Deploying SQL Server Databases with Octopus Deploy

Overview

This 1-day crash-course is designed to support Octopus Deploy evaluators and new users who wish to deploy SQL Server updates. The objective is to coach students to both understand the different tools and techniques and to rapidly acquire experience configuring their own Octopus instance to deploy SQL Server updates, in the minimum time, for the minimum cost.

Students achieve this by setting up their own Proof of Concept (PoC), from scratch, in an environment that they personally own and have complete control over. This allows students to hold on to their PoC indefinitely and to refer to it as they roll out Octopus Deploy within their organization.

This class aims to be accessible for anyone who is considering using Octopus Deploy to manage their database deployments, regardless of their technical stack or hosting strategy. However, to create any PoC we need to build a sample application, from some real source code, and we need to deploy it somewhere. Therefore, as well as SQL Server, students will be working with:

- GitHub
- Amazon Web Services (AWS)
- Windows Servers

Students should not be put off if these tools/technologies don't match their tech stack. Most students won't have used all of the technologies above and some won't have used any. This class assumes zero prior knowledge/experience with any of them. If students have little to no prior knowledge about these technologies, they can treat this class as a mini crash-course on some new tech.

In fact, this class **assumes** that a student's systems will not match this PoC exactly. The class is designed to teach core Octopus Deploy and database deployment tools/techniques in such a way that students will understand how to apply them to their own databases, whether the source code/servers are hosted in TFS, SVN, on Linux servers, on their own servers or in Azure/GCP etc.

Please note: Students are expected to create accounts with GitHub and Amazon Web Services (AWS), as well as an Octopus Cloud instance, **before** starting the class. (Ideally by at least 24 hours.) Please see the Prerequisites section for more details. Students are also encouraged to complete Lab 1 before starting the class.

Contents

LAB 0: Prerequisites *(Try to complete 24 hours before class!)*

You DO NOT need to install anything on your Windows, Linux or Mac computer.

This entire class can be completed through a web browser. Hence, there is no need to install anything on your local machine. You can build your Proof of Concept (PoC) using any standard Windows, Linux or Mac desktop or laptop computer that has a reliable internet connection.

However, there are a few optional applications that you **may find useful:**

- If you would like to clone the GitHub repo and explore the scripts on your local machine in your preferred IDE, that's completely fine. However, since this is entirely optional, installing Git, as well as any preferred Git clients and/or IDEs, is left as an optional task for the user. (At the users own risk.) Using Git on a local machine is only recommended for people who are already familiar with Git. If you are not familiar with Git and would like to explore the code locally, rather than installing Git, you are advised to download the code in .zip format and to explore it using whatever tools you are already familiar with.
- You may wish to remote desktop/RDP into the Amazon Web Services (AWS) instances that you'll be building as part of your PoC. You may find this useful either to explore your PoC or, if you have any problems, you might want to log on to check the logs or troubleshoot/experiment with the various startup or deployment scripts. If using a Windows machine, all the software you need is already packaged with the operating system. If using a Linux or Mac machine, you may wish to install your preferred remote desktop tool (at your own risk).
- You may wish to install your preferred SQL Server tooling on your local machine for convenience. This is not essential, since as part of our PoC, we'll be deploying a "jumpbox" server which will have SQL Server Management Studio (SSMS) pre-installed. (Note: if you intend to use the Jumpbox, it will be necessary to RDP to it. See the point above.) If you decide to install any SQL Server tooling on your local machine, you do so at your own risk.

You DO need to set up accounts for GitHub, Octopus Deploy and Amazon Web Services (AWS).

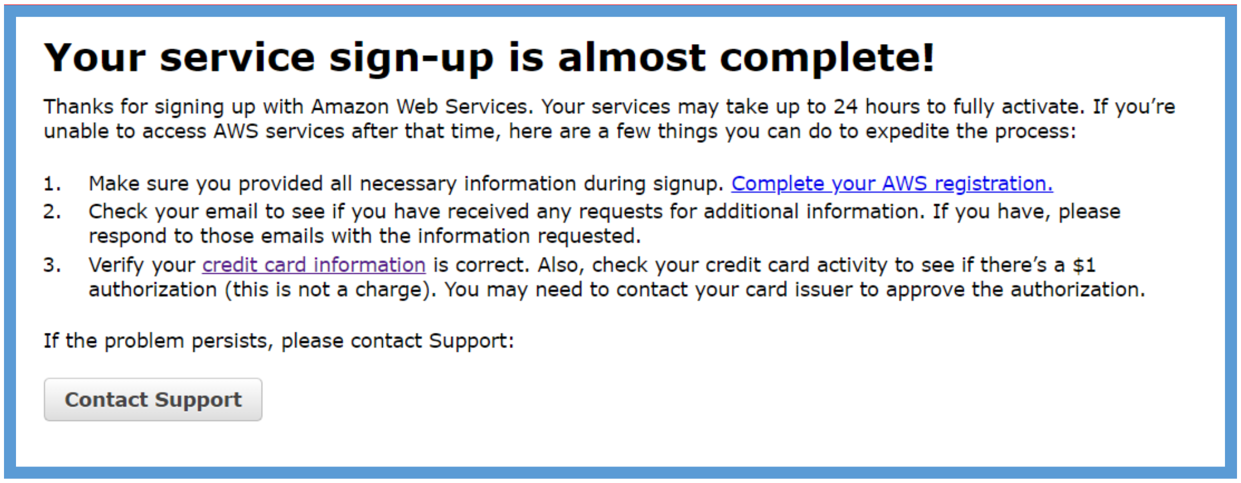
To complete this class, you will need to use your own GitHub, Octopus Deploy and Amazon Web Services (AWS) accounts. All of these are either free or fairly inexpensive (<$1 USD/day for this PoC). In order to make the best use of the class time, you are requested to create these accounts in advance.

It is strongly recommended that you use brand new and/or personal accounts, rather than existing company accounts to build your PoC. This has a few advantages:

- You are safe to experiment – and safe to fail, confident in the knowledge that you cannot break any important company assets.
- You can be sure that you are not going to exceed any of the free limits and run into a paywall.
- It removes the risk that you'll hit an unexpected conflict with prior work.
- You will personally own your PoC, and you'll have complete control to spin it up and tear it down whenever you like. This is a personal reference project for you to own indefinitely.

It is strongly recommended that you set up your AWS, Octopus and GitHub accounts at least 24 hours before attempting the following labs.

Setting up your AWS account in plenty of time is especially important, since it can take a while (and, occasionally, up to 24 hours) for some of the AWS features to become enabled on new accounts. If you don't have 24 hours to spare, observe the advice AWS provides to speed up the process:



Your service sign-up is almost complete!

Thanks for signing up with Amazon Web Services. Your services may take up to 24 hours to fully activate. If you're unable to access AWS services after that time, here are a few things you can do to expedite the process:

1. Make sure you provided all necessary information during signup. Complete your AWS registration.
2. Check your email to see if you have received any requests for additional information. If you have, please respond to those emails with the information requested.
3. Verify your credit card information is correct. Also, check your credit card activity to see if there's a $1 authorization (this is not a charge). You may need to contact your card issuer to approve the authorization.

If the problem persists, please contact Support:

If your AWS account is not yet fully activated when you try to complete the rest of this class, be aware that you may have to wait before some of the services that you will need are enabled. If they are not yet enabled, some of the Octopus Runbooks that we'll build in later labs may fail.

If any Runbooks/scripts do fail, read the logs and look for any errors referencing the fact that your account is a new account or that some AWS Services are not enabled for your account. In this case, you may just need to be patient and try again a bit later.

GitHub

If you already have a GitHub account, you can use your existing account and skip to the Octopus Deploy section below. Otherwise:

1. Follow this link to the GitHub website: https://github.com/
2. Follow the instructions to sign up and create a free account. (You will not need any paid features to complete this class.)
3. You should not need to provide any credit card details.

Octopus Deploy

If you already have an Octopus Deploy Cloud account, you might be able to use that. However, **it is strongly recommended to use either a new account, or an account that does not contain anything important**, for the following reasons:

- Octopus Deploy Cloud accounts are limited to 10 deployment targets. You will need to use at least 2 targets to complete this PoC so if you are already using 9 or 10 targets you will need to either temporarily disable some of your existing targets, or you will need to create a new account.

- This class requires that you have admin access to your Octopus Deploy Cloud account. If you are not an admin on your existing account, you will not be able to complete this PoC.
- This class will add a new project to the "default space". If you do not want to use the default space in your existing Octopus Cloud Account, consider creating your own Octopus Deploy Cloud account.
- The cleanup scripts for this class have not been explicitly tested for Octopus instances that contain other projects/infrastructure. It's possible, depending on your set up, that they will delete your existing assets, as well as the PoC infrastructure. If you choose to run them, you do so at your own risk.

To create a new Octopus Deploy Cloud account:

1. Follow this link to the Octopus Deploy Website: https://octopus.com/
2. Follow the "Get Started" instructions to create yourself a new "Cloud" account. *(Note that Octopus Deploy can either be hosted on-prem or in the cloud. For this PoC we'll use a cloud instance.)*

Amazon Web Services (AWS)

1. Follow this link to the Amazon Web Services website: https://aws.amazon.com/
2. Follow these instructions to create an AWS account using the AWS free tier. *(Note that you will need to provide credit card details. While the "free" tier does not have a monthly fee, you will be billed for the services you use on a pay as you go basis. It should not cost you more than $1USD/day to build your PoC and you can delete your PoC, and stop paying, at any time.)*
3. Decide which AWS Region you would like to build your PoC in. (AWS default is us-east-2, Ohio. Default for this PoC is eu-west-1, Ireland, because it's one of five carbon neutral regions, and marginal price/latency improvements are unlikely to be major concerns for this PoC.)
4. Consider setting up a cost budget. (More information here. See "4. Set up a cost budget".) The PoC we create in this class should not cost more than $1/day to run. You might like to experiment with scaling it up and down and you might want to run it for a few days or try building it in different ways, but in either case, setting a monthly budget of $10, $20 or $30 (for example) should result in ongoing email notifications if you forget to shut down your PoC when you are finished or if your spending increases unexpectedly.

Single page cheat-sheet

A significantly more concise version of the full lab manual. This is likely to be useful if attending a live class or watching recorded demos and following instructions. However, if solely working from the lab manual, you might find it easier to use the full version.

LAB 1: Forking the repo

Repo URL:
https://github.com/dlmconsultants/my_sql_octopus_poc

GitHub secrets:

- OCTOPUS_URL
  (Like: https://my_name.octopus.app/)
- OCTOPUS_APIKEY
  (Like: API-12345667890ABCDEFGHIJ)

LAB 3: Preparing AWS

AWS user:

- Username: octopus
- Programmatic access. (Save keys!)
- Attach policy: AdministratorAccess
- Tag key: my_sql_octopus_poc
- Tag value: NULL

AWS Secrets: *(Tag with my_sql_octopus_poc)*

- Key: OCTOPUS_THUMBPRINT
  Value: Lookup Octopus / Configuration
- Key: OCTOPUS_APIKEY
  Value: New API Key ***(Sensitive!)***
- Key: STUDENT_SQL_PASSWORD
  Value: D3vOpsRocks! ***(Sensitive!)***
- Key: OCTOPUS_SQL_PASSWORD
  Value: 5re4lsoRocks! ***(Sensitive!)***
- Key: SYSADMIN_SQL_PASSWORD
  Value: S4fePa55word!!! ***(Sensitive!)***

LAB 5: Octopus Projects and Variables

Project name: my_sql_octopus_poc

Project variables:

- Name: AWS_ACCOUNT
  Value: Change type / AWS Account
- Name: OCTOPUS_APIKEY
  Value: New API Key ***(Sensitive!)***
- Name: OCTOPUS_SQL_PASSWORD
  Value: 5re4lsoRocks! ***(Sensitive!)***

LAB 6: Creating an Octopus Runbook

Step type: Script/Run a Script
Execution Location: Run once on a worker
Script Source: Script file inside a package
Script File in Package:

- Package ID:
  my_sql_octopus_poc_infra
- File name:
  provision_environment_runbook.ps1

Repeat for: kill_environment_runbook.ps1

LAB 7: Running a Runbook

RDP creds: student / D3vOpsRocks!
Startup log and files: C:\StartUp

LAB 8.1: Defining a SQL Server Deploy Process

Create a "SQL Server / SQL - Deploy a DACPAC from Package Parameter" step (Community Step Template):

- Name: Deploy AdventureWorks dacpac
- Role: my_sql_octopus_poc-DbJumpbox
- DACPAC Package Name:
  AdventureWorks.dacpac
- Target Servername: `#{SQLSERVER_IP}`
- Target Database: AdventureWorks
- Username: octopus
- Password: `#{OCTOPUS_SQL_PASSWORD}`
- DACPAC Package / Package ID:
  AdventureWorks

LAB 11: Deploying your database(s)

Before you deploy, headphones on, [click here.]()

LAB 1: Forking the repo

Objective

Fork your own copy of the GitHub code repository.

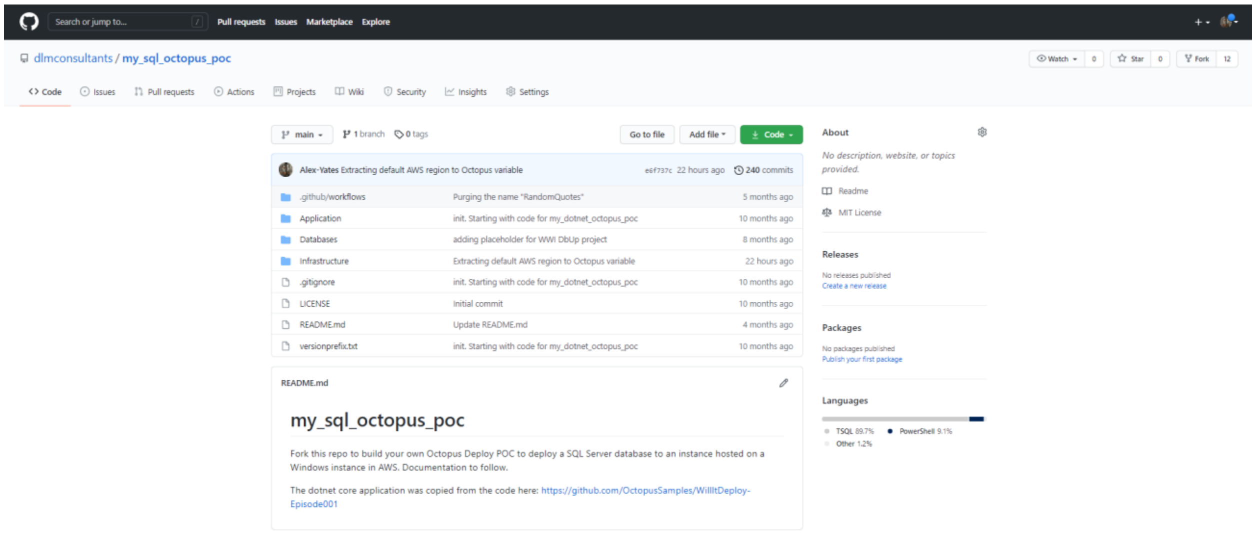
This creates a sort of source control branch, where you can safely experiment and make mistakes. This will also act as your personal reference project, since you will have complete control over the code, and you will be able to review it whenever you like.

The code in your fork will be public. (If you would like to make it private you can duplicate the repo and make it private under the "Settings" tab, but that's outside the scope of these training materials). However, your fork does not know how to communicate with your Octopus Cloud account, and you do not want to publish any passwords in your public code.

In order to enable GitHub to communicate securely with Octopus, without compromising your security, you will create a couple of GitHub "secrets" to hold your Octopus credentials.

Steps

1. Ensure you are signed into your GitHub account.
2. Follow the link below to access the code for today's class:
   https://github.com/dlmconsultants/my_sql_octopus_poc



Alex-Yates Extracting default AWS region to Octopus variable e6f737c 22 hours ago 240 commits

.github/workflows	Purging the name "RandomQuotes"	5 months ago
Application	init. Starting with code for my_dotnet_octopus_poc	10 months ago
Databases	adding placeholder for WWI DbUp project	8 months ago
Infrastructure	Extracting default AWS region to Octopus variable	22 hours ago
.gitignore	init. Starting with code for my_dotnet_octopus_poc	10 months ago
LICENSE	Initial commit	10 months ago
README.md	Update README.md	4 months ago
versionprefix.txt	init. Starting with code for my_dotnet_octopus_poc	10 months ago

README.md

my_sql_octopus_poc

Fork this repo to build your own Octopus Deploy POC to deploy a SQL Server database to an instance hosted on a Windows instance in AWS. Documentation to follow.

The dotnet core application was copied from the code here: https://github.com/OctopusSamples/WillItDeploy-Episode001

About

No description, website, or topics provided.

Readme

MIT License

Releases

No releases published
Create a new release

Packages

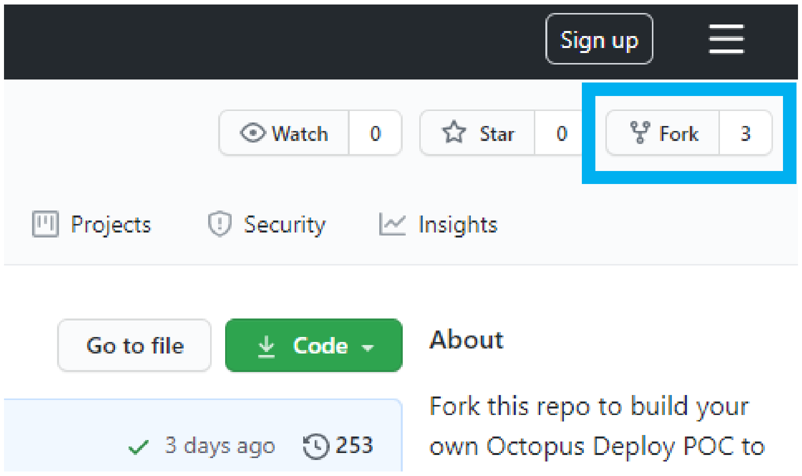
No packages published
Publish your first package

Languages

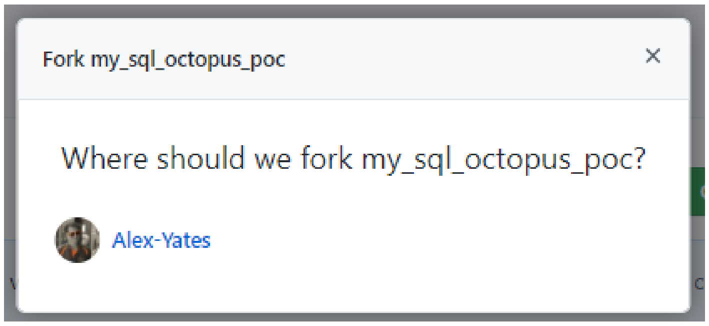
TSQL 89.7% PowerShell 9.1% Other 1.2%

Tip:
All screenshots in this lab manual were taken in 2021. Also, most of the websites are designed to be responsive. Hence, do not be surprised if they look different in different screen resolutions or if the websites have been updated since this lab manual was written.

3. From the URL in step 2 (above), click the **Fork** button in the top-right corner of the page to "fork" the repo into your own GitHub account.

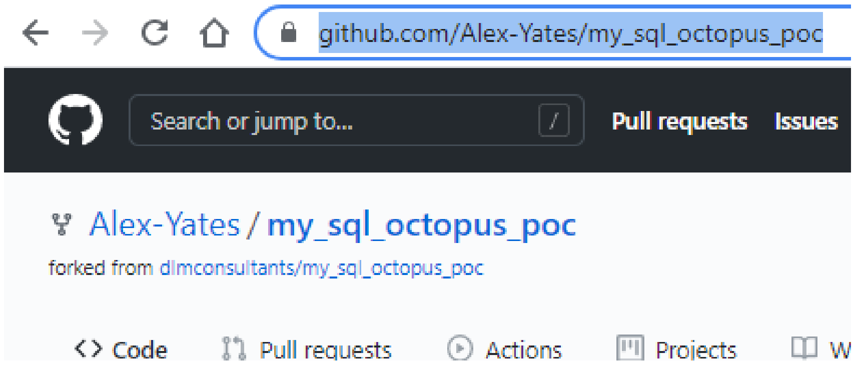


4. When asked where you would like to fork the repo to, select your personal account. In my case **Alex-Yates**:

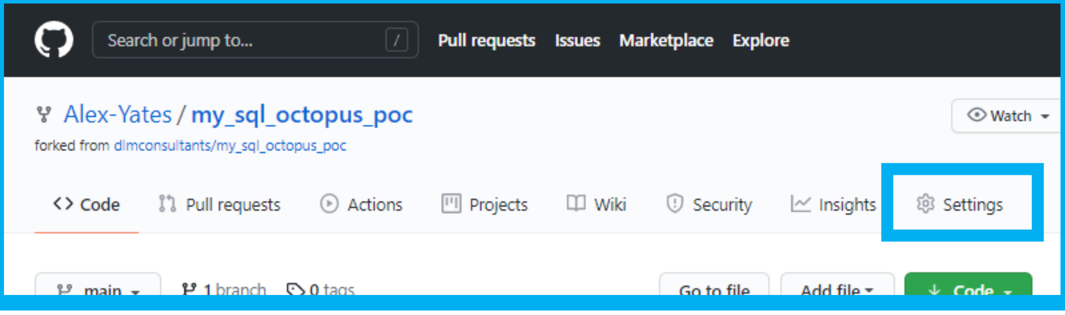


5. Observe that a copy of the repository has been made under your own account with the name "[your-account]/my_sql_octopus_poc". This is a safe place for you to play with the code. You can make changes (and break things) here, without affecting anyone else.

   You may wish to bookmark the URL for your fork, since you'll need to return to it often:

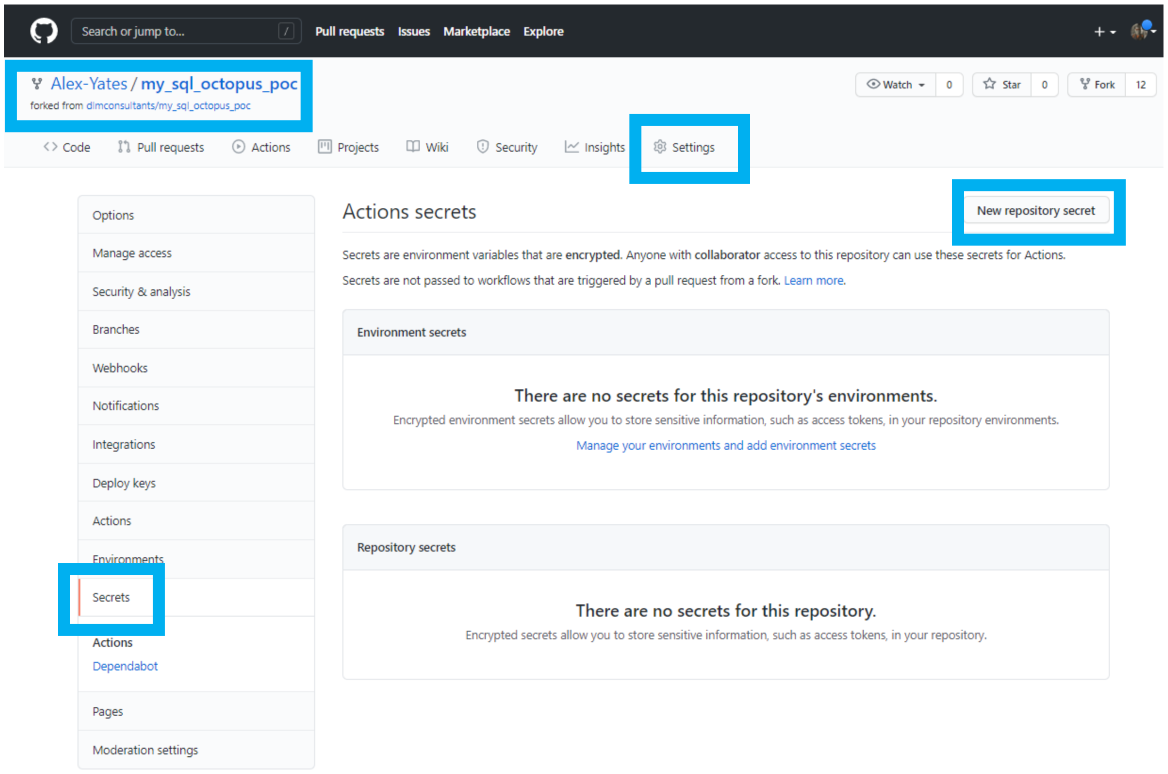

github.com/Alex-Yates/my_sql_octopus_poc
Search or jump to...
Pull requests
Issues
Alex-Yates / my_sql_octopus_poc
forked from dlmconsultants/my_sql_octopus_poc
Code
Pull requests
Actions
Projects

Tip:
If you ever break the code, and you cannot fix it, you can delete your fork under the "Settings" tab. Then you can repeat this lab to recreate a clean fork.



Now we need to tell your GitHub account the address of your Octopus Server and how to authenticate against it. However, since we care about security, we don't want to save this information in the source code itself. Instead, we'll create a couple of GitHub secrets. Later on, our GitHub Actions will be able to retrieve our Octopus information securely, without exposing it in either the source code or logs.

6. Go to the **Settings** tab in your repo and select **Secrets**, then click **New repository secret**.



Secrets are environment variables that are encrypted. Anyone with collaborator access to this repository can use these secrets for Actions.

Secrets are not passed to workflows that are triggered by a pull request from a fork. Learn more.

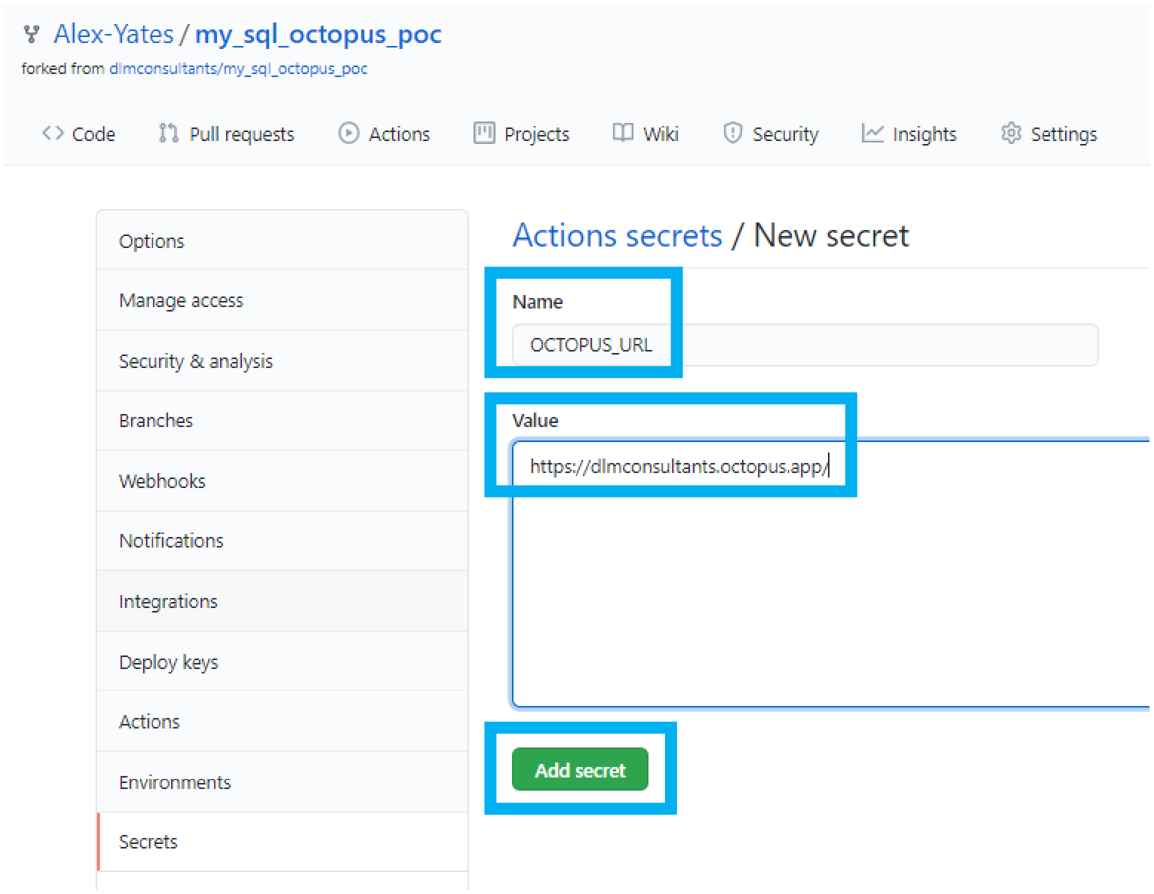
There are no secrets for this repository's environments.

Encrypted environment secrets allow you to store sensitive information, such as access tokens, in your repository environments.

There are no secrets for this repository.

Encrypted secrets allow you to store sensitive information, such as access tokens, in your repository.

7. Create a secret as follows, then click **Add secret**:
   a. Name: **OCTOPUS_URL**
   b. Value: https://my_name.octopus.app/
   (Note: To find your personal Octopus Server URL, log into your Octopus Deploy Cloud account and check the URL)

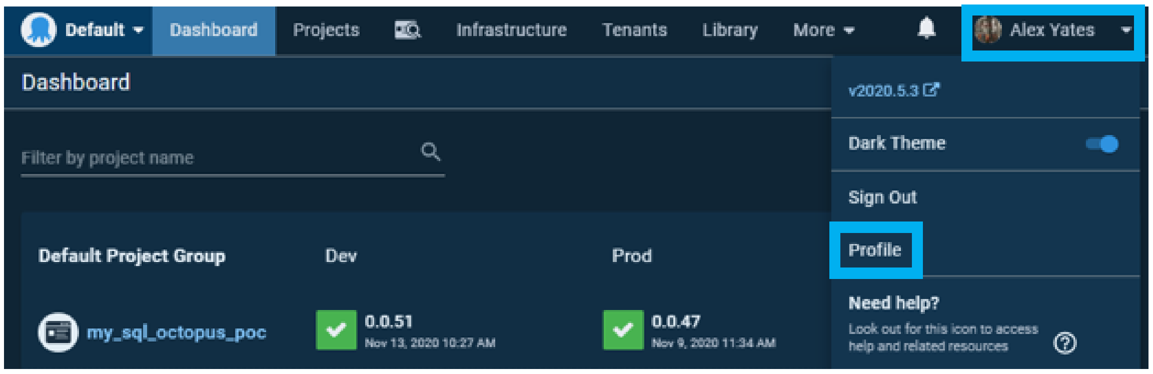


Name: OCTOPUS_URL

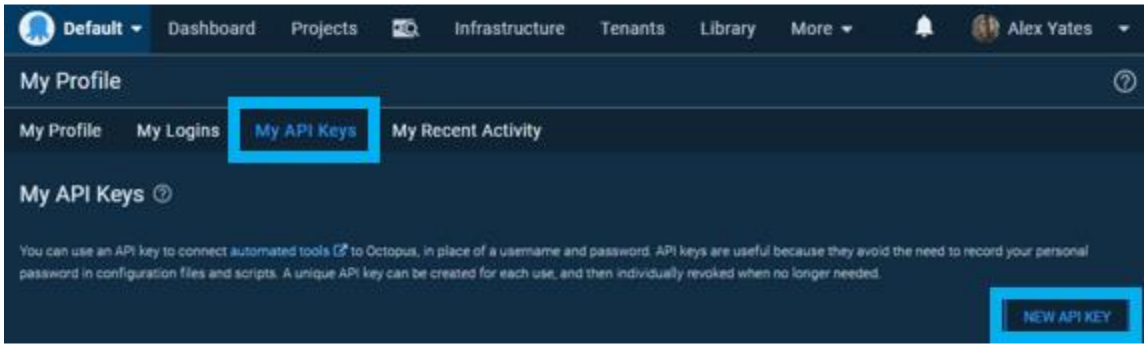
Value: https://dlmconsultants.octopus.app/

8. You need to generate a new Octopus Deploy API Key.

   In a separate tab, log into your Octopus Deploy Cloud account. If you see a "getting started" tutorial, you can exit the tutorial. Click on the dropdown next to your username in the top-right corner and select **Profile**:

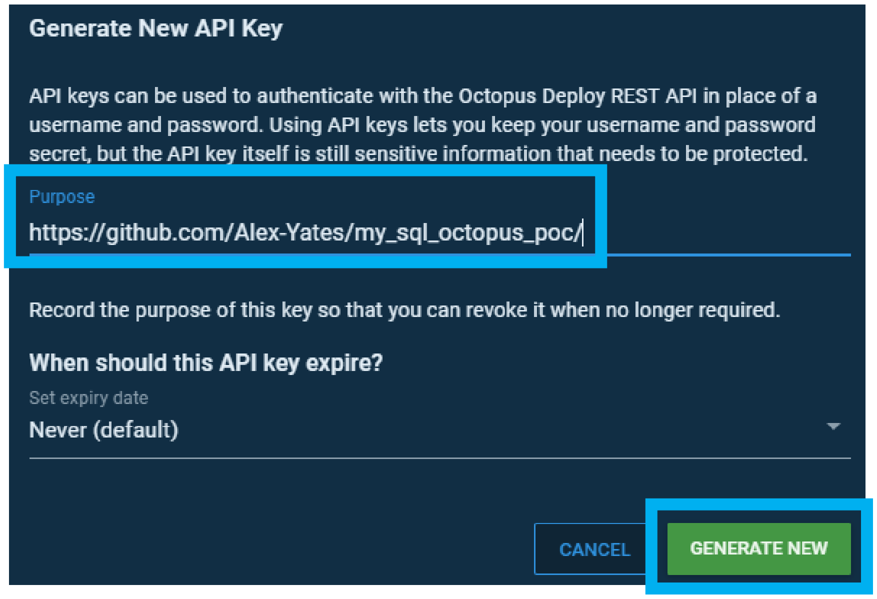


9. Select **My API Keys** and create a new one, giving it the name or URL of your GitHub fork for reference:



My API Keys

You can use an API key to connect automated tools to Octopus, in place of a username and password. API keys are useful because they avoid the need to record your personal password in configuration files and scripts. A unique API key can be created for each use, and then individually revoked when no longer needed.



Generate New API Key

API keys can be used to authenticate with the Octopus Deploy REST API in place of a username and password. Using API keys lets you keep your username and password secret, but the API key itself is still sensitive information that needs to be protected.

Purpose
https://github.com/Alex-Yates/my_sql_octopus_poc/

Record the purpose of this key so that you can revoke it when no longer required.

When should this API key expire?

Set expiry date
Never (default)

10. Click **GENERATE NEW** and make a note of the API key that Octopus Deploy generates for you. Octopus API keys have 36 characters and start with “API-”. (Note: it’s important to include those four leading characters.)

***Tip*:**
This is the only time Octopus will reveal your API key, so make a temporary note of it somewhere safe. If you ever lose it, or it becomes compromised, you can delete the API key in your Octopus profile and generate a new one.

11. Repeat step 7 (above) to create a second GitHub secret. This time, providing your Octopus Deploy API key, enabling your GitHub repo to authenticate against your Octopus Deploy Cloud account.
    a. Name: **OCTOPUS_APIKEY**
    b. Value: API-12345667890ABCDEFGHIJKLMNOPQRSTUV *(for example)*

Further reading

Documentation:

- Working with forks (GitHub docs): https://docs.github.com/en/free-pro-team@latest/github/collaborating-with-issues-and-pull-requests/working-with-forks
- Encrypted secrets (GitHub docs): https://docs.github.com/en/free-pro-team@latest/actions/reference/encrypted-secrets

Opinion:

- The definitive guide to forks and branches in git (Pluralsight article): https://www.pluralsight.com/blog/software-development/the-definitive-guide-to-forks-and-branches-in-git
- Your branching strategy should mirror your reality (Alex Yates blog post): http://workingwithdevs.com/branching-reality/

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

LAB 2: Creating packages

Objective

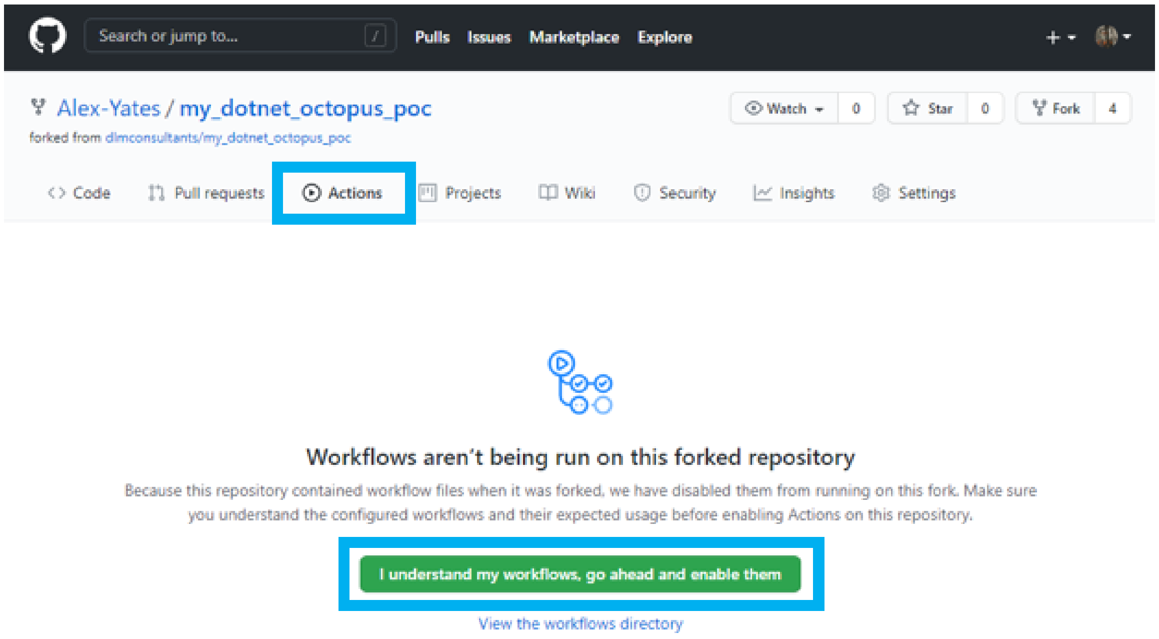
Use GitHub Actions to:

1. Build the necessary source code and binaries for the application and the infrastructure.
2. Pack the necessary files into a pair of NuGet packages.
3. Publish those NuGet packages to the Octopus Deploy built-in package repository.

In later labs, these packages will be used to deploy the infrastructure and application.

Steps

1. Navigate to your forked repository in GitHub and select Actions. Acknowledge that workflows are disabled by default on forks, then click the green button to enable them for this fork:


Search or jump to...
Pulls
Issues
Marketplace
Explore
Alex-Yates / my_dotnet_octopus_poc
forked from dlmconsultants/my_dotnet_octopus_poc
Watch 0
Star 0
Fork 4
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings

Workflows aren't being run on this forked repository

Because this repository contained workflow files when it was forked, we have disabled them from running on this fork. Make sure you understand the configured workflows and their expected usage before enabling Actions on this repository.

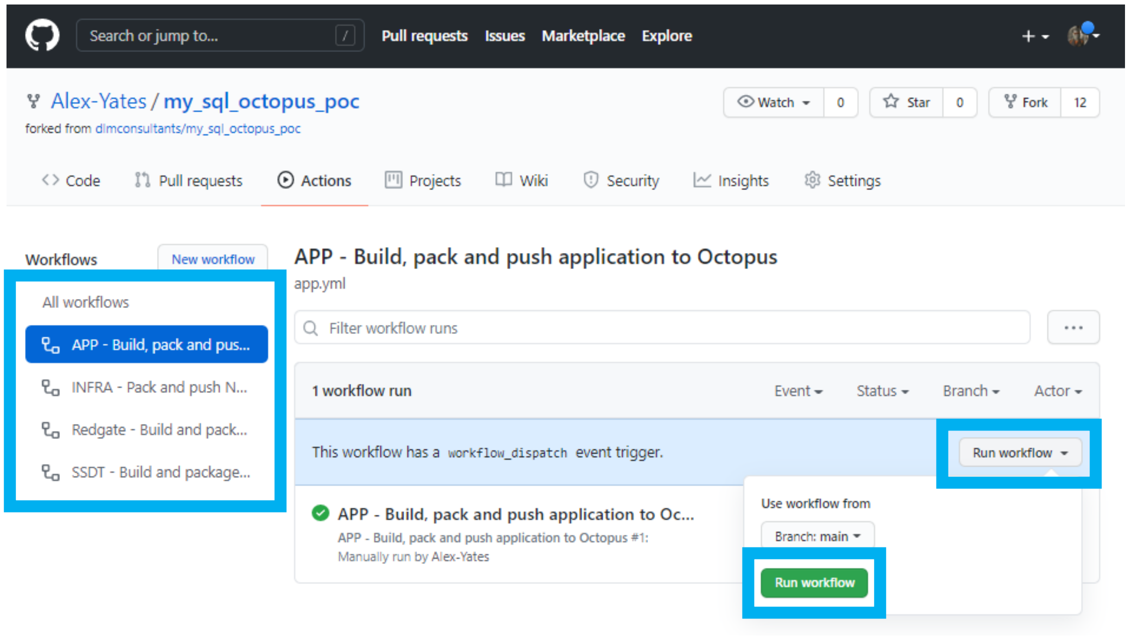
I understand my workflows, go ahead and enable them
View the workflows directory

<u>Tip</u>:
Workflows (the automation scripts that are executed by GitHub Actions) are disabled by default on forks. GitHub asks you to verify that you understand what the automation scripts do before you run them.

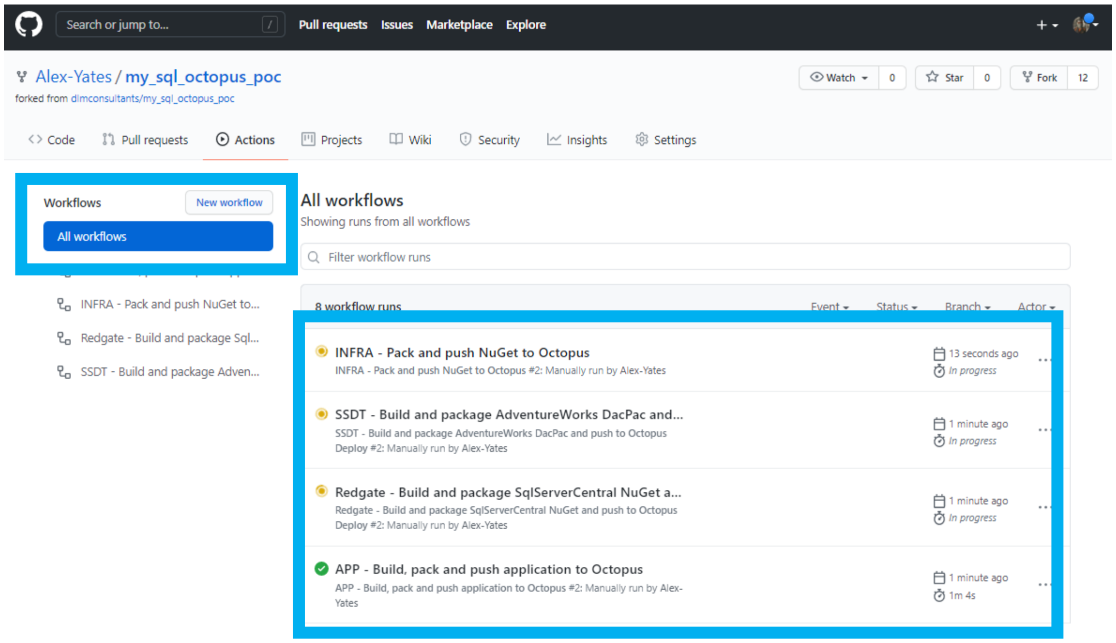
It's like a disclaimer: You are confirming that you understand the workflows, and that you take responsibility for them, before you run them.

If you would like to read the code before enabling workflows, you can do so by clicking the "View the workflows directory" text, which is immediately below the green button. (See screenshot above.)

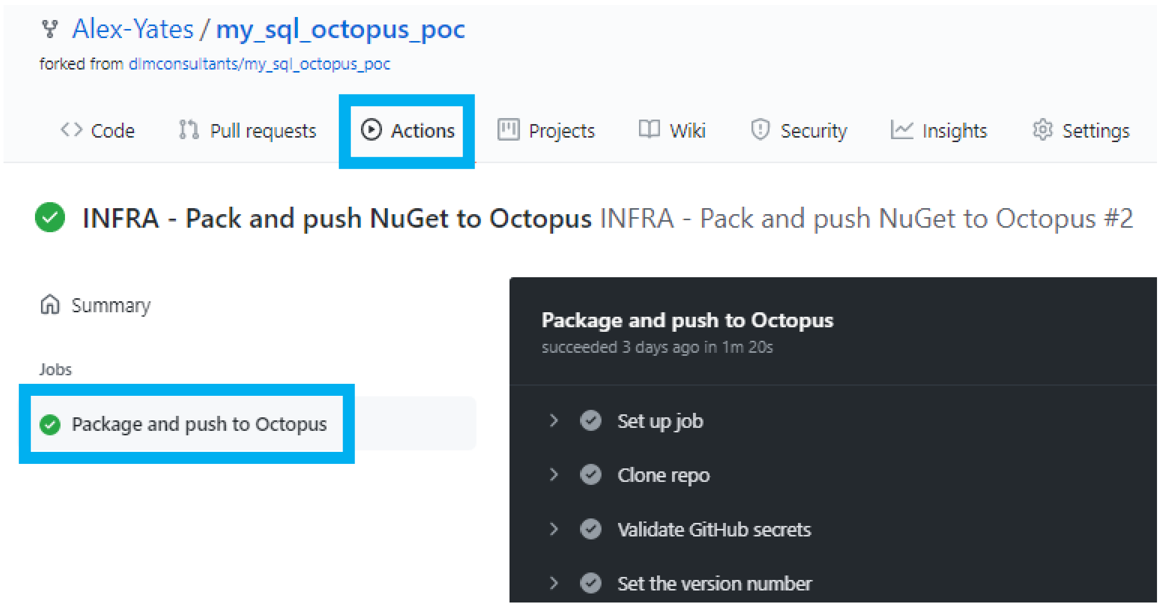
2. From the **Actions** tab on your forked repository, observe several workflows: **APP**, **INFRA, Redgate** and **SSDT**. Select each in turn and run them.


Search or jump to...
Pull requests
Issues
Marketplace
Explore
Alex-Yates / my_sql_octopus_poc
forked from dlmconsultants/my_sql_octopus_poc
Watch 0
Star 0
Fork 12
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Workflows
New workflow
All workflows
APP - Build, pack and pus...
INFRA - Pack and push N...
Redgate - Build and pack...
SSDT - Build and package...
APP - Build, pack and push application to Octopus
app.yml
Filter workflow runs
1 workflow run
Event
Status
Branch
Actor
This workflow has a workflow_dispatch event trigger.
Run workflow
APP - Build, pack and push application to Oc...
APP - Build, pack and push application to Octopus #1: Manually run by Alex-Yates
Use workflow from
Branch: main
Run workflow

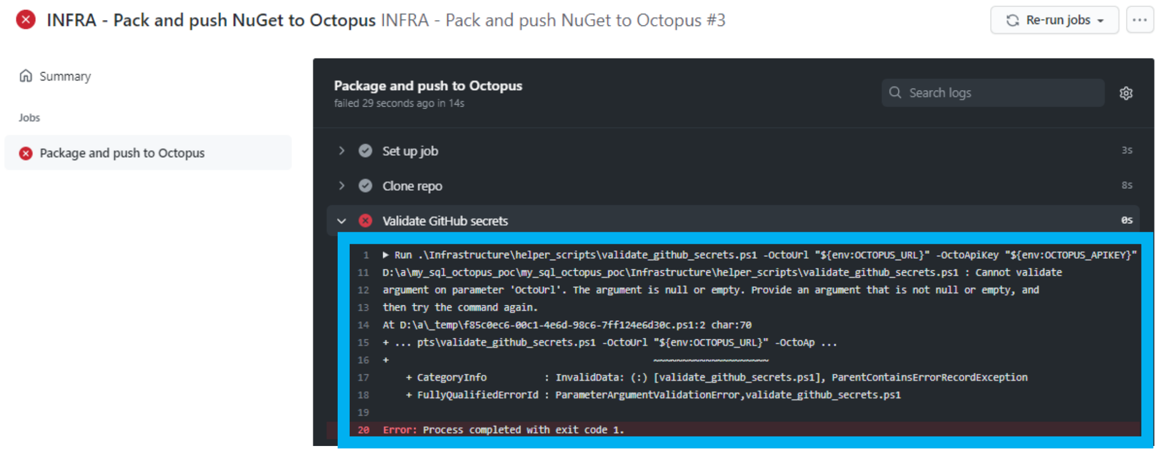
3. Once you have triggered all workflows, select **All workflows** either from the sidebar or the dropdown (depending on resolution) and observe that both Actions are running.


Search or jump to...
Pull requests
Issues
Marketplace
Explore
Alex-Yates / my_sql_octopus_poc
forked from dlmconsultants/my_sql_octopus_poc
Watch 0
Star 0
Fork 12
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Workflows
New workflow
All workflows
INFRA - Pack and push NuGet to...
Redgate - Build and package Sql...
SSDT - Build and package Adven...
All workflows
Showing runs from all workflows
Filter workflow runs
8 workflow runs
Event
Status
Branch
Actor
INFRA - Pack and push NuGet to Octopus
INFRA - Pack and push NuGet to Octopus #2: Manually run by Alex-Yates
13 seconds ago
In progress
SSDT - Build and package AdventureWorks DacPac and...
SSDT - Build and package AdventureWorks DacPac and push to Octopus Deploy #2: Manually run by Alex-Yates
1 minute ago
In progress
Redgate - Build and package SqlServerCentral NuGet a...
Redgate - Build and package SqlServerCentral NuGet and push to Octopus Deploy #2: Manually run by Alex-Yates
1 minute ago
In progress
APP - Build, pack and push application to Octopus
APP - Build, pack and push application to Octopus #2: Manually run by Alex-Yates
1 minute ago
1m 4s

4. Watch the workflows run to ensure they finish successfully. You can click on them to watch the live logs if you wish. You should expect them to run for a few minutes each.


Alex-Yates / my_sql_octopus_poc
forked from dlmconsultants/my_sql_octopus_poc
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
INFRA - Pack and push NuGet to Octopus INFRA - Pack and push NuGet to Octopus #2
Summary
Jobs
Package and push to Octopus
Package and push to Octopus
succeeded 3 days ago in 1m 20s
Set up job
Clone repo
Validate GitHub secrets
Set the version number

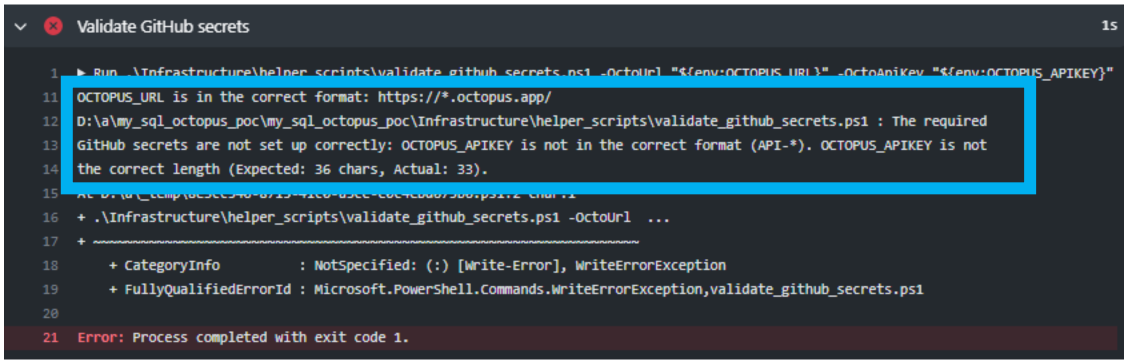
If the workflows fail, look through the logs to investigate why. The most likely reason is that you've failed to update your secrets correctly and GitHub is unable to authenticate against Octopus Deploy (see Lab 1, steps 7-11). For example, if the GitHub secret is missing:


INFRA - Pack and push NuGet to Octopus INFRA - Pack and push NuGet to Octopus #3
Re-run jobs
Summary
Jobs
Package and push to Octopus
Package and push to Octopus
failed 29 seconds ago in 14s
Search logs
Set up job 3s
Clone repo 8s
Validate GitHub secrets 0s

```
▶ Run .\Infrastructure\helper_scripts\validate_github_secrets.ps1 -OctoUrl "${env:OCTOPUS_URL}" -OctoApiKey "${env:OCTOPUS_APIKEY}"
D:\a\my_sql_octopus_poc\my_sql_octopus_poc\Infrastructure\helper_scripts\validate_github_secrets.ps1 : Cannot validate
argument on parameter 'OctoUrl'. The argument is null or empty. Provide an argument that is not null or empty, and
then try the command again.
At D:\a\_temp\f85c0ec6-00c1-4e6d-98c6-7ff124e6d30c.ps1:2 char:70
+ ... pts\validate_github_secrets.ps1 -OctoUrl "${env:OCTOPUS_URL}" -OctoAp ...
+                                              ~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidData: (:) [validate_github_secrets.ps1], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ParameterArgumentValidationError,validate_github_secrets.ps1

Error: Process completed with exit code 1.
```

And if the secret is not in the correct format:

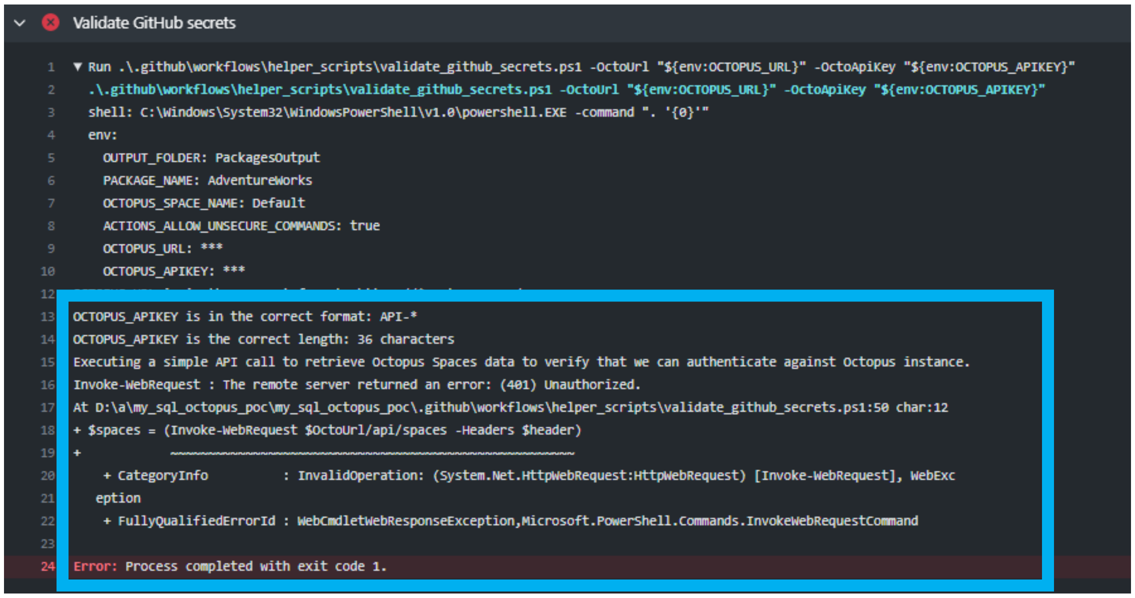


```
▶ Run .\Infrastructure\helper_scripts\validate_github_secrets.ps1 -OctoUrl "${env:OCTOPUS_URL}" -OctoApiKey "${env:OCTOPUS_APIKEY}"
OCTOPUS_URL is in the correct format: https://*.octopus.app/
D:\a\my_sql_octopus_poc\my_sql_octopus_poc\Infrastructure\helper_scripts\validate_github_secrets.ps1 : The required
GitHub secrets are not set up correctly: OCTOPUS_APIKEY is not in the correct format (API-*). OCTOPUS_APIKEY is not
the correct length (Expected: 36 chars, Actual: 33).
[illegible]
+ .\Infrastructure\helper_scripts\validate_github_secrets.ps1 -OctoUrl  ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
    + FullyQualifiedErrorId : Microsoft.PowerShell.Commands.WriteErrorException,validate_github_secrets.ps1

Error: Process completed with exit code 1.
```

And if the secret is in the correct format, but GitHub is still failing to authenticate against

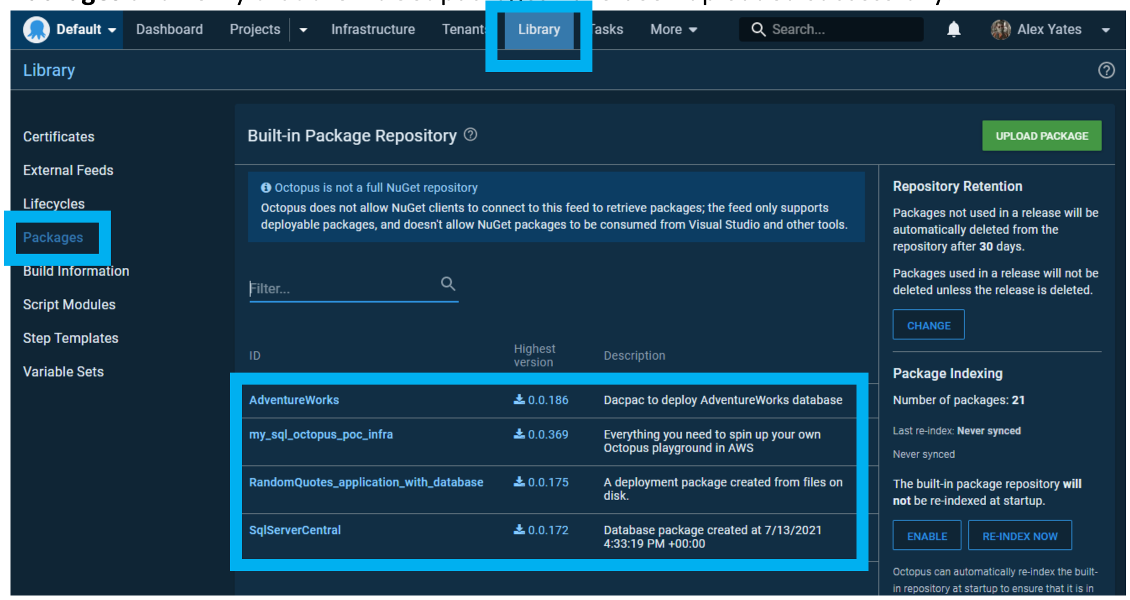
Octopus Deploy. (For example, if you have deleted your API Key in Octopus Deploy or copied it incorrectly):


Validate GitHub secrets

```
▼ Run .\.github\workflows\helper_scripts\validate_github_secrets.ps1 -OctoUrl "${env:OCTOPUS_URL}" -OctoApiKey "${env:OCTOPUS_APIKEY}"
  .\.github\workflows\helper_scripts\validate_github_secrets.ps1 -OctoUrl "${env:OCTOPUS_URL}" -OctoApiKey "${env:OCTOPUS_APIKEY}"
  shell: C:\Windows\system32\WindowsPowerShell\v1.0\powershell.EXE -command ". '{0}'"
  env:
    OUTPUT_FOLDER: PackagesOutput
    PACKAGE_NAME: AdventureWorks
    OCTOPUS_SPACE_NAME: Default
    ACTIONS_ALLOW_UNSECURE_COMMANDS: true
    OCTOPUS_URL: ***
    OCTOPUS_APIKEY: ***
OCTOPUS_APIKEY is in the correct format: API-*
OCTOPUS_APIKEY is the correct length: 36 characters
Executing a simple API call to retrieve Octopus Spaces data to verify that we can authenticate against Octopus instance.
Invoke-WebRequest : The remote server returned an error: (401) Unauthorized.
At D:\a\my_sql_octopus_poc\my_sql_octopus_poc\.github\workflows\helper_scripts\validate_github_secrets.ps1:50 char:12
+ $spaces = (Invoke-WebRequest $OctoUrl/api/spaces -Headers $header)
+            ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (System.Net.HttpWebRequest:HttpWebRequest) [Invoke-WebRequest], WebExc
   eption
    + FullyQualifiedErrorId : WebCmdletWebResponseException,Microsoft.PowerShell.Commands.InvokeWebRequestCommand

Error: Process completed with exit code 1.
```

5. If your workflows have run successfully, head over to Octopus Deploy and select **Library / Packages** and verify that the NuGet packages have been uploaded successfully:



Built-in Package Repository

Octopus is not a full NuGet repository

Octopus does not allow NuGet clients to connect to this feed to retrieve packages; the feed only supports deployable packages, and doesn't allow NuGet packages to be consumed from Visual Studio and other tools.

Filter...

ID	Highest version	Description
AdventureWorks	0.0.186	Dacpac to deploy AdventureWorks database
my_sql_octopus_poc_infra	0.0.369	Everything you need to spin up your own Octopus playground in AWS
RandomQuotes_application_with_database	0.0.175	A deployment package created from files on disk.
SqlServerCentral	0.0.172	Database package created at 7/13/2021 4:33:19 PM +00:00

Repository Retention

Packages not used in a release will be automatically deleted from the repository after **30** days.

Packages used in a release will not be deleted unless the release is deleted.

Package Indexing

Number of packages: **21**

Last re-index: **Never synced**

Never synced

The built-in package repository **will not** be re-indexed at startup.

Octopus can automatically re-index the built-in repository at startup to ensure that it is in

Further reading

Code:

- The workflow source code*: https://github.com/dlmconsultants/my_sql_octopus_poc/tree/main/.github/workflows

** While the link above points at the source repo, you can see the code on your personal fork by navigating to .github/workflows under the “Code” tab on your GitHub fork.*

Documentation:

- Integrating Octopus Deploy with GitHub Actions (Octopus docs): https://octopus.com/docs/packaging-applications/build-servers/github-actions

Opinion:

- GitHub Actions now supports CI/CD, free for public repositories (Nat Friedman, GitHub blog): https://github.blog/2019-08-08-github-actions-now-supports-ci-cd/
- Publishing a package to Octopus with GitHub Actions (Ryan Rousseau, Octopus blog): https://octopus.com/blog/publishing-a-package-to-octopus-with-github-actions
- Can GitHub Actions replace your CI server? (Matthew Casperson, Octopus blog): https://octopus.com/blog/can-github-actions-replace-your-ci-server

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

LAB 3: Preparing AWS

Objective

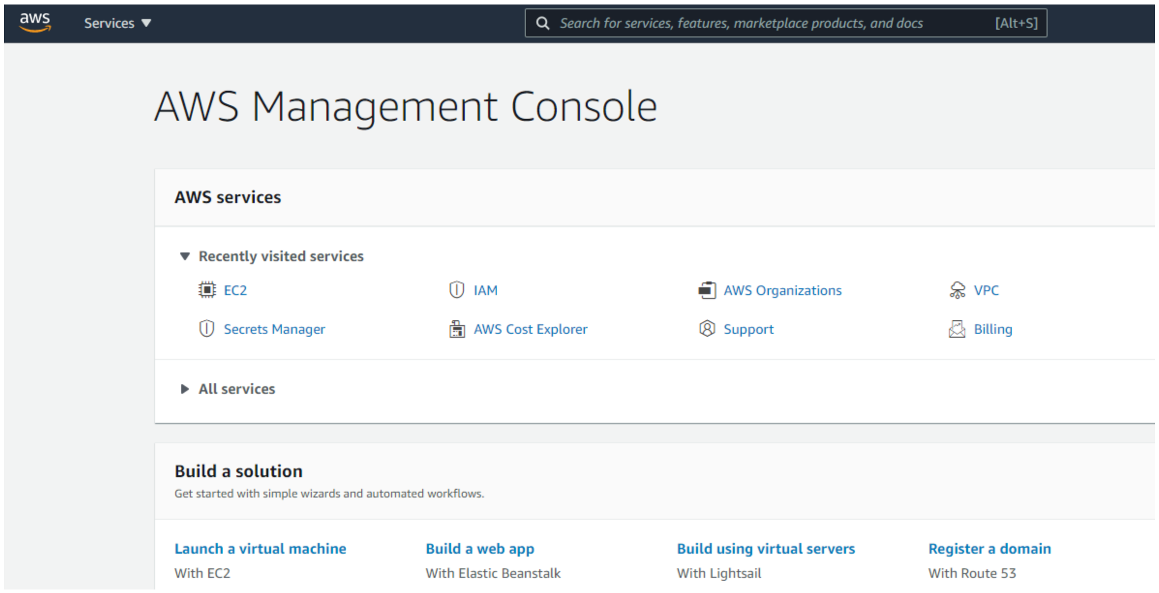
Complete a little security preparation to enable us to build EC2 instances.

We need to create:

- An IAM user for Octopus Deploy
- Some AWS Secrets, to store passwords etc

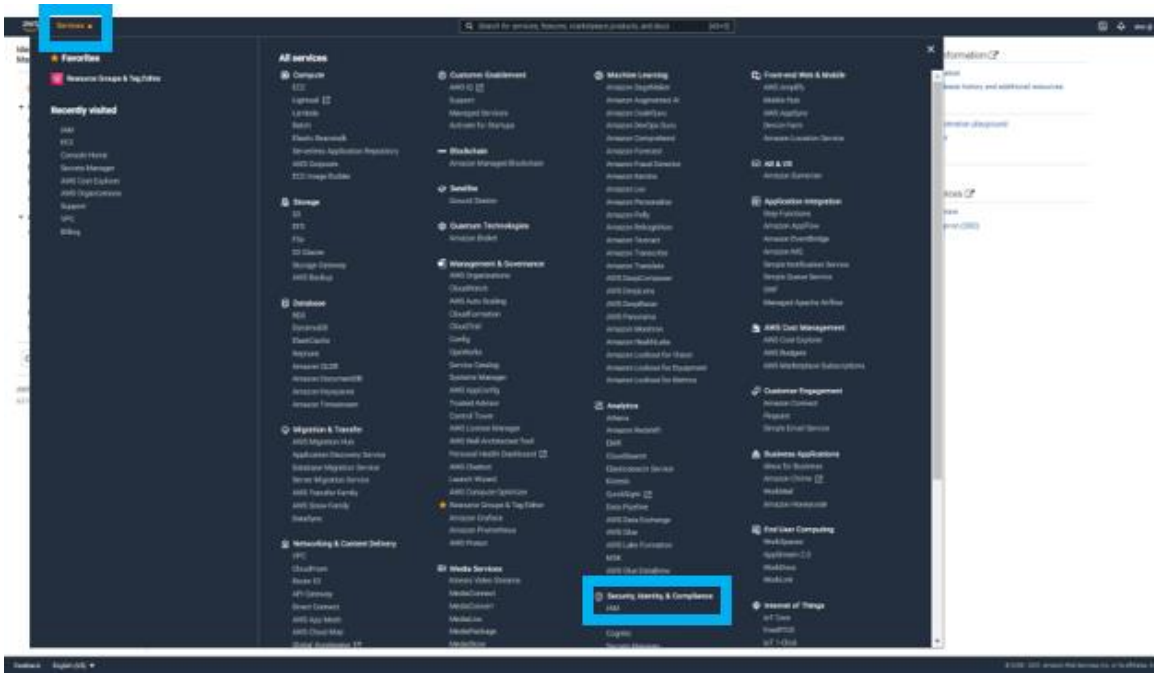
Steps: Creating an IAM user for Octopus Deploy

1. In a web browser, log in to Amazon Web Services and navigate to the AWS Management Console: https://console.aws.amazon.com/.

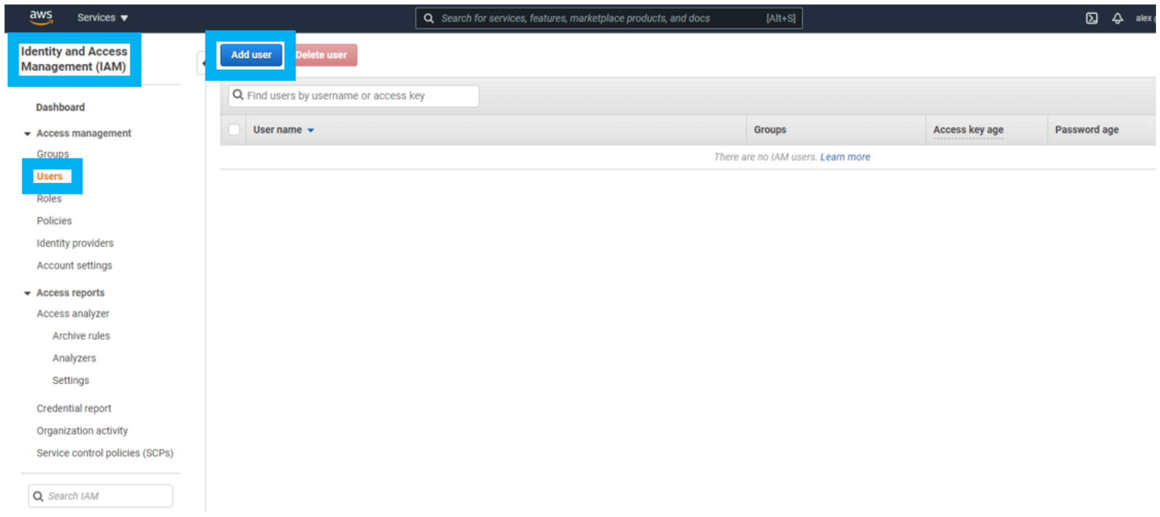


2. From the **Services** dropdown in the top left corner, select **IAM** *(Identity and Access Management)*. You will find it under the **Security, Identity, & Compliance** heading. (You will probably need to scroll to find it!) Alternatively, type "iam" into the search bar in the middle at

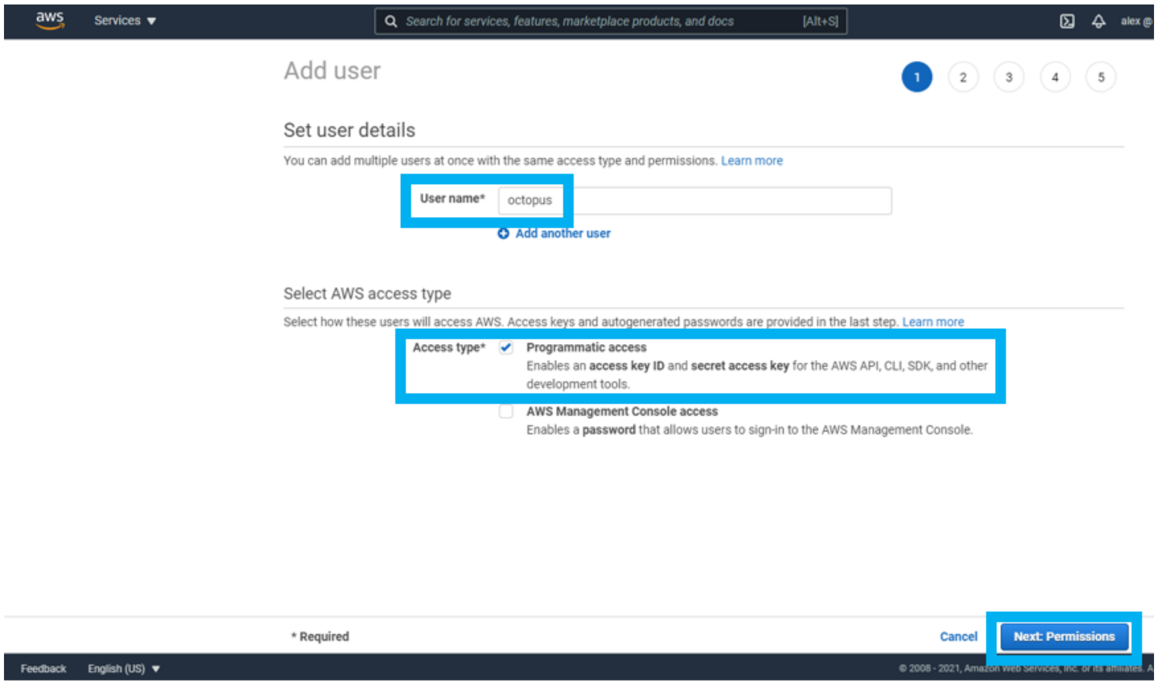
the top:



3. Within **Identity and Access Management (IAM)** select **Users** and click **Add user**:



4. Create a user called **octopus** with **Programmatic** access to AWS and click **Next: Permissions**.



Add user

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name* octopus

Add another user

Select AWS access type

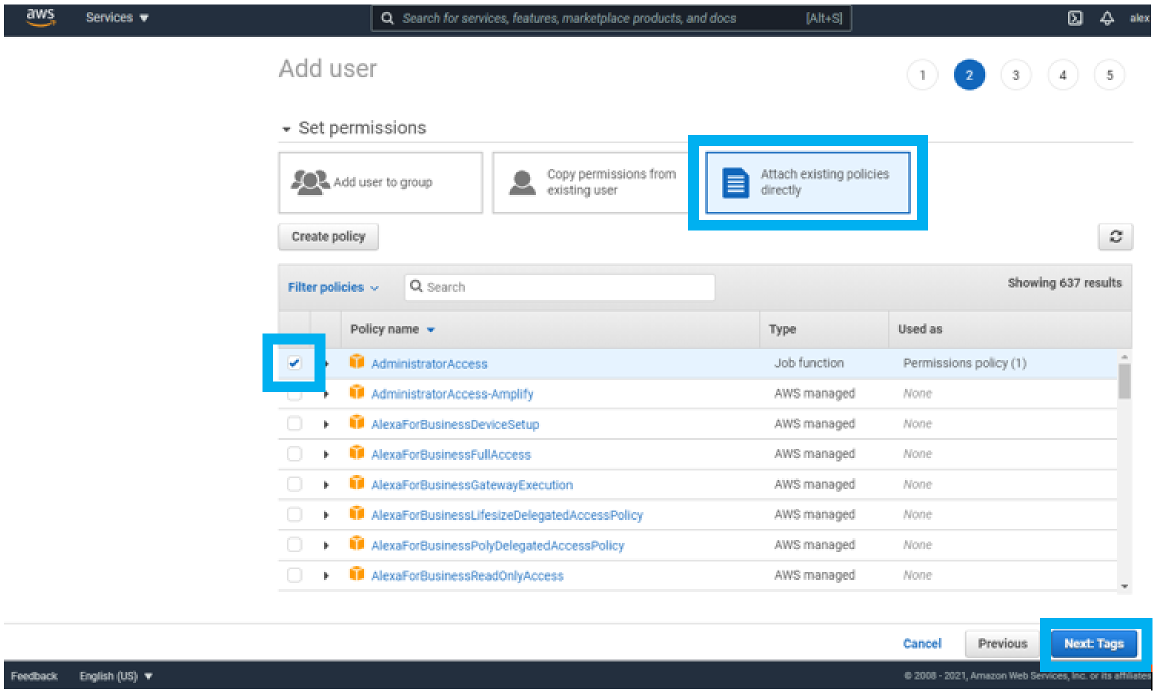
Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type* [✓ Programmatic access] Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

AWS Management Console access — Enables a **password** that allows users to sign-in to the AWS Management Console.

* Required

5. On the Permissions screen, select **Attach existing policies directly**, check the box next to **AdministratorAccess** and click **Next: Tags**. (Note, see the warning about this below).



Add user

Set permissions

Add user to group
Copy permissions from existing user
Attach existing policies directly
Create policy
Filter policies
Search

Showing 637 results

	Policy name	Type	Used as
✓	AdministratorAccess	Job function	Permissions policy (1)
	AdministratorAccess-Amplify	AWS managed	None
	AlexaForBusinessDeviceSetup	AWS managed	None
	AlexaForBusinessFullAccess	AWS managed	None
	AlexaForBusinessGatewayExecution	AWS managed	None
	AlexaForBusinessLifesizeDelegatedAccessPolicy	AWS managed	None
	AlexaForBusinessPolyDelegatedAccessPolicy	AWS managed	None
	AlexaForBusinessReadOnlyAccess	AWS managed	None

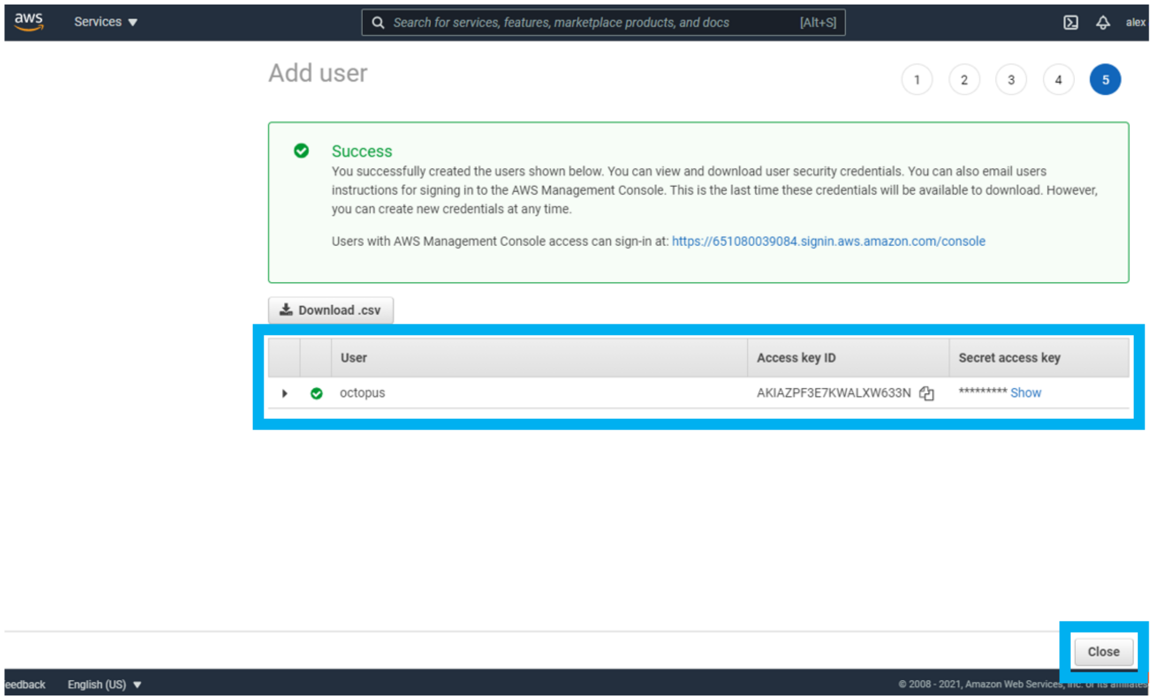
***Warning*:**
This is not good security practice. We are deliberately taking a shortcut to save time and complexity.

As the AWS console highlights, it's better to manage permissions using groups. Also, making the octopus user an Administrator is an egregious dereliction of the principle of least privilege. Most security folks would (rightly) be uncomfortable with allowing full admin rights to Octopus Deploy.

*However, this is a class about Octopus Deploy that is designed to be accessible to folks with zero prior GitHub, Octopus Deploy and AWS experience. This means we are explicit about providing the *minimum* possible GitHub and AWS detail to facilitate the simplest path to a working PoC and to give us the time to focus on Octopus features in more depth.*

This class takes the view that it is better to be bold and obvious when we are taking shortcuts, rather than rushing through important topics such as security. If you would like to learn more about better security practices in AWS, check out the Further Reading for this lab.

6. On the tags screen, add a tag with the key "**Project**" and the value "**my_sql_octopus_poc**" and click **Next: Review**.
7. On the Review screen, double-check all the details and click **Create User**.
8. You'll be presented with an **Access key ID** and a **Secret access key**. It's important that you make a note of these credentials in a safe place since you will not get another opportunity to check these. When you have made a note of the credentials for the Octopus user, click Close.


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
alex
Add user
1
2
3
4
5

Success
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://651080039084.signin.aws.amazon.com/console

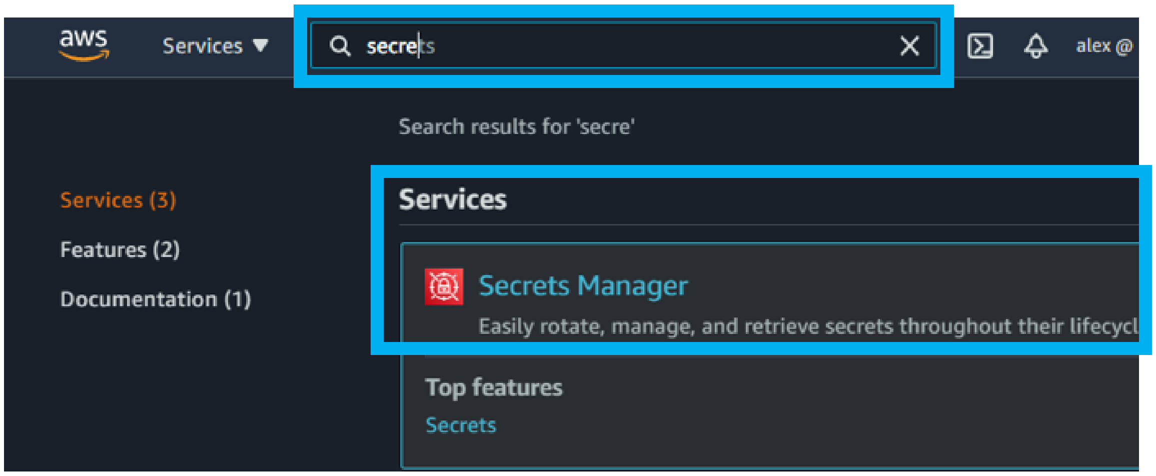
User	Access key ID	Secret access key
octopus	AKIAZPF3E7KWALXW633N	********* Show

Note:
As well as creating a user for Octopus Deploy, it's also good practice to create a non-root, non-admin user for yourself. This is not included for this class because our focus is on building an end-to-end deployment pipeline in as few steps as possible. We are deliberately restricting ourselves to the absolute minimum required AWS configuration.

However, if you are considering using AWS for any professional work, you should consider creating a more appropriate user for yourself. For more details, check out the Further Reading for this lab.

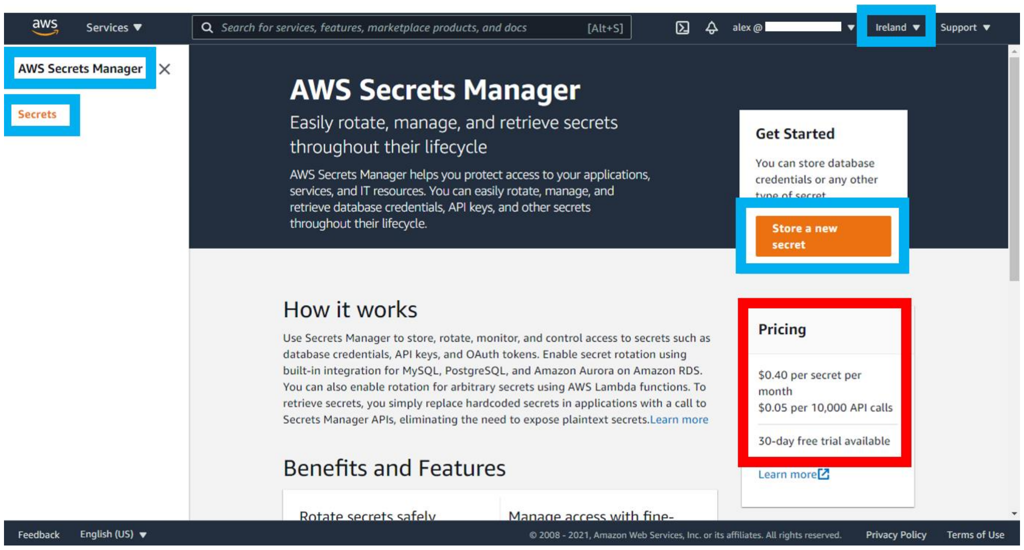
Steps: Safely store credentials with AWS Secrets Manager

9. From the AWS console, select the **Services** (top left) and **Secrets Manager**. You will find Secrets Manager under the **Security, Identity and Compliance** header. (You will probably need to scroll.) Alternatively, you can type "Secrets" into the search bar at the top.



10. Decide which AWS region you want to build your PoC in. You can read more about AWS regions here. The scripts you'll use later will default to **eu-west-1** (Ireland) because it's one of five carbon neutral AWS regions and it's unlikely that miniscule differences in price or latency are going to be a significant issue for this PoC. Unless you have a strong preference, you are recommended to build your PoC in eu-west-1 (Ireland). (There will be additional steps to follow if you pick a different region.) Whichever region you wish to use, you must be consistent for this PoC, so decide now.
11. While the AWS IAM user we created above exists at a global level, secrets, like instances, live within a specific AWS region. From the SecretsManager page, check the dropdown in the top-right corner and select your preferred region.

12. Click the **Store a new secret** button (top-right).



AWS Secrets Manager

Easily rotate, manage, and retrieve secrets throughout their lifecycle

AWS Secrets Manager helps you protect access to your applications, services, and IT resources. You can easily rotate, manage, and retrieve database credentials, API keys, and other secrets throughout their lifecycle.

Get Started

You can store database credentials or any other type of secret

Store a new secret

How it works

Use Secrets Manager to store, rotate, monitor, and control access to secrets such as database credentials, API keys, and OAuth tokens. Enable secret rotation using built-in integration for MySQL, PostgreSQL, and Amazon Aurora on Amazon RDS. You can also enable rotation for arbitrary secrets using AWS Lambda functions. To retrieve secrets, you simply replace hardcoded secrets in applications with a call to Secrets Manager APIs, eliminating the need to expose plaintext secrets. Learn more

Pricing

$0.40 per secret per month
$0.05 per 10,000 API calls

30-day free trial available

Learn more

Benefits and Features

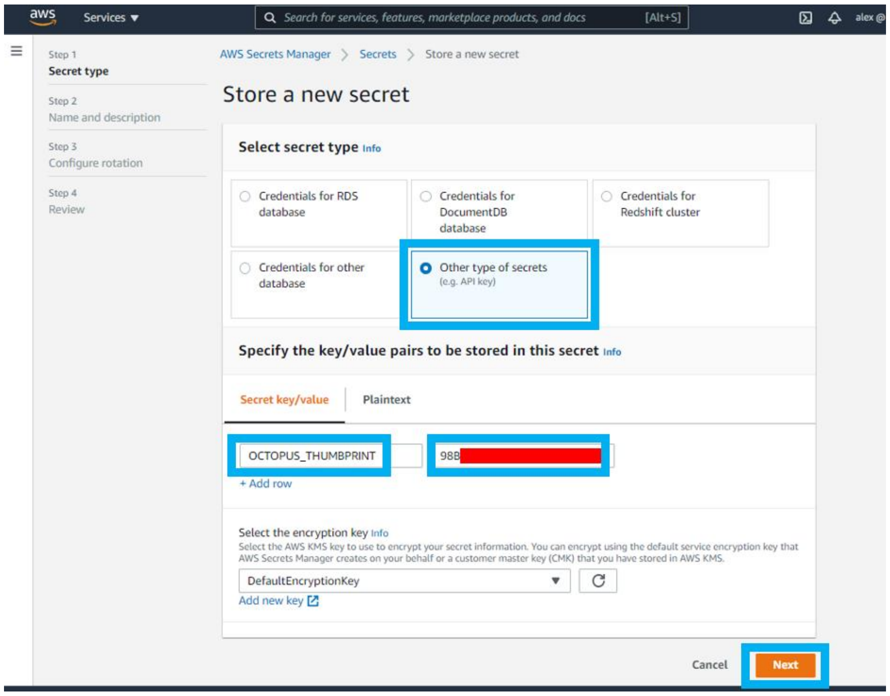
Rotate secrets safely
Manage access with fine-

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Warning:
Each secret will cost you USD $0.40/month, plus a tiny fraction of a cent for each API call. While this isn't a lot of money, you should make a note to delete your secrets when you are finished.

For more information about AWS pricing, see Appendix 1: Licensing considerations.

13. Select **Other type of secrets**, enter the following, then click **Next**:
    a. Key: **OCTOPUS_THUMBPRINT**
    b. Value: Your Octopus Thumbprint
    *(You can find your Thumbprint under **Configuration / Thumbprint** in Octopus Deploy.)*


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
alex @
Step 1
Secret type
Step 2
Name and description
Step 3
Configure rotation
Step 4
Review
AWS Secrets Manager > Secrets > Store a new secret
Cancel
Next

Store a new secret

Select secret type Info

- Credentials for RDS database
- Credentials for DocumentDB database
- Credentials for Redshift cluster
- Credentials for other database
- [✓ Other type of secrets (e.g. API key)]

Specify the key/value pairs to be stored in this secret Info

Secret key/value | Plaintext

OCTOPUS_THUMBPRINT | 98B[redacted]

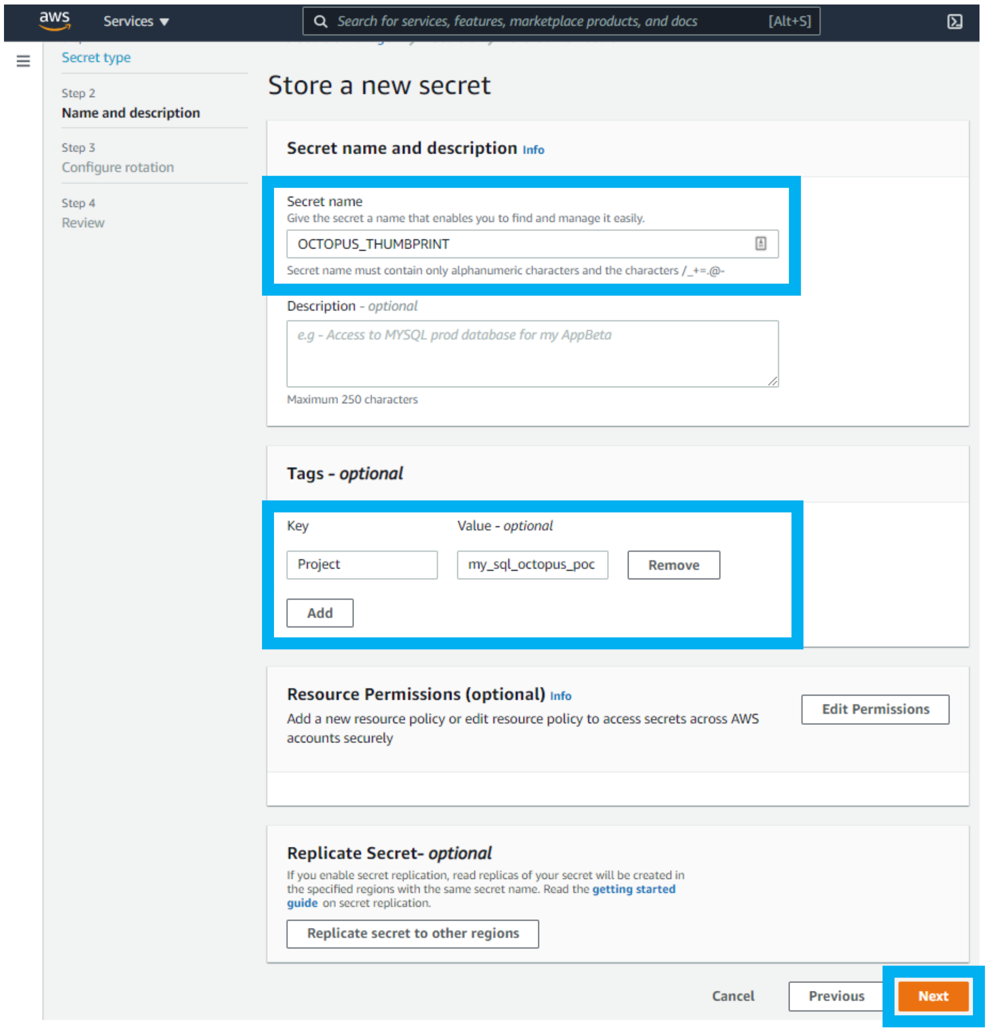
+ Add row

Select the encryption key Info
Select the AWS KMS key to use to encrypt your secret information. You can encrypt using the default service encryption key that AWS Secrets Manager creates on your behalf or a customer master key (CMK) that you have stored in AWS KMS.

DefaultEncryptionKey

Add new key

14. Give the secret the name **OCTOPUS_THUMBPRINT**, tag it with the key “Project” and the value “**my_sql_octopus_poc**” and click **Next**.



Store a new secret

Secret name and description

Secret name: OCTOPUS_THUMBPRINT

Description - *optional*: e.g - Access to MYSQL prod database for my AppBeta

Tags - *optional*

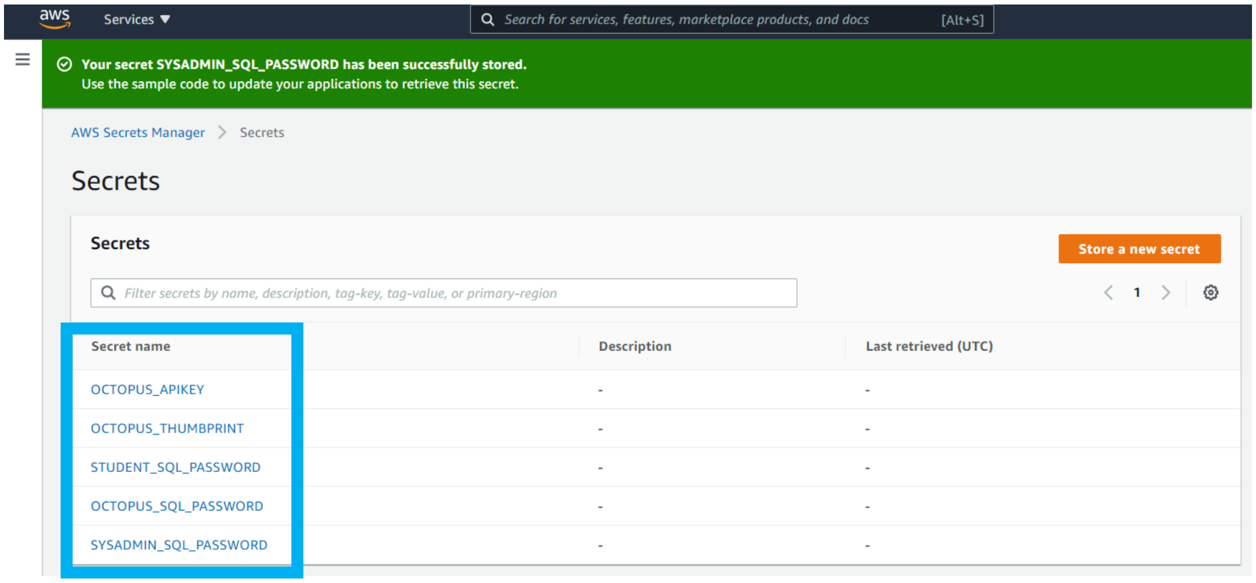
Key	Value - *optional*
Project	my_sql_octopus_poc

Resource Permissions (optional)

Replicate Secret- *optional*

Secret type
Step 2
Name and description
Step 3
Configure rotation
Step 4
Review
Give the secret a name that enables you to find and manage it easily.
Secret name must contain only alphanumeric characters and the characters /_+=.@-
Maximum 250 characters
Remove
Add
Add a new resource policy or edit resource policy to access secrets across AWS accounts securely
Edit Permissions
If you enable secret replication, read replicas of your secret will be created in the specified regions with the same secret name. Read the getting started guide on secret replication.
Replicate secret to other regions
Cancel
Previous
Next

15. Leave all the default (disabled) automatic rotation values and click **Next**.
16. Review your secret and click **Store**.
17. Repeat steps 3-7 above to create additional secrets with the following values:
    a. Key/Name: **OCTOPUS_APIKEY,** Value: *(Generate a new one, like Lab 1)*
    b. Key/Name: **STUDENT_SQL_PASSWORD,** Value: **D3vOpsRocks!**
    c. Key/Name: **OCTOPUS_SQL_PASSWORD,** Value: **5re4lsoRocks!**
    d. Key/Name: **SYSADMIN_SQL_PASSWORD,** Value: **S4fePa55word!!!**


Your secret SYSADMIN_SQL_PASSWORD has been successfully stored.
Use the sample code to update your applications to retrieve this secret.

AWS Secrets Manager > Secrets

Secrets

Secret name	Description	Last retrieved (UTC)
OCTOPUS_APIKEY	-	-
OCTOPUS_THUMBPRINT	-	-
STUDENT_SQL_PASSWORD	-	-
OCTOPUS_SQL_PASSWORD	-	-
SYSADMIN_SQL_PASSWORD	-	-

Further reading

Money:

- Appendix 1: Licencing considerations (This document): [Appendix 1: Licencing considerations]
- AWS Secrets Manager vs. Hashicorp Vault vs. AWS Parameter Store: https://hackernoon.com/aws-secrets-manager-vs-hashicorp-vault-vs-aws-parameter-store-bcbf60b0c0d1
- Appendix 2: Shutting down your PoC (This document): Appendix 2: Shutting down your PoC

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

LAB 4: Preparing Octopus

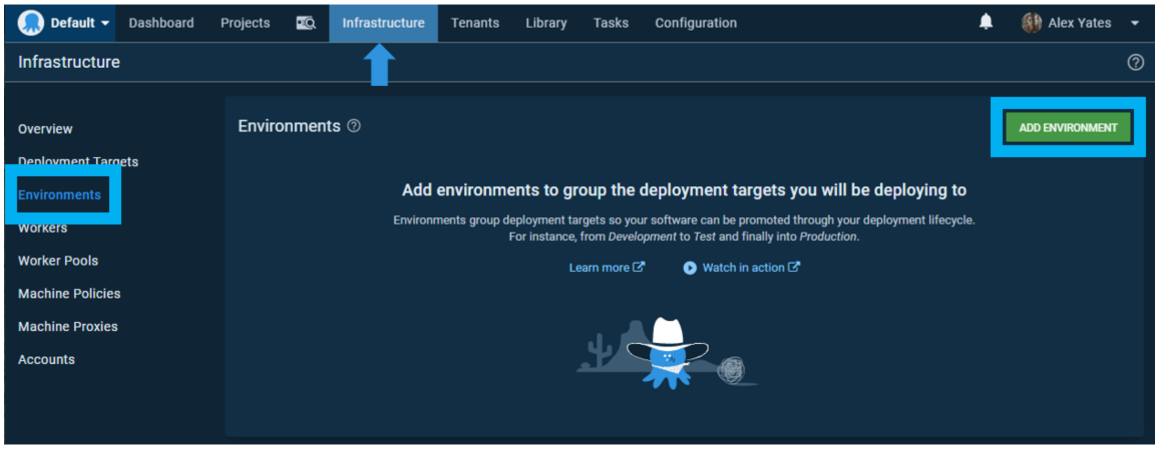
Objective

Perform a couple of tasks to prepare your Octopus Deploy account to automate your deployments:

1. Declare two environments: Dev and Prod
2. Enable Octopus to communicate with AWS

Steps: Declare two environments: Dev and Prod

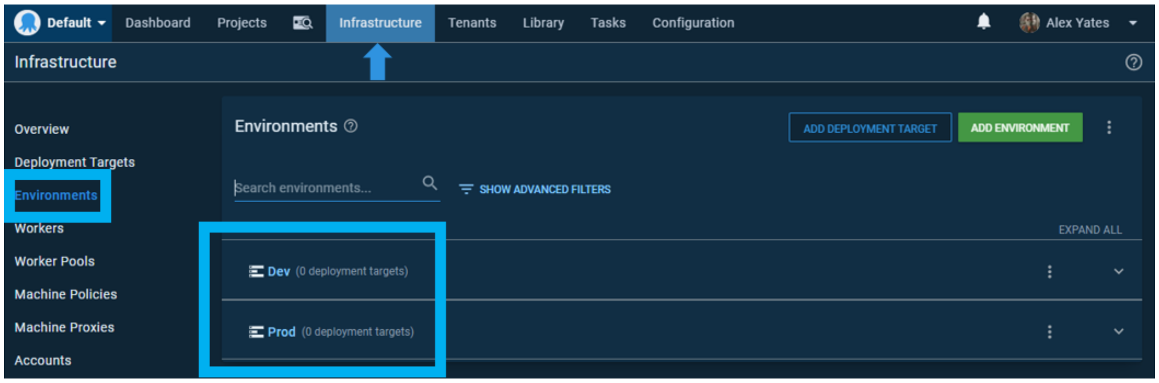
1. Log on to your Octopus instance in a web browser. If you see a "getting started" tutorial, you can exit out of it.
2. Select the **Infrastructure** tab, select **Environments** and click **ADD ENVIRONMENT**.



Add environments to group the deployment targets you will be deploying to

Environments group deployment targets so your software can be promoted through your deployment lifecycle. For instance, from *Development* to *Test* and finally into *Production*.

3. Give the environment the name "**Dev**" and click **Save**.
4. Repeat steps 2 and 3 to create a "**Prod**" environment.
5. If you click on **Infrastructure / Environments** again, you should see both your environments. For now, they are empty. However, when we provision our AWS instances, we'll be able to add our infrastructure into these environments.

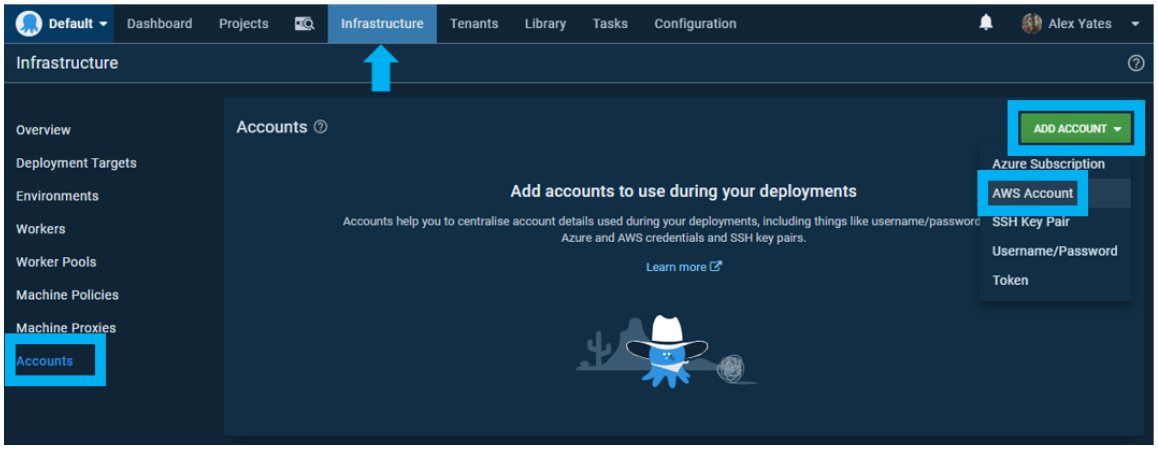


Dev (0 deployment targets)

Prod (0 deployment targets)

Steps: Enable Octopus to communicate with AWS

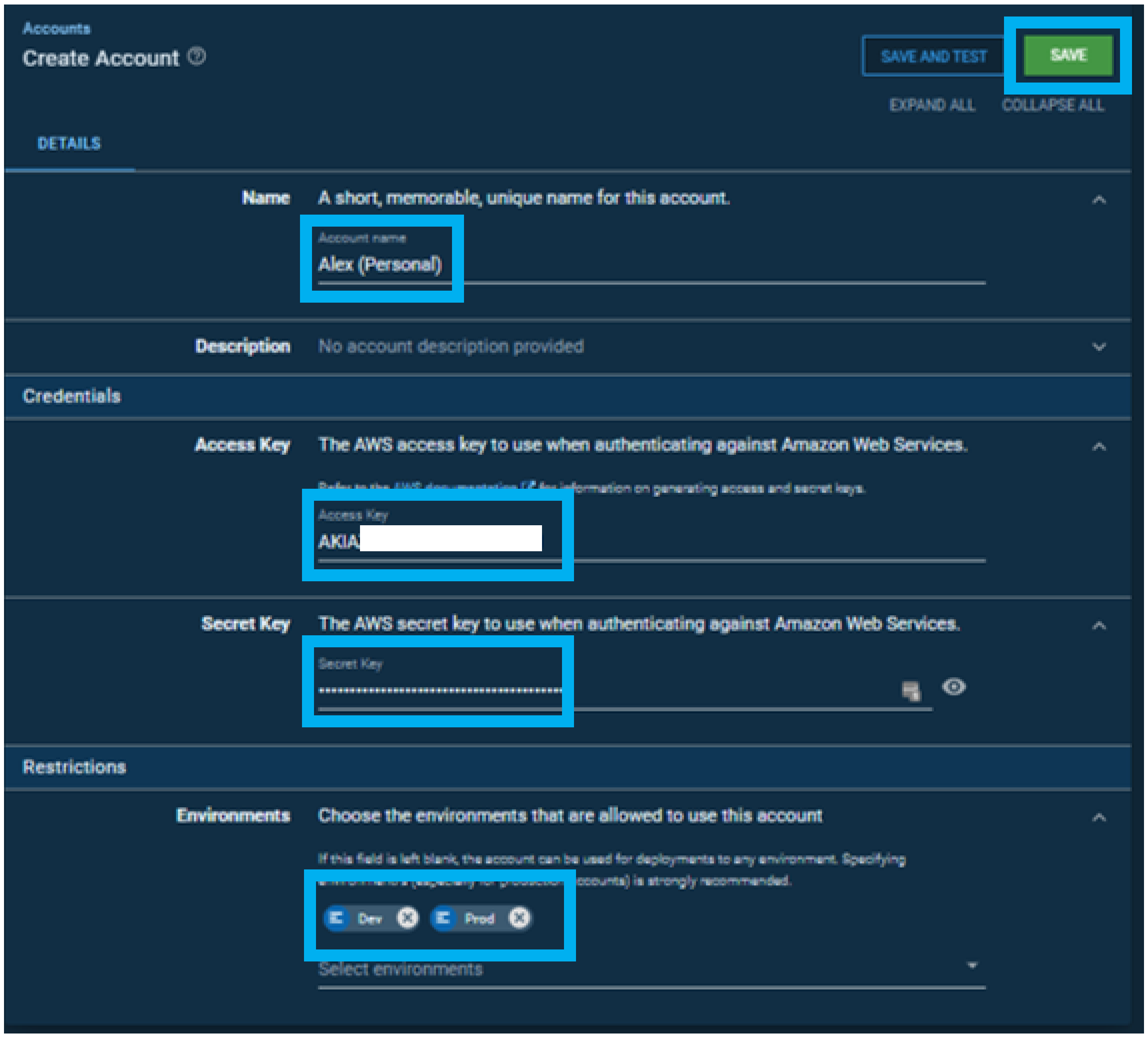
6. Navigate to your Octopus Deploy instance and select **Infrastructure / Accounts**. Click **ADD ACCOUNT** and select **AWS Account**:



Add accounts to use during your deployments

Accounts help you to centralise account details used during your deployments, including things like username/password, Azure and AWS credentials and SSH key pairs.

7. Add a name for your account (see tip below for guidance) as well as the Access and Secret Keys for the octopus user that we created in AWS in Lab 3, Step 8 (above). Then use the dropdown under **Environments** to add both Dev and Prod. Finally, click **SAVE**.



Name: A short, memorable, unique name for this account.

Account name: Alex (Personal)

Description: No account description provided

Access Key: The AWS access key to use when authenticating against Amazon Web Services.

Refer to the AWS documentation for information on generating access and secret keys.

Access Key: AKIA[redacted]

Secret Key: The AWS secret key to use when authenticating against Amazon Web Services.

Secret Key: ••

Environments: Choose the environments that are allowed to use this account

If this field is left blank, the account can be used for deployments to any environment. Specifying environments (especially for production accounts) is strongly recommended.

Tip:
Your account name will be personal to you. It can be whatever you like. Imagine if you used different AWS accounts for each of your environments, services, clients, or internal departments. The account name should help you to easily distinguish between multiple AWS accounts. In this case, you are probably referencing your personal AWS account.

Further reading

Documentation:

- Managing Environments with Octopus Deploy (Octopus documentation): https://octopus.com/docs/infrastructure/environments
- Managing AWS Accounts with Octopus Deploy (Octopus documentation): https://octopus.com/docs/infrastructure/deployment-targets/aws

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

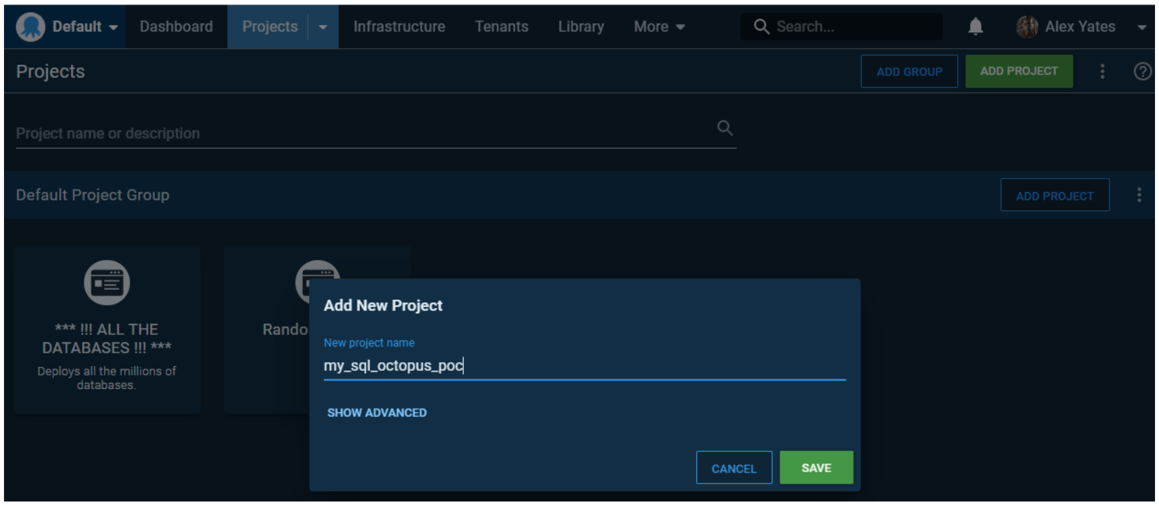
LAB 5: Octopus Projects and Variables

Objective

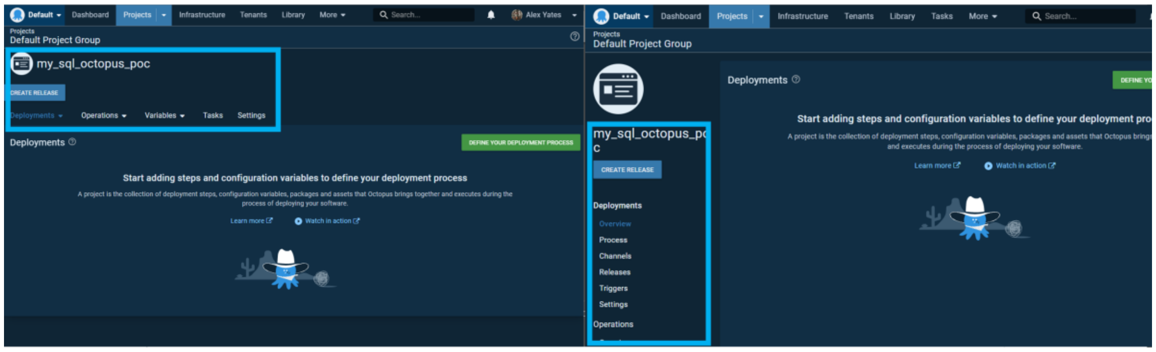
Create a Project in Octopus to hold all of the automation steps and configuration for your PoC. Then, create a set of Project Variables to hold various configuration data.

Steps

1. Open your Octopus Deploy account in a web browser and select the **Projects** tab. Then click **ADD PROJECT**, call it **my_sql_octopus_poc** and click **SAVE**.

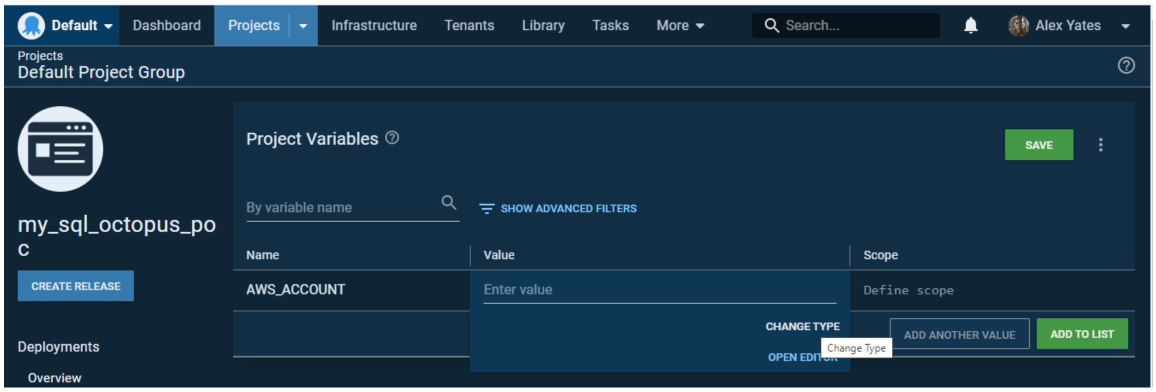


2. From within the my_sql_octopus_poc Project, observe the sub-menu that may appear either on the left or across the top, depending on your screen resolution.



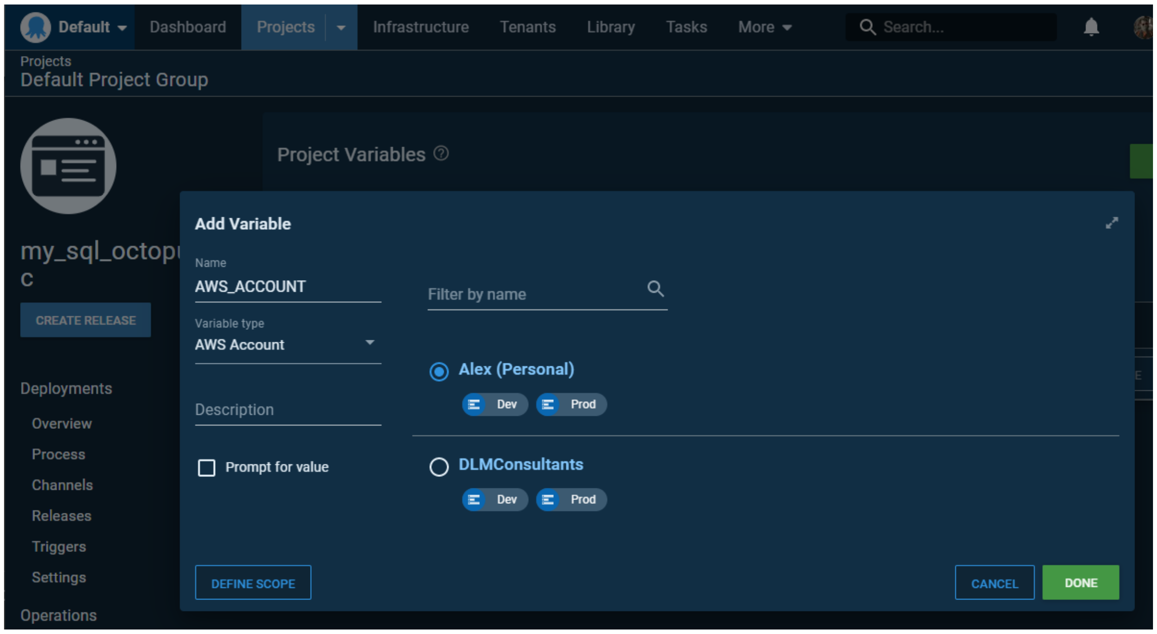
3. From the sub-menu, select **Variables**, and observe an empty table of variables. You need to create a variable to hold a reference to your AWS Account. Under the Name column type **AWS_ACCOUNT**. Click on the corresponding Value field, select **CHANGE TYPE**, and then select

AWS Account.



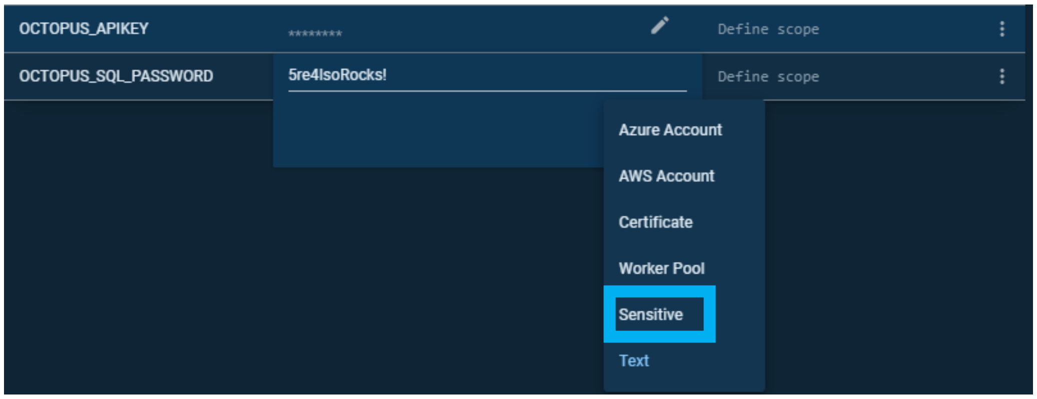
Name	Value	Scope
AWS_ACCOUNT	Enter value	Define scope

4. Select the AWS Account that you set up in Lab 4 and click **DONE**, then click **SAVE**.



Name: AWS_ACCOUNT
Variable type: AWS Account
Description

5. Create additional variables with the following values:
   a. Name: **OCTOPUS_APIKEY**, Value: *Create a new one, like in Lab 1.*
   b. Name: **OCTOPUS_SQL_PASSWORD**, Value: **5re4lsoRocks!**
   c. (Only required if not using eu-west-1) Name: **AWS_REGION**, Value (e.g.): *us-east-2*
6. Click on the values for the **OCTOPUS_APIKEY** and **OCTOPUS_SQL_PASSWORD** variables in turn and select **CHANGE TYPE / Sensitive** to ensure the values are protected.



OCTOPUS_APIKEY	********	Define scope
OCTOPUS_SQL_PASSWORD	5re4lsoRocks!	Define scope

Further reading

Documentation:

- Octopus Deploy Projects (Octopus documentation): https://octopus.com/docs/projects
- Octopus Deploy Variables (Octopus documentation): https://octopus.com/docs/projects/variables

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

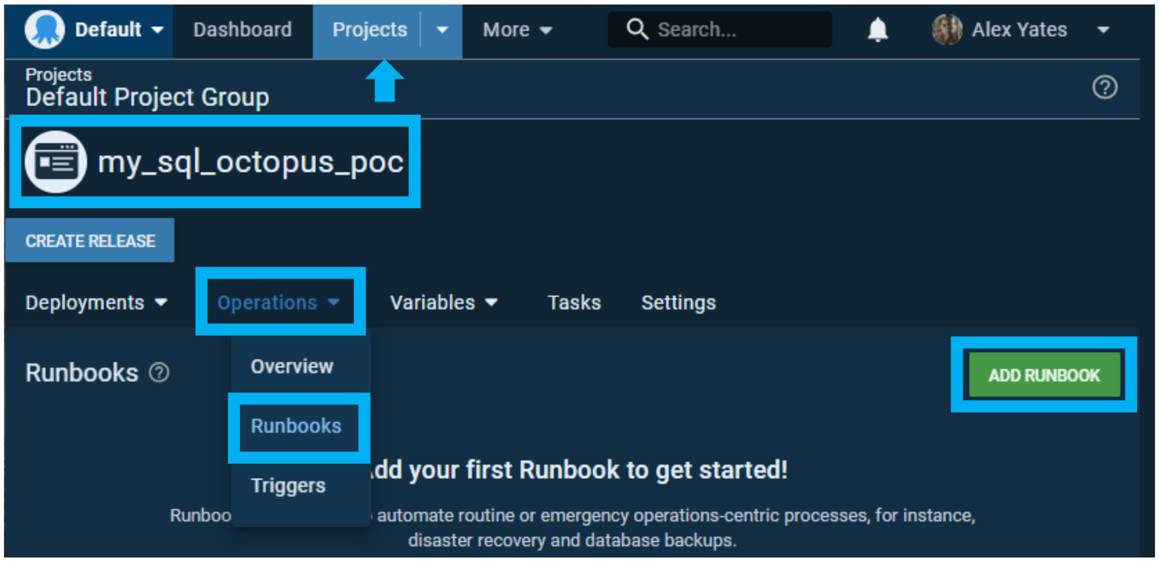
LAB 6: Creating an Octopus Runbook

Objective

Create an Operations Runbook in Octopus Deploy to deploy the infrastructure for your database and a corresponding web application.

Steps

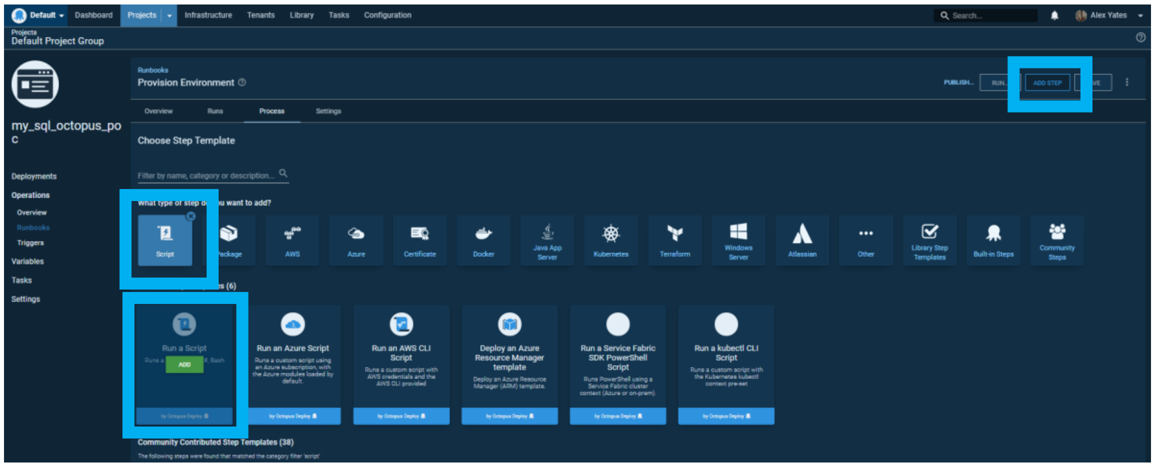
1. Open your Octopus Deploy account in a web browser. Select the Projects tab and the my_sql_octopus_poc Project. From the sub-menu in the Project (which might be across the top or on the left, depending on screen resolution), select Operations / Runbooks, and then **ADD RUNBOOK**.



Add your first Runbook to get started!

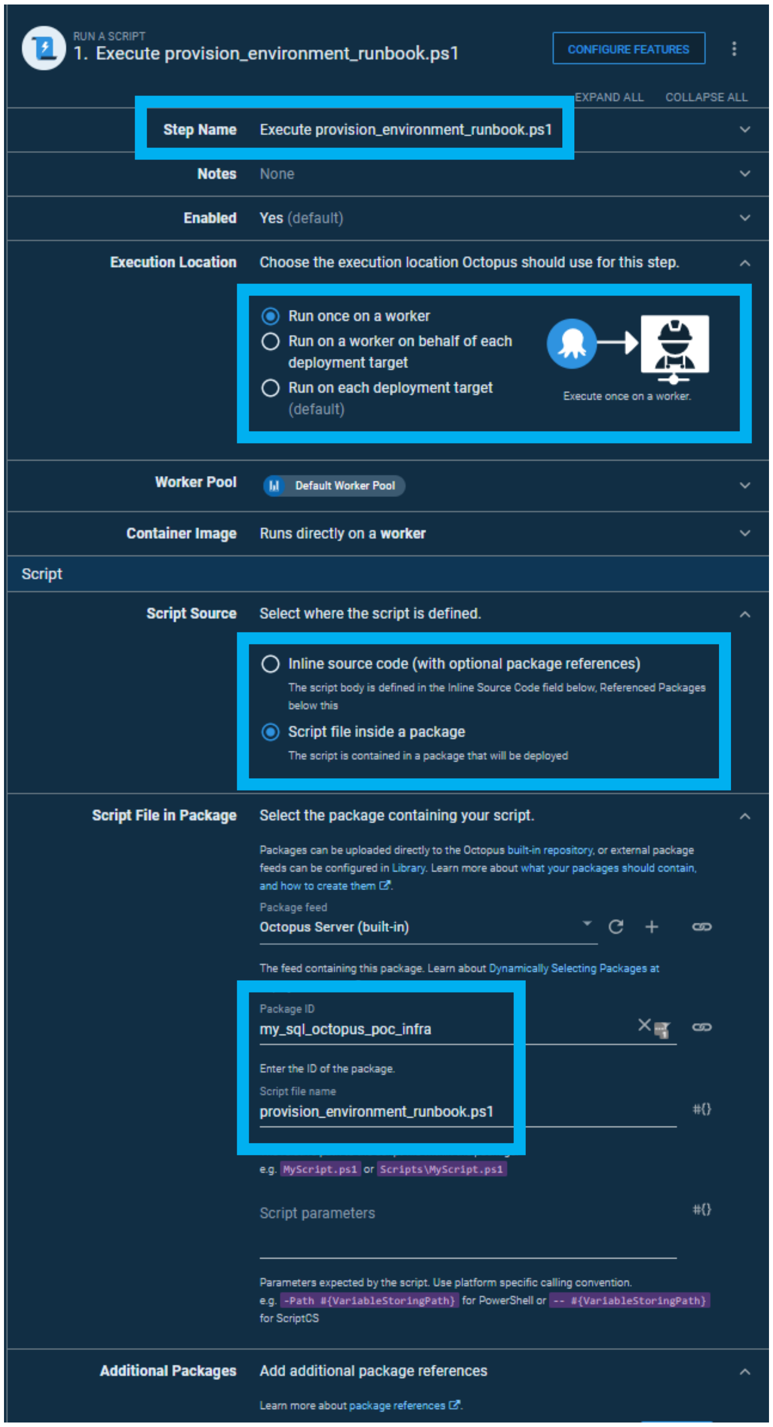
Runbooks automate routine or emergency operations-centric processes, for instance, disaster recovery and database backups.

2. Give your new Runbook the name “Provision Environment” and click **SAVE**.
3. Click the **DEFINE YOUR RUNBOOK PROCESS** button in the top-right corner, and then the **ADD STEP** button. From the various step options, select Script and Run a Script.



4. Provide the following configuration, leaving the defaults settings unless specified below:
   - Step Name: Execute provision_environment_runbook.ps1
   - Execution Location: Run once on a worker

- Script Source: Script file inside a package
- Script File in Package:
    1. Package ID: my_sql_octopus_poc_infra
    2. Script file name: provision_environment_runbook.ps1


RUN A SCRIPT
1. Execute provision_environment_runbook.ps1

Step Name Execute provision_environment_runbook.ps1

Notes None

Enabled Yes (default)

Execution Location Choose the execution location Octopus should use for this step.

- [✓ Run once on a worker]
- Run on a worker on behalf of each deployment target
- Run on each deployment target (default)

Worker Pool Default Worker Pool

Container Image Runs directly on a **worker**

Script

Script Source Select where the script is defined.

- Inline source code (with optional package references)
  The script body is defined in the Inline Source Code field below, Referenced Packages below this
- [✓ Script file inside a package]
  The script is contained in a package that will be deployed

Script File in Package Select the package containing your script.

Packages can be uploaded directly to the Octopus built-in repository, or external package feeds can be configured in Library. Learn more about what your packages should contain, and how to create them.

Package feed
Octopus Server (built-in)

The feed containing this package. Learn about Dynamically Selecting Packages at

Package ID
my_sql_octopus_poc_infra

Enter the ID of the package.

Script file name
provision_environment_runbook.ps1

e.g. `MyScript.ps1` or `Scripts\MyScript.ps1`

Script parameters

Parameters expected by the script. Use platform specific calling convention.
e.g. `-Path #{VariableStoringPath}` for PowerShell or `-- #{VariableStoringPath}` for ScriptCS

Additional Packages Add additional package references

Learn more about package references.

5. Double check your configuration matches the screenshot above and click **SAVE** (top left). (Note, to save space in the screenshot above, each section where the default settings are left

unchanged has been minimized using the arrow on the right. This makes no functional difference.)

Further reading

Theory:

- Operations Runbooks: Putting the Ops in DevOps (Rob Pearson, Octopus blog): https://octopus.com/blog/operations-runbooks
- Octopus Workers (Octopus blog) https://octopus.com/blog/octopus-workers

Documentation:

- Runbooks (Octopus documentation): https://octopus.com/docs/runbooks
- Workers (Octopus documentation): https://octopus.com/docs/infrastructure/workers
- Dynamic Worker Pools (Octopus documentation): https://octopus.com/docs/infrastructure/workers/dynamic-worker-pools

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

LAB 7: Running a Runbook

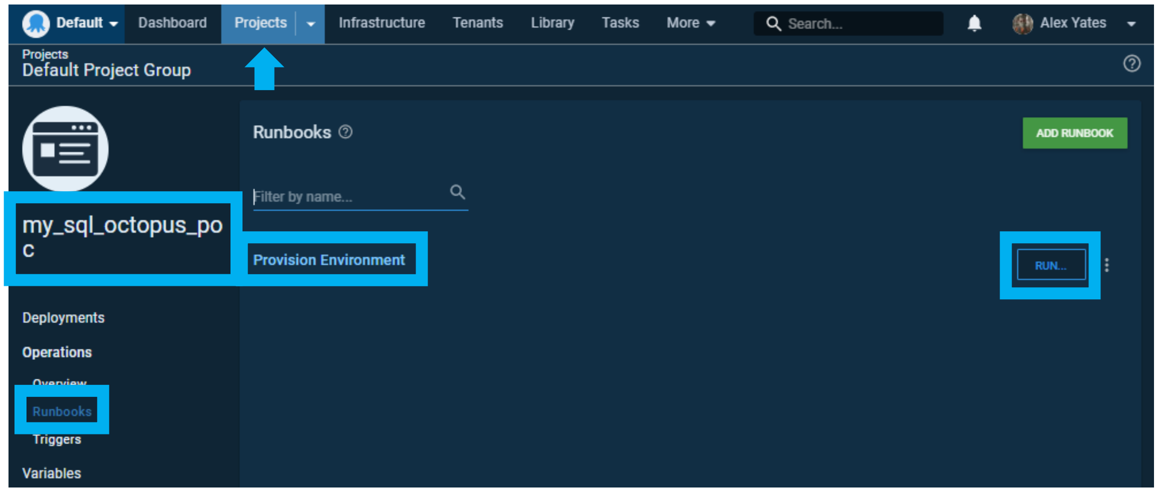
Objective

Run the Runbook created in Lab 6 to build all the infrastructure we need to deploy a database and some associated web applications.

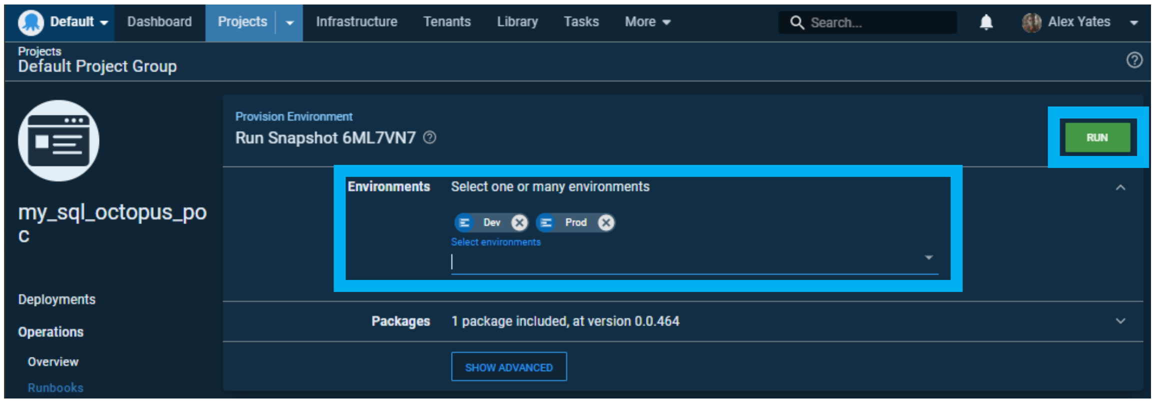
Once deployed successfully to both Dev and Prod, create a "Kill Environment" Runbook to gracefully terminate an environment, and validate that it works.

Steps: Run the Provision Environment Runbook

1. Open your Octopus Deploy account in a web browser. Select the **Projects** tab and the **my_sql_octopus_poc** Project. From the sub-menu, select **Operations / Runbooks**. Observe your **Provision Environment** runbook and click the **RUN** button next to it.

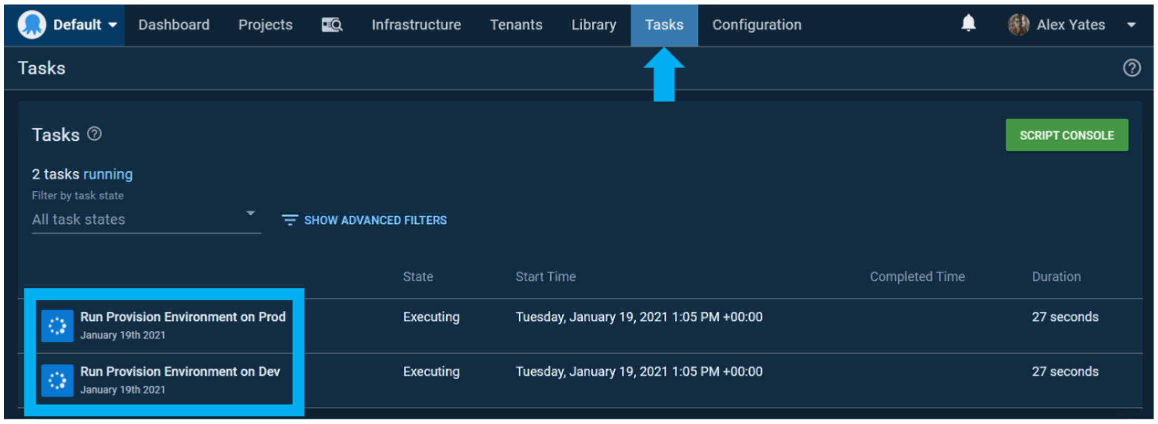


2. Select both the Dev and Prod environments and click **RUN**.



Octopus Deploy
Training produced by DLM Consultants

3. After starting your Runbook, notice that you are taken to the Tasks tab. Observe that your Runbook is running in both Dev and Prod:



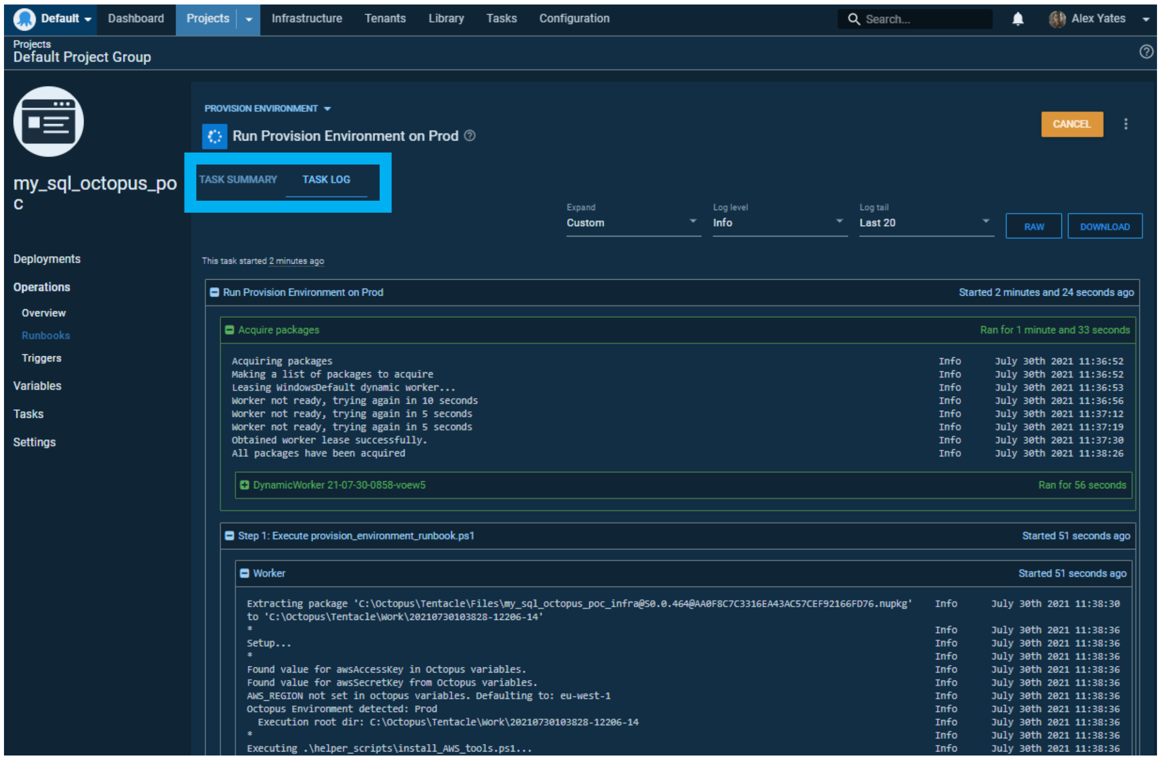
Tasks

2 tasks running

Filter by task state: All task states

	State	Start Time	Completed Time	Duration
Run Provision Environment on Prod January 19th 2021	Executing	Tuesday, January 19, 2021 1:05 PM +00:00		27 seconds
Run Provision Environment on Dev January 19th 2021	Executing	Tuesday, January 19, 2021 1:05 PM +00:00		27 seconds

4. Click on one of the running tasks and observe that you can toggle between **TASK SUMMARY** and **TASK LOG** to watch the live logs:



Run Provision Environment on Prod

Expand: Custom
Log level: Info
Log tail: Last 20

This task started 2 minutes ago

Run Provision Environment on Prod — Started 2 minutes and 24 seconds ago

Acquire packages — Ran for 1 minute and 33 seconds

```
Acquiring packages                                  Info    July 30th 2021 11:36:52
Making a list of packages to acquire                Info    July 30th 2021 11:36:52
Leasing WindowsDefault dynamic worker...            Info    July 30th 2021 11:36:53
Worker not ready, trying again in 10 seconds        Info    July 30th 2021 11:36:56
Worker not ready, trying again in 5 seconds         Info    July 30th 2021 11:37:12
Worker not ready, trying again in 5 seconds         Info    July 30th 2021 11:37:19
Obtained worker lease successfully.                 Info    July 30th 2021 11:37:30
All packages have been acquired                     Info    July 30th 2021 11:38:26
```

DynamicWorker 21-07-30-0858-voew5 — Ran for 56 seconds

Step 1: Execute provision_environment_runbook.ps1 — Started 51 seconds ago

Worker — Started 51 seconds ago

```
Extracting package 'C:\Octopus\Tentacle\Files\my_sql_octopus_poc_infra@S0.0.464@AA0F8C7C3316EA43AC57CEF92166FD76.nupkg'   Info    July 30th 2021 11:38:30
to 'C:\Octopus\Tentacle\Work\20210730103828-12206-14'
*                                                                           Info    July 30th 2021 11:38:36
Setup...                                                                    Info    July 30th 2021 11:38:36
*                                                                           Info    July 30th 2021 11:38:36
Found value for awsAccessKey in Octopus variables.                          Info    July 30th 2021 11:38:36
Found value for awsSecretKey from Octopus variables.                        Info    July 30th 2021 11:38:36
AWS_REGION not set in octopus variables. Defaulting to: eu-west-1           Info    July 30th 2021 11:38:36
Octopus Environment detected: Prod                                          Info    July 30th 2021 11:38:36
  Execution root dir: C:\Octopus\Tentacle\Work\20210730103828-12206-14      Info    July 30th 2021 11:38:36
*                                                                           Info    July 30th 2021 11:38:36
Executing .\helper_scripts\install_AWS_tools.ps1...                         Info    July 30th 2021 11:38:36
```

Octopus Deploy
Training produced by DLM Consultants

Tip:

The Provision Environment Runbook should take about 15 minutes the first time you run it. Subsequent runs may be faster.

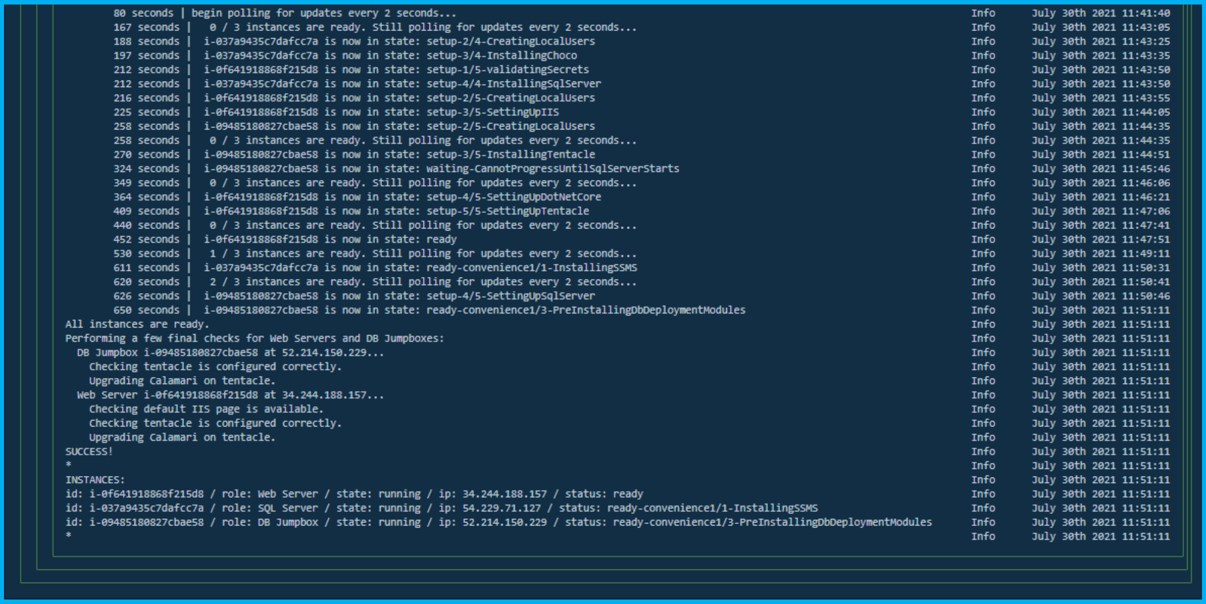
First Octopus will lease a dynamic worker on which to run your scripts. For the first run this will probably take a couple of minutes. Any subsequent runs within the next hour or so will be faster since Octopus will re-use the existing worker.

Next the provision_environment_runbook.ps1 script will execute "install_AWS_tools.ps1" to install various PowerShell modules. On new workers this will take a couple of minutes, but subsequent runs on the same worker will be much quicker since the modules will already be installed.

Then the runbook runs a few relatively quick scripts to set up some AWS networking and security bits, including an IAM Role, Security Group and Key Pair.

Finally, the runbook executes the "build_servers.ps1" script which builds your instances. It only takes a couple of minutes to start all the instances, but then it waits in a while loop for roughly 10 minutes to verify that all the instances boot up and that all the startup scripts run successfully. (For example, we need to wait until SQL Server has been installed and configured.)

You can monitor progress in the logs:



```
        80 seconds | begin polling for updates every 2 seconds...                                              Info    July 30th 2021 11:41:40
       167 seconds |   0 / 3 instances are ready. Still polling for updates every 2 seconds...                 Info    July 30th 2021 11:43:05
       188 seconds |  i-037a9435c7dafcc7a is now in state: setup-2/4-CreatingLocalUsers                        Info    July 30th 2021 11:43:25
       197 seconds |  i-037a9435c7dafcc7a is now in state: setup-3/4-InstallingChoco                           Info    July 30th 2021 11:43:35
       212 seconds |  i-0f641918868f215d8 is now in state: setup-1/5-validatingSecrets                         Info    July 30th 2021 11:43:50
       212 seconds |  i-037a9435c7dafcc7a is now in state: setup-4/4-InstallingSqlServer                      Info    July 30th 2021 11:43:50
       216 seconds |  i-0f641918868f215d8 is now in state: setup-2/5-CreatingLocalUsers                        Info    July 30th 2021 11:43:55
       225 seconds |  i-0f641918868f215d8 is now in state: setup-3/5-SettingUpIIS                              Info    July 30th 2021 11:44:05
       258 seconds |  i-09485180827cbae58 is now in state: setup-2/5-CreatingLocalUsers                        Info    July 30th 2021 11:44:35
       258 seconds |   0 / 3 instances are ready. Still polling for updates every 2 seconds...                 Info    July 30th 2021 11:44:35
       270 seconds |  i-09485180827cbae58 is now in state: setup-3/5-InstallingTentacle                        Info    July 30th 2021 11:44:51
       324 seconds |  i-09485180827cbae58 is now in state: waiting-CannotProgressUntilSqlServerStarts          Info    July 30th 2021 11:45:46
       349 seconds |   0 / 3 instances are ready. Still polling for updates every 2 seconds...                 Info    July 30th 2021 11:46:06
       364 seconds |  i-0f641918868f215d8 is now in state: setup-4/5-SettingUpDotNetCore                       Info    July 30th 2021 11:46:21
       409 seconds |  i-0f641918868f215d8 is now in state: setup-5/5-SettingUpTentacle                         Info    July 30th 2021 11:47:06
       440 seconds |   0 / 3 instances are ready. Still polling for updates every 2 seconds...                 Info    July 30th 2021 11:47:41
       452 seconds |  i-0f641918868f215d8 is now in state: ready                                               Info    July 30th 2021 11:47:51
       530 seconds |   1 / 3 instances are ready. Still polling for updates every 2 seconds...                 Info    July 30th 2021 11:49:11
       611 seconds |  i-037a9435c7dafcc7a is now in state: ready-convenience1/1-InstallingSSMS                 Info    July 30th 2021 11:50:31
       620 seconds |   2 / 3 instances are ready. Still polling for updates every 2 seconds...                 Info    July 30th 2021 11:50:41
       626 seconds |  i-09485180827cbae58 is now in state: setup-4/5-SettingUpSqlServer                        Info    July 30th 2021 11:50:46
       650 seconds |  i-09485180827cbae58 is now in state: ready-convenience1/3-PreInstallingDbDeploymentModules Info  July 30th 2021 11:51:11
All instances are ready.                                                                                       Info    July 30th 2021 11:51:11
Performing a few final checks for Web Servers and DB Jumpboxes:                                                Info    July 30th 2021 11:51:11
  DB Jumpbox i-09485180827cbae58 at 52.214.150.229...                                                          Info    July 30th 2021 11:51:11
    Checking tentacle is configured correctly.                                                                 Info    July 30th 2021 11:51:11
    Upgrading Calamari on tentacle.                                                                            Info    July 30th 2021 11:51:11
  Web Server i-0f641918868f215d8 at 34.244.188.157...                                                          Info    July 30th 2021 11:51:11
    Checking default IIS page is available.                                                                    Info    July 30th 2021 11:51:11
    Checking tentacle is configured correctly.                                                                 Info    July 30th 2021 11:51:11
    Upgrading Calamari on tentacle.                                                                            Info    July 30th 2021 11:51:11
SUCCESS!                                                                                                       Info    July 30th 2021 11:51:11
*                                                                                                              Info    July 30th 2021 11:51:11
INSTANCES:                                                                                                     Info    July 30th 2021 11:51:11
id: i-0f641918868f215d8 / role: Web Server / state: running / ip: 34.244.188.157 / status: ready                Info    July 30th 2021 11:51:11
id: i-037a9435c7dafcc7a / role: SQL Server / state: running / ip: 54.229.71.127 / status: ready-convenience1/1-InstallingSSMS    Info    July 30th 2021 11:51:11
id: i-09485180827cbae58 / role: DB Jumpbox / state: running / ip: 52.214.150.229 / status: ready-convenience1/3-PreInstallingDbDeploymentModules    Info    July 30th 2021 11:51:11
*                                                                                                              Info    July 30th 2021 11:51:11
```

5. Watch the build logs and verify that the Runbook Run completes successfully. If you experience any errors, read the logs to see if they explain why it failed. A few common reasons for failure, including some tips to fix the issues are included below. If you cannot work out why the runbook failed, try running it a second time to see if it was a one-off:

Common reasons for failure:

Failure to authenticate with AWS:

- Try again. Sometimes, especially if using a new AWS account, the first run fails because your account is still being set up. Often, simply re-running the runbook will resolve the issue.
- Verify your Octopus variables are configured correctly. (See Lab 5.)
- Verify your AWS Account is configured correctly in Octopus Deploy. (See Lab 4. Go to **Infrastructure / Accounts**, select your AWS Account and click **SAVE AND TEST**.)
- Verify your octopus user in AWS is configured correctly. (Log into AWS and head to **IAM / Users**. Click on the octopus user look under **Security credentials**. From there you can create a new access / secret key pair if necessary. Then you can update your AWS Account in Octopus with the new keys.)
- Verify your octopus user in AWS has AdministratorAccess. (See Lab 3).

Missing NuGet Packages:

- Ensure your Runbook Process is configured properly (see Lab 6), and that it references the correct NuGet package.
- If the NuGet package does not exist in the built-in Octopus package repository (see **Library / Packages** in the Octopus UI), re-run your GitHub Actions to create the necessary NuGet Packages and upload them to Octopus Deploy. (See Lab 2)

Incorrect path to provision_environment_runbook.ps1:

- Ensure your Runbook Process is configured properly (see Lab 6), and that you have typed the name of the script correctly.
- Check that the NuGet package contains the correct scripts.
    - Download the NuGet package from the Octopus UI (see **Library / Packages**)
    - Rename the file extension to .zip and extract the package. Alternatively, if you have NuGet Package Explorer installed, use that to open the package.
    - Verify that the provision_environment_runbook.ps1 script exists in the root of the NuGet package.

Instance setup fails:

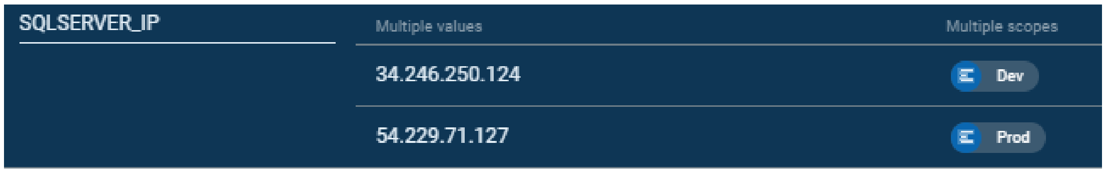
- All the instances are created successfully, but one or more either post a startup status with an error message, or they take significantly longer than expected to start (see logs for estimated timings). If the logs in Octopus give you enough information to understand and fix the error, great. If not, you may wish to remote desktop/RDP on to the offending instances and review the more detailed logs available on the instance itself, located at C:/startup.
    - Assuming the instances successfully completed the "CreatingLocalUsers" step (step 2) you should be able to RDP onto the instance IP address using the credentials user: "student" / password: "D3vOpsRocks!". (This is an admin user).
    - If the CreatingLocalUsers setup script either was not run or it failed, refer to Appendix 6: Getting the Admin Password for an AWS instance.

Common reasons for failure (continued)…

Insufficient Octopus Targets (New Octopus Tentacles will start failing to register with Octopus Deploy)

- The Web Server and DB Jumpbox instances will install an Octopus Tentacle and attempt to register it with the Octopus Server. Each tentacle counts as a target with respect to your Octopus licence. (Free instances are entitled to a maximum of 10 targets). If you have used up all the targets permitted under your licence, the Tentacle registration will fail.
    - You can free up space by navigating to the “Infrastructure” tab in Octopus Deploy, selecting Deployment Targets, and either Disabling or Deleting any targets that are not required.
    - Note: Terminating instances in AWS will NOT automatically delete them as a target in Octopus Deploy. For example, if you have run the Provision Environment runbook several times, and you have not been deleting your old targets in Octopus Deploy, you will eventually run out of targets, regardless of whether the AWS instances are still running.

6. Once your Runbook reaches the while loop and starts waiting for all the instances to start up, open up your Octopus Project Variables in another tab. Notice that the runbook has added a new variable to hold the SQL Server IP address, and that there are unique values scoped to each environment. This has been automated using the Octopus Deploy API, as documented here.



SQLSERVER_IP	Multiple values	Multiple scopes
	34.246.250.124	Dev
	54.229.71.127	Prod

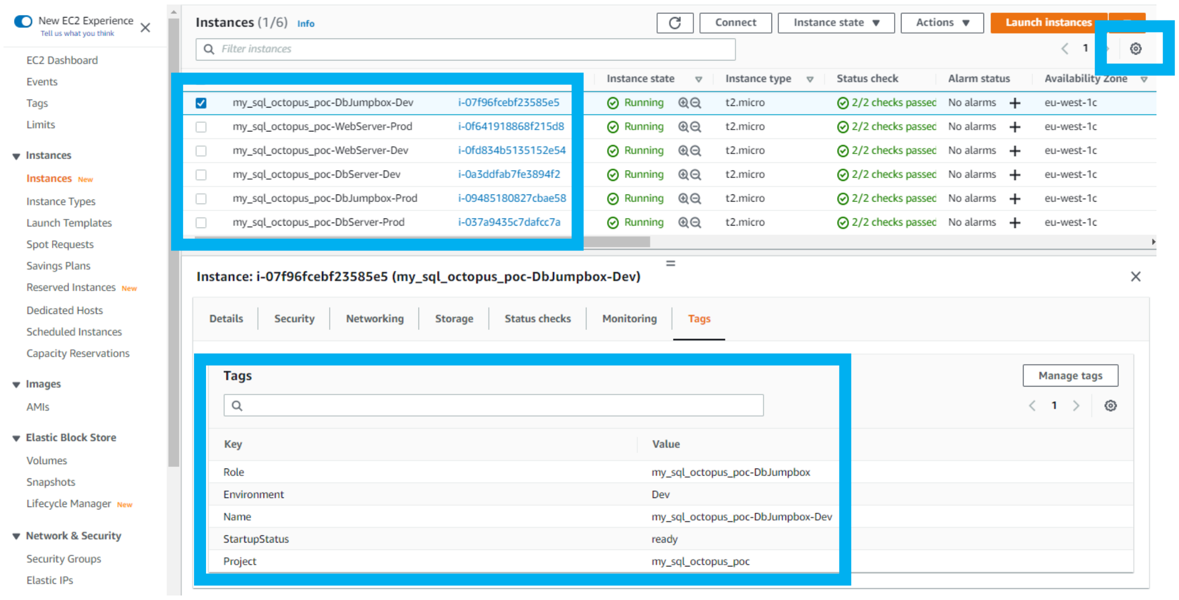
Tip:
To keep the AWS configuration as simple as possible, we are accepting the random IP addresses that AWS assigns to us and using them for our SQL Server connection strings.

For real world applications this is considered a bad practice. It would be better to use static IP addresses and DNS records to avoid hardcoding IP addresses into code/config.

However, that would require a more complicated AWS set-up. This is an Octopus Deploy class that is explicit about providing the minimum AWS content to optimize for attendees with zero prior knowledge to build an end-to-end deployment pipeline as quickly and easily as possible.

7. Log into AWS, head to **EC2 / Instances**, and observe that various instances have been started. If you select an instance and look under Tags, you should notice that your instances have been

tagged with various useful information.



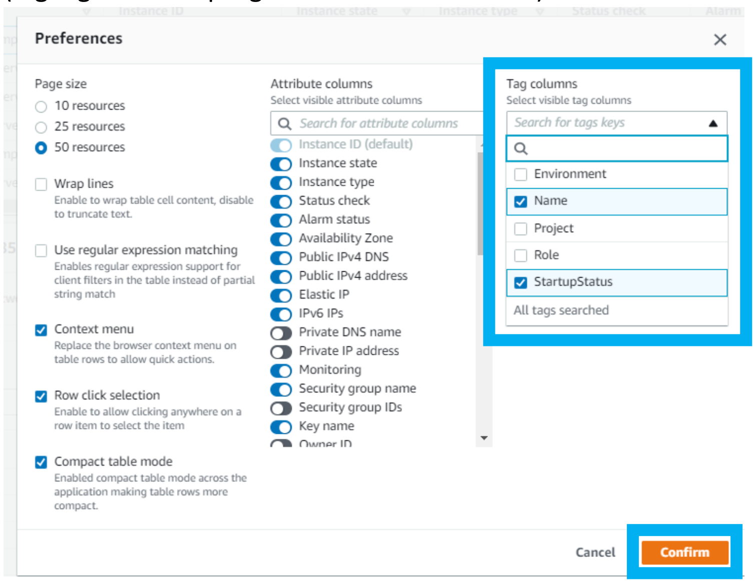
		Instance state	Instance type	Status check	Alarm status	Availability Zone
✓ my_sql_octopus_poc-DbJumpbox-Dev	i-07f96fcebf23585e5	Running	t2.micro	2/2 checks passec	No alarms	eu-west-1c
my_sql_octopus_poc-WebServer-Prod	i-0f641918868f215d8	Running	t2.micro	2/2 checks passec	No alarms	eu-west-1c
my_sql_octopus_poc-WebServer-Dev	i-0fd834b5135152e54	Running	t2.micro	2/2 checks passec	No alarms	eu-west-1c
my_sql_octopus_poc-DbServer-Dev	i-0a3ddfab7fe3894f2	Running	t2.micro	2/2 checks passec	No alarms	eu-west-1c
my_sql_octopus_poc-DbJumpbox-Prod	i-09485180827cbae58	Running	t2.micro	2/2 checks passec	No alarms	eu-west-1c
my_sql_octopus_poc-DbServer-Prod	i-037a9435c7dafcc7a	Running	t2.micro	2/2 checks passec	No alarms	eu-west-1c

Instance: i-07f96fcebf23585e5 (my_sql_octopus_poc-DbJumpbox-Dev)

Tags

Key	Value
Role	my_sql_octopus_poc-DbJumpbox
Environment	Dev
Name	my_sql_octopus_poc-DbJumpbox-Dev
StartupStatus	ready
Project	my_sql_octopus_poc

8. Note that by default the "Name" tag is the first column in the instances table. However, if you would like to add additional columns, such as "StartupStatus", you can do so using the cog icon (highlighted in top-right of screenshot above).



Preferences

Page size
- 10 resources
- 25 resources
- [✓ 50 resources]

- [ ] Wrap lines — Enable to wrap table cell content, disable to truncate text.
- [ ] Use regular expression matching — Enables regular expression support for client filters in the table instead of partial string match
- [x] Context menu — Replace the browser context menu on table rows to allow quick actions.
- [x] Row click selection — Enable to allow clicking anywhere on a row item to select the item
- [x] Compact table mode — Enabled compact table mode across the application making table rows more compact.

Attribute columns
Select visible attribute columns
Search for attribute columns
- Instance ID (default): on
- Instance state: on
- Instance type: on
- Status check: on
- Alarm status: on
- Availability Zone: on
- Public IPv4 DNS: on
- Public IPv4 address: on
- Elastic IP: on
- IPv6 IPs: on
- Private DNS name: off
- Private IP address: off
- Monitoring: on
- Security group name: on
- Security group IDs: off
- Key name: on
- Owner ID: off

Tag columns
Select visible tag columns
Search for tags keys
- [ ] Environment
- [x] Name
- [ ] Project
- [ ] Role
- [x] StartupStatus

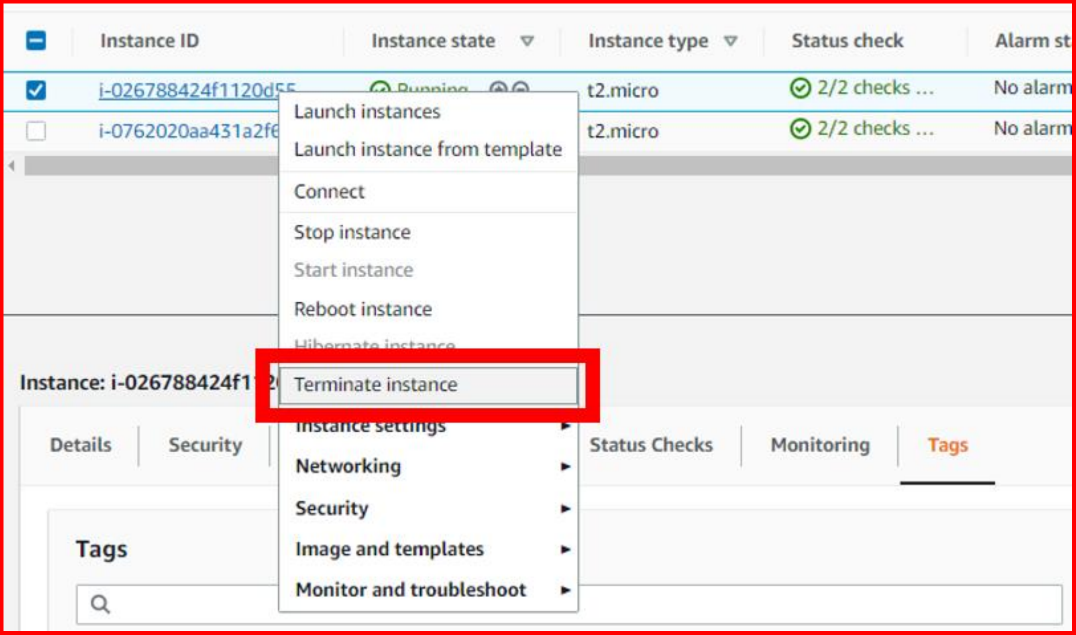
All tags searched

Cancel Confirm

Warning:
You are now paying a little under $0.02 USD/hr for each of these instances. Not a lot over the course of a few hours, but if you forget about them your annual bill will be around $1,000 USD for all six instances.

Do not forget to shut down or terminate your instances when you are finished!

*You can terminate them either by running the Kill Environment runbook (see below) or by right-clicking them in the AWS console and clicking **Terminate instance**.*



For more information about AWS pricing, see Appendix 1: Licensing considerations.

Steps: Create and run a Kill Environment Runbook

The Kill Environment Runbook does the following:

- Deletes all EC2 instances associated with your Octopus project in a given environment
- Uses the Octopus API to deregister each of the EC2 instance(s) as Deployment Targets

We have not yet covered Tentacles and Deployment Targets. For now, just be aware that Octopus Deploy is licenced based on the number of targets. (Free licences are limited to 10 targets.) Even if you terminate your AWS instances, if you do not also clean up old targets in Octopus Deploy, you'll run out of permitted targets and your Provision Environment Runbook will fail.

9. Repeat Lab 6: Creating an Octopus Runbook. However, this time call the Runbook "Kill Environment" and reference the script "kill_environment_runbook.ps1" instead.

10. Repeat the first part of this Lab, but run your "Kill Environment" runbook only in the Prod environment. (Do not kill the "Dev" instance. We will need it to complete Lab 8.)
11. Once the runbook has completed successfully, head to the **Infrastructure** tab in Octopus Deploy and verify that all the Prod targets have gone.
12. Log into AWS, head to **EC2 / Instances**, and verify that your Prod instances have been terminated, but that your "Dev" instances are still running. If this is not the status of your AWS instances, run the appropriate Octopus Runbooks ("Provision Environment" or "Kill Environment") in the appropriate environments ("Dev" or "Prod") to correct it.

Further reading

Code:

- Appendix 3: The infrastructure scripts explained (This document):
  Appendix 3: The infrastructure scripts explained

Theory:

- How do I associate a static public IP address with my EC2 Windows or Linux instance? (Amazon Knowledge Center)
  https://aws.amazon.com/premiumsupport/knowledge-center/ec2-associate-static-public-ip/
- Elastic Load Balancing (Amazon documentation)
  https://aws.amazon.com/elasticloadbalancing/

Pricing:

- Amazon EC2 Pricing (Amazon):
  https://aws.amazon.com/ec2/pricing/
- Appendix 1: Licensing considerations (This document):
  Appendix 1: Licensing considerations

Additional material:

Appendix 6: General DevOps reading materials (This document):
Appendix 6: General DevOps reading materials

LAB 8: Defining a SQL Server Deployment Process

Objective

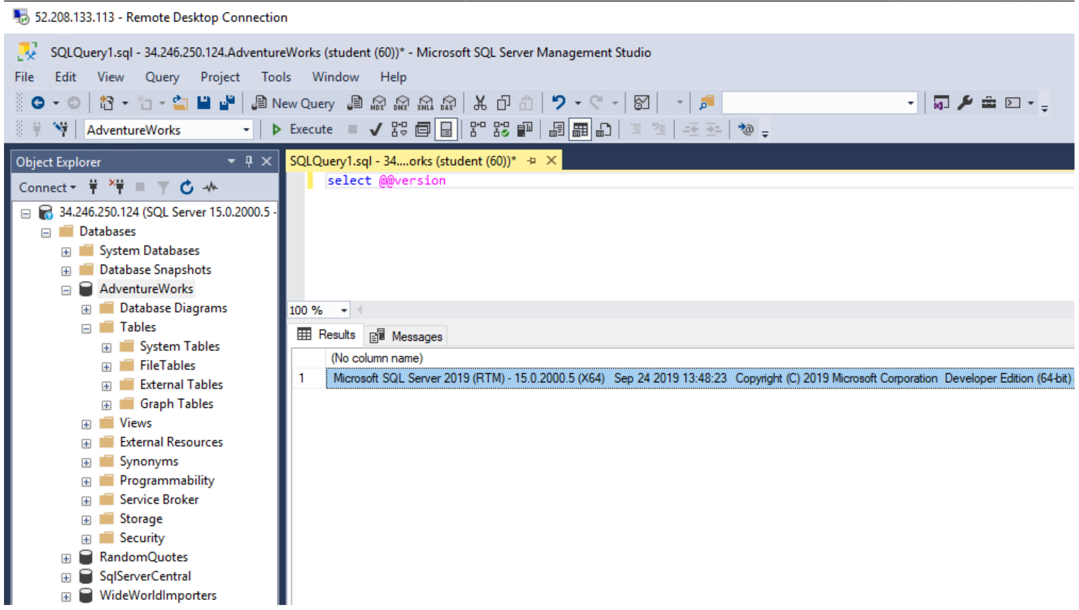
Configure a Deployment Process that will deploy database schema updates to the SQL Server instance(s) you started in Lab 7.

Initial steps (Optional):

Before configuring an Octopus Deployment Project, let's take a look at what we have already.

We'll use your tool of choice to connect to your Dev SQL Server instance. You should be able to connect using the public IPv4 address of your SQL Server AWS instance. You can find the *public* IPv4 address either in the AWS console, your Octopus Deploy Project Variables, or the logs for the Provision Environment Runbook Run. (See Lab 7). You should be able to authenticate using SQL Auth with the username "student" and the password "D3vOpsRocks!".

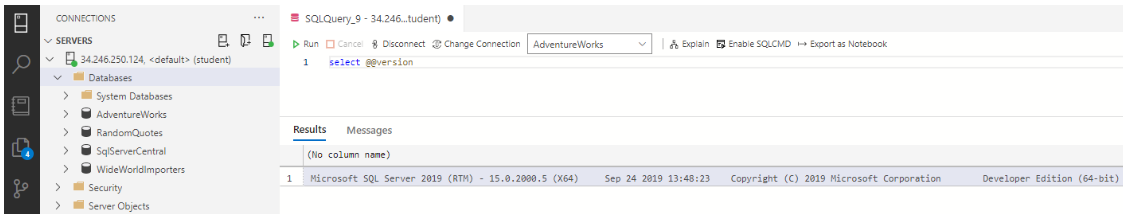
1. If you do not have any SQL Server tooling installed on your local machine, you can RDP to your Jumpbox in AWS and use SQL Server Management Studio (SSMS) from there. (It was pre-installed for you as part of the initial setup). You should be able to RDP to the jumpbox using the public IPv4 address for the jumpbox in either the AWS Console or the Provision Environment Runbook Run logs and you should be able to authenticate using the "student"/"D3vOpsRocks!" credentials:



```
select @@version
```

```
(No column name)
1  Microsoft SQL Server 2019 (RTM) - 15.0.2000.5 (X64)  Sep 24 2019 13:48:23  Copyright (C) 2019 Microsoft Corporation  Developer Edition (64-bit)
```

2. Alternatively, if you have your preferred SQL Server tooling installed on your local machine, you may prefer to use that instead. For example, using Azure Data Studio from my local machine:



```
select @@version
```

```
(No column name)
1  Microsoft SQL Server 2019 (RTM) - 15.0.2000.5 (X64)   Sep 24 2019 13:48:23   Copyright (C) 2019 Microsoft Corporation   Developer Edition (64-bit)
```

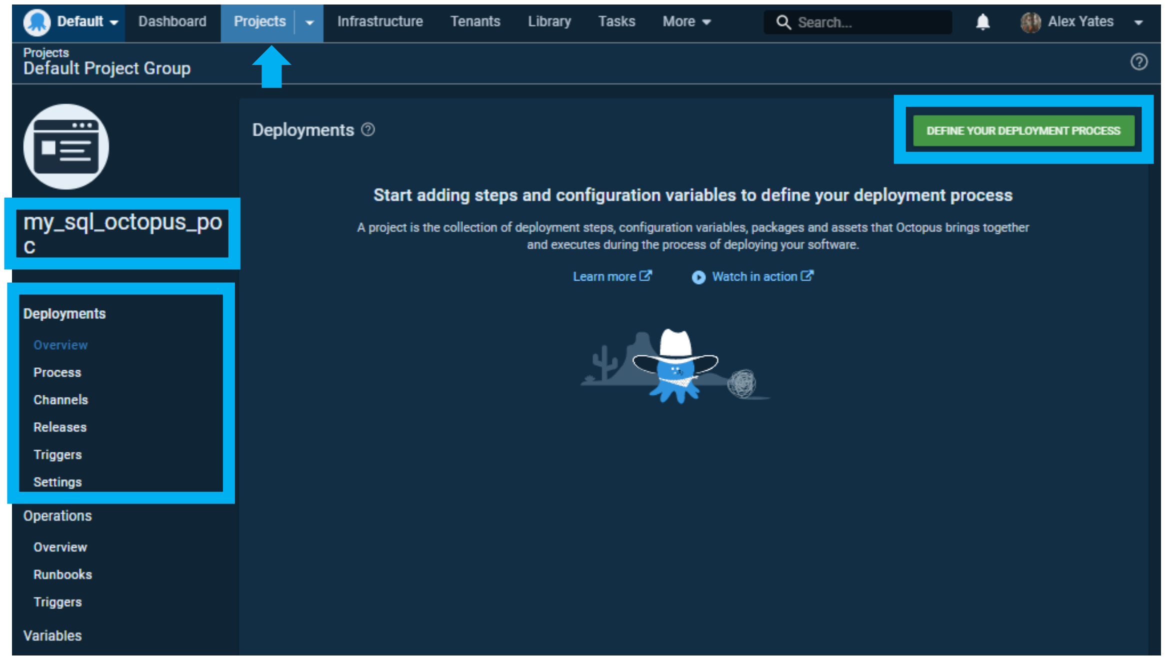
3. However you choose to connect, notice that 4 databases have been created for us by the Provision Environment Runbook: AdventureWorks, RandomQuotes, SqlServerCentral and WideWorldImporters. Each database is currently empty. None of them contain any data, tables, views or any other objects. What follows in this Lab are 4 different ways to set up a database deployment with Octopus Deploy, using 4 different database tooling options: SSDT, Redgate, DbUp and Entity Framework. (Other tools are available.)

Tip:
For a real-world project, our implementation would be a bit of a security nightmare. The decisions to open SQL Server up to the public internet and to go for SQL Auth over Windows Auth were made with simplicity in mind. This class is deliberate about aiming for the shortest and simplest path to a working SQL Server deployment pipeline PoC, without getting sidetracked.

Discussing and implementing production grade security is outside the scope of this class. This class takes the stance that it's better to be deliberate and obvious about the shortcuts we are taking, rather than doing a rushed job of covering such an important topic – especially for relative beginners. Database security is a pretty daunting topic!

LAB 8.1: SSDT projects

1. Open your **my_sql_octopus_poc** project in Octopus Deploy and select **Deployments**. Then click **DEFINE YOUR DEPLOYMENT PROCESS**.



Start adding steps and configuration variables to define your deployment process

A project is the collection of deployment steps, configuration variables, packages and assets that Octopus brings together and executes during the process of deploying your software.

2. Click **ADD STEP**. This step will run a script to deploy the dacpac in your AdventureWorks NuGet package against a target database:
   a. Step Template: **Community Steps / SQL Server / Deploy a DACPAC from Package Parameter**
   b. Step Name: Deploy AdventureWorks dacpac
   c. On Targets in Roles: my_sql_octopus_poc-DbJumpbox

   d. DACPACPackageName: AdventureWorks.dacpac
   e. Target Servername: #{SQLSERVER_IP}
   f. Target Database: AdventureWorks
   g. Username: octopus
   h. Password: #{OCTOPUS_SQL_PASSWORD}
   i. DACPAC Package, Package ID: AdventureWorks
3. Click **SAVE**.

LAB 8.2: Redgate SQL Source Control projects

- To do

LAB 8.3: DbUp Projects

- To do

LAB 8.4: Entity Framework Projects

- To do: RandomQuotes website with DB backend

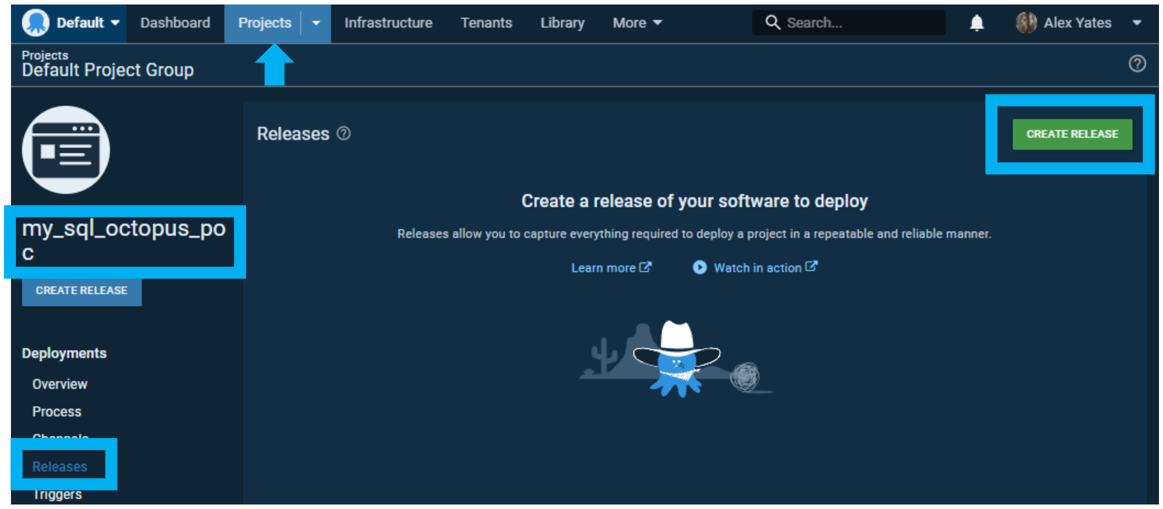
LAB 9: Deploying your database(s)

Objective

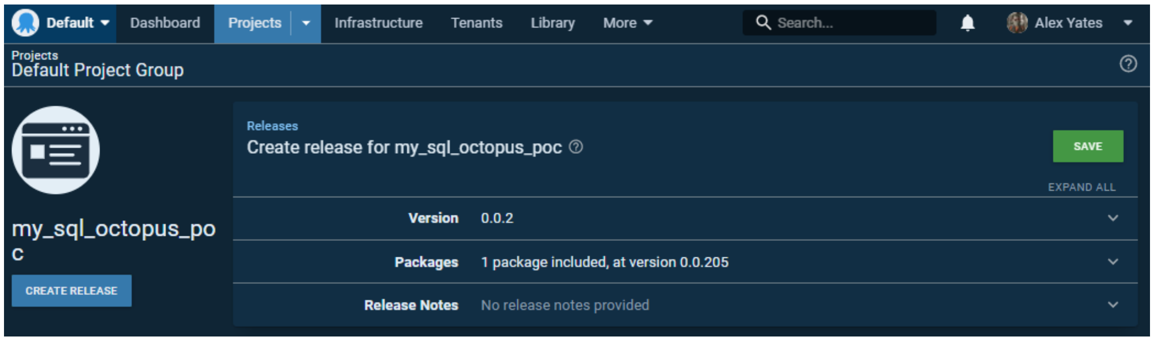
Deploy your databases.

Steps

1. Open your Octopus Deploy account in a web browser. Select the **Projects** tab and the **my_sql_octops_poc** Project. From the sub-menu, select **Releases** and **CREATE RELEASE**.

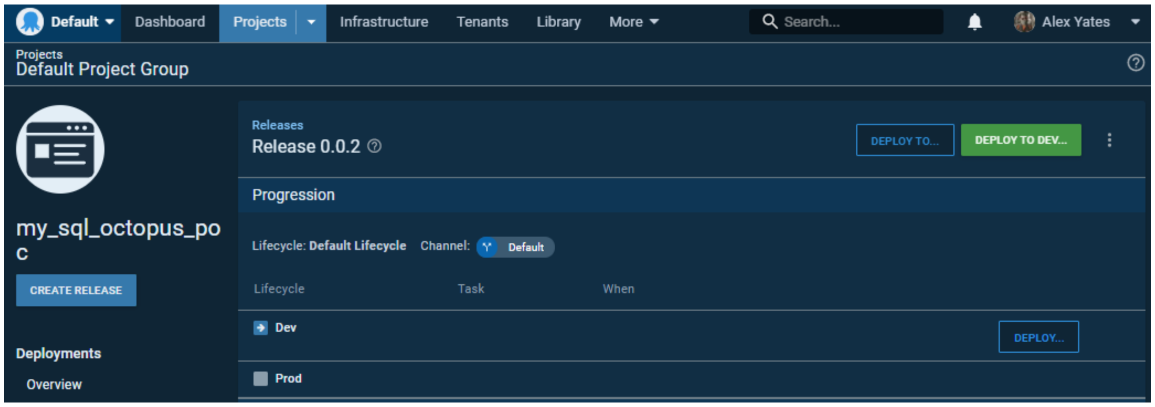


2. Accept all the default values and click **SAVE**.



Version 0.0.2
Packages 1 package included, at version 0.0.205
Release Notes No release notes provided

3. *(Optional, but highly recommended.)* Put on some headphones and open this link in another tab.
4. Click **DEPLOY TO DEV...**, and then **DEPLOY**.



5. Pour yourself a cup of [tea] [coffee] [soda] [beer] [whisky] (delete as appropriate). When the deployment finishes, connect to your SQL Server instance again and take a look at the tables for the databases you just deployed.
6. Optionally, re-run your Provision Environment Runbook to build a set of prod infrastructure and deploy your release there. (Note: If you do not currently have a production environment, you won't yet be able to deploy to it.

Congratulations! You've successfully pushed it.

Tip:
*By default your deployment project will use the "Default Lifecycle". This will mean that releases must be deployed to Dev, before they can be deployed to Prod. If you would like to specify a different Lifecycle (for example, to allow you to deploy releases to any environment, in any order), you can create a different Lifecycle and specify that under **Deployments / Process**.*

More info: https://octopus.com/docs/releases/lifecycles

Further reading

Documentation:

- Community step templates (Octopus documentation): https://octopus.com/docs/releases
- Lifecycless (Octopus documentation): https://octopus.com/docs/releases/lifecycles

Shutting down your PoC

- Appendix 2: Shutting down your PoC (This document): Appendix 2: Shutting down your PoC

Pricing:

- Appendix 1: Licencing considerations (This document): Appendix 1: Licencing considerations

Additional material:

- Appendix 6: General DevOps reading materials (This document): Appendix 6: General DevOps reading materials

Appendix 1: Licensing considerations

We've used three services to produce this proof of concept (PoC):

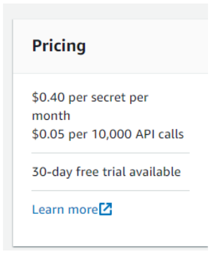
1. GitHub
2. Octopus Deploy
3. Amazon Web Services (AWS)

You may have also created some Redgate deployment processes using Redgate trial licencing.

While the PoC can be completed for under $1/day, there are likely to be licensing considerations if you wish to use these services in a commercial/production environment. You will also want to understand what services you did pay for during your PoC so that you can be sure that you've stopped paying for those services when you are finished.

In this section we'll summarize how the licensing works for each service, starting with AWS, since if you have completed the PoC, you will have necessarily paid for some AWS Services.

AWS – The free tier (and what you have paid for)



Pricing

$0.40 per secret per month
$0.05 per 10,000 API calls

30-day free trial available

Learn more

GitHub

Octopus Deploy

Redgate Trial licence

Appendix 2: Shutting down your PoC

Deleting your PoC manually

Deleting your PoC using an Octopus Runbook

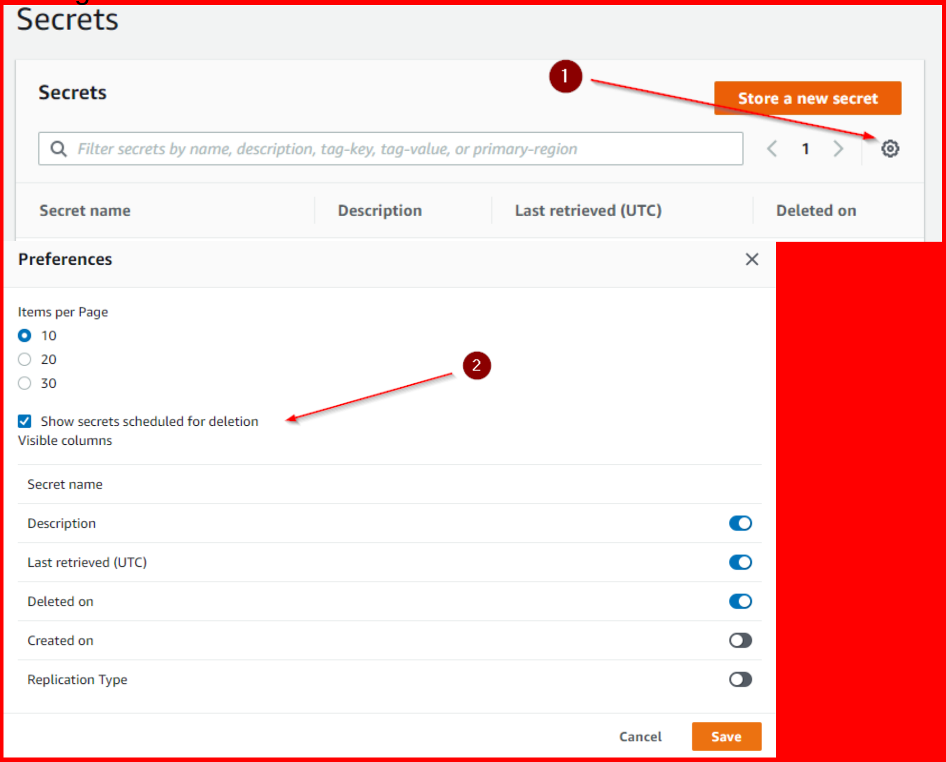
TO DO:

HARD DELETE SCRIPT TO KILL EVERYTHING THAT COSTS MONEY

WARNING!:

This will delete any AWS object with a tag that contains the string "RandomQuotes". If you have any other objects in the AWS region you used for your PoC that have tags containing the string "RandomQuotes", this script will attempt to delete them all.

Similarly, this script will delete any Octopus Tentacles with a role that contains the string "RandomQuotes" – even if the tentacle is also used for other purposes.

- TZ: Canceling secret deletion:
    - 

Items per Page
- ● 10
- ○ 20
- ○ 30

☑ Show secrets scheduled for deletion

Visible columns

Secret name	
Description	on
Last retrieved (UTC)	on
Deleted on	on
Created on	off
Replication Type	off

    -

Appendix 3: The infrastructure scripts explained

Appendix 4: Using a different AWS region

Appendix 5: Using a non-default Octopus Deploy space

Appendix 6: Getting and Admin Password for AWS instances

- Assuming the instances successfully completed the "CreatingLocalUsers" step you should be able to RDP to the instance IP address using the credentials user: student / password: D3vOpsRocks!
- If the CreatingLocalUsers setup script either was not run or it failed, you will not be able to RDP to this instance. Instead, observe that a my_sql_octopus_poc keypair has been created automatically. Then delete it and recreate one with the same name. Create it in the .pem format and make a note of the key. Then delete all your instances and re-run your runbook. This time, you can use your Key Pair to decrypt the Administrator password for your instances and you can use those credentials to open an RDP session.

Appendix 7: General DevOps reading materials

Appendix 8: Additional support options

DLM Consultants

These training materials, including all code, slides, and lab manuals, were created by Alex Yates from DLM Consultants. DLM Consultants are an official Octopus Deploy training and consulting partner who offer various DevOps training, consulting, coaching, and mentoring services.

DLM Consultants believe in supporting the existing people within an organization to adopt DevOps ideas and practices, rather than becoming an outsourced “DevOps team”. We believe in investing in people, and we enjoy supporting any IT folks with general DevOps topics. We also have specialized experience working with senior management and database specialists to modernize cultural and technical practices.

DLM Consultants are based in the UK but offer their services internationally, both in-person (when safe and responsible) and remotely. They have worked with many customers across the globe, including folks from Europe, North America, Africa, Asia and Australasia.

If you would like to contact DLM Consultants for additional support, you can find contact details on their website:

www.dlmconsultants.com

Other training partners

As well as DLM Consultants, Octopus Deploy has a large selection of official training and consulting partners around the world. You can see the full selection of Octopus Deploy partners on the Octopus Deploy website:

www.octopus.com/partners

Contact

For our full list of scheduled Database DevOps classes:

DLM Workshops
www.dlmconsultants.com/dlm-workshops

To contact the presenter for further assistance or training/consulting requirements:

Alex Yates
alex.yates@dlmconsultants.com
www.dlmconsultants.com

Happy deployments!