

An introduction to canary deployments —Through PCF—

A guide for
engineers,
entrepreneurs
and beginners

```
...LIMIT -loadedClasses=26448 -poolType=metaspace -stack
ons="$JAVA_OPTS") && echo JVM Memory Configuration: $CALCULATED_
LATED_MEMORY" && MALLOC_ARENA_MAX=2 SERVER_PORT=$PORT eval exec
java-buildpack/open_jdk_jre/bin/java $JAVA_OPTS -cp $PWD/.
ringframework.boot.loader.WarLauncher
```

	cpu	memory	disk	details
0-29T22:23:08Z	180.9%	661.1M of 2G	195.7M of 1G	
0-29T22:23:43Z	292.3%	805.5M of 2G	195.8M of 1G	
0-29T22:23:08Z	239.1%	699.3M of 2G	195.7M of 1G	

```
a>d:
```

TABLE OF CONTENTS

REAL WORLD IMPACT OF DEPLOYMENTS	3
BACKGROUND ON SOFTWARE DEPLOYMENTS	3
WHAT IS A CANARY DEPLOYMENT?	4
WHAT ARE CONFIGURATION FILE CHANGES AND HOW DO YOU CHANGE THEM?	6
STEPS TO PERFORM CONFIGURATION CHANGES	6
HOW TO PERFORM DATABASE CHANGES.....	6
STEPS TO PERFORM DATABASE CHANGES FOR DEPLOYMENT.....	7
HOW TO PERFORM A CANARY DEPLOYMENT.....	9
STEPS TO DEPLOY:	10
CONGRATULATIONS ON YOUR DEPLOYMENT	14

REAL WORLD IMPACT OF DEPLOYMENTS

Take a moment to remember which websites you visit the most frequently. Perhaps you log into social media sites like Facebook and Twitter every morning or maybe you purchase products off of Amazon every two weeks. Have you noticed how these websites have evolved or changed since you first started using them?

If you recall, Facebook used to have a poke feature to get your friends attention, however they later removed that due to complaints of annoyance. Meanwhile, Amazon only used to have an "Add to Cart" option, but now they've introduced an additional "Buy Now" feature that allows for instant purchases. These changes are usually introduced to the website during periods of low user activity such as during the nighttime for each time zone. This introduction of changes is referred to as a deployment where the website or applications software is replaced on the cloud server with new software. In this guide, we will discuss some background on software deployments and how the deployment process takes place.



BACKGROUND ON SOFTWARE DEPLOYMENTS

Your software application (API) will be hosted on a cloud platform to which you will be deploying the latest version of your software for your consumers to use. Software is regularly deployed in order to present the latest changes made by programmers and software engineers to its customer base.



Sometimes there is one cloud platform to which the API is deployed like Amazon Web Services, Google Cloud Services or Azure, while some companies deploy to various different cloud platforms to suit their various APIs. For multiple cloud platforms, it helps to use cloud-agnostic platform services like Pivotal Cloud Foundry (PCF) that can host APIs belonging to various cloud platforms. If your company uses PCF or if you are starting a business that uses PCF, it helps to understand the process behind these types of software deployments.

The entire deployment process usually involves configuration file changes, database changes, and the actual deployment of software from testing environments to the final production stage. A type of deployment called "Canary Deployments" are used to gradually deploy the APIs to production. The people who deploy software are either software or development operations engineers, so if you plan to be or are currently employed in such a position, it is important to understand deployments. The three points listed below are key pieces of information to begin understanding the deployment process.

- Your consumers will be accessing this latest software, so the deployment should run smoothly.
- Deployment procedures involve testing the software prior to deployment to make sure it works as expected.
- Deployments roll out the latest software in a staged release

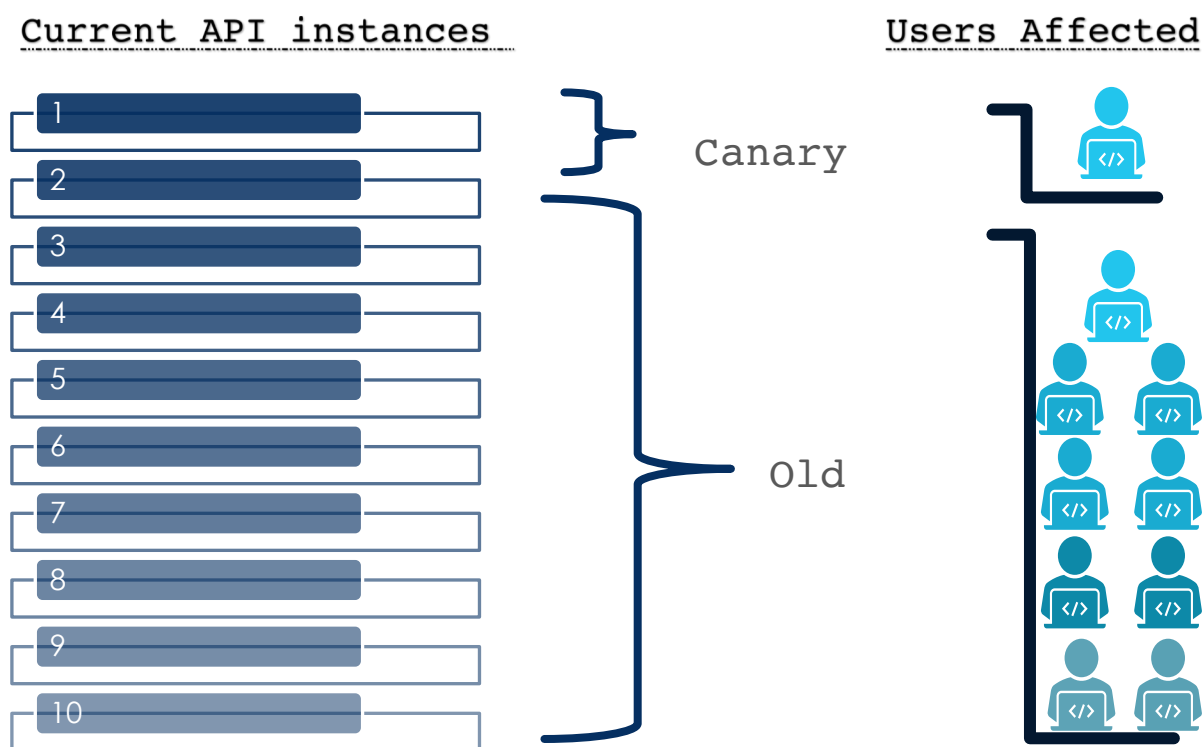
WHAT IS A CANARY DEPLOYMENT?

Software applications (APIs) are divided into several instances on their API cloud hosting platform. Each API instance is rolled out to a small subset of API users. Some of these users can include client company employees, customers, or the general population. When you deploy a new version of an API or roll back to an older version, you must do so very gradually by scaling down the old API and scaling up the new API. This is why we use canary deployments.

A canary deployment is a final production deployment strategy where you deploy only 1 instance of the latest API (called the API-Canary) and decrease the currently running API by one instance (let's say from 10 instances to 9). This way if something goes wrong with your API-canary, it can easily be deleted without affecting many users.

In the example below, you can see that 1 canary API instance on the left affects only 1/10 of the users on the right whereas instances 2-10 affect 9/10 of the users on the right.

Upon initial deployment, most users are still using the old API.



However, if we directly deployed 9 instances of the latest API-Canary and it contained a fault, we would have complaints from thousands of users and would need to spend extra time rolling back to the old API (since deletion and scaling up of instances is a gradual process). On the cloud platform, you will have the new canary API and the old API running simultaneously for a short period of time. The 1 instance of the latest canary API is rolled out to only a small subset of the users of the software and the other larger number of instances of the old API is rolled out to the rest majority of the users.

Note

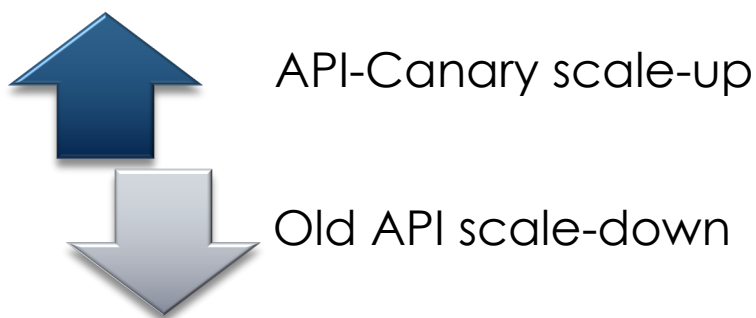
In your deployment files, the latest version of your API should be called your API name with a hyphen and “canary” both appended to the name like so -> “APIName-canary” to distinguish it from the current running API. Refer to the name changes in the manifest file below.

```

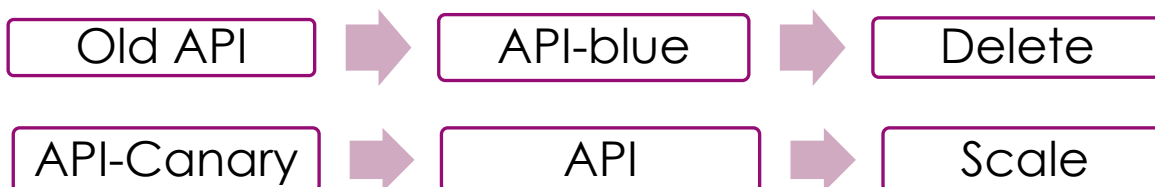
1 applications:
2   - name: DataProcessorAPI-canary
3     memory: 800MB
4     route: http://dataprocessorapi-canary.shared-domain.example.com
5     path: ./dataprocessor.1.0.233.jar
6     services:
7       - pcf-appd-instance
8       - account-service
9     env:
10      APPDYNAMICS_AGENT_APPLICATION_NAME: DataProcessorAPI-canary
11      APPDYNAMICS_AGENT_TIER_NAME: DataProcessorService

```

When your “API-Canary seems to be working after post deployment testing (usually 15 minutes), you can scale up the number of instances of the API-Canary to the same as the old API and scale down the number of instances of the old API to one.



For example, right after deployment we had 1 instance of the new canary API and 9 instances of the old API. After testing out the new canary API, we scale up the canary API to 9 instances and scale down the old API to 1 instance. Then, after giving a minute for the new API scaling adjustment, we can delete the old API.



Before deploying your API, you should edit your API's configuration files and make any database changes. When you are ready to deploy your API, you should prepare a manifest file in yaml format and download the latest API version in a ".jar" file.

Sample jar file name...

```
dataprocessor-1.0.233.jar
```

The manifest file contains information about the amount of memory the API will use, the number of instances of the API being deployed, the name of the API, and the ".jar" file being uploaded.

WHAT ARE CONFIGURATION FILE CHANGES AND HOW DO YOU CHANGE THEM?

Configuration file changes are the API settings described in an XML file on a separate platform like "Bitbucket" or "GitHub." As part of a deployment you might be asked to add new configurations or change existing ones. Some examples of configuration settings are "isAuthorized": true, "amount": 0.0, "statusCode": "95724950."

To edit the configuration files, it is ideal to clone the repository in Bitbucket or other platform to your local directory, make changes there and push your changes up to the repository in GitHub. If you don't have cloning abilities, you can simple edit the existing configuration file in Bitbucket.

STEPS TO PERFORM CONFIGURATION CHANGES

1. For each configuration you will add/change, perform a ctrl-f search of the variable to see if it's already existing.
2. If the variable exists, comment out the entire line and copy the new line directly underneath it.
3. If the variable does not exist, simply add the new variable to the most related configurations or add it at the very bottom
4. Avoid removing the old configurations in case you need to remove the changes in the future.

HOW TO PERFORM DATABASE CHANGES

You will need a database server setup for your organization in which you can connect to the appropriate environment (production, testing, etc..). Some examples are DBeaver, SQLite, and MySQL. Prior to deployment, make sure you check the database to see whether the tables in which you want to add columns already do not contain the columns you plan to add by performing a search query. For the database changes given to you, you will need to create a series of commands to reverse all changes incase the deployment is to be rolled back or reverted to the original state of the API.

STEPS TO PERFORM DATABASE CHANGES FOR DEPLOYMENT

CHECK DATABASE TO SEE IF NEW CHANGES ARE NECESSARY

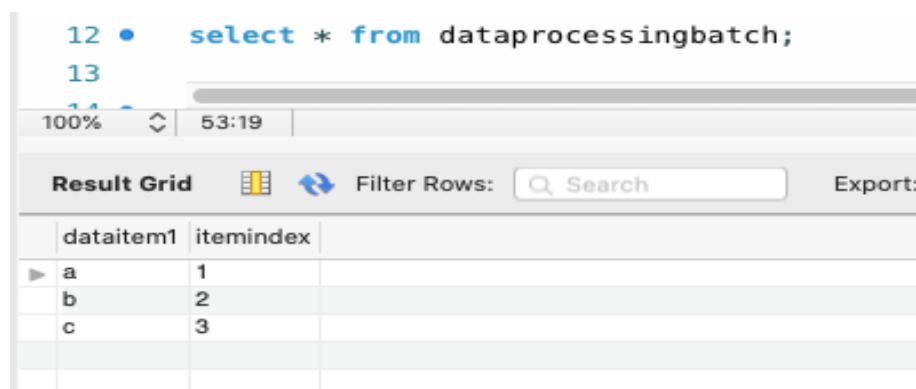
1. Read the SQL statements provided by your database engineers and check if it contains insertions (keywords in statement include "insert"), deletions (keywords are "delete") or creations (keywords are "create").
2. If you are creating a new table, check to see if the already table exists by writing the command

```
"Select * from {insert_table_name}"
```

with the table name you are supposed to create (the "*" indicates all columns).

3. Run the command by highlighting it and clicking the lightning bolt button up top.
4. If the table's contents show up, you may need to bring this issue up with any engineers or involved in these changes. If the table does not show up, then that is good and you are ready to create the table yourself.

We see that the table "dataprocessingbatch" exists because it lists all contents below. This is an issue if you were required to create a table.



```
12 • select * from dataprocessingbatch;
13
```

100% 53:19

Result Grid Filter Rows: Search Export:

	dataitem1	itemindex
▶	a	1
	b	2
	c	3

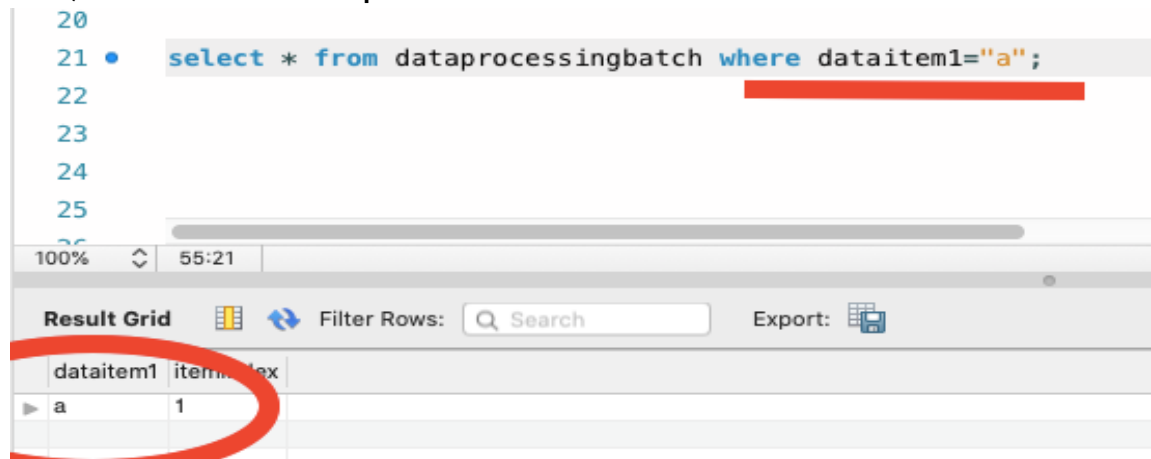
5. On the other hand, if you were required to delete a table, then it is good news if the table shows up in your query, as you may then proceed to delete the table.
6. If you are required to insert new columns in a table, check to see whether these already exist in your table by running the below query with either one or more columns you want to search for in your table.

```
"select {column_name 1, column_name 2, ...} from {table_name}"
```


- If you are required to insert new rows in a table, check to see whether these already exist by running the below query. Use "AND" in between each condition you are searching for.

```
"select * from {table_name} where {column_name 1 = 'value' } AND {column_name 2 = 'value'}
```

Here, we searched for the specific row where the column "dataitem1" had the value "a."



WRITE COMMANDS TO REVERSE DATABASE CHANGES INCASE OF ROLLBACK

- Open a Notepad++ document or any equivalent text editor
- Copy your official database commands to this editor for your personal reference
- A few spaces below the official commands start writing the reversal commands according to the following steps.
- If your official database commands involve creating tables, your reversal command should delete the table like the statement below.

```
Drop table {table_name}
```

- If your official commands involve deleting tables, there is no command than can bring back all values in an extensively large table, so double check this command with engineers and management.
- If your official commands involve inserting columns, your reversal command should delete the columns using the below statement.

```
Alter table {table_name} drop column {column_name_to_drop}
```


7. If your official commands involve inserting rows, your reversal command should delete the rows using the below statement.

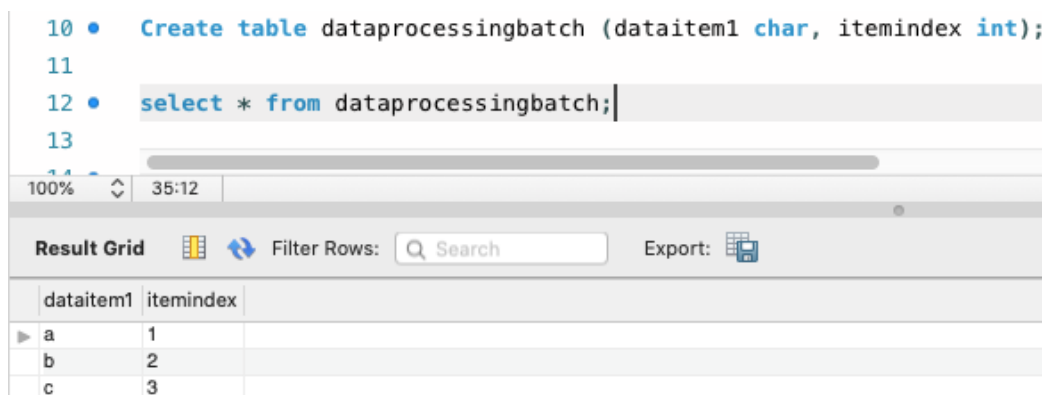
```
Delete from {table_name} where {column_name} = 'value' AND {column_name} = 'value'}
```

8. Save your reversal commands with an identifiable name like "db_rollback_Aug05_2020.txt" in your API deployment folder in case you need to run these commands during a deployment rollback.

RUN YOUR OFFICIAL DATABASE COMMANDS FOR THE DEPLOYMENT

1. Copy and paste any SQL statements provided by the SME into any database page.
2. Highlight each individual line of command and select on the lightning bolt button to run them one by one.
3. Perform a final search on this on all database changes to see if they have been added.

Check for existence of table after creation of table



HOW TO PERFORM A CANARY DEPLOYMENT

Once you have done all configuration file updates and database changes, you can proceed with the canary deployment. Canary deployments ought to be performed when the API user traffic is low. This means when no one is using the API, which is usually sometime after work hours (after 5 pm) and before offshore time (if the API is used in another country with a different timezone). While some platforms like Gitlab and Jenkins allow you to automate a canary deployment through simply clicking a button, sometimes you will need to resort to a manual deployment where you download the API to be deployed and run a command. In a manual canary deployment you need to create your own manifest file and download the new version of the API to be deployed from the deployment pipeline. These two files respectively would be titled something like "manifest.yml" and "{APIName version number}.jar." Before deployment, each of your API folders should contain a manifest and jar file ready to go.

STEPS TO DEPLOY:

SETUP YOUR FILES FOR DEPLOYMENT

1. Setup your file system with the appropriate folders and subfolders to allow for a smooth deployment.
2. Create a folder called "Deployments" in your local folder using the Unix command **"mkdir Deployments."** This will help you to store all your deployment files.
3. Enter the deployments folder with the command **"cd Deployments."**
4. Use the command **"mkdir {folder name}"** to create a new folder for you upcoming deployment. Make the folder name as the date of deployment as shown in the file list below.

Organize deployments in mail folder by date of deployment

```
/Users/Sahana/Deployments ls -l
total 0
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:43 June2nd2020
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:44 May1st2020
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:43 May22nd2020
drwxr-xr-x  8 Sahana  staff  256 Oct 13 15:41 Oct13th2020
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:43 Sept12th2020
/Users/Sahana/Deployments
```

5. Enter the folder with the date of your upcoming deployment using **"cd {folder name}."**
6. Create different folder names for each API you will deploy on this date using the command **"mkdir {folder name}."** Make the name of your folder as the API name as shown below.

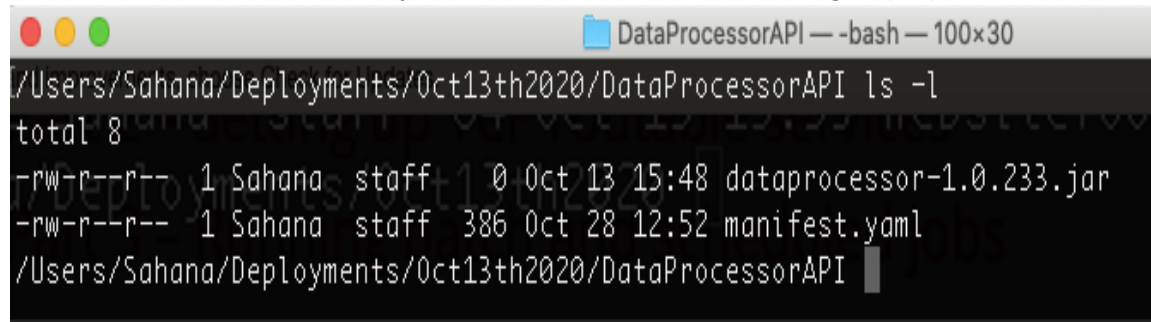
Within deployment date folder, organize by each API being deployed.

```
/Users/Sahana/Deployments/Oct13th2020 ls -l
total 0
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:33 DataProcessorAPI
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:37 ImportBatchAPI
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:39 ManagerAPI
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:41 RequestLookupAPI
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:41 StagingTableAPI
drwxr-xr-x  2 Sahana  staff   64 Oct 13 15:39 WebsiteToolAPI
/Users/Sahana/Deployments/Oct13th2020
```

7. Download the jar file for your API into the corresponding folder with the API name. Your jar file should be available in the location to which you deployed your API to the test environment (Jenkins, Gitlab, etc...). Do this for all APIs you are deploying.

8. Copy your company's manifest file to the same API folder. The contents of your API folder should be similar to the below picture with the jar file and manifest file together. Also copy your company's manifest file into each folder for every API you are deploying that day.

Inside API folder, store the API jar file and manifest file containing deployment information

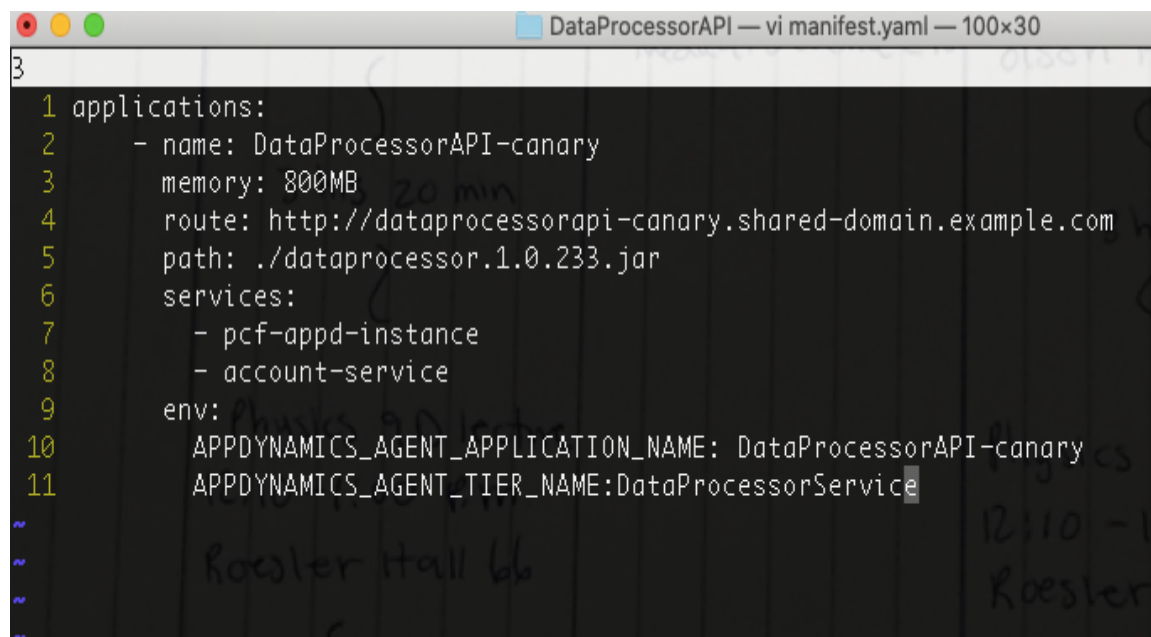


```

DataProcessorAPI — -bash — 100x30
/Users/Sahana/Deployments/Oct13th2020/DataProcessorAPI ls -l
total 8
-rw-r--r--  1 Sahana  staff   0 Oct 13 15:48 dataprocessor-1.0.233.jar
-rw-r--r--  1 Sahana  staff 386 Oct 28 12:52 manifest.yaml
/Users/Sahana/Deployments/Oct13th2020/DataProcessorAPI

```

9. Open your manifest file and set it up like so. Fill in the API name, host name, memory, disk space, instances, jar file path, build pack, and list of services used. You can browse these details by clicking on the API in the PCF website.



```

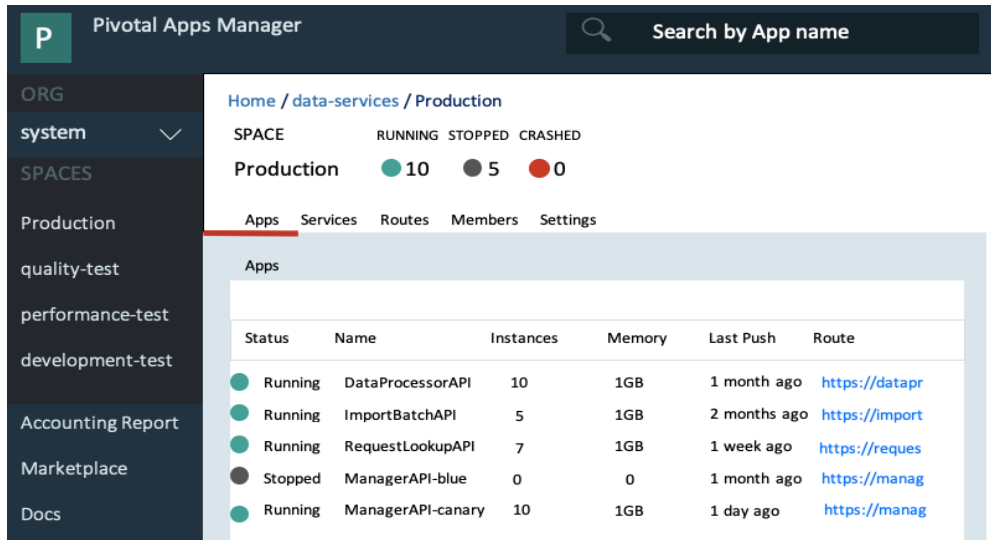
DataProcessorAPI — vi manifest.yaml — 100x30
3
1 applications:
2   - name: DataProcessorAPI-canary
3     memory: 800MB
4     route: http://dataprocessorapi-canary.shared-domain.example.com
5     path: ./dataprocessor.1.0.233.jar
6     services:
7       - pcf-appd-instance
8       - account-service
9     env:
10      APPDYNAMICS_AGENT_APPLICATION_NAME: DataProcessorAPI-canary
11      APPDYNAMICS_AGENT_TIER_NAME:DataProcessorService
~
~
~
~

```

START THE DEPLOYMENT PROCESS

1. Open the API hosting website (PCF in this example) so you can view it immediately after deployment.

List of APIs in PCF's production environment



The screenshot shows the Pivotal Apps Manager interface. The left sidebar contains a navigation menu with 'system' selected. The main content area shows the 'Production' space with a summary of 10 running, 5 stopped, and 0 crashed apps. Below this is a table of apps.

Status	Name	Instances	Memory	Last Push	Route
Running	DataProcessorAPI	10	1GB	1 month ago	https://dataprocessorapi
Running	ImportBatchAPI	5	1GB	2 months ago	https://importbatchapi
Running	RequestLookupAPI	7	1GB	1 week ago	https://requestlookupapi
Stopped	ManagerAPI-blue	0	0	1 month ago	https://managerapi-blue
Running	ManagerAPI-canary	10	1GB	1 day ago	https://managerapi-canary

2. Navigate to the folder of the first API you want to deploy. Double check that this folder has the manifest file and the jar file for deployment.
3. Login to PCF with your username and password first before deploying using the format below. "cf login -a URL."

***The URL is essentially the one in the address bar when you login to the PCF website to your space.*

```

/Users/Sahana cd Deployments/
/Users/Sahana/Deployments cd Oct13th2020/
/Users/Sahana/Deployments/Oct13th2020 cd DataProcessorAPI/
/Users/Sahana/Deployments/Oct13th2020/DataProcessorAPI ls -l
total 8
-rw-r--r-- 1 Sahana staff 0 Oct 13 15:48 dataprocessor-1.0.233.jar
/Users/Sahana/Deployments/Oct13th2020/DataProcessorAPI cf login -a http://cf-space.production.com

```

4. Enter your PCF username and password when it prompts you and enter the space name as "Production" or whichever environment is production for you upon the next prompt.

***Verify once to see that you are logged into the correct space by entering the command "cf target." Then it will show you the current space in PCF you will be deploying to.*

- Run the command **cf push manifest.yaml** to start the deployment.

```

/Users/Sahana/Deployments/Oct13th2020/DataProcessorAPI ls -l
total 0
-rw-r--r--  1 Sahana  staff  0 Oct 13 15:48 dataprocessor-1.0.233.jar
-rw-r--r--  1 Sahana  staff  0 Oct 13 15:49 manifest.yaml
/Users/Sahana/Deployments/Oct13th2020/DataProcessorAPI cf push manifest.yaml

```

- Go onto PCF and check to see that the API is being deployed.

***Some APIs may crash immediately after deployment, which is not an issue. Simply wait for the API to get back up and running.*

- Then go to the old API (the non-canary one) and scale it down by one instance by clicking on the blue "scale" button and decreasing the instances.

***Depending on your API hosting platform, the name of the old API might be automatically changed to "APIName-blue," as "blue" indicates the old version.*

web				
Instances	2	Memory Allocated	128 MB	Disk Allocated 1 GB
Autoscaling <input type="checkbox"/>				
#	CPU	Memory	Disk	Uptime
0	0%	50.6 MB	107.26 MB	15 min

- Monitor your canary API for crashes or 400 and 500 errors through any logging system in the API hosting site or any your company has for checking API status updates like Splunk. You may click on the API in the production environment, then navigate to the "Logs" button to check for errors.

Pivotal Apps Manager Search by App name

ORG system

DataProcessorAPI Overview Services Routes **Logs** Tasks Settings

Last Push: 12:07pm

Events

- scaled app 1 instance
- stopped app
- started app
- created app

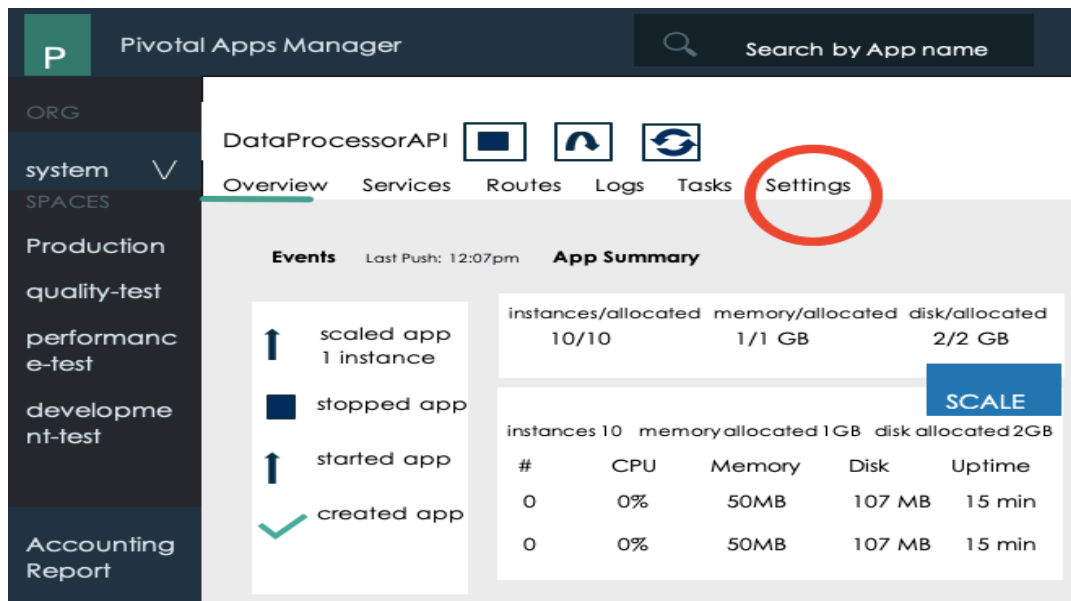
App Summary

instances/allocated	memory/allocated	disk/allocated
10/10	1/1 GB	2/2 GB

SCALE

#	CPU	Memory	Disk	Uptime
0	0%	50MB	107 MB	15 min
1	0%	50MB	107 MB	15 min

9. If the API has a website component, check to see that all changes are reflected on the website.
10. If the API does not produce any 400 or 500 errors in the logs, and all post deployment testing is successful, click on the “settings” tab and select the option to change the API name and remove the “-canary” after the API name.



11. Then delete the API-blue by clicking on the “settings” tab and selecting “Delete API” at the bottom.

CONGRATULATIONS ON YOUR DEPLOYMENT

I hope this article offers you the tips and tricks you need to successfully get through your deployments. When doing a deployment for the first time, it is important to have a more experienced person with you to ensure the changes go through precisely. Use this article as a guide to stay organized, employ efficient strategies and to understand the reasoning behind the steps you're taking. By now you would have a more thorough understanding of when, why and how deployments take place.