*OPS-DEVOPS*

1. **Prerequisites**:
   1. **Aws cli** should install in ***encode*** server, ***live*** server, ***dev*** server and in ***local*** also from where you run the **setupDeployent.sh** file.
   2. Create one **Iam** user which has the below permissions.
      1. [AmazonEC2FullAccess](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonEC2FullAccess)
      2. [IAMFullAccess](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FIAMFullAccess)
      3. [AmazonS3FullAccess](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonS3FullAccess)
      4. [AWSCodeDeployFullAccess](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSCodeDeployFullAccess)
      5. [AWSCodeDeployRole](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2Fservice-role%2FAWSCodeDeployRole)
      6. [AWSCodePipeline\_FullAccess](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSCodePipeline_FullAccess)
      7. [AWSCloudFormationFullAccess](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSCloudFormationFullAccess)
      8. [EC2InstanceConnect](https://console.aws.amazon.com/iam/home" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FEC2InstanceConnect)
   3. **Configure aws** with generated *secret access key* and *access key* from new user at same four places mentioned in first point.
   4. Cerate one bucket called masterops and upload the folders that given by devops team.

1. Steps that will be run very **first** time:
   1. Installing ***code deploy agent*** on each server:
      1. Run the ***./codeDeployAgent.sh {servername}*** script on each server.

Ex: ./codeDeployAgent.sh ops-dev

       ./codeDeployAgent.sh ops-prod

1. It will ask your **OS** name , please enter either *ubuntu* or *centOs*.
2. Creating **pipeline** and **stack** for respective project and also your first deployment:
   1. Aws should **configure** here.
   2. Project repo must be **cloned** here and after it's clone, ***./setupDeployment.sh*** should run from inside repo folder.
   3. Run ***./setupDeployment --init*** to generate the ***devops.json***
      1. Fill the ***devops.json*** with respective values
      2. Push that ***devops.json*** to your project repo

1. Run .***/setupDeployement.sh {branchName}*** script from your local

Ex: ./setupDeployment.sh master

       ./setupDeployment.sh development

1. Firstly you have to run for the ***./setupDeployment.sh development*** that targets your **development** server.
   * It clone the project repo in the development server inside the **public\_html** and also install the **vendor** folder and **robots.txt sitemap.xml** files with the respective permission.
   * At last It will give you one ***.gitlab-ci.yml file*** , if you not have with you otherwise it will give you  text if you already have that file, so copy that text to your existing .gitlab-ci.yml  file.
   * After that **push** the .gitlab-ci.yml  file in your project repo.
   * Modify the **devops.json** and **add** the things you want to send from dev server to live server in action property
   * Now run the ***imageUpload pipeline*** again from the gitLab action.

1. After that you have to run ***./setupDeployment master*** to target your **build** server and **live** server.
   * It will do **encryption** of php files if its true in devops.json
   * Running **gulp**
   * Zip the **artifact**
   * Take **backup** if its true
   * And **unzip** the artifact and made the respective folder their like **cache** ,**robots.txt** and **sitemap.xml**
   * Also install **vendor**
   * And also unzip the things that present on the additional **artifact**

 3. Steps that you have to perform **after setup** all this:

1. Upload the **devops.json** and enter the path of files and folder that you want to send from the dev server to live server.
2. From now onwards you have to run the pipeline through **git lab actions**, for running the **imageUpload** pipeline run the pipeline called imagesUpload-job .
3. For running the build pipeline you have to run the pipeline called **build-job**