Documentation of IL API

**Introduction:**

This is the documentation for the IL API. It includes the build and release pipeline processes. Build pipelines include Nuget-related tasks, artifact generation, etc. Once the build is successful, the release pipeline starts triggering it.

**Build and Release pipeline for IL API Services :-**

**IPERFORMECCMApiIllinois :-**

**Scope:**

* CI/CD is used because code can be easily deployed into a specific environment in a short period of time.
* Through the build pipeline, we can generate the artifacts, and through the release pipeline, we are able to deploy them in a specific location with the required changes.

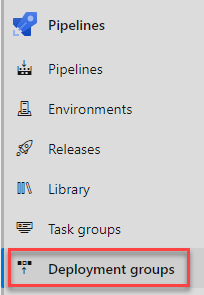
**CI/CD Architecture:**

**A diagram of a software process

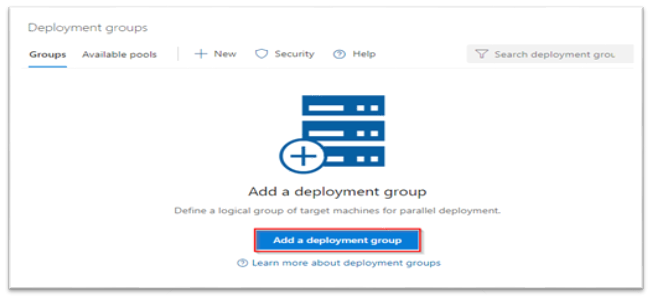
Description automatically generated**

**Deployment Group:**A deployment group is a logical set of deployment target machines that have agents installed on each one.

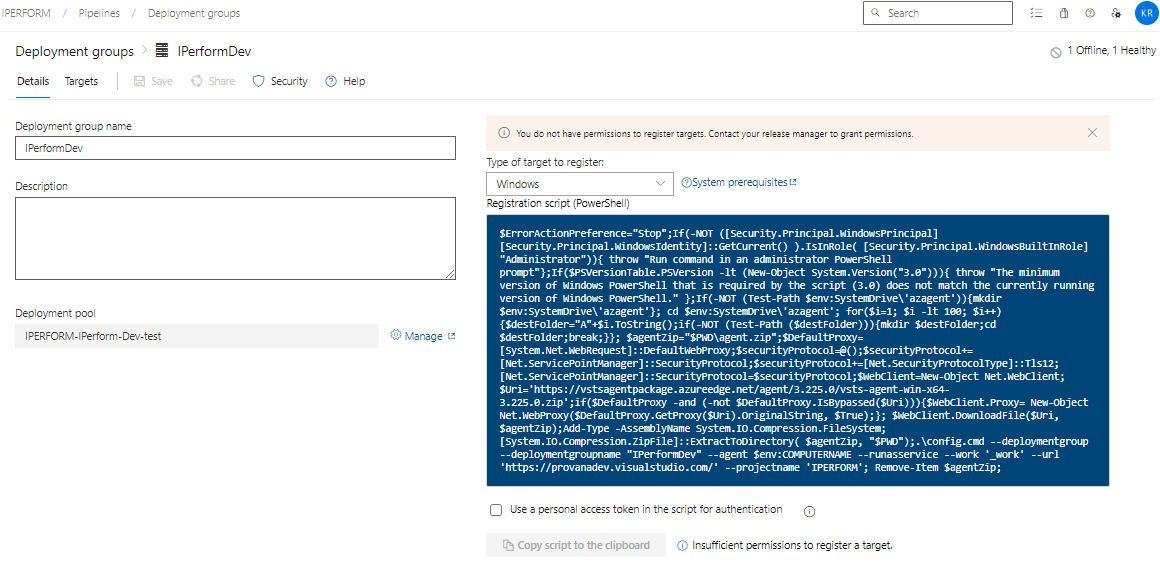
1. Create a deployment group.

****

1. Select Add a deployment group.

****

1. Enter a Deployment group name and then select Create. A registration script will be generated. Select the Type of target to register and then select Use a personal access token in the script for authentication. Finally, select Copy script to the clipboard.



At this stage our deployment group is created but not online yet.

1. Log onto each of your target machines and run the script from an elevated PowerShell command prompt to register it as a target server. When prompted to enter tags for your agent, press *Y* and enter the tag(s) you will use to filter subsets of the servers.

A blue screen with white text

Description automatically generated

At this stage, our deployment group creation gets completed and now if you go and check the status of the deployment group in the Azure DevOps, it should show as Online.

A screenshot of a computer

Description automatically generated

**Build Pipeline for IPERFORMECCMApiIllinois :-**

Steps:

1) Navigate the page to Pipelines and click on the “New Pipeline”.

**Graphical user interface, text, application, email

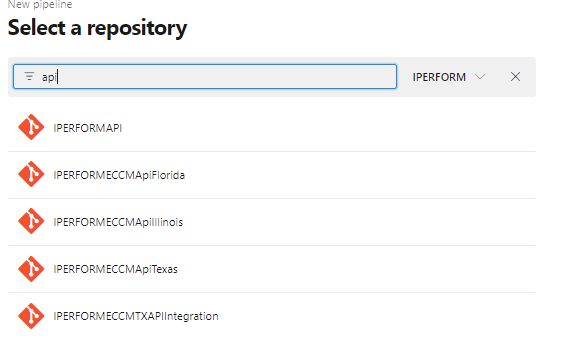
Description automatically generated**

2) Click on the Azure Repos Git as our code is in azure repository.

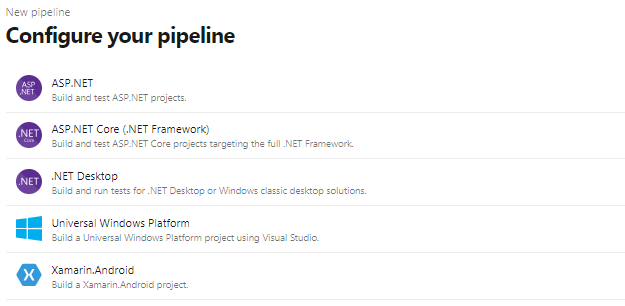
A screenshot of a computer

Description automatically generated

1. Select the repository for which we have to create build pipeline.

****

4)then configure your pipeline as per your requirements.



**Pipelines:**

We are using Azure pipelines (Microsoft hosted agent) as the agent pool for this pipeline. Microsoft-2019 OS version will be used in the agent.

Background pattern

Description automatically generated with low confidence

**Variables:**

you can create variables that can be used in every step of the pipeline. For builds, you could set the username and password of an external service that you want to call. Azure DevOps already comes with a couple of predefined variables like the build configuration or platform.

|  |  |  |
| --- | --- | --- |
| Variable | Description | Used in Task |
| BuildPlatform | CPU configs | Build Solution |
| BuildConfiguration | Configuration | Build Solution |

A screenshot of a computer program

Description automatically generated

**Tasks:**

**NuGetToolInstaller@0** - Acquires a specific version of NuGet from the internet or the tools cache and adds it to the PATH. Use this task to change the version of NuGet used in the NuGet tasks.

A screenshot of a computer

Description automatically generated

**NuGetCommand@2** – Use this task to restore, pack, or push NuGet packages, or run a NuGet command.

A close-up of a computer screen

Description automatically generated

**SonarQubePrepare@4** - This task use prepare a SonarQube analysis Configuration.

A screenshot of a computer code

Description automatically generated

**VSBuild@1** - The Visual Studio Build step builds your application using the solution file

A picture containing text, screenshot, font, line

Description automatically generated

**SonarQubePublish@4** - This Use this task to publish SonarQube's Quality Gate result on the Azure DevOps build result.

A close-up of a computer code

Description automatically generated

**sonar-buildbreaker@8** - This task for has better error handling and you probably want to browse to SonarQube if this task makes your build fail.

A picture containing text, font, screenshot, line

Description automatically generated

**VSTest@2** - Use this task to run unit and functional tests (Selenium, Appium, Coded UI test, etc.

Graphical user interface, text, application

Description automatically generated

**Publish Artifact**

Publish Artifact is necessary for an automated deployment. This step publishes all the files which you want to deploy later in the release pipeline.

A screen shot of a computer code

Description automatically generated

**Git Tag –**

A simple task that tags a commit.

**A computer code with black text

Description automatically generated with medium confidence**

**Validation**

So, here “drop’ folder should include the following files:

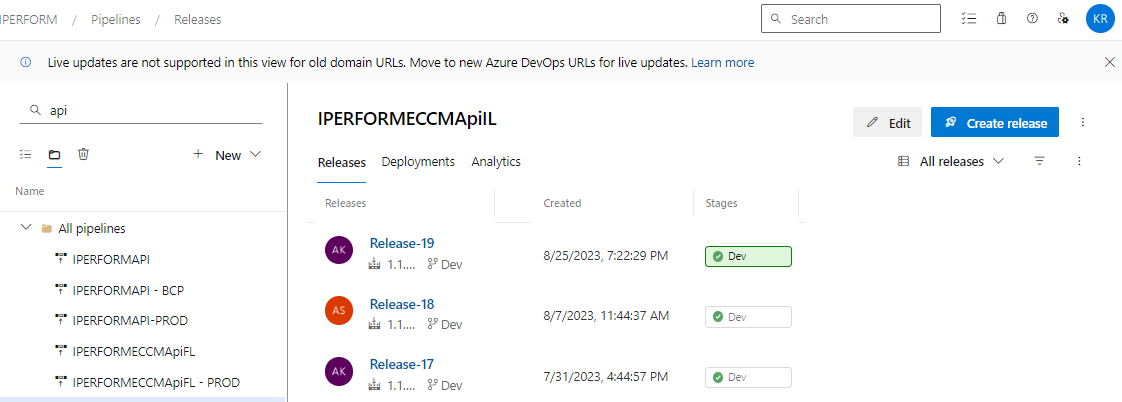
A screenshot of a computer

Description automatically generated

**Release pipeline for IPERFORMECCMApiIllinois :**

**Steps:**

1. Navigate to release pipelines and click on “New“so as to create a new release pipeline which will be IPERFORMECCMApiIL in this case.



1. Select an empty job because we need to add customized jobs later which will support our project’s needs and functionality.

A screenshot of a computer

Description automatically generated

1. Before customizing the jobs, we will need to add the artifact that was built through our build pipeline. So, click on “Add an artifact” and then select the build pipeline which produced the artifact that is going to get used in this release pipeline.

A screenshot of a computer

Description automatically generated

1. As soon as the artifact is added, we must customize jobs according to the functionality of the project.

Graphical user interface, application, PowerPoint

Description automatically generated

**Continuous deployment trigger:**

Continuous deployment triggers allow you to create a release every time a new build artifact is available.

A screenshot of a computer

Description automatically generated

Note -A release will be triggered only for a build that is from one of the branches selected here.  For example, selecting "DEV" will trigger a release for every build from the DEV branch.

**Pull request trigger**

Enabling this will create a release every time a selected artifact is available as part of a pull request workflow.

A screenshot of a computer

Description automatically generated

**Pre-deployment conditions :**

Pre-deployment gates ensures there are no active issues in the work item or problem management system before deploying a build to an environment.

A screenshot of a computer

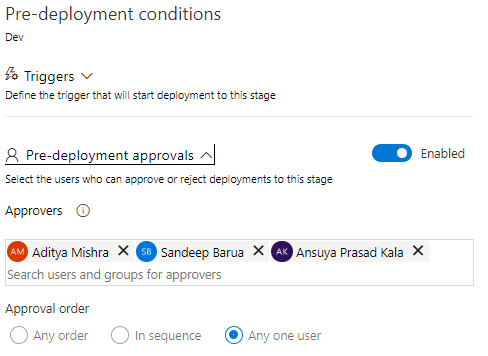
Description automatically generated

Note –

* Triggers - Set the trigger that will start the deployment to this stage automatically. Select "Release" to deploy to the stage every time a new release is created. Use the "Stage" option to deploy after deployments to selected stages are successful. To allow only manual deployments, select "Manual".
* Artifact filters - Select artifact condition(s) to trigger a new deployment. A release will be deployed to this stage only if all artifact conditions match.

**Pre-deployment approvals** –

Select the users who can approve or reject deployments to this stage.



**Tasks:**

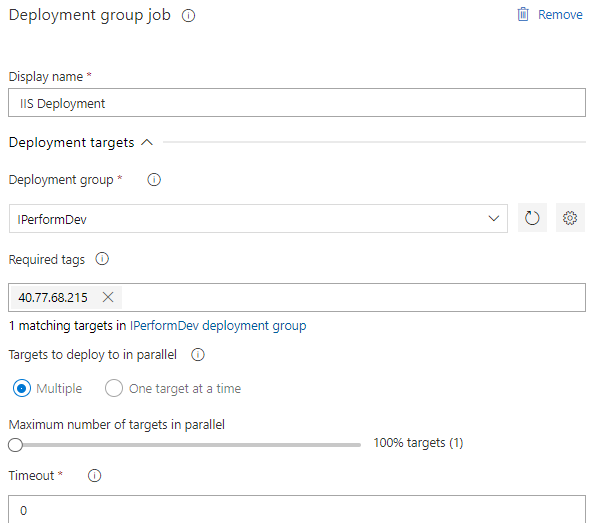
1. We must provide Stage Name, Azure subscription and App type and Name.

A screenshot of a computer

Description automatically generated

**Agent Job:**

A job is a logical grouping of tasks that defines the runtime target on which the tasks will execute. An agent job executes tasks on an agent in an agent pool.



**IIS Web App Manage**  - This task use for Create or update websites, web apps, virtual directories, or application pools.

A screenshot of a computer

Description automatically generated

**IIS Web App Deploy**  - This task use for Deploy a website or web application using Web Deploy.

A screenshot of a computer

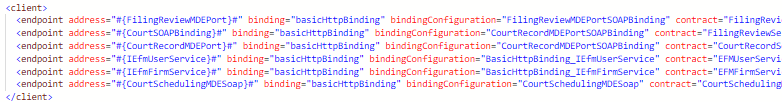
Description automatically generated

**Replace Tokens -** Replace tokens in Web.config.

A screenshot of a computer

Description automatically generated

Here we can see the use of replace tokens which uses ‘#’ in the web.config.



**Command Line Script  -** This task use for Run a command line script using Bash on Linux and macOS and cmd.exe on Windows.

A screenshot of a computer

Description automatically generated

**Variables:**

A screenshot of a computer

Description automatically generated

**Validation :**

Once the release is successful, We check the url with public ip working or not.

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| **30/8/2023** | 1.0 | **Draft and Initial** | **Kamal Rajput** |