|  |  |
| --- | --- |
|  |  |



***FIRST DRAFT***

**Deployment Automated – 2.2**

**FedEx Proprietary and Confidential**

**Printed Copies for Reference Only**

Prepared by: ITCC

Manager: Chris Kellmeyer

Last Revision: 3/6/2020

**Table of Contents**

[1. Document Information and Revision History 3](#_Toc34321651)

[2. Introduction 4](#_Toc34321652)

[3. Scope 4](#_Toc34321653)

[4. Request / Task lifecycle 4](#_Toc34321654)

[5. Prerequisites 4](#_Toc34321655)

[6. Pipeline Flow 4](#_Toc34321656)

[7. Explaining the architecture: 5](#_Toc34321657)

[8. Flow Diagram 6](#_Toc34321658)

# Document Information and Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Revision Notes** |
| 1.1 | 3/5/2020 | Eshwar Anne | First draft |
| 1.2 |  |  |  |
| 1.3 |  |  |  |
| 1.4 |  |  |  |
| 1.5 |  |  |  |
| 1.6 |  |  |  |
| 1.7 |  |  |  |

# Introduction

* 1. This document explains, and plots executable steps required to deploy in Dev test and production environment. This process is completely automated and has multiple checkpoints to verify the process.

# Scope

* 1. To be able to understand the new deployment process and execute it using gitlab CI/CD pipelines.

# Request / Task lifecycle

Change curl commands on c0001195.test.cloud.fedex.com

*Push the changes for respective servers*

Run the pipeline for new Applications

Run the pipline for existing applications.

# Prerequisites

* 1. Access to gitlab (dev/test, gcp-prod, MC deployment server, predeployment repositories)
  2. Access to gcp servers and c0001195.test.cloud.fedex.com

# Pipeline Flow

* 1. The Parent pipeline triggers (predeployment pipeline) the on-prem (mc-deployment-server-uf) and gcp pipeline (gcp-prod-deployment-repo) and 3 files (cicd\_gcp.sh, cicd\_onprem.sh, indexes\_list.txt) should be changed on c0001195.test.cloud.fedex.com.
  2. The deployment should be divided in 2 parts:
     1. Deploying new apps

If you are deploying new apps, log on to c0001195.test.cloud.fedex.com and under /opt/predeployment/ for cicd\_gcp.sh, cicd\_onprem.sh we would be editing the variables for the curl command.

"variables[app]=$index" -F "variables[new]=yes"

For new apps the variable “new” should be yes.

* + 1. Updating existing apps

For existing applications, the variable “new” should be no.

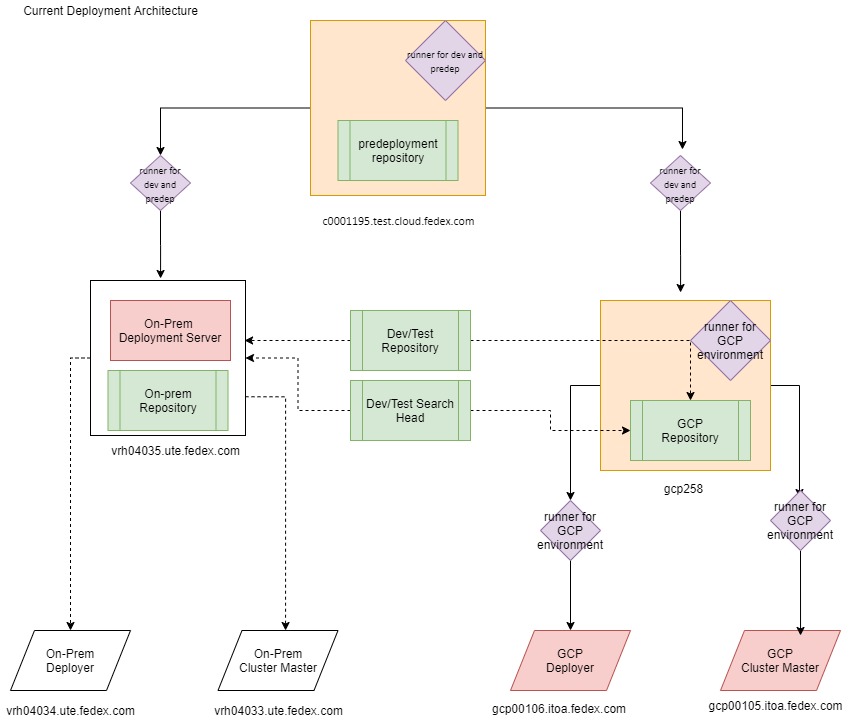
"variables[app]=$index" -F "variables[new]=no"

* + 1. Update indexes\_list.txt with the respective indexes.
  1. Once the files are updated on the host, add the changes to the commit and commit the change to the branch and push it to the repository.
  2. Once the changes are pushed, the pipeline would be triggered for the respective applications for both on-prem and gcp. Each triggered pipeline can be monitored separately by opening the pipeline page in each gitlab repository.
  3. If everything goes well, you should see a green checkmark on each job generated. If something fails, we would see a red cross for that specific job. If you want to check the logs for any job just click on the job, and again on the pop-up, this help to troubleshoot any failures in the job.
  4. If any of the pipeline fail the testing part of job, we can get notified with the test log to specify why the pipeline failed.
  5. Once the pipelines are run successfully, login into deployment-server (on-prem) and reload the serverclass to CM and Deployer of On-prem. Once all the changes are accounted for, use the appropriate the command to push the changes to indexers and search heads. (this will be veined once dual feeding is stopped)
  6. In GCP login to deployer and Cluster master to validate the changes. Once all the changes are accounted for, use the appropriate the command to push the changes to indexers and search heads.

# Explaining the architecture:

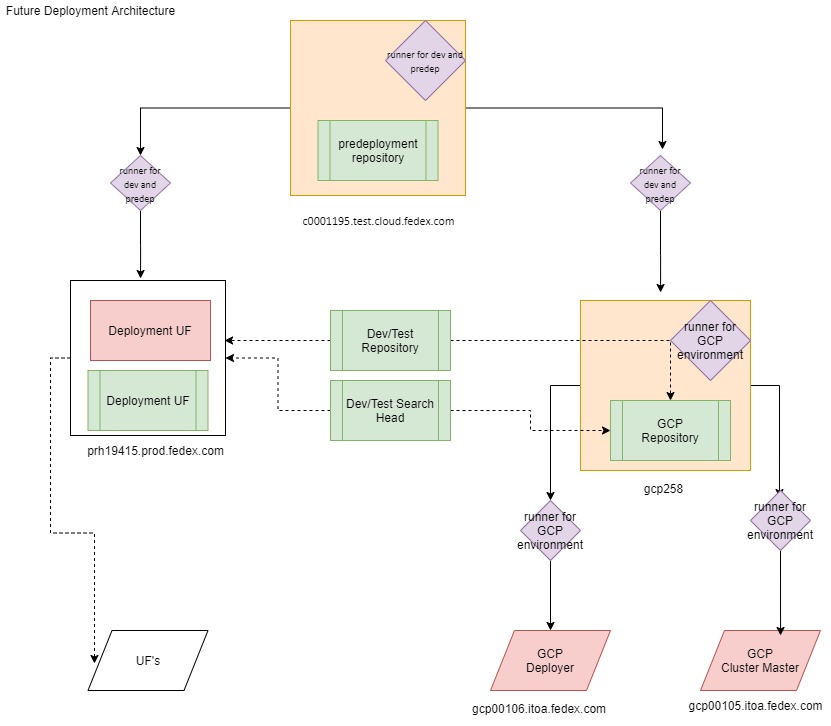
* 1. All the configurations are being copied over from the dev/test environment. The inputs and props are from the dev/test Deployment server, the search app is from the dev/test search head.
  2. The pre-deployment repository is on a cloud test box which acts as a runner for the repository and a place to run the curl command scripts. This repository is the top-level pipeline which triggers the end pipelines using curl commands and control the behavior by changing respective variables.
  3. Once the parent pipeline is triggered, the end pipelines may be triggered multiple times depending on how many indexes are listed in the indexes\_list.txt. Each time a pipeline is triggered, 3 jobs are generated.
  4. The 3 jobs are testing building and staging the deploy. The test job validates the quality of the props from dev/test, the build job copies the sets up the staging part of the pipeline. Finally, the staging deployment job, makes all the changes to the target server and add to the change bundle.
  5. Once the changes are made, validate the changes and push the changes from respective servers and using the respective commands.

# CurrentFlow Diagram



* Red boxes are Target machines that the pipeline would eventually be would run on.
* Green Boxes are repositories used.
* Purple boxes are the runners used.
* Solid lines are the flow path.
* Dotted lines are the repository being used on different servers.

# Future Deployment Flow Diagram:



----------------------------------END OF THIS DOCUMENT------------------------------------------