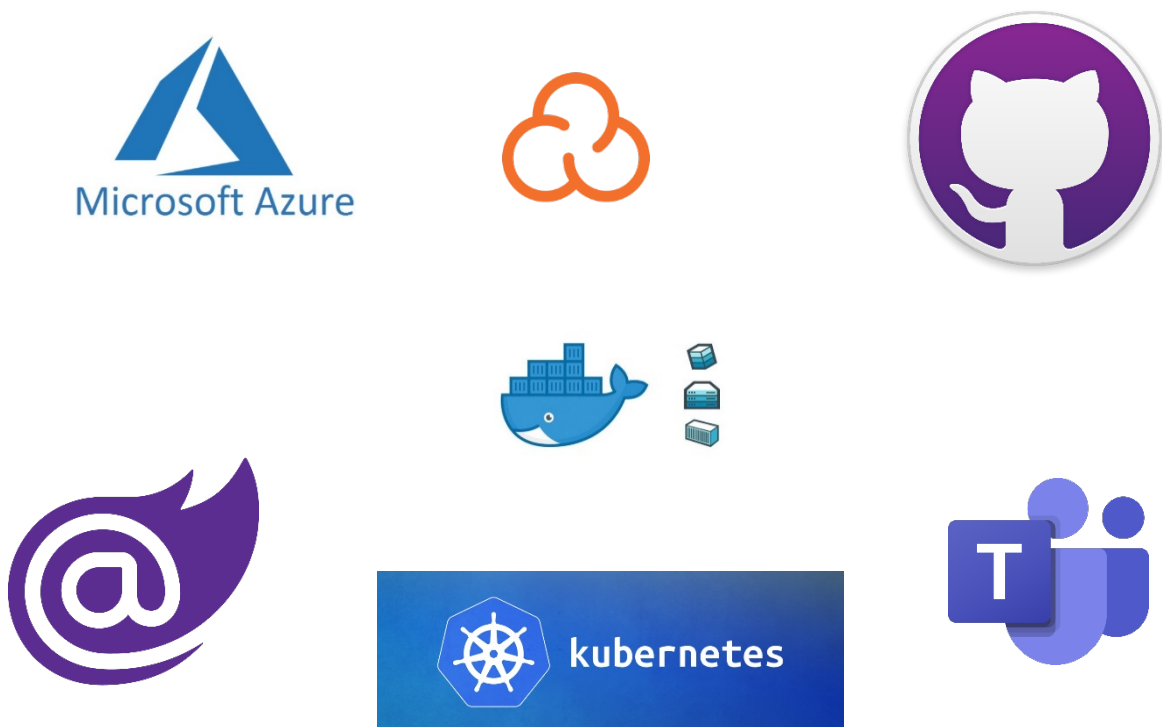
1 Introduction

1.1 Overview

1.1.1 System Components

The components used for the Advanced Development Factory (ADF) range from open source to cloud offerings. The one thing that all of these components have in common is that they are cross-platform in both development of ADF to deployment and consumption by the end user.

1.1.1.1 System Components that will be utilized.



- Docker
- GitHub
- Kubernetes
- Microsoft Azure
- Microsoft Azure DevOps
- Microsoft Teams

- Blazor
- Razor
- SonarCloud

1.2 Infrastructure

1.2.1 The system Infrastructure varies based on development team. The DevSecOps team has provided an general infrastructure via....

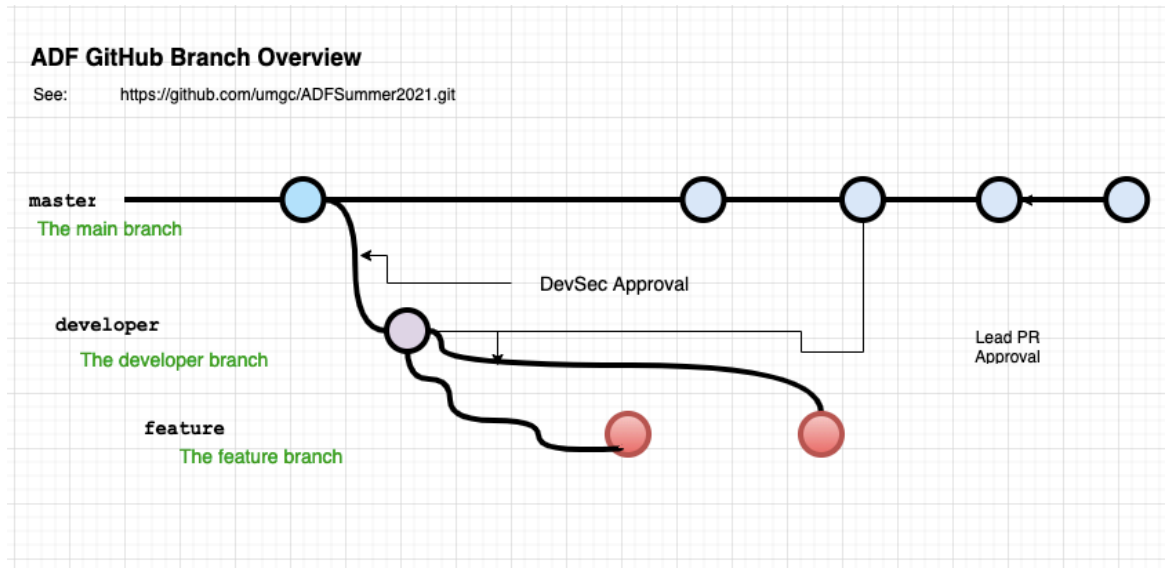
GitHub. GitHub is used to provide repositories, pipelines using GitHub Actions, and pull request review.

GitHub repositories use the Git system; a popular version control system. GitHub Actions allows you to write scripts that are executed when triggered by repository actions. For example, a script can automatically run and analyze the code whenever a new pull request is submitted.

Pull request review is set up so that DevSecOps and the project PM receive a message whenever a new pull request is created. They then are asked to go in, examine the result of the pipeline (for example static code analysis results from SonarCloud), review the changes made, and either approve or deny the request. After this the creator of the pull request is allowed to merge.

2 GitHub

GitHub Branch Overview



2.1 Master Branch: Main Branch of working components that have been previously reviewed and merged to the branch

2.2 Developer Branch: Branch of ideally working components to be reviewed and merged to the main branch

2.3 Feature Branch: Branch for developing features and creating pull request for working components to be reviewed and merged to the developer branch

2.4 GitHub Initial Configuration

Initial GitHub configuration was provided by course mentors, in which case you submit your already established GitHub account and are assigned teams. New GitHub Account can be created at <https://github.com> for users that do not have a GitHub account

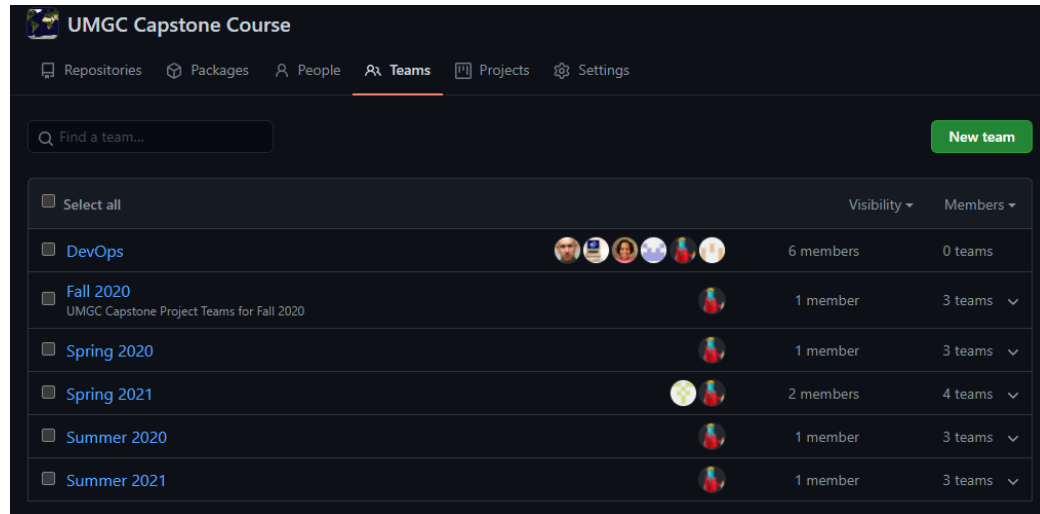
2.5 Creation of Development Teams

2.5.1 Create Team

Using the UMGC service account do the following in order to create desired teams in GitHub. The mentors can provide the DevSecOps team members the UMGC service account credentials. Safe guard and use wisely.

- Go to Profile →
- Your organizations →

- UMGC →
- Teams →
- **Choose team** <team name> (Semester).
 - If the semester is not yet created then click the “New team” green button near the upper right of the page.
 - Enter Team name →
 - Description →



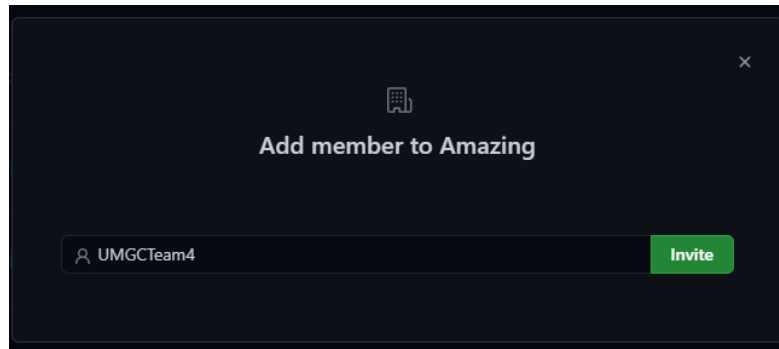
- Choose semester under **Parent team** →
- Create team.

Teams are created for this project.



2.5.2 Add Members to the GitHub team

- Select the desired team to add member →
- Select the “Members” page →
- Add Member →

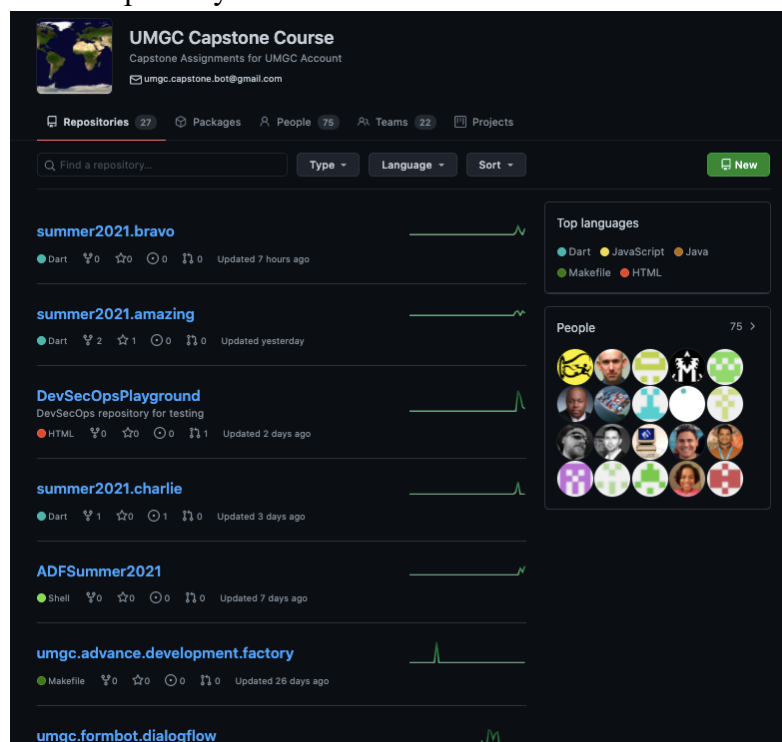


- Invite
- Confirm Invite to this team

2.6 Repository Creation

Login to UMGC SWEN Capstone Course admin GitHub account by means of provided credentials.

- Go to **Profile** →
- Your organizations →
- UMGC →
- New →
- Enter **Repository name** exercising the naming convention **<semester>.<teamname>** →
- Choose **Public** →
- Create repository



2.7 Assign repository to development teams

From the UMGC SWEN Capstone Course admin GitHub account by means of provided credentials.

- Go to **Profile** →
- Your organizations (UMGC) →
- Teams →
- Choose semester →
- Choose team →
- Click Repositories →
- Add repository →
- enter repository name →
- Add repository to team.
- To modify access controls for the repository
 - For DevSecOps team, set the access level to **Admin**
 - For development teams, set the access level to **Write**

3 GitHub Client Installation

Any Git client is capable. However, the specifications in this document will be for the operation of GitHub using the command line or using GitHub Desktop. Commonly used client examples are described below.

Proceed to the following website to view and execute the GitHub command line tools. GitHub CLI is GitHub on the command line. It facilitates Pull Requests, Issues, and Other GitHub concepts in the terminal from your local computer.

<https://github.com/cli/cli#installation>

3.1.1 MacOS

3.1.1.1 Option using homebrew

- brew install gh

3.1.1.2 Proceed to the following website to view and execute the GitHub Desktop Installation Guide for MacOS (You can install GitHub Desktop on macOS 10.10 or later):

- Please follow the "Install on macOS" section.
- <https://central.github.com/deployments/desktop/desktop/latest/darwin>

3.1.2 Windows

3.1.2.1 GitHub Desktop install options

- Installer: [GitHub Installer](#)

3.1.2.2 Option using [WinGet](#)

winget install gh

3.1.3 Linux

Debian based distributions support apt-get and will be used here as an example.

- `sudo apt-get install git`

RPM based distributions can use yum:

- `sudo yum install git`

3.2 Initial Configuration

3.2.1 Verify Git Installation.

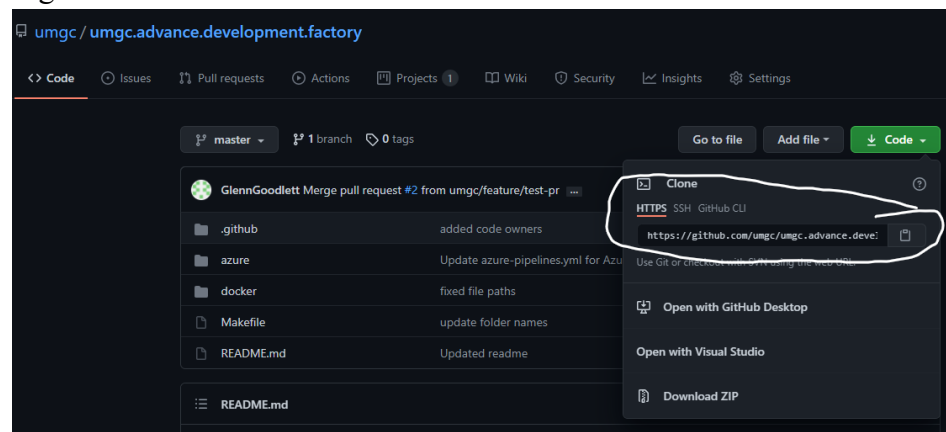
- `git --version`

3.2.2 Set Git Global Configuration.

- `git config --global user.name "<name>"`
- `git config --global user.email "<email>"`

3.2.3 Routine Git Hub Tasks

- Clone the remote repository to local desktop/repository.
 - `git clone <repository URL>`
 - Location to repository URL
 - Click the green “Code” button



3.2.4 Check out the development branch.

- `git checkout <branch name>`
- `git pull origin develop`

3.2.5 Create a new feature branch.

- `git checkout -b <feature name>`

3.2.6 Publish To Remote

To publish the new branch you created in GitHub and make it accessible for everyone in your team, run the following command:

- `git push -u <remote> <branch-name>`

3.2.7 Update code, add it, commit and push changes.

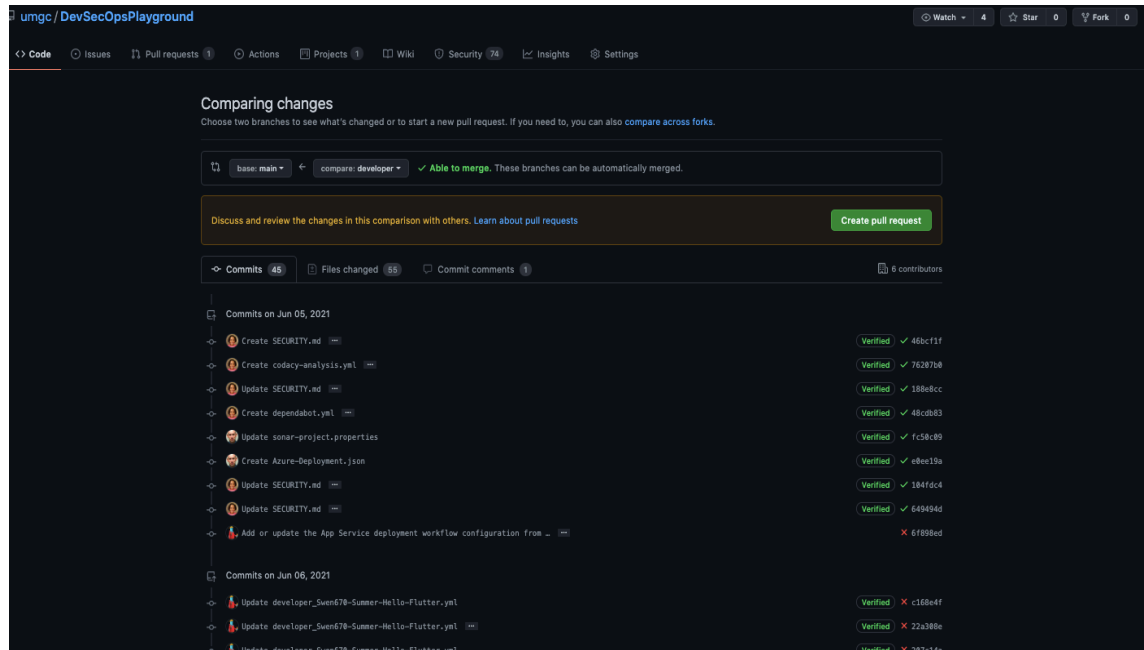
- `git status`
- `git add -A`
- `git commit -m "<description of the change>"`
- `git push -u origin <feature name>`
- The following naming convention for feature branch name is common: `<username>/<feature>`, i.e. `jersoe/delbutton`

4 Submitting Code to be merged

Once you have pushed your local branch to origin, you'll have the option to create a pull request.

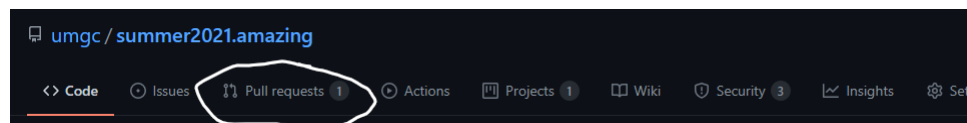
4.1 Create pull request

- Login to GitHub website and navigate to the repository.
- From the tabs listed on the top ribbon Choose **Pull request** →
- Click New pull request →
- Choose base <branch> →
- Choose compare <branch> →
- Click Create pull request →
- Insert title →
- Leave a comment describing pull request
- Click Create pull request →

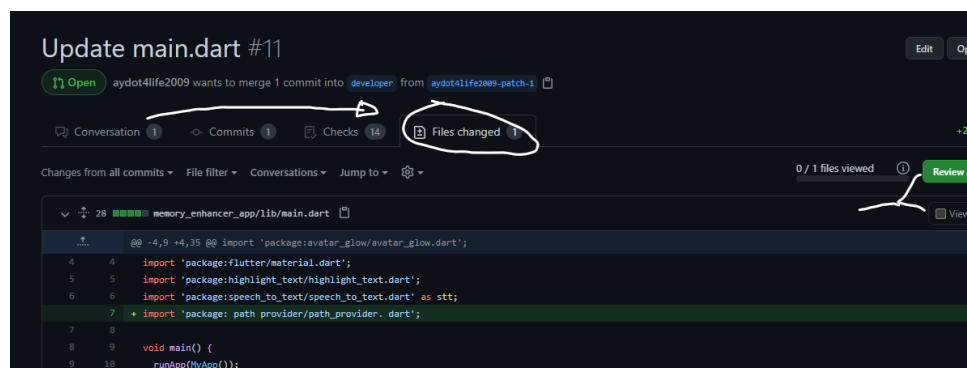


4.2 Merge pull request

- Login to GitHub website and navigate to the repository.
- Click Pull Request →



- Choose the pull request →
- Review the changes →

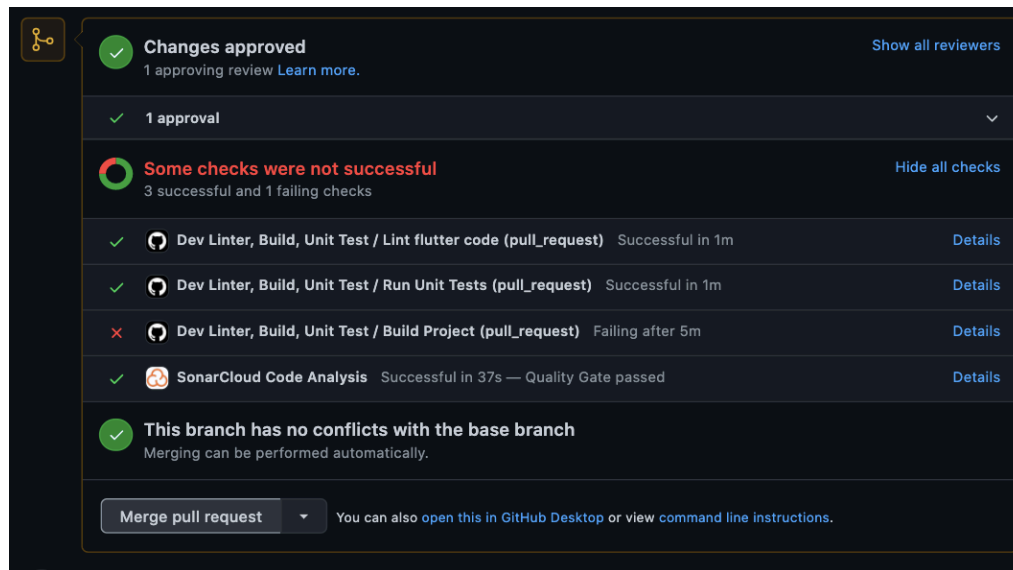


- **NOTE:** General rule of thumb is not to merge a branch into any other branch unless
 - Both branches are yours
 - The owner of the branch receiving the merge has approved your branch to be merged
 - The owner of the branch to be merged has approved their branch to be merged into another

- In other words, if the branch in the PR to be merged is not yours, do not merge it.
- The author may not be done with the branch, although approved.

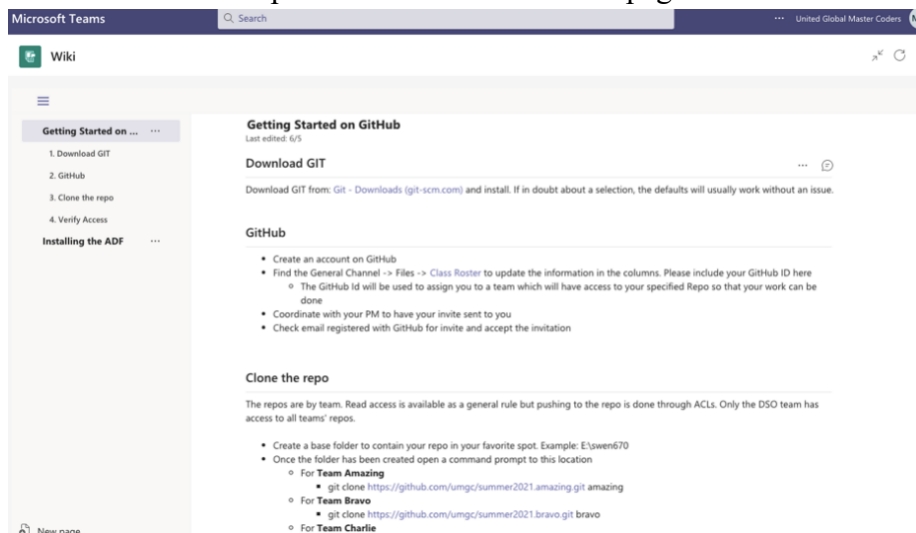
4.3 Merge a Pull Request

- Click the “Merge pull request” button →
- Confirm the merge request →



5 Microsoft Teams

- Create signup sheet for developer access to repositories
- Utilize the DevSecOps channel Wiki to create a page



- Gather development teams to provide their information to be added to their respective team/repository.

Name	Perferred Name	Role	Contact Email	Contact Phone	School User Account	GitHub ID	Timezone
Lead PM							
Malcolm Freeman		Program Manager	mfreeman@student.umgc.edu	301.980.2118	mfreeman@student.umgc.edu		
DevSecOps (DSO)							
Michelle Morfitt	Michelle		mmorfitt@student.umgc.edu	301.317.2934	mmorfitt@student.umgc.edu	mmorfitt	EST
Robert Wilson	Rob		rwilson127@student.umgc.edu	310.680.6823	rwilson127@student.umgc.edu	rwilson127	PDT
Artem Sourat		Developer				ar5004	CST
Team A							
Christian Ahmed		Developer	c.ahmed@student.umgc.edu	202.351.8889	c.ahmed@student.umgc.edu	c-ahmed	
Mitchell Oshansky		Test Engineer	mitch.oshansky@protonmail.com	301.310.1893	mitch.oshansky@student.umgc.edu	mitchUMGC	EST
Shawn Kelly		Team PM	skelly@student.umgc.edu	301.541.6100	skelly@student.umgc.edu	shawnkelly	EST
Monodou Drammeh	Mo	Developer	mdrammeh33@student.umgc.edu	(301) 305-2484	mdrammeh33@student.umgc.edu	mdrammeh1	EST
Nicholas Bello		Test Engineer	nbell01@student.umgc.edu			nbell01	EST
Ayodeji Famulohin		Business Analyst	afamulohin@yahoo.com			ayodeji1999	
Team B							
Orlando Martin		Test Engineer					
Tyler Puchinsky		Developer	tpuchinsky@student.umgc.edu	(443) 720-8702	tpuchinsky@student.umgc.edu	tylerpuchinsky	EST
Alex Bailey		Test Engineer	abail@student.umgc.edu	240-481-7205	abail@student.umgc.edu	alexbailey	EST
Benjamin Cushing	Ben	Business Analyst	bcushing@proton.net			benjamincushing	
Raul Hernandez		Team PM					
Team C							
Madison Dunning		Developer	mdunning14@gmail.com	(361) 232-6951	mdunning14@student.umgc.edu	MDunning14	CST until June 5, then PST until June 24, then CST
Michael Le		Team PM	ml@lewis108@yahoo.com				EST
Prince & Antwa	Prince	Business Analyst	prince@student.umgc.edu	2815087084	prince@student.umgc.edu	prince	EST
Adoagye			adadagye@student.umgc.edu				
Odinus Kimbi		Developer	okimbi@student.umgc.edu	301.795-8059	okimbi@student.umgc.edu	okinbi	EST
Deborah Jena		Developer	djenastudent@gmail.com				EST
Austin Johnson		Developer	ajohnson@student.umgc.edu	858-204-9628	ajohnson@student.umgc.edu	ajohnson04	EST
Domin Savits		Test Engineer					EST

- Create DevSecOps guidelines for development teams
- Utilize the DevSecOps channel Wiki to produce a page with strategies for development teams

DevSecOps Team Channel Posts Files Wiki Meet

Verify Access

It is important that you can push the changes back to GitHub. If you are unfamiliar with pull / push with Git please contact one of the DSO team members and coordinate for help.

- Check out a new branch. Example: `git checkout -b dev/willyw548/verify_access`
- After the repo has been downloaded by edit the Readme file in the at the root of the repo (Example- E:\swen670\amazing\readme.md) by adding your name to appropriate location
- As you are editing, don't forget to **commit!**
- Once done editing, it is time to push!
 - One the initial push back to GitHub, you will likely be prompted to authenticate. This is with the same information that you use to login to GitHub and the DSO team will not be able to assist you.
 - Example: `git push origin dev/willyw548/verify_access`
- If your push is successful your changes will be made available to create a PR!

Go ahead and create the PR!

PR's need approval!

Add more commits by pushing to the `dev/willyw548/verify_access` branch on `umgc/summer2021/amazing`.

Review required
At least 1 approving review is required by reviewers with write access. [Learn more](#)

Merging is blocked
Merging can be performed automatically with 1 approving review.

- Advise development teams of reference available in team wikis.

6 Build Pipeline – GitHub Actions

The section below describes the setup and configuration of the tools necessary for the DEVSECOPS team to manage and create the components of the CI/CD pipeline.

6.1 ADF Pipeline with GitHub Actions

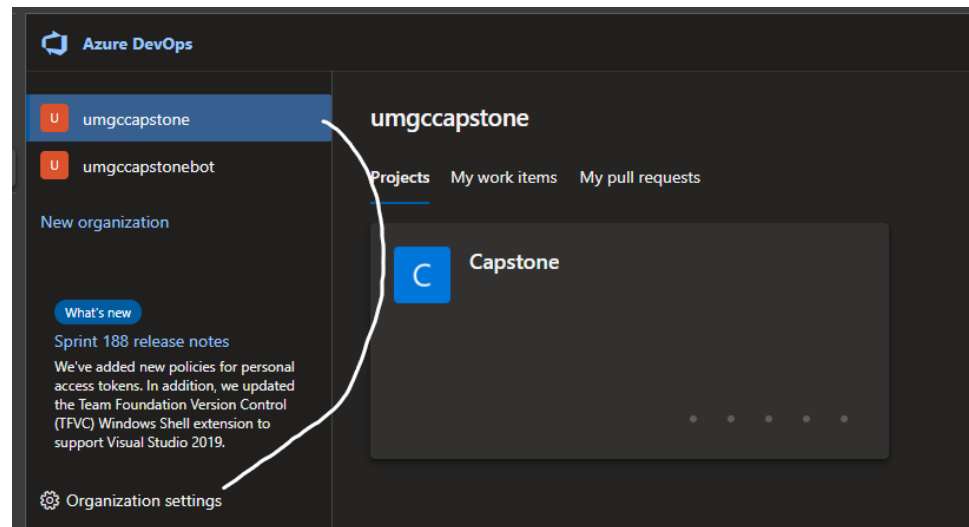
Previous semesters have incorporated teams using Microsoft Azure DevOps. However, teams now operate using GitHub because of the public accountability and accessibility of the repos. Pipelines integrate well with Azure. The association with Azure DevOps and the UMGC

federation actually limits the visibility of what is happening from an instructor/mentor level as well as create a hard dependency on the project and the student who setup the operation.

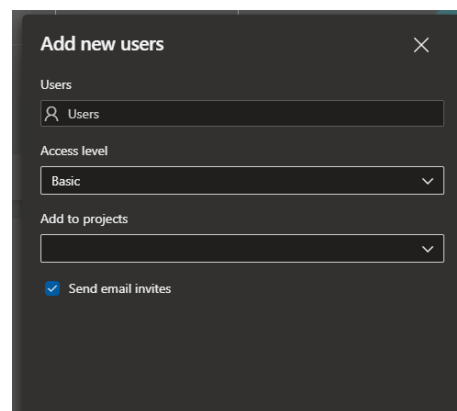
If Microsoft Azure DevOps is a requirement the following steps can be implemented. Otherwise section 6 may be skipped entirely.

6.1.1 Add Users

- Visit site: <https://azure.microsoft.com/en-us/services/devops/> and log in to admin UMGC SWEN Capstone account
- Under Azure DevOps Organizations, click **dev.azure.com/umgccapstone**
- Click Organization settings



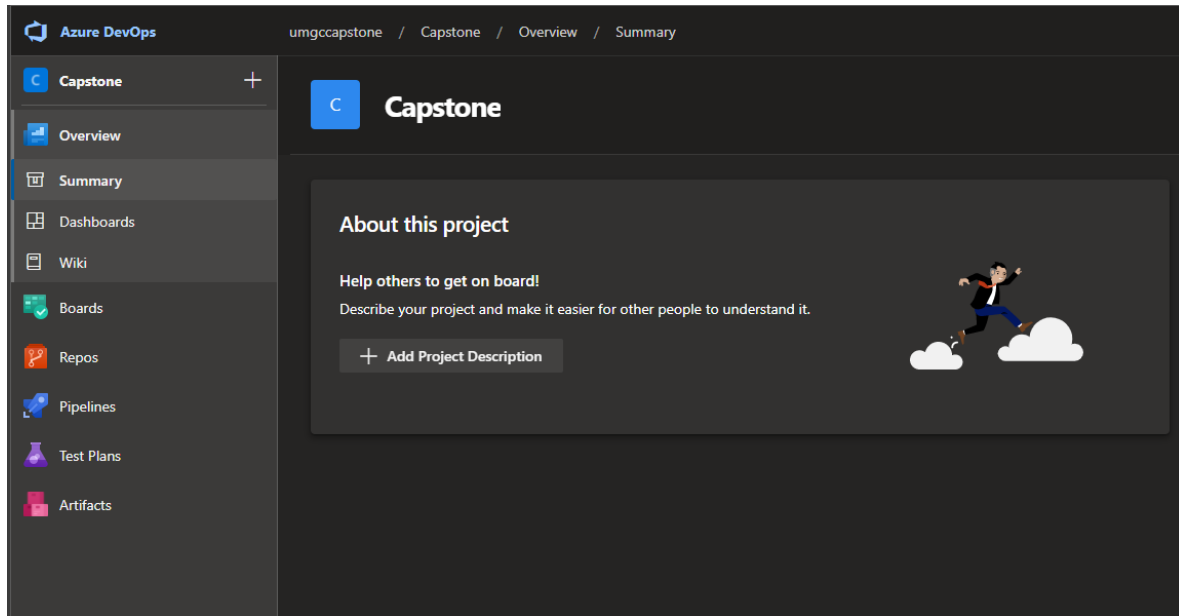
- Click Users →
- Enter email address of DEVSECOPS team members →



- choose **Basic** Access level →
- under Add to projects
- Add users →
- choose Capstone →
- Add

6.2 Access the Capstone Azure DevOps Project

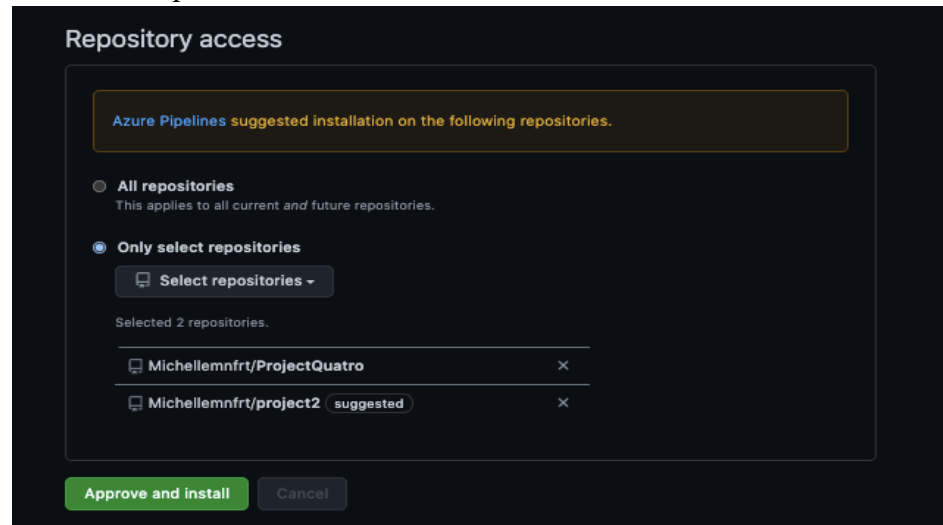
- Go to <https://azure.microsoft.com/en-us/services/devops/> and log in with your DEVSECOPS team member account
- Under Azure DevOps Organizations, expand **dev.azure.com/umgccapstone**
- Under Projects, Click **Capstone**



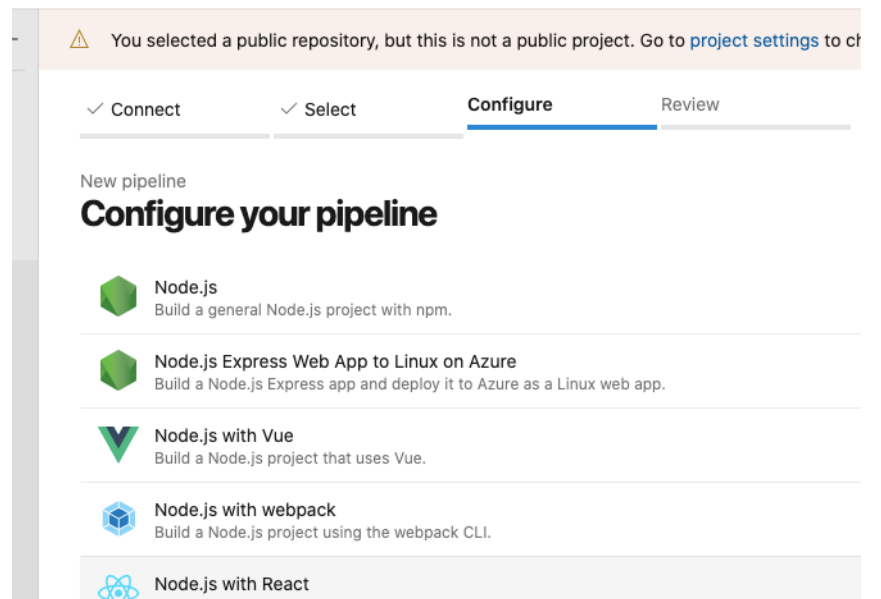
6.2.1 Creation of a continuous integration pipeline

- In the right-side menu, click Pipelines → Pipelines → New pipeline
- Click **GitHub**
- Confirm password
- Choose **All repositories** from the search criteria dropdown
- Search for the repository for which to create a pipeline and Choose it (enter admin account password if prompted)

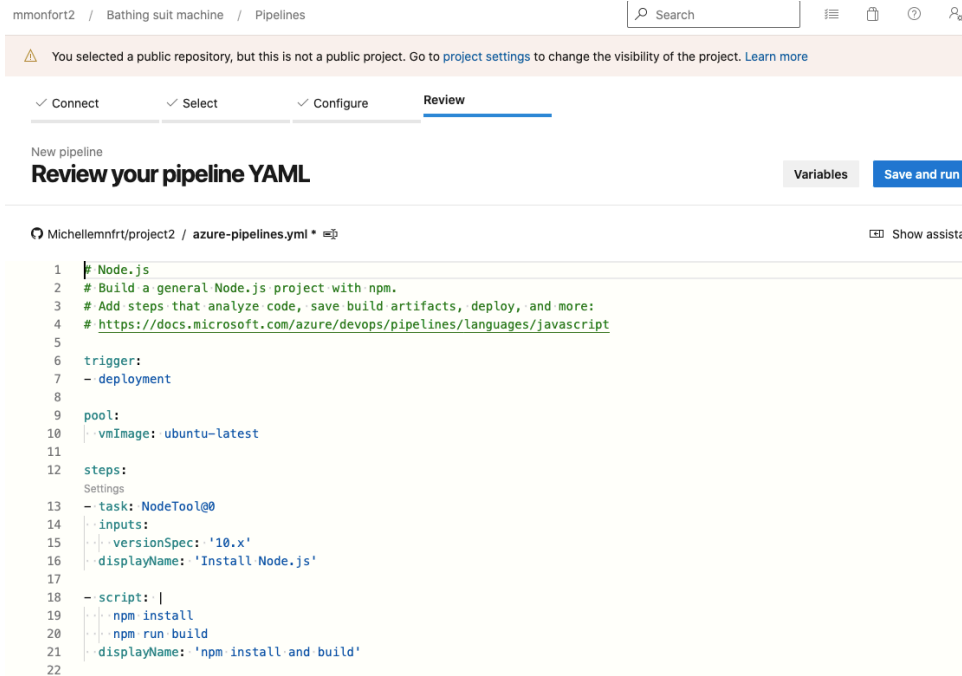
- Approve and install the access to GitHub repository for Azure DevOps



- Figure 1. GitHub – Approve Azure DevOps Access
- Confirm Microsoft account
- Create the pipeline based on the programming language or services being used



- Figure 2. Azure DevOps – Create Pipeline
- Edit or add the initial pipeline YAML template



- Save and run

7 Docker

7.1 Get and run Docker images

7.1.1 Ensure that docker images are installed

- docker images

7.1.2 Pull docker image to local machine

- docker pull <image>

7.1.3 Download and execute docker image

- docker run <image>

7.1.4 Get a particular version of docker

- docker pull <image>:<version>

7.1.5 Bind Port to Host

When running more than one version of the same image, the same port will be used, you will need to bind your host port to the container's port using the following command:

- docker run -p<hostport>:<imageport> <image>

7.1.6 To run docker image in detached mode

- `docker run -d`

7.1.7 Delete an image

- `docker rmi <image id>`

7.2 Actions with Docker containers

7.2.1 Check running docker containers

- `docker ps`

7.2.2 Check both running and stopped images

- `docker ps -a`

7.2.3 Stop docker image

- `docker stop <container id>`

7.2.4 Start docker image again

- `docker start <container id>`

7.2.5 Delete container

- `docker rm <container id>`

7.3 Docker network

7.3.1 List networks

- `docker network ls`

7.3.2 Create docker network for image communication

- `docker network create <network name>`

7.3.3 Tell container to run on a specific docker network

- `docker run <any other options> --net <network name>`

7.4 Docker Compose file (running multiple containers with different configurations)

Reference the following for examples: <https://docs.docker.com/compose/>

7.4.1 Start images using Compose configurations

- `docker-compose -f <yaml file name> up`

7.4.2 Shut down images using Compose

- `docker-compose -f <yaml file name> down`

7.5 Create a Dockerfile (creating a Docker image)

Reference the following for examples: <https://docs.docker.com/engine/reference/builder/>

7.5.1 Create docker image

- `docker build -t <name of your app>:<tag name, such as 1.0 or version-1, etc> <location of Dockerfile, such as root directory> .>`

Note: Rebuild image after every modification to the dockerfile

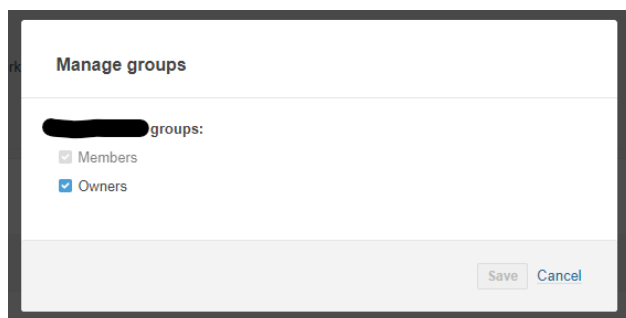
8 Build Development Team Pipelines

GitHub actions contain the capability to build the projects, upload them as artifacts, set up a release schedule, run unit tests, and submit to SonarCloud without leaving the GitHub ecosystem.

8.1 SonarCloud

8.1.1 Create a SonarCloud account

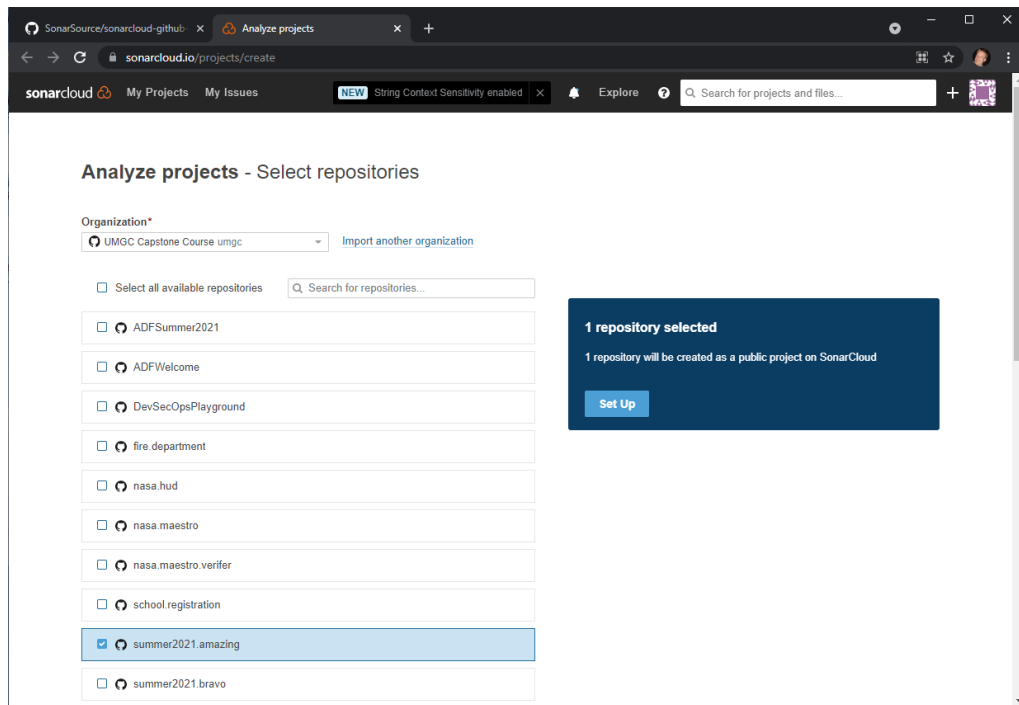
- Sign up to SonarCloud using your GitHub account at <https://sonarcloud.io>
- Assign yourself as owner along with the other DevSecOps members
- Open a private browser session
- Login into GitHub using UMGC service account
- Open a new private tab and navigate to <https://sonarcloud.io>
- Login using GitHub
- Choose yourself from the list and make owner



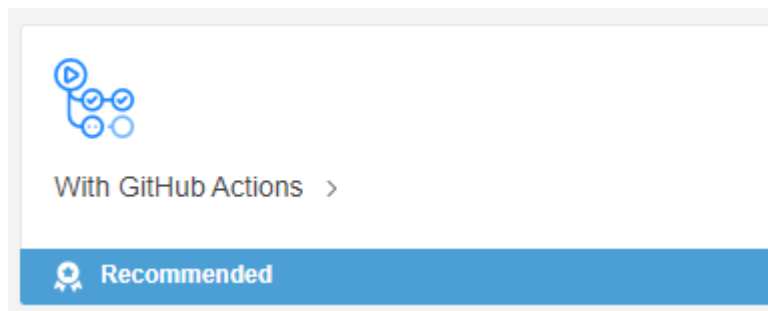
- Log out of service account

8.1.2 Link SonarCloud to a GitHub repository

- Back to your account in SonarCloud. Navigate to <https://sonarcloud.io/projects/create>
- Choose the repository you wish to setup



- Click the “Set Up” button
- Choose with GitHub Actions



- Follow the instructions to create the GitHub Secrets. The below screenshots should help guide you

Analyze with a GitHub Action

1 Create a GitHub Secret

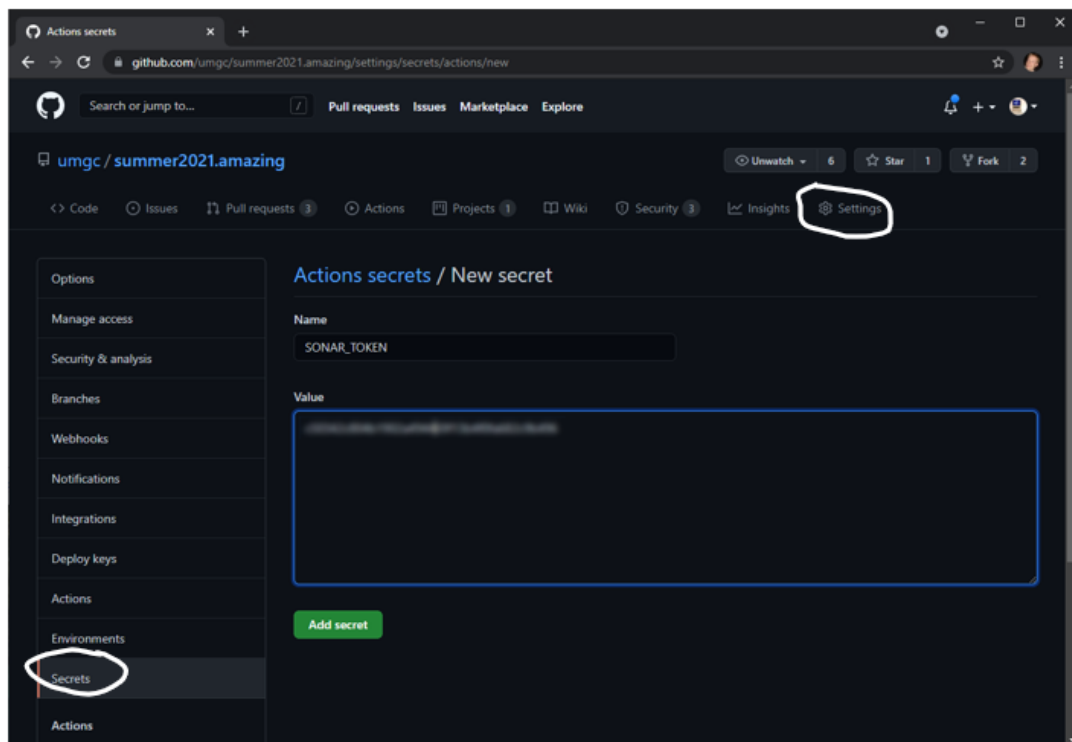
In your GitHub repository, go to [Settings > Secrets](#) and create a new secret with the following details:

1 In the Name field, enter `SONAR_TOKEN`

2 In the Value field, enter

[Continue](#)

- In a different window open GitHub
- The secrets are under the Settings menu item for the repository then secrets



- Leave the Sonar Cloud configuration window open. You will return to this spot later

8.1.3 GitHub Analysis Files

- Add analysis_options file from [UMGC/DevSecOpsPlayground](#) to root of repository of new repo

- Add sonar-project.properties from [UMGC/DevSecOpsPlayground](#) to root of repository of new repo

8.1.4 From the SonarCloud set up continue

- Choose other and continue
- You should be on step 3. This new “properties” file is much like the property file you’ve just added to the new repository. But you’ll need to replace the first 5 lines of text from the website to the file in the repository.

3 Create a `sonar-project.properties` file

Create a configuration file in the root directory of the project `sonar-project.properties`

```

sonar.projectKey=umgc_summer2021.amazing
sonar.organization=umgc

# This is the name and version displayed in the SonarCloud UI.
#sonar.projectName=summer2021.amazing
#sonar.projectVersion=1.0

# Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.
#sonar.sources=.

# Encoding of the source code. Default is default system encoding
#sonar.sourceEncoding=UTF-8

```

Copy

Is it done? You should see the page refresh itself in a few moments with your analysis results if everything runs successfully.

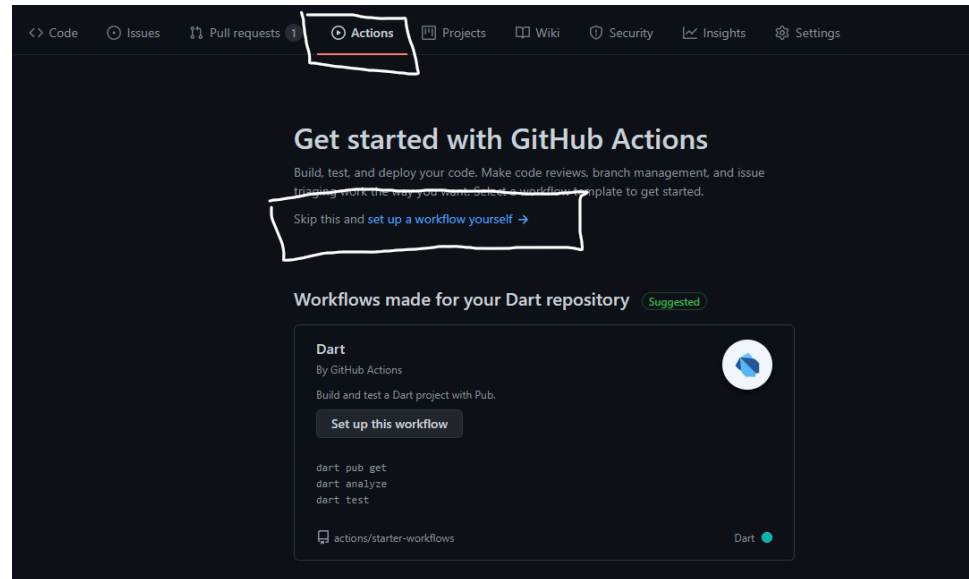
And then what? Each new push triggers an analysis by SonarCloud.

Check these useful links while you wait: [What is a Quality Gate?](#), [Configure your Quality Profiles](#)

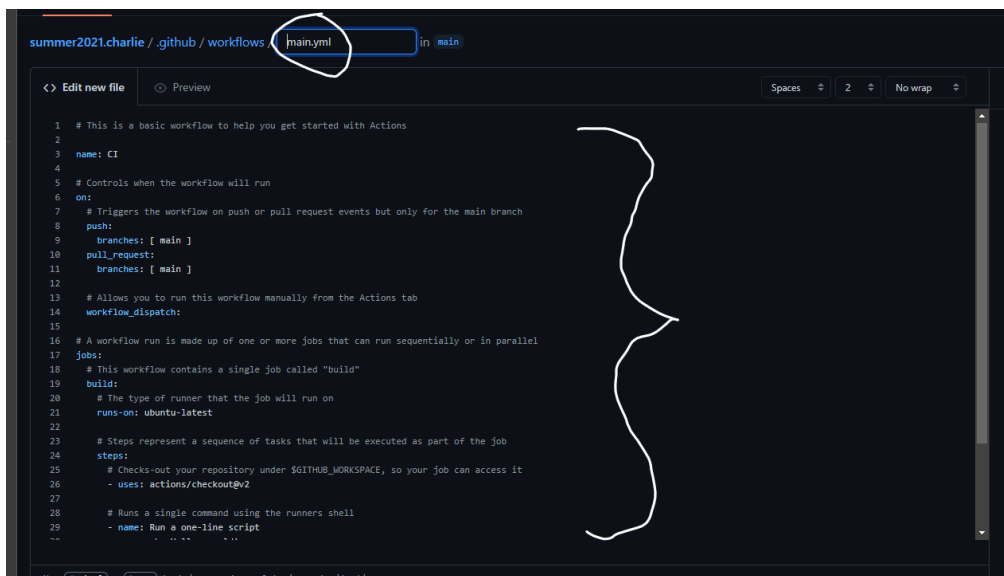
8.2 GitHub Actions

GitHub Actions is the GitHub equivalent of Azure Pipelines. Here you can test, analyze, build, and deploy all within the same place your code is.

- From GitHub and the repository to build the pipeline for
- Click “Actions”
- Then set up a workflow yourself



-
- Rename from main.yml to stable.yml. Then highlight the boilerplate code and delete it.

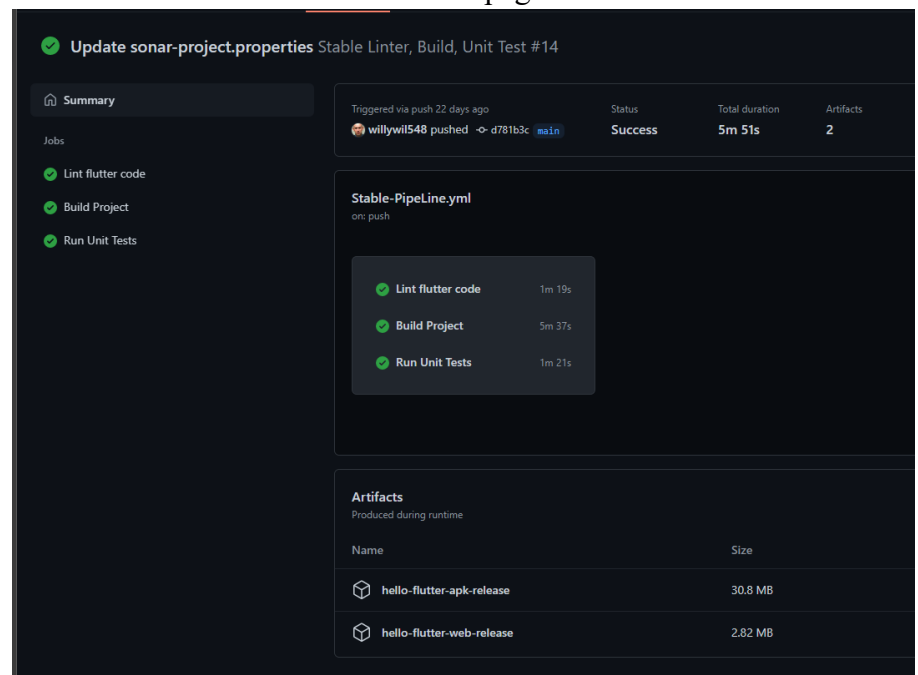


- Retrieve prebuilt pipeline from the [UMGC/DevSecOpsPlayground](https://github.com/UMGC/DevSecOpsPlayground), copy the contents and paste into new workflow you are setting up.
- Note: This pipeline assumes the pubspec.yaml file is at the root of the repository. If not then it may have to be modified to be inclusive to the target directory. The analysis file and sonar-project.properties files assume the Flutter project is based in the root of the repository (the pubspec.yaml).
- This finishes the set up of the build pipeline.

- The build pipeline established a Linter, Builder, Artifacts, and Unit Testing. The build artifacts can be obtained after a successful build by taking the following actions:
- Click Actions
- Click Workflow. Example is Stable Workflow that has completed.



- Artifacts will be at the bottom of the page

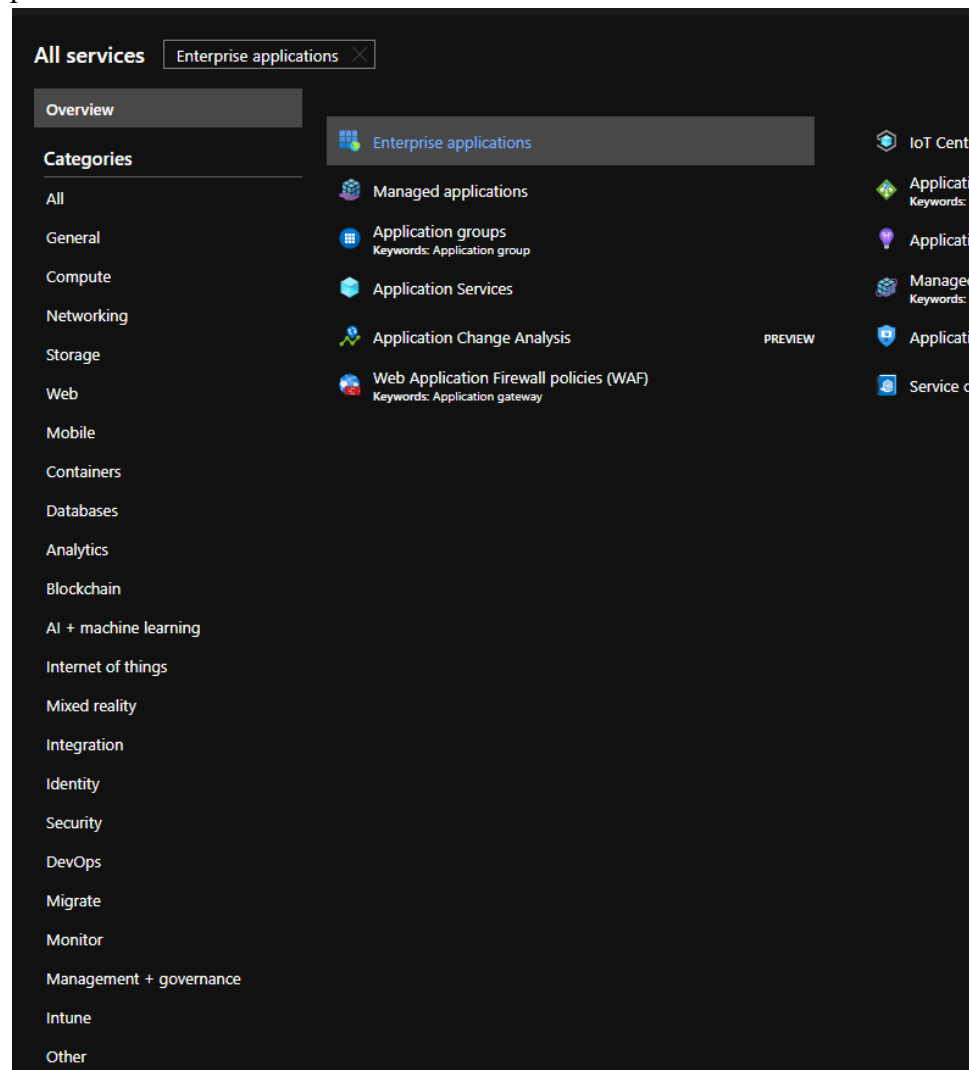


8.3 Configure Enterprise Application

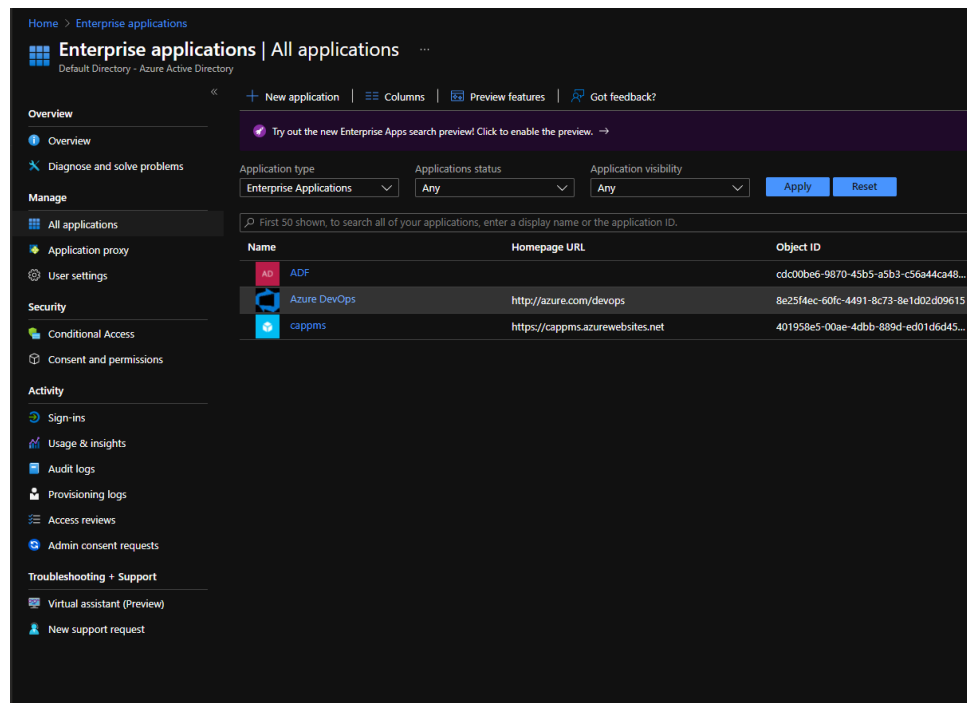
Create or reuse an enterprise application. Enterprise applications are a good place for service accounts and general authentication requirements to be met. An application can be scoped for a specific purpose and contain a limited number of usable accounts and secrets.

- Select Enterprise applications from the Azure service section. This is not to be confused with the Azure Market

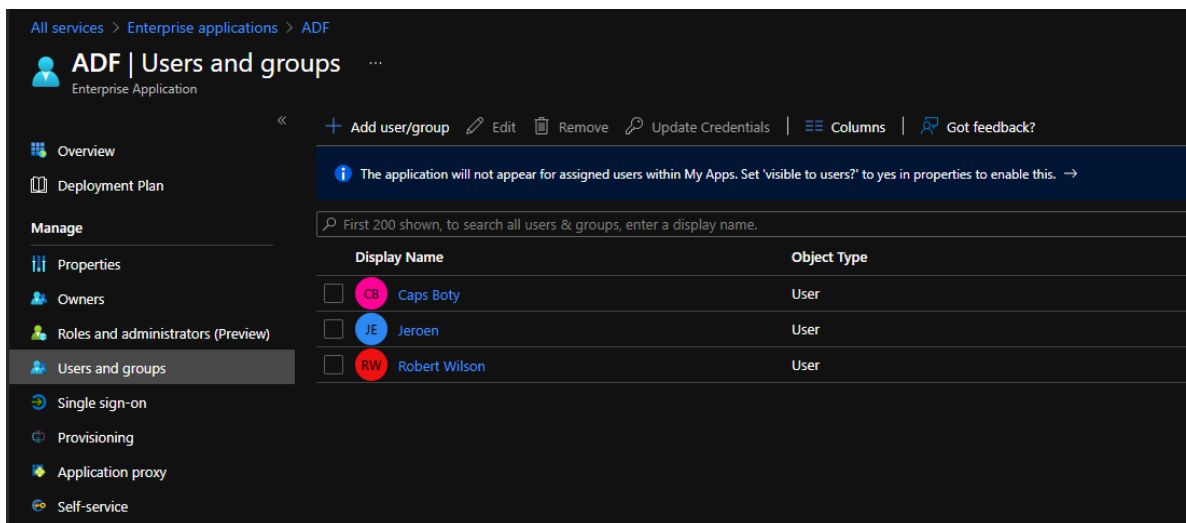
place.



- Select the desired application. In this example we will select ADF. Feel free to create a new application for the desired purpose.

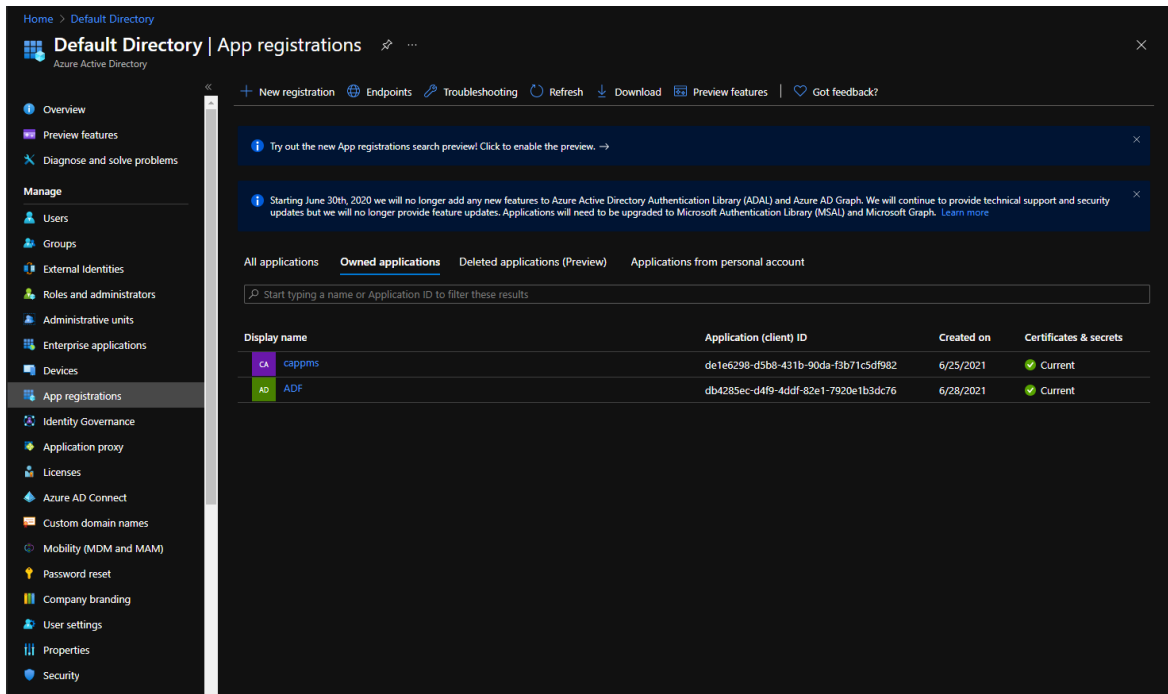


- Add some users to help the chat bot to manage the instance.

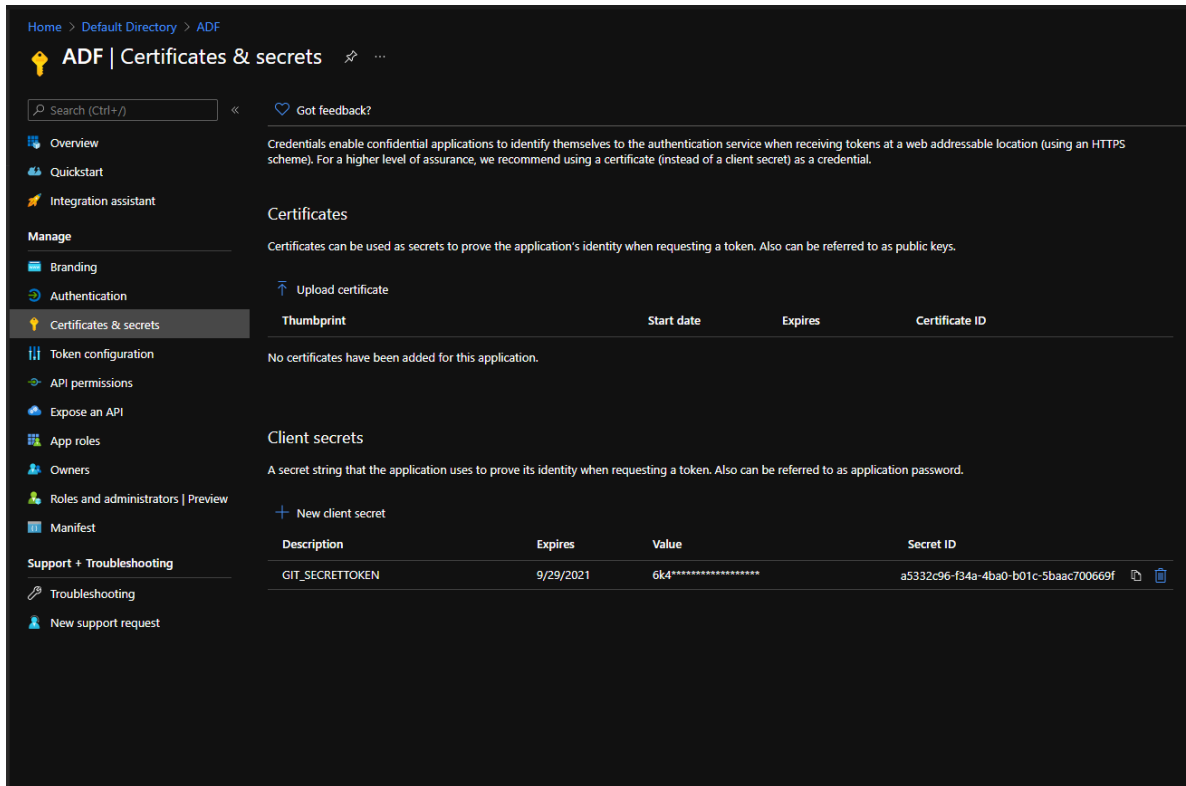


8.4 Setup secrets for AKS

Go to the Azure Directory Service and find App registration. From here select the desired application.



- Create the secrets for the AKS cluster you are about to create. Make note of the secrets value as you won't be able to get it again. You will also need the Application (client) id from the overview or the previous screen. Put these items aside as you will need them in the next step.

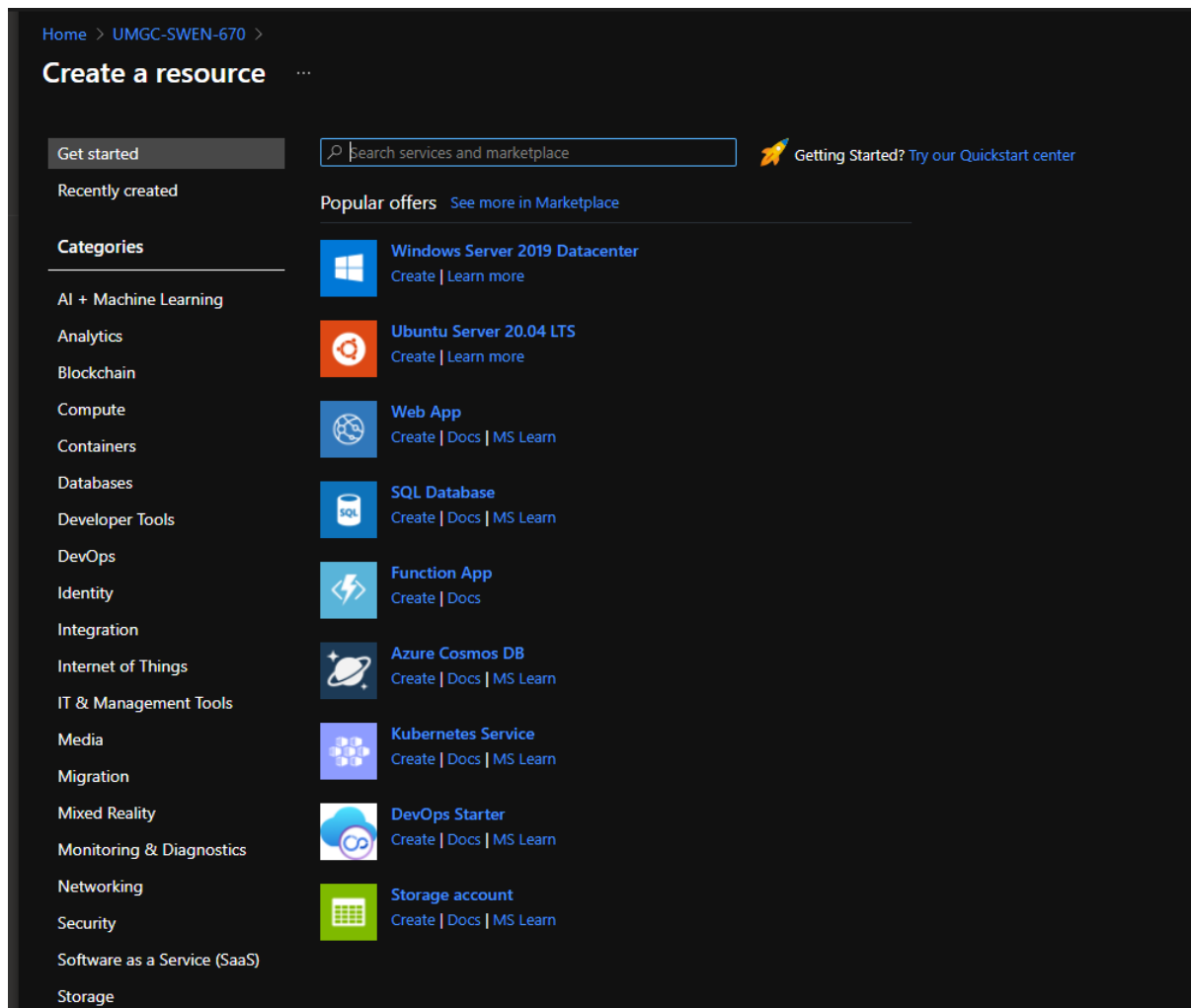


8.5 Docker and Azure Kubernetes Service (AKS)

This section involves the setup and configuration of the tools required of the DevSecOps team to create and manage the deployment components of the application.

8.5.1 Create a Kubernetes cluster

- Login to Azure Portal (<https://portal.azure.com>)
- Choose Create a resource
- From the icons on the header choose **Kubernetes Service**
- From the icons on the header choose **Create**
- Choose Create Kubernetes cluster



- Choose **Resource Group** or **create new** (A resource group is a collection of resources that share the same lifecycle, permissions, and policies)
- Insert **Kubernetes cluster name** (The name can contain only letters, numbers, underscores and hyphens. The name must start and end with letter or number. Kubernetes service name must be unique in the current resource group)
- Choose Region
- Choose Availability Zones
- Choose Kubernetes Versions

[Home](#) > [Resource groups](#) > [UMGC-SWEN-670](#) > [Create a resource](#) >

Create Kubernetes cluster ...

[Basics](#) [Node pools](#) [Authentication](#) [Networking](#) [Integrations](#) [Tags](#) [Review + create](#)

Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment, making it quick and easy to deploy and manage containerized applications without container orchestration expertise. It also eliminates the burden of ongoing operations and maintenance by provisioning, upgrading, and scaling resources on demand, without taking your applications offline. [Learn more about Azure Kubernetes Service](#)

Project details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ

Resource group * ⓘ [Create new](#)

Cluster details

Kubernetes cluster name * ⓘ ✓

Region * ⓘ

Availability zones ⓘ

Kubernetes version * ⓘ

Primary node pool

The number and size of nodes in the primary node pool in your cluster. For production workloads, at least 3 nodes are recommended for resiliency. For development or test workloads, only one node is required. If you would like to add additional node pools or to see additional configuration options for this node pool, go to the 'Node pools' tab above. You will be able to add additional node pools after creating your cluster. [Learn more about node pools in Azure Kubernetes Service](#)

Node size * ⓘ **Standard B2ms**
2 vcpus, 8 GiB memory

[Review + create](#) [< Previous](#) [Next : Node pools >](#)

- Go to the authentication page
- Input the service principal client id generated in a previous step
- Input the service principal secret generated in a previous step

Home > Resource groups > UMGC-SWEN-670 > Create a resource >

Create Kubernetes cluster ...

Basics Node pools **Authentication** Networking Integrations Tags Review + create

Cluster infrastructure
The cluster infrastructure authentication specified is used by Azure Kubernetes Service to manage cloud resources attached to the cluster. This can be either a [service principal](#) or a [system-assigned managed identity](#).

Authentication method ☒ Service principal ☐ System-assigned managed identity

Service principal client ID * ⓘ db4285ec-d4f9-4ddf-82e1-7920e1b3dc76 ✓

Service principal client secret * ⓘ ✓

Kubernetes authentication and authorization
Authentication and authorization are used by the Kubernetes cluster to control user access to the cluster as well as what the user may do once authenticated. [Learn more about Kubernetes authentication](#)

Role-based access control (RBAC) ⓘ ☒ Enabled ☐ Disabled

AKS-managed Azure Active Directory ⓘ ☐

Node pool OS disk encryption
By default, all disks in AKS are encrypted at rest with Microsoft-managed keys. For additional control over encryption, you can supply your own keys using a disk encryption set backed by an Azure Key Vault. The disk encryption set will be used to encrypt the OS disks for all node pools in the cluster. [Learn more](#)

Encryption type (Default) Encryption at-rest with a platform-managed key ▾

[Review + create](#) [< Previous](#) [Next : Networking >](#)

- Finally, Choose **Review + Create**
- Accept remaining defaults and Choose **Create**

8.5.2 Configure Deployment

- Using the service account, select your new created Kubernetes cluster from the specified resource.

Home > **UMGC-SWEN-670** Resource group

Search (Ctrl+/)

Overview

Activity log

Access control (IAM)

Tags

Events

Settings

Deployments

Security

Policies

Properties

Locks

Cost Management

Cost analysis

Cost alerts (preview)

Budgets

Advisor recommendations

Monitoring

Insights (preview)

Alerts

Metrics

Diagnostic settings

Logs

Advisor recommendations

Essentials

Subscription (change) : Azure for Students

Subscription ID : 6bf49353-f56e-4ce0-85a6-3d55c8ad4240

Location : East US 2

Tags (change) : Click here to add tags

Filter for any field...

Type == all X Location == all X Add filter

Showing 1 to 6 of 6 records. Show hidden types

No grouping List view

Name	Type	Location
adf-aks	Kubernetes service	East US 2
capmms-db	SQL server	East US 2
capmms	App Service	Central US
capmms	Application Insights	Central US
capmms-db (capmms-db/capmms-db)	SQL database	East US 2
ServicePlan4d355a2-9260	App Service plan	Central US

Previous Page 1 of 1 Next

- Select Deployment Center on the left side.
- Select GitHub and then next.

Home > UMG-SWEN-670 > adf-aks

adf-aks | Deployment center (preview)

Kubernetes service

Search (Ctrl+/)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Security

Kubernetes resources

Namespaces

Workloads

Services and ingresses

Storage

Configuration

Settings

Node pools

Cluster configuration

Networking

Deployment center (preview)

Policies

Properties

Locks

Monitoring

Insights

Launch a docker app to Azure Kubernetes cluster in a few quick steps

Configure a DevOps pipeline to deploy your application updates to this Kubernetes cluster

1 Source 2 Repository 3 Application 4 Resources

Select the code location

Azure Repos

Unlimited free private repos

GitHub

Home to the world's largest community of developers

Bitbucket Cloud

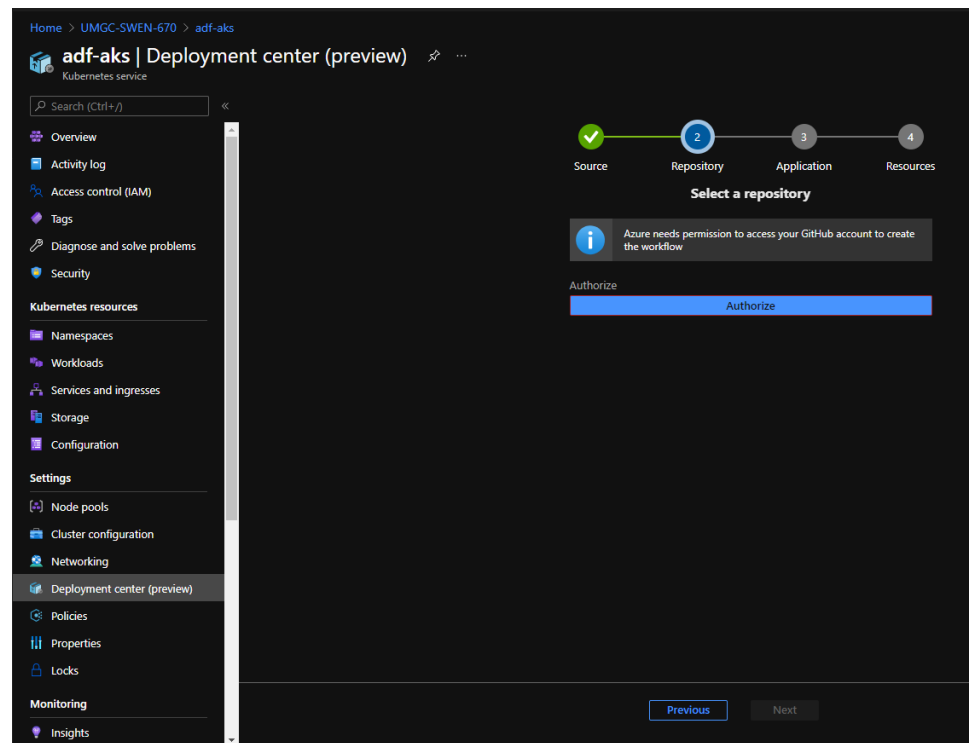
Hosted by Atlassian

External Git

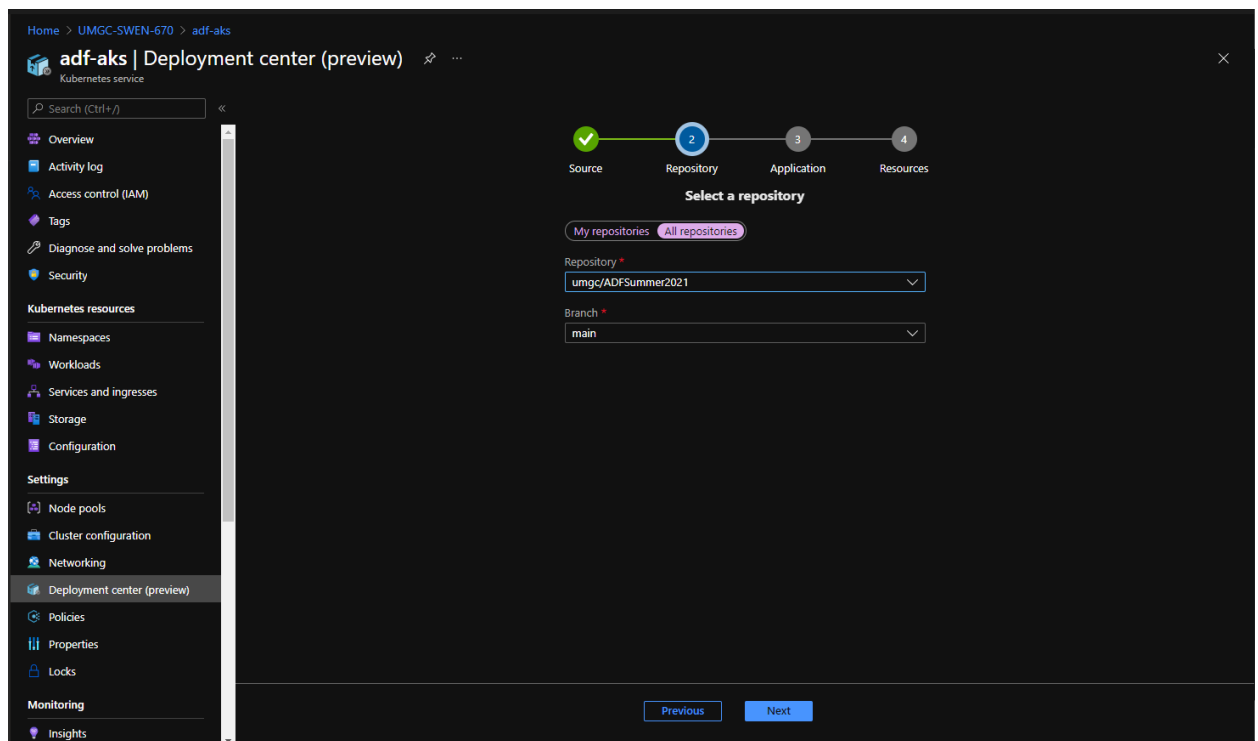
Deploy from a public or private Git repo

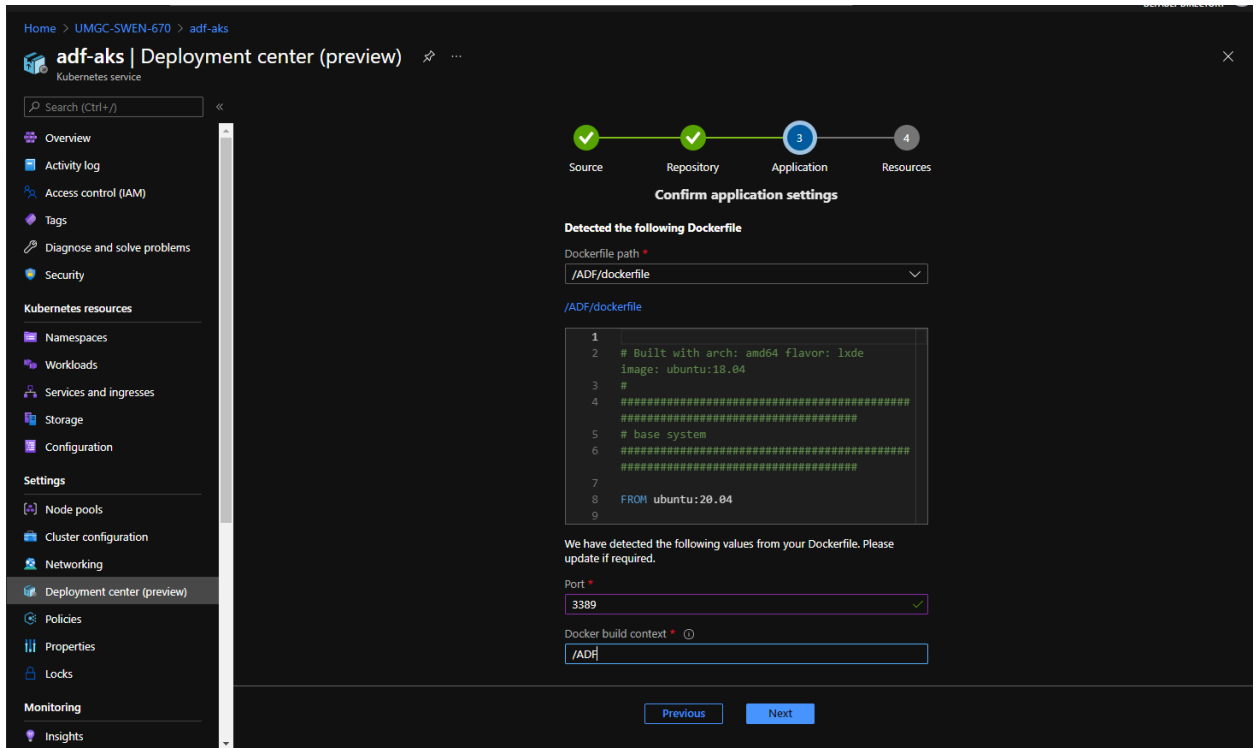
Previous Next

- Authorize Azure to GitHub.

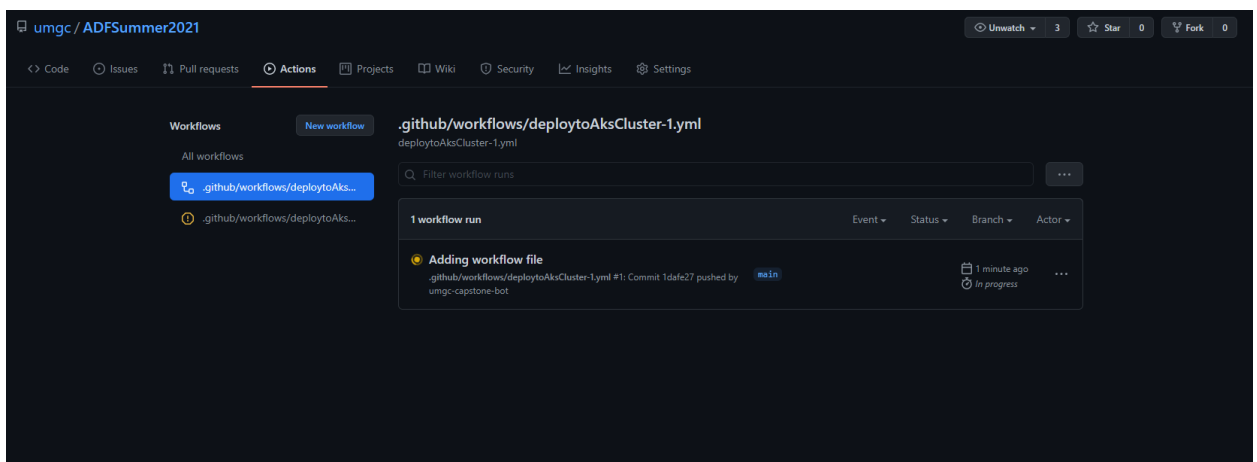


- Select the repository to build the image from.





- Verify the dockerfile and update the port.



- Verify workflow has been added to GitHub Actions.

9 License

The software is unrestricted in use. All components and guides have no guarantees and no warranties the product is provided exclusively "AS-IS". Code may be outdated and contain significant bugs, or not perform as intended, or fail in effectiveness. University of Maryland is not responsible for any shortcomings and the user is solely responsible for the use and implementation.

